Lightweight Monocular 3D Vehicle Detection in Calibrated and Uncalibrated Scenarios

Eric Henriksson Martí

Abstract

This paper explores the 3D vehicle detection problem based exclusively on monocular images, placing emphasis on creating the simplest possible yet functional models for the task. The main concept behind the CenterNet system is taken as a foundation for this purpose, involving the detection of vehicle centers and the regression of other parameters that provide further location and pose information. Several lightweight architectures with U-Net-like feature extractors are tested to implement this concept in a simplified manner, which use only a fraction of the parameters that constitute the original CenterNet model. A case with calibrated cameras is first addressed, where several model variants capable of predicting the location and orientation of 3D bounding boxes enclosing vehicles are designed for the KITTI dataset. The feasibility of estimating this same information when dealing with uncalibrated cameras is later explored through the simplification of the 3D detection problem into 2D. Images from the Ko-PER traffic monitoring dataset are first transformed into their bird's-eye view (BEV) versions using homographies, and vehicles are then detected as rotated bounding boxes in a 2D plane.

Index Terms

Monocular 3D object detection, Objects as points, Fully Convolutional Networks (FCN), Projective geometry, Calibrated cameras, Uncalibrated cameras, Homography

I. Introduction

DETECTION systems capable of estimating the position, dimensions and orientation of objects in 3D space have received a lot of attention within the computer vision research community over the past years [1][2]. Being able to delimit vehicles with oriented 3D bounding boxes is a crucial task in fields like urban autonomous driving, and it has also proven to be beneficial in other applications related to traffic monitoring. These kinds of detection systems have had a tendency to rely on sparse input data from LiDAR sensors, which can provide reliable depth measurements in good visibility conditions. However, due to their expensiveness and considerable size, LiDAR sensors are not a viable option in many scenarios.

Considering the convenient access to cameras in current times, estimating aspects like depth and orientation from monocular images is a valuable alternative to the use of high-end sensors and the difficulties these involve. That being said, monocular 3D object detection is considered a very challenging task due to its reliance on only a projected representation of the real world as an input. With the appearance of Deep Learning, however, creating systems capable of associating certain visual cues present in images to measures of 3D location and pose has become a feasible reality. Still, these systems struggle to reach the same level of performance as that of others with more complex inputs.

Most of the research in learning-based monocular 3D vehicle detection, especially that related to autonomous driving, assumes that the calibration parameters of the cameras from which images originate are known. Having access to these camera matrices makes it possible to determine the line in space on which a point in an image lies, which in turn helps establish a relationship between the 2D and the 3D domain. Note that these systems learn to relate visual cues to 3D information for a particular calibration setup, meaning that they are not capable of providing reliable predictions when inferring on images from a camera with a different calibration.

While having access to camera calibration information and 3D bounding box annotations is common in fields like autonomous driving, there are plenty of other applications where neither of these elements can be easily obtained. The calibration of most traffic monitoring and security cameras for instance tends to be unknown. It is also not possible to annotate their footage with 3D bounding boxes, as unlike the case of 2D bounding boxes that can easily be drawn on images, 3D annotations need some form of point clouds to be placed in space. This makes the problem of uncalibrated monocular 3D vehicle detection an even bigger challenge, and one that can only be achieved to a certain extent through significant simplifications.

This paper is meant as a practical exploration on how the task of monocular 3D vehicle detection can be accomplished through the design and use of the simplest possible fully-convolutional networks (FCNs) built completely from scratch. The purpose of the study is not to outperform existing solutions, but rather to evaluate to which level these can be simplified in terms of size and inference speed while maintaining a decent functionality. The idea of representing objects as points and then regressing other parameters related to location and pose introduced in the CenterNet paper [3] is taken as inspiration and used

Author: Eric Henriksson Martí, eric.henriksson@autonoma.cat
Advisor 1: Dimosthenis Karatzas, Computer Vision Centre, Universitat Autònoma de Barcelona
Advisor 2: Marçal Rossinyol, AllRead Machine Learning Technologies S.L.
Submitted: September 2022

as a foundation for the design of these networks. Instead of the modified DLA-34 that the original CenterNet employs for its 3D detector, a simpler U-Net type architecture is used as a feature extractor. Several variations of this architecture with different layer depths and widths are tested and the performance of the system is compared to that of similar models.

The KITTI dataset is first targeted, which is comprised of images of urban scenes taken from car-mounted cameras with known calibration parameters. Only objects belonging to the car category are considered. Predictions are carried out for the location of the vehicle center points in the images, their width and height in said images, their position relative to the camera in terms of depth, and their local orientation around the axis perpendicular to the road. Since the intra-class variations in the width, length and height of car objects are rather small, the average dimensions of car objects in the dataset are assumed for the detections for the sake of simplicity.

Next, the Ko-PER dataset is addressed, which contains footage from two static traffic monitoring cameras overlooking an intersection from a higher angle. While this dataset also includes calibration matrices for each of the involved cameras, it is assumed this information is not available and an alternative approach for an uncalibrated case is tested. Images are converted to bird's-eye view (BEV) versions of themselves with a homography, through the correspondence of key-points between them and a map of the intersection available in the dataset. The 3D detection problem is then instead treated like a 2D detection problem consisting of the estimation of rotated bounding boxes. Predictions of vehicle centers and their global orientation are carried out here from a BEV perspective, while the width and length of said vehicles is once again taken directly from intra-class average values. Due to its simplification to 2D, this approach assumes vehicles are placed on top of a completely flat surface and does not provide any estimations in terms of their position in the vertical axis.

II. State of the art

This section presents an overview of public work related to the topic at hand. It first addresses the different kinds of 2D object detectors available at this point in time, as some of these are the foundation on which many 3D bounding box detectors are built. Next, the state of the art in terms of object pose recovery methods is summarized, along with the main concepts these methods are based on. Considering the objectives of the study, a particular focus is placed on the subject of 3D bounding box detectors that are only provided RGB monocular images as input. Finally, the most common metrics and datasets used in 3D detection are also briefly introduced.

A. 2D object detection

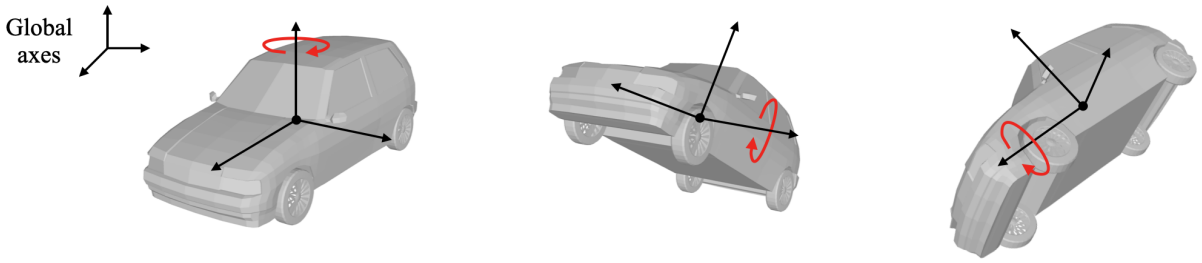
The detection of objects in images and their delimitation through 2D bounding boxes is one of the areas of computer vision that has received the most attention during recent years. There are generally two approaches to the task of 2D object detection. The first is the use of traditional machine vision methods that rely on motion [4] and/or background modelling [5] to distinguish objects of interest from the background of an image. The other is based on employing deep convolutional neural networks (CNNs) that are capable of learning image features in a way that enables classification and regression tasks. The latter has proven to be the most successful at performing accurate predictions under a variety of challenging conditions, and has thus been the main focus of modern research in the field since its introduction.

When it comes to deep learning-based detectors, some of the first successful attempts came with the idea of region classification initially proposed by R-CNN [6] object detector. R-CNN relies on extracting many region proposals from an image, cropping and resizing these to a set format and classifying them with a deep network. Fast R-CNN [7] improves on R-CNN by first extracting the image features and then cropping these for each region proposal, rather than having the extraction be carried out for each region proposal independently. Nevertheless, both methods suffer from slow computation times due to their dependence on low-level region proposal mechanisms.

Faster R-CNN [8] later introduced the use of implicit anchors in object detection. It uses a region proposal network (RPN) that exclusively generates region proposals by ranking sets of bounding boxes with fixed shapes (anchors), while a different network takes care of detecting objects in these. The use of a RPN that shares computational efforts with the base object detection network enables generating region proposals much faster compared to algorithms like selective search.

Further major improvements in inference times arrived with one-stage detection systems, reducing the multi-step process of detectors like Faster R-CNN to a singular network that can be trained end-to-end. YOLO [9] divides images into a grid of $S \times S$ smaller cells. Cells that are are considered to include the centre of an object in question are responsible for the prediction of said object. A set of k bounding boxes and a confidence score is predicted for each of these cells, where the confidence score indicates how likely it is for that cell to include an object and how well the bounding box fits it (based on an IOU metric). SSD [10] and RetinaNet [11] are some other popular one-stage detectors. The former makes use of convolutional layers of different sizes to predict boundary boxes and classes directly from multi-scale feature maps. The latter introduces the concept of focal loss to deal with class imbalances between foreground and background, and uses a feature pyramid network (FPN) [12] in its backbone.

As an alternative to the use of anchors, some works propose the use of keypoint estimation for object detection. CornerNet [13] is for instance built on the idea of detecting two of the corners of a bounding box as keypoints. ExtremeNet [14] instead detects the four points that delimit the top, left, bottom and right-most of an object, as well as its center. Both of these methods require a



a) Azimuth rotation (θ) b) Elevation rotation (ϕ) c) Roll rotation (α)

Fig. 1: Most 3D vehicle detectors assume gravity-aligned local axes (a), as other orientations (e.g., b & c) are not common.

stage of combinatorial grouping after the keypoint detection that slow down their computation. CenterNet [3] presents a simpler approach, as it only predicts an individual center point for each object and avoids the need for grouping and post-processing steps. Since this project builds on CenterNet to a great extent, a more detailed explanation of this detector is later provided in Section II-F.

B. Object pose recovery

The problem of object pose recovery usually refers to either carrying out 3D bounding box detection or full 6D pose estimation [15]. 3D bounding box detectors are usually meant to generate amodal 3D bounding boxes [16] at a category level, and assume images are gravity aligned. These bounding boxes are thus parameterized with center $T = (t_x, t_y, t_z)$, size $D = (d_x, d_y, d_z)$ and orientation $R(\theta)$ around the gravity axis. Methods for full 6D pose estimation on the other hand work at an instance-level and do not consider any alignment constraints at all. These instead focus on finding an object's 3D translation $T = (t_x, t_y, t_z)$ and its 3D rotation $R(\theta, \phi, \alpha)$.

C. 3D bounding box detection and representation

The work reflected in this report can be considered to be part of the 3D bounding box detection case, as vehicles are generally delimited with these kinds of box representations and can be considered to be placed on a flat ground (null elevation and roll angles, as shown in Figure 1).

If it is assumed the origin of an object's coordinate frame is in the center of its corresponding 3D bounding box, the coordinates of said bounding box's vertices can be expressed as $\mathbf{X}_1 = [d_x/2, d_y/2, d_z/2]^T$, $\mathbf{X}_2 = [-d_x/2, d_y/2, d_z/2]^T$, ..., $\mathbf{X}_8 = [-d_x/2, -d_y/2, -d_z/2]^T$, in the object's coordinate frame. Knowing the pose of the object in the camera's coordinate frame (R, T) and the intrinsic parameters K of the camera, it is common practice to obtain the projection of a 3D point $\mathbf{X}_i = [X, Y, Z, 1]^T$ in the object's coordinate frame on an image $\mathbf{x}_i = [x, y, 1]^T$ using Equation 1 below. Figure 2 displays the relationship between a point of a 3D bounding box and its projection on the image plane.

$$\mathbf{x}_i = K[R\ T]\mathbf{X}_i \tag{1}$$

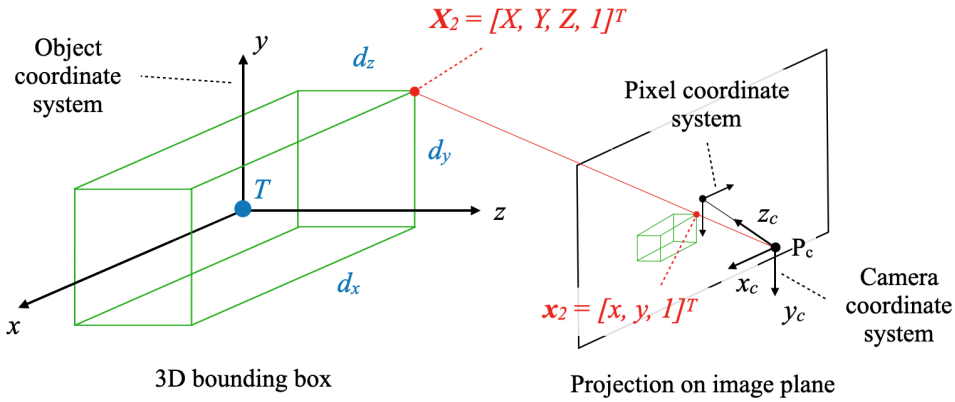


Fig. 2: Visual representation of the relationship between a point of a 3D bounding box and its projection in the image plane.

D. 3D detection methods

While 3D detection methods all share the same goal, these can differ from each other in terms of what they take in as an input. Some detectors only require monocular RGB images to perform object pose predictions, while others make use of additional depth and/or 3D shape information. There are also a variety of methods that use LiDAR data and/or stereo images, which generally have the potential to perform more accurate estimations. Despite the fact this study is based on object detection using only monocular RGB images, a short summary of the main representatives from all these different kinds of methods is still provided.

- **RGB images only**: The problem of object pose estimation from 2D images was initially addressed as a *perspective n-point problem* (PnP), exploiting geometrical information and establishing correspondences between 2D keypoints in an image and their corresponding position in 3D space [17]. Other approaches include building 3D models of the objects and finding the 3D pose in the image that matches the model the best [18][19].

  While several other methods have been proposed over time to deal with this problem, the appearance of CNNs has lead to a significant improvement in the performance of 3D detectors, and methods built on these networks have quickly become the state of the art. Most of the methods using CNN build on 2D detectors like the ones presented in Section II-A, introducing extensions that enable 3D pose predictions by exploiting appearance and projective geometry information present in the images.

  Certain methods consider geometric constraints to ease the estimation of 3D pose and object dimensions [20][21][22]. Deep3DBox [20] for instance proposes enforcing constraints related to projective geometry; the idea that the projection of the 3D bounding box should fit tightly within its corresponding 2D bounding box detection. CenterNet [3] extends its 2D detection capabilities to 3D detection by implementing what can be considered a one-stage version of Deep3DBox, making its inference times significantly faster (more details available in Section II-F).

  OFTNet [23] maps image-based features into an 3D voxel map through an orthographic feature transform. These voxel map features are later simplified to a 2D bird's-eye view (BEV), which helps support the estimation of detections by making use of integral image representations. The basic version of ROI-10D [24] suggests a new loss to lift 2D detections, orientation and scale to 3D, in a way that enables end-to-end training. MonoGRNet [21] presents a CNN that includes sub-networks for 2D detection, depth estimation, 3D location estimation and local corner regression. The model is trained in three different stages; the first one focusing on the backbone and the 2D detector, the second on the other modules responsible for geometric mechanisms, and a final end-to-end stage.

  Some recent studies have focused on trying to avoid the need for calibrated cameras in monocular detection systems [25][26], particularly in traffic monitoring applications. These are addressed in more detail in Section 15.

- **RGB images + depth information**: Accompanying image inputs with depth estimations has the potential to help methods achieve more accurate predictions. ROI-10D [24] includes an extension to its base version that makes use of disparity information obtained with SuperDepth [27], a self-supervised system for monocular depth estimation. There are other methods that utilize a similar approach [28][29][30], employing depth estimation models [31][32] trained on larger datasets and using these to augment the input RGB images. In [30], it is additionally proposed to use a multi-level fusion scheme, combining the disparity results from a stand-alone depth estimation model with the features from the targeted images at different levels for a more accurate 3D localization.

- **RGB images + 3D shape information**: Having access to the estimated shapes of the objects that are being targeted can also contribute to a system's ability to perform accurate predictions. 3D R-CNN [33] exploits shape priors specific to each object class by learning a low dimensional shape space from datasets of CAD models. It introduces what it calls a Render-and-Compare loss, which makes it possible to drive the optimization of 3D shape and pose with 2D supervision. In [34], the accuracy of 3D estimations through the use of bounding boxes is questioned, and a 3D scene representation is designed that encompasses information regarding the 3D shape of various objects. This allows for a finer perception of 3D geometry and occlusions. Deep-MANTA [35] presents a coarse-to-fine object proposal mechanism also relying on CAD models and annotated 3D parts, which too leads to a robust model capable of localizing vehicle parts even when they are suffering from occlusion. The use of shape priors to assist in 3D pose and shape recovery is also prevalent in [36], where these are encoded using keypoints learnt from a small keypoint annotated dataset. [37] focuses on optimizing the projection consistency between generated 3D hypotheses and 2D pseudo-measurements at two different scales. It employs a morphable wireframe model to provide 3D shape and pose, and considers unsupervised monocular depth and a ground plane constraint in addition to vehicle shape priors.

- **LiDAR or stereo-based information**: Certain methods employ stereo images, which naturally provide richer information than regular monocular ones. 3DOP [38] is for instance a stereo-based method that tries to minimize en energy function that takes into account object size priors, ground plane, point cloud densities and other prior knowledge about the scene. Stereo R-CNN [39] is yet another method that utilizes stereo imagery through the adaptation of the Faster R-CNN network

to stereo inputs, enabling the simultaneous detection and association of object in the pairs of images. It produces stereo bounding boxes, keypoints, dimensions and viewpoint angles that are formed in a learnt 3D prediction module.

There is also a variety of methods that use data originated by LiDAR sensors. MV3D [40] proposes a sensory-fusion network that takes in LiDAR point clouds and RGB images as inputs. It is comprised of two subnetworks, one dedicated to 3D object proposal generation and another to feature fusion from several views. Frustrum PointNets [41] on the other hand use LiDAR point clouds exclusively, aligning candidate points from 2D detections to estimate the appropriate 3D bounding boxes. PointRCNN [42] also operates on raw point clouds, introducing a two-stage framework in which the first stage involves the generation of bottom-up 3D proposals through foreground/background segmentation, and where the second stage adjusts these in canonical coordinates. RoarNet [43] yet again uses RGB images as inputs in addition to LiDAR point clouds. It first estimates the 3D poses of objects from images through a 2D detector that provides feasible candidates, and then the point clouds corresponding to the candidate regions are processed recursively to determine the final 3D bounding boxes.

In [15], object pose recovery methods are differentiated into categories according to what problem modelling approaches they employ. The first category includes models that approach the pose estimation task as a classification problem. These models consist of a offline training phase where a classifier is trained on real or synthetic data. Training data is annotated with pose parameters akin to those introduced in Section II-B, and may or may not include information regarding the dimensions of the objects. In the online testing phase, an image is fed to the system as input. Methods built on 2D detectors first extract a 2D bounding box around the objects of interest, which are later lifted to 3D. Some cases, depending on the input, apply a pre-processing step and generate 3D hypotheses. In 6D pose estimation methods, features are extracted from the input and processed by the trained classifier, which estimates the pose. Certain methods apply a refinement process to the output of the classifier, and eventually provide final pose predictions after some filtering.

The second category is comprised of methods that rely on the regression of object pose parameters. The general process these methods follow is similar to that of classification ones, the only difference being the use of a trained regressor instead of a classifier. Since classification and regression are not mutually exclusive, the third category considered in [15] includes methods that combine both means within a singular architecture. Some of these first perform the classification and then refine the results through a regression-based step. Others instead first perform regression and then classification, or carry out both in a single-shot manner. Figure 3 summarizes the overall process followed by classification, regression, and classification & regression methods.

The fourth category involves methods that are based on template matching, estimating the object's pose parameters through the matching of template annotations and representations in the feature space. They consist of an offline feature extraction

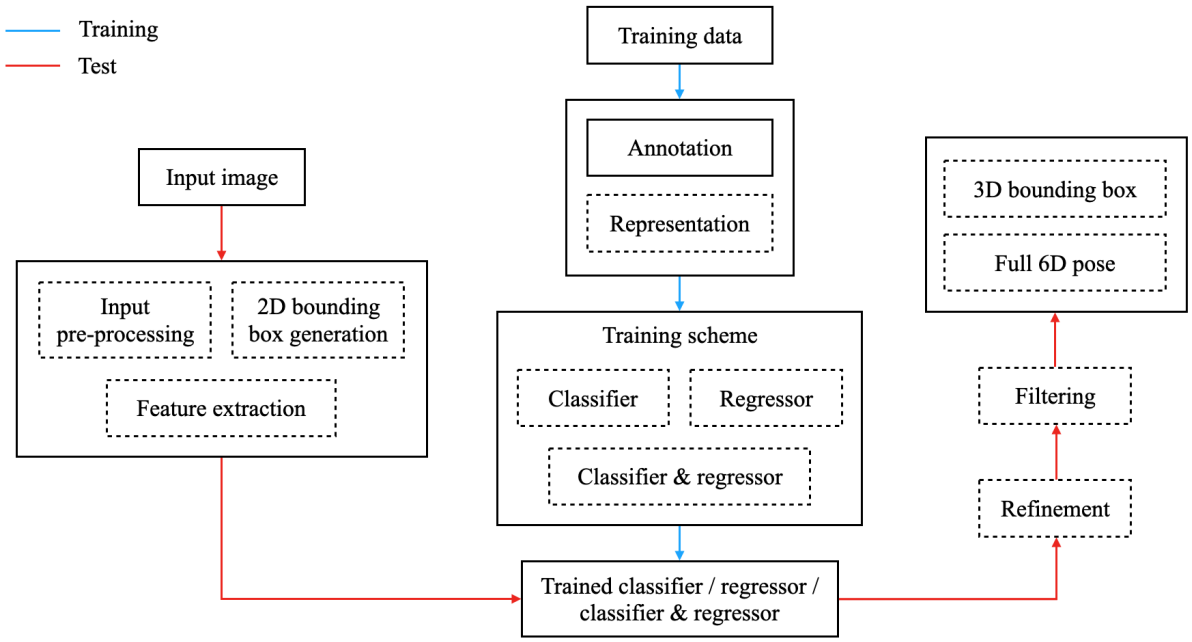


Fig. 3: Overall scheme of object pose recovery methods based on classification, regression, or both, as seen in [15]. Blocks marked with a solid border are essential to all methods, while the ones marked with dashed borders are dependent on the specific method at hand.

phase and an online testing phase. In the former, 3D or 6D pose annotated templates that are part of the training data are characterized as feature descriptors. While features are originally hand-crafted based on available shape, geometry and appearance data, current state of the art methods involve the learning of said features using neural networks. During the testing phase, a sliding window is moved over an input image. This window is represented with a feature descriptor, and is compared based on a distance measure in the feature space with the stored templates. When a match where the distance is the lowest is found, the window is assigned the pose parameters of the matching template.

The final category is based on methods that utilize point-pair feature matching. This approach first involves an offline phase where the 3D model of an object of interest is represented through point-pair features (PPF) stored in a hash table. In the followup online phase, PPFs extracted from a test image are compared to the mentioned model representation. The best matching candidates then vote for the final pose parameters.

An extensive categorization of most 3D bounding box detectors and 6D pose estimation methods based on the described problem modelling approaches is available in [15]. It provides extensive information about the type of input, the applied pre-processing, the training data, the considered parameters, and other relevant aspects that characterize each method.

E. Camera calibration

As indicated in Section II-C, the monocular vehicle detection methods on which this study is based on usually require knowledge regarding the intrinsic parameters of the cameras that have taken the images being used as inputs. Some studies in recent years have however proposed means to avoid this calibration requirement. This is particularly relevant in traffic monitoring situations, where unlike in applications like autonomous driving, the parameters that characterize the used cameras are in many cases unknown. The common way to calibrate cameras is by extracting vanishing points from images [44], which can then be used to determine the position and orientation of said cameras [45][46]. When it comes to the intrinsic parameters, the focal length can be approximated based on the assumption that certain kinds of vehicles have similar dimensions. One of the problems with this approach is the fact it also assumes vehicles move in straight and mutually parallel trajectories towards or from the vanishing point.

In [25], the problem of estimating the 3D pose of vehicles is addressed through the utilization of homographies between the road plane and image plane, which can be determined without the need for camera intrinsics or extrinsics. Having these, the problem is mainly reduced to the estimation of rotated bounding boxes from bird's-eye view (BEV) images generated via inverse perspective mapping. This method appears to generalize well, delivering good performance on new camera and environment arrangements that are not included in the training. However, its proposed technique also requires rather extensive computation times that make it unsuitable for certain application cases.

Building on the described homography-based method to avoid the need for regular calibration matrices, [22] additionally removes the need for 3D box annotations and high computation resources by further exploiting the consistency in the 3D to 2D projective geometry and vehicle dimension priors. The homography and the scale of the BEV images is in this case obtained using satellite images available in public mapping services. An overview of the process followed by this method is displayed in Figure 4.

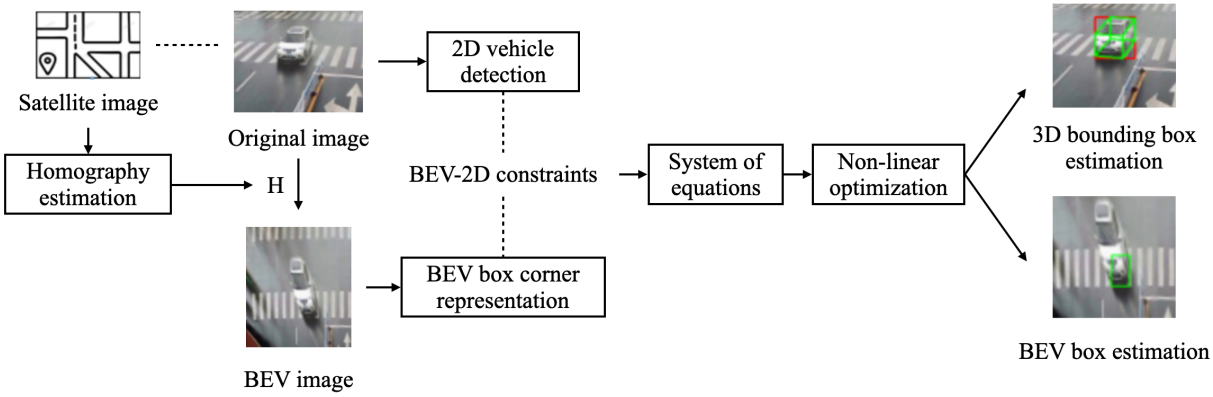


Fig. 4: Overview of the monocular 3D bounding box estimation method proposed in [22]. A homography matrix is first found that relates the image plane to its BEV. 2D detection is performed on the original image, and its BEV representation is built. A system of equations is obtained enforcing BEV-2D constraints related to the tight fit of the BEV box within the 2D box. Non-linear optimization is then used to solve the system and estimate the final BEV and 3D boxes.

F. CenterNet - Objects as points

The CenterNet detector introduced in [3] is designed with the idea of avoiding the anchor-based region mechanisms prevalent in most modern detectors, where numerous potential object locations are proposed and then classified. It instead makes use of keypoint estimation, locating objects through their centers and then regressing all other properties, which among other things eliminates the need for post-processing. This constitutes a method that is differentiable end-to-end, simpler, very fast and that delivers state of the art accuracy levels.

Its basic version focuses on 2D detection. Considering $I \in \mathbb{R}^{W \times H \times 3}$ an input image of width W and height H, CenterNet intends to create a keypoint heatmap $\hat{Y} \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times 3}$, where C is the number of considered object categories and R the output stride that downsamples the prediction (usually set to a default value of 4). In said heatmap, a value of $\hat{Y}_{x,y,c} = 1$ indicates the presence of the center of an object, while $\hat{Y}_{x,y,c} = 0$ corresponds to the background. Locating objects thus becomes a matter of predicting this kind of heatmap, identifying peaks that could be considered potential object centers. As a preparation for training, a low resolution equivalent $\tilde{p} = [\frac{p}{R}]$ is computed for each ground truth keypoint $p \in \mathbb{R}^2$. Ground truth keypoints are then placed in a heatmap $Y \in [0,1]^{\frac{W}{R} \times \frac{H}{R} \times 3}$ employing a Gaussian kernel $Y_{xyc} = exp(-\frac{(x-\tilde{p}_x)^2+(y-\tilde{p}_y)^2}{2{\sigma_p}^2})$ with a standard deviation σ_p that adapts to the object size. A penalty-reduced pixel-wise logistic regression with focal loss is utilized as a training objective.

In order to compensate for the discretization errors that stem from the mentioned output stride, a local offset $\hat{O} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ is predicted for each keypoint, and trained using an L1 loss. The size of the 2D bounding boxes that enclose objects is estimated in parallel through a regression process. A singular size prediction map $\hat{S} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ is used for all object categories, and a L1 loss is used in training at the center points. The training objective of the model is a weighted sum of the localization, offset and size losses. All predictions share a common fully-convolutional backbone network that divides into a head for each modality.

During inference, the highest 100 peaks for each category are extracted. For a keypoint in (x_i, y_i), its detection confidence score is taken directly from the keypoint value $\hat{Y}_{x_i y_i c}$. A bounding box is then constructed with its center on the point in question (with the appropriate correction established by the offset prediction), and its dimensions are dictated by the prediction of the size module.

The 2D detection version of CenterNet is easily extendable to 3D detection. Said extension basically requires three additional attributes per center point on which to perform regression: depth, 3D dimensions and orientation. Separate heads are added to the model for each of these. Depth is a scalar value per center point that involves the consideration of a channel $\hat{D} \in [0,1]^{\frac{W}{R} \times \frac{H}{R}}$. Since it is hard to regress to directly, an inverse sigmoidal transformation of the output as suggested in [47] is employed. The depth estimator is trained using a L1 loss. The 3D dimensions are just an extension of the 2D size to three scalars. The values of these are regressed to directly in the corresponding head $\hat{\Gamma} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 3}$ using a L1 loss. Finally, while orientation can be expressed with a scalar, CenterNet instead makes use of a multi-bin orientation estimation mechanism introduced in [20] to facilitate regression, which leads to its respective head $\hat{A} \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 8}$ being encoded via 8 scalars. Figure 5 displays an overview of the setup for a CenterNet model targeting the problem of 3D bounding box detection.

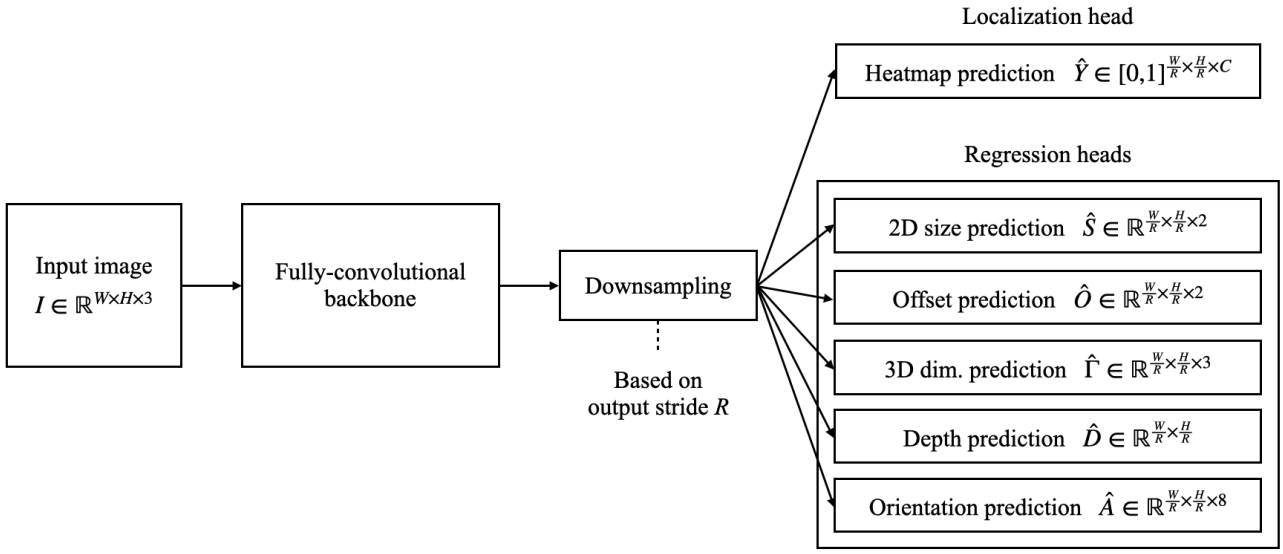


Fig. 5: Overview of the CenterNet model for 3D detection. Images are passed in a single forward pass through a common fully convolutional architecture and a down-sampling process based on an output stride R, and then onto different prediction heads responsible for regressing the parameters that define a 3D bounding box.

TABLE I: Summary of most common datasets for 3D vehicle bounding box detection (*M*: Monocular, *S*: Stereo).

Dataset	Modality	Camera positions	# Images	# Object categories
KITTI	RGB(*M/S*), LiDAR	Dynamic (road footage)	15k	3
ApolloScape	RGB(*M*), LiDAR	Dynamic (road footage)	140k	8+
NuScenes	RGB(*M*), LiDAR	Dynamic (road footage)	40k	23
A*3D	RGB(*M*), LiDAR	Dynamic (road footage)	39k	7
Cityscapes 3D	RGB(*M/S*)	Dynamic (road footage)	5k	8
Ko-PER	Monochrome(*M*), LiDAR	Static (traffic monitoring)	2×4.8k	4
PASCAL 3D+	RGB(*M*)	Dynamic (varied)	30.9k	12

G. Datasets for 3D bounding box detection

Despite the extensive work necessary to annotate 3D bounding boxes, there are several datasets for 3D bounding box detection which are commonly employed in studies to evaluate the performance of detectors. The most common of them, particularly in the field of autonomous driving, is KITTI [48], consisting of 14,999 images of road footage taken with cameras and a LiDAR sensor mounted on a car. Three different classes are considered in this set: *car*, *pedestrian* and *cyclist*; with a total of 80,256 object annotations. Some other similar datasets that provide 3D bounding box annotations elaborated with the help of LiDAR sensors include ApolloScape [49], NuScenes [50], A*3D [51] and Waymo Open [52]. These generally provide a higher number of annotated frames than KITTI, and consider additional object classes. Cityscapes 3D [53] is yet another similar dataset also designed for autonomous driving; the difference compared to the previous being the fact its 3D labeling is based on stereo imaging rather than LiDAR sensors.

In contrast to the wide variety of datasets available for autonomous driving, there is a shortage of 3D bounding box detection datasets for traffic monitoring applications. Ko-PER [54] is one of the only options in this field, offering monochrome footage of a traffic intersection from two different camera angles where vehicles have been annotated with 3D bounding boxes through the use of LiDARs. The main sequence of the footage is organized into four different parts with approximately 1,200 annotated frames each for every camera.

In terms of other datasets aimed at general 3D object detection and pose estimation without a specific application in mind, PASCAL3D+ [55] provides a total of 30,899 images, with 3D annotations for the 12 object categories (*aeroplane*, *bicycle*, *boat*, *bottle*, *bus*, *car*, *chair*, *dining table*, *motorbike*, *sofa*, *train*, and *tvmonitor*) originally found in PASCAL VOC 2012 [56] dataset. Table I provides a summary of the mentioned datasets, with information regarding their data modalities, size, number of object categories and number of annotated objects.

III. METHOD

The methodology followed in order to tackle the problem of lightweight monocular 3D vehicle detection for both calibrated and uncalibrated cases is explained in this section. For each separate case, details regarding the used datasets, the assumptions that have been made and the architectures of the detection systems constructed from scratch are provided.

A. Detection with calibrated cameras

The first detection system has been created to work on the KITTI dataset or similar alternatives. Only the monocular images included in said dataset have been employed, alongside the annotations needed to delimit the ground truth 3D bounding boxes. One of the peculiarities of the 3D annotations in this dataset is the assumption of objects laying on a completely flat ground, a common aspect as mentioned in Section II-C. Elevation and roll angles are thus treated as being null, and only the azimuth is considered to be relevant. Annotated objects present in the images are divided in "Easy", "Moderate" and "Hard" groups, according to how challenging it should be for a detector to correctly identify and delimit them. This difficulty relates to the size of the objects in question as well as the level of occlusion and the truncation they are subject to. It is worth noting that despite the fact the KITTI dataset contains three different kinds of object categories, only objects belonging to the car category have been considered here.

To prepare the images from the dataset for the model that is to be trained to perform predictions on these, these are resized from their original size of 1242×375 to 1166×352, maintaining their aspect ratio. A padding of 217 per side is then added to every image using replicated borders, resulting in dimensions of 1600×352. The intention behind this side-padding is to account for object centers that would normally fall outside of the image's frame.

As hinted at the beginning of the paper, the detection system designed for this particular problem is based on the objects as points concept of the CenterNet. Like the original CenterNet model, it consists of a feature extraction architecture with some down-sampling that feeds several prediction heads. While it shares this general structure and also utilizes the same output stride of $R = 4$, several alterations have been applied to the original architecture in an attempt to simplify it. Instead of the

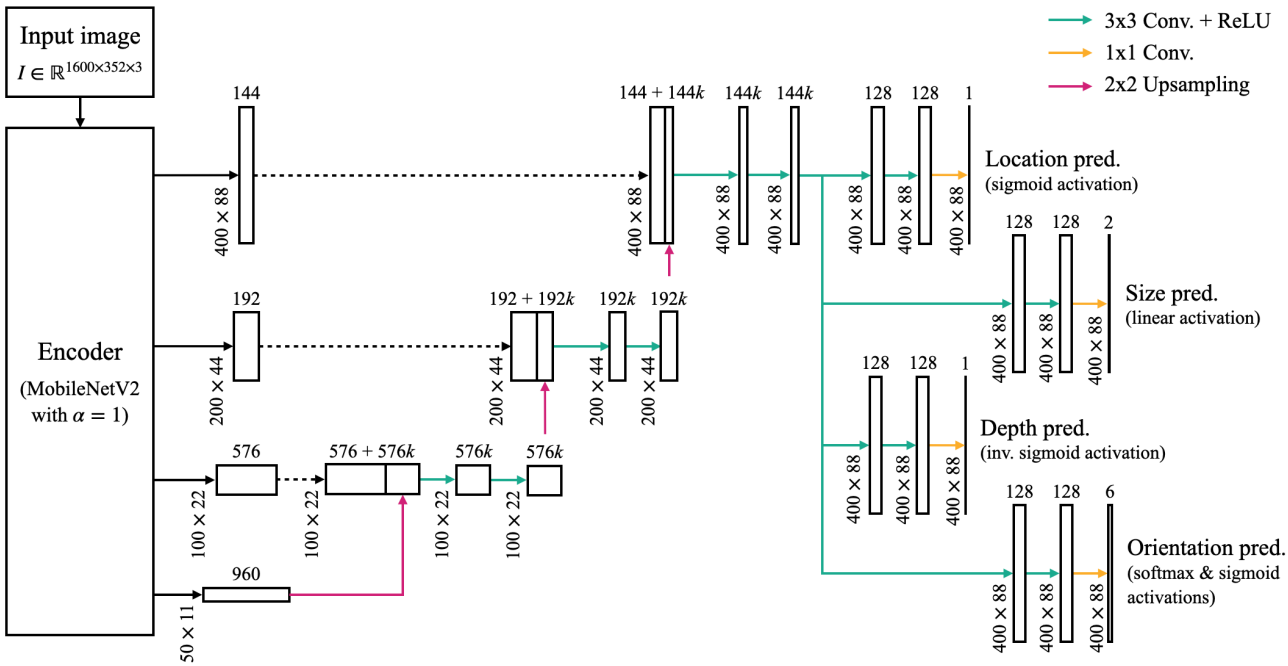


Fig. 6: Overview of the architecture proposed for the 3D vehicle detection model, consisting of a U-Net-like backbone with an output with a stride of 4 that connects to four different prediction heads. The displayed example uses a MobileNetV2 as an encoder, but the general structure of the system remains the same for other encoder options.

modified DLA-34 architecture used by the CenterNet, a model like the one depicted in Figure 6 is here employed. Similarly to U-Net-like architectures [57], its backbone is a FCN consisting of a contracting path with progressive down-sampling blocks and an expansive one with up-sampling operations, where different resolution levels from each part are connected through concatenation. Lightweight models pre-trained on ImageNet [58], i.e. the MobileNetV2 [59] and the EfficientNet [60], are used for the mentioned contracting path. Features from some of their intermediate layers corresponding to different down-sampling levels (strides of 4, 8, 16 and 32) are extracted and fed to the expanding part of the architecture. For the up-sampling steps of the expanding part, both bilinear interpolation and transposed convolutions are tested.

Only four of the six prediction heads of the original CenterNet for 3D detection are here maintained. Considering the low output stride of 4, the potential benefits of including offset prediction are deemed to be irrelevant for the intended purpose of the study and this aspect is thus not included in the system. In addition, the prediction head responsible for the estimation of 3D dimensions is also discarded. Instead, considering the low intra-class variation of car widths, lengths and heights, the average values of these measures are considered for all car instances. This leaves the system with prediction heads for the location of 2D object centers, 2D dimensions, depth, and orientation angle. Each prediction head consists of double 3×3 and 1×1 convolutions separated by a ReLU. The inclusion of 2D prediction capabilities is justified by the fact some of the evaluation metrics for the KITTI dataset rely heavily on predicting accurate 2D bounding boxes. A version of the system that predicts 3D object centers and does not include a prediction head for the estimation of 2D dimensions is also tested for comparison purposes.

The ground truth maps for each of the prediction heads are built from the KITTI object annotations following two steps. First, similarly to what is explained in Section II-F, a map for object location $Y \in [0,1]^{\frac{W}{R}\times\frac{H}{R}\times1}$ is built where Gaussian kernels with peak values of 1 are placed in object centers. As observed in the example in Figure 7, the standard deviation of these is made adaptive to the size of the objects in the image, or in other words, cars closer to the camera are assigned a wider Gaussian. This map will later help the system learn to estimate the level of confidence for every pixel of an image $\hat{Y}_{xy}$ to correspond to an object center of the car class. This is achieved using a sigmoid activation in the output and a penalty-reduced pixel-wise logistic regression with focal loss, which is detailed in Equation 2. α and β are hyper-parameters set to 2 and 4 as suggested in [13] and N is the number of object centers in the image.

$$L_{loc} = \frac{-1}{N} \sum_{xy} \begin{cases} (1-\hat{Y}_{xy})^{\alpha} \log(\hat{Y}_{xy}) & \text{if } Y_{xy} = 1 \\ (1-Y_{xy})^{\beta} (\hat{Y}_{xy})^{\alpha} \log(1-\hat{Y}_{xy}) & \text{otherwise} \end{cases} \tag{2}$$

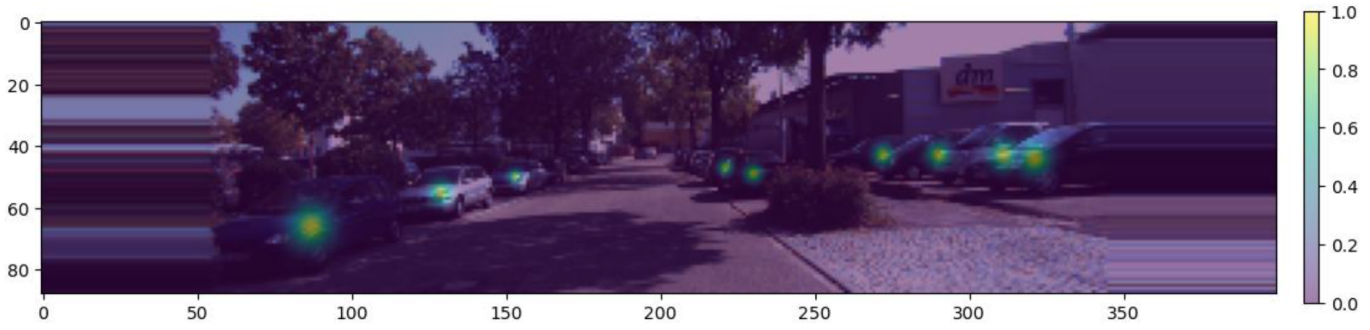


Fig. 7: Example of a ground truth location map where Gaussian kernels have been applied on object centers with peak values of 1 and standard deviations adaptive to the size of said objects.

To create the ground truth maps for the other prediction heads, a mask is created based on a threshold of 0.5 applied to the described location map. Parameters representative of 2D dimensions, depth and orientation for each object can this way be applied with the help of the mask to the small regions that surround the centers of said objects. For 2D object width and height, values are made relative to the dimensions of the input image, meaning they constitute a map $S \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 2}$ that includes values ranging from 0 to 1. A simple L1 loss is used to learn these values, as shown in Equation 3.

$$L_{size} = \frac{1}{N} \sum_{k=1}^{N} \left| \hat{S}_k - S_k \right| \tag{3}$$

When it comes to depth, absolute values are employed in its corresponding map $D \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R}}$. The depth predictor utilizes an inverse sigmoidal activation as suggested in [47] to facilitate its regression. As indicated in Equation 4, the training for this parameter is then once again based on a L1 loss.

$$L_{dep} = \frac{1}{N} \sum_{k=1}^{N} \left| \frac{1}{\sigma(\hat{d}_k)} - 1 - d_k \right| \tag{4}$$

For the estimation of the global azimuth θ, the local orientation θ_l of cars is instead taken into consideration when building the ground truth maps to be predicted by the system. The difference between the two is that while the global azimuth only depends on the object's orientation relative to the camera, the local angle also takes into account where the object is located relative to said camera. This difference is illustrated in Figure 8, which explains how the local orientation is determined with respect to the ray that goes from the camera to the center of the object and how the global azimuth is thus computed as $\theta = \theta_{ray} + \theta_l$. What makes the local orientation more convenient for the system to predict is the fact it is closely related to appearance. A car with a certain global orientation will look differently in an image depending on where it is located, and using local orientation together with positioning information makes it possible to effectively interpret these variations.

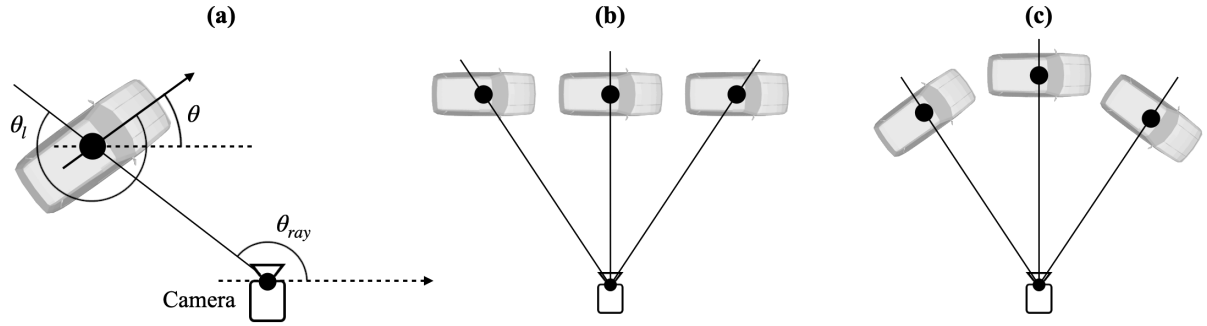


Fig. 8: (a) Relationship between local and global orientations. (b) Example showing three cars with equal global orientation that would look differently from the perspective of the same camera. (c) Example showing three cars with equal local orientation that would have the same appearance from the perspective of the same camera.

A multi-bin approach similar to that introduced in [20] and to the method employed by the CenterNet is implemented for the prediction of the mentioned local orientation. Instead of regressing its value directly, estimation is treated as a problem of combined classification and regression. The whole range of possible angles is divided into two bins with a slight overlap, one going from $\frac{-7\pi}{6}$ to $\frac{\pi}{6}$ and the other from $-\frac{\pi}{6}$ to $\frac{7\pi}{6}$. The orientation of an object is then represented through three scalars per bin, resulting in a total of six maps: $A \in \mathbb{R}^{\frac{W}{R} \times \frac{H}{R} \times 6}$. For every bin, the first scalar is a value of either 0 or 1 that indicates whether the object's orientation falls into the range of said bin. This value is evaluated during training with a softmax loss. The other two relate to sin and cos of the in-bin angle offset $\Delta\theta_{li}$, which are evaluated with an L1 loss whenever the angle is deemed to belong to the bin in question. These two aspects are weighted and combined into the loss function shown in Equation 5, where $\hat{b}_i$ refers to the probability of the angle belonging to bin i, c_i is the ground truth value of either 0 or 1 reflecting this belonging, and m_i is the center of bin i. During inference, the estimated offset $\Delta\theta_{li}$ is applied to the center of the bin with the maximum confidence.

$$L_{ori} = \frac{1}{N}\sum_{k=1}^{N}\sum_{i=1}^{2}(softmax(\hat{b}_{k_i}, c_{k_i}) + w \times c_{k_i} \,|(sin(\Delta\theta_{l_{k_i}}), cos(\Delta\theta_{l_{k_i}})) - (sin(\theta_{l_k} - m_{k_i}), cos(\theta_{l_k} - m_{k_i}))|) \quad (5)$$

Once all parameters are predicted by the system, these are passed through a decoder to obtain the 2D bounding boxes enclosing the objects in the image and the positioning and orientation of the corresponding 3D bounding boxes in space. The 2D bounding boxes are obtained by adding the width and height values predicted at the location of the k detections with the highest confidence values. A mechanism employed by the CenterNet to get rid of redundant overlapping detections is used. It involves applying a maxpool operation to the predicted locations heatmap (in this case one with a kernel of 10×10), which elevates the values of the pixels surrounding peaks. Next, an boolean element-wise comparison is performed between the input and the output of the maxpool, bringing all the values in the map to 0 except for the local maxima, which maintain a value of 1. The result of this comparison is then multiplied with the original heatmap, leaving only the peaks of interest above zero value.

To determine the 3D bounding boxes, based on the projective geometry concepts illustrated in Section II-C, the provided camera matrices and the object centers located in the image plane are used to first infer the lines on which the objects in question lie. Note that this approach assumes the centers of the object 2D bounding boxes are similar to the positions of the real 3D object centers when projected on the image plane, when in reality this is not always the case [61]. Next, the estimated depth values are used to establish in which point of said lines the objects are actually located. The pre-established 3D dimensions of the car class are then enforced at the determined points in space according to the predicted azimuths.

In the case of the version of the detector that directly estimates 3D object centers and skips the regression of 2D width and height, the 2D bounding boxes are instead obtained by simply drawing a rectangle that tightly encloses the projection of the 3D bounding box.

B. Detection with uncalibrated cameras

To address the uncalibrated case, the previously mentioned Ko-PER dataset containing footage from two elevated traffic cameras monitoring an intersection has been used to train the detection system. Once again, only the objects belonging to the car category are taken into account. Despite the fact this dataset provides camera matrices, these are ignored due to the assumption of no calibration. Instead, considering how the dataset provides a map of the monitored intersection, 8 point correspondences between the cameras' images and this map have been manually established. The planar relationship between images and the map is then calculated with the use of RANSAC, which selects the best set of points and then employs non-linear optimization of distance residuals to define an homography. While 4 point correspondences would technically be enough to define this homography, making use of additional points should help find a better fit. Having access to the homographies for each of the camera angles, it is then possible to transform camera images into a BEV version that fits the provided map, as showcased in Figure 9. The images from both cameras are stitched together into single BEV images of 736×736 on which the ground truth box annotations can easily be placed. Since the cameras only cover part of the extent of the map and some of the annotated cars fall outside of this scope, a region of interest (ROI) is enforced to filter out any car instances beyond this visibility field. Under a flat-earth assumption, the resulting images can then be used to train the system as a 2D detection problem of rotated bounding boxes.

The same objects as points concept, architecture and loss functions as for the calibrated case are employed for this particular problem. The only difference lies in the used prediction heads, which are now limited to one for the location of 2D object centers and another for the estimation of the azimuth. While the same multi-bin approach for orientation estimation as in the calibrated detector is also used here, the global orientation or azimuth θ is predicted directly instead of the local orientation θ_l. This has to do with how unlike the aspect of an object with the same azimuth can change depending on where it is located relative to the camera in a normal image, in BEV versions this aspect remains generally unaltered. The average width and length of cars in the dataset are again used directly to determine the size of the detected 2D bounding boxes.

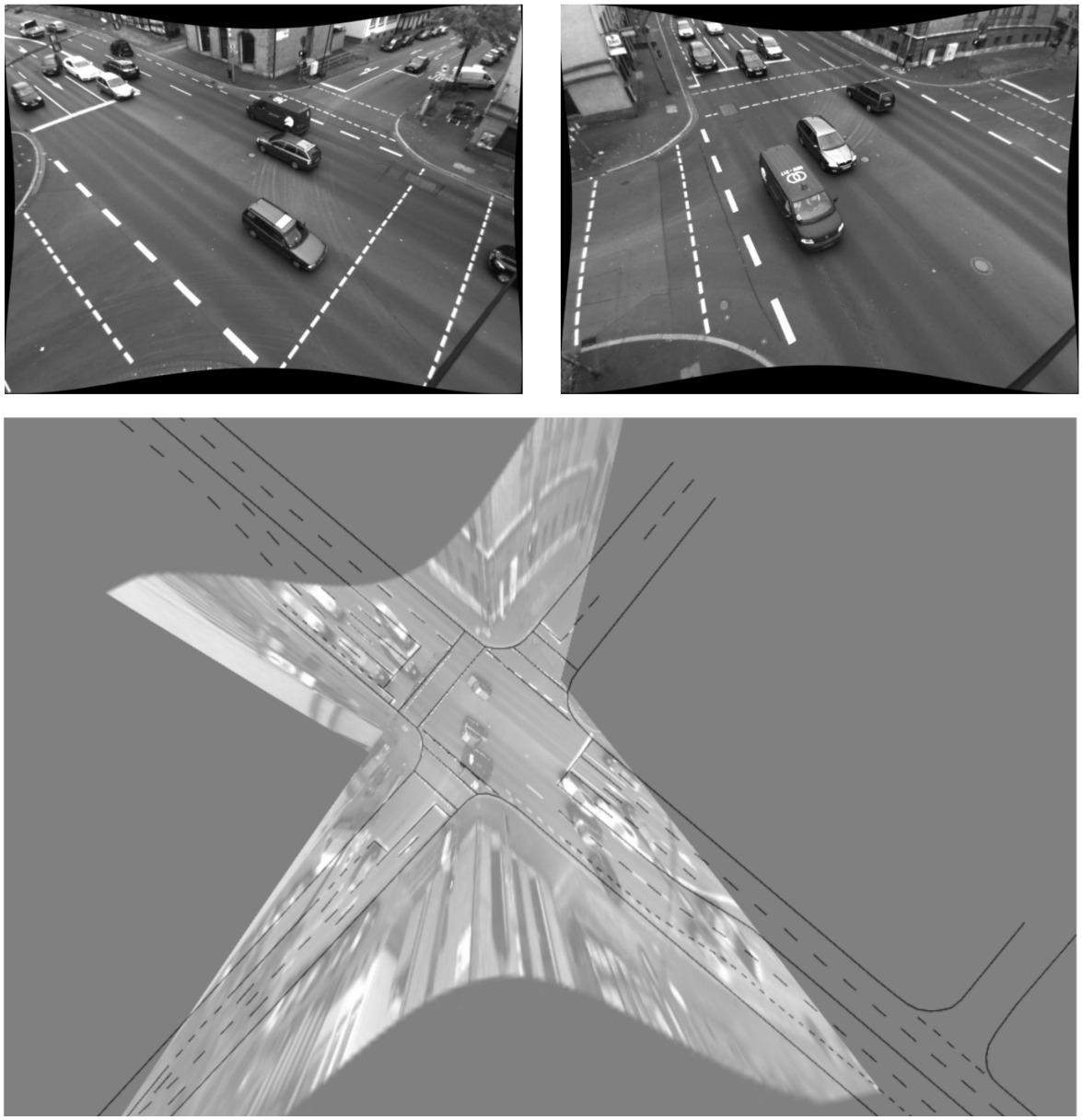


Fig. 9: Practical example of a given timestamp where the two images of the same intersection taken from different camera angles are stitched together into a single BEV image after having established point correspondences with a map of the scene and having determined homography-based transformations from these. The stitched image is showcased with some transparency to show how the transformed images fit the used map.

The decoding of the predicted parameter maps is similar to that used in the calibrated case. 2D bounding boxes are drawn from the identified location keypoints according to the mentioned average size values, and next they are rotated based on the predicted azimuth. Since the small imprecisions derived from the homography transform can result in the system outputting small plateaus instead of clean peaks at the detected object centers, the mechanism for eliminating redundant detections explained in the calibrated case can at times be rendered insufficient. A small non maximum suppression (NMS) algorithm is thus added at the start of the decoding process, which iterates through the detections in order of confidence, and whenever the location of an object center is less than 15px away from one of the already observed ones, it updates the latter with the average location of the two and discards the new one.

If the detected 2D bounding boxes are to be displayed in a projected manner in the original camera images, it is possible to revert the homography transformation to bring back the BEV images to their original form. Note however that having approached the detection problem in 2D, no information in terms of height is available and full 3D bounding boxes can in principle not be drawn.

IV. EXPERIMENTS

A series of experiments have been carried out in an attempt to determine how certain aspects related to the architecture and layer settings of the proposed detection systems influence their performance. In addition, the potential benefits of some types of data augmentation techniques on this performance has also been explored. All of the details behind these tests are explained throughout this section, and information on how the training process has been approached is also provided.

A. Architecture-related tests

Variations to certain elements of the architecture introduced in Section III are tested, including the use of different kinds of models for the encoder and layer widths, different types of up-sampling methods, and two kinds of activation functions for the hidden layers. Unless specified otherwise, experiments by default are carried out using 2×2 transposed convolutions for the up-sampling steps and ReLU activation functions in the hidden layers. The presented tests have been for the most part evaluated on the detection system for the calibrated case, and the overall insights from these have later been assumed valid for the uncalibrated detector due to the similarity between both systems.

- **Encoder model**: First, several lightweight models pre-trained on ImageNet are employed for the encoder of the feature extraction part of the proposed architecture. The selected models for this are the MobileNetV2 and the EfficientNetB0 available in Keras Applications. Values of 0.35, 0.5 and 1 are tested for the width multiplier (α) parameter of the MobileNetV2, which increases or decreases the filters to be included in every layer of the model proportionally to its value. Note that these models are only used up until their output of stride 32, meaning that their entire parameter count is not utilized. Table II summarizes the number of parameters and depth of the mentioned encoder options. The number of filters for each of the four extracted outputs at different stride levels is also detailed.

  The number of filters used in the up-sampling and convolution layers of the decoder is made dependant to those they receive from the encoder. However, in order to try to keep the size of the resulting system as low as possible, only a fraction k of the number of filters extracted from the encoder at a given level are used in the corresponding up-sampling and convolution operations. This k is treated as another parameter to adjust, for which values of 0.5, 0.75 and 1 are tested. For instance, for a MobileNetV2 with $\alpha = 1$ encoder that provides 576 filters at its stride 16 output, if k is set to 0.5, the up-sampling operation which creates the features to be concatenated to these will be limited to 288 filters, and the same amount of filters will be used for the two convolutions that follow.

- **Up-sampling method**: For the encoder option with the best trade-off between detection accuracy and parameter count, the default 2×2 transposed convolutions for the up-sampling operations are compared to simpler alternatives like nearest neighbor interpolation and bilinear interpolation. The purpose of this particular experiment is to determine if the added learning capacity of transposed convolutions compared to other interpolation methods actually contributes to a better detection performance of the resulting system.

- **Activation of hidden layers**: Leaky ReLU is tested as an alternative to the default ReLU activation functions used in the system's hidden layers (excluding those belonging to the encoder). Some authors report how the addition of a slight negative slope in the negative range of the ReLU can consistently help neural networks reach lower error levels in small datasets [62]. The idea behind this experiment is thus to determine if the use of leaky ReLU activations with a negative slope coefficient set to 0.3 could improve the system's performance, which would most likely indicate a mitigation of the effect of dying neurons in the system.

TABLE II: Summary of the different pre-trained models tested as encoders and their extracted features at different stride levels. The parameter count is for the version of these models cropped after the activation of their stride 32 blocks. The same applies to the depth column, which refers to the number of layers in the models with parameters.

Encoder model		**Parameters**	**Depth**	**Number of filters**			
				Stride 4	**Stride 8**	**Stride 16**	**Stride 32**
MobileNetV2	α = 0.35	224,224	100	48	96	192	336
	α = 0.5	417,888	100	96	96	288	480
	α = 1	1,528,656	100	144	192	576	960
EfficientNetB0		3,144,899	119	144	240	672	1152

Fig. 10: Example of two frames taken from the KITTI object detection dataset. While all scenes are taken during daytime and in clear weather conditions, it can be appreciated how vehicles can still appear in significantly different lighting conditions due to shadows cast by their surroundings.

B. Data augmentation

To try to make the most out of the available training data, two kinds of data augmentation techniques are tested while trying different encoders for the calibrated case. The first is to randomly flip 50% of images horizontally, which should in principle help the system generalize how it interprets the appearance of the vehicles in images, especially when it comes to predicting orientation. The second consists in randomly altering the brightness value of the images used in training within a range of 80% to 120% of the original brightness value. This has been considered a suitable measure due to how there are significant variations in lighting conditions throughout the scenes included in the KITTI dataset, as observed in the examples of Figure 10. Since most of the footage of the used datasets originates from specific cameras with consistent characteristics, data augmentations that apply alterations to images in terms of noise, color values, etc. have been considered irrelevant for the case at hand. Also, augmentation methods like cropping, scaling, or shearing have been ruled out due to how they would interfere with the 3D measurements. For the uncalibrated detection case, random vertical flipping is also tested in addition to horizontal flipping. On the other hand, brightness variations are not included, as the lighting conditions of the Ko-PER dataset remain mostly consistent through each of its sequences.

C. Training process

Due to the designed detection systems' reliance on multiple outputs with different loss functions, and the rather limited amount of available data, the approach to the training process of the models has been essential for reaching the intended results. For the calibrated case, the 7,481 training images of the KITTI dataset have been divided into training and validation sets of 3,682 and 3,799 images respectively, following the standard split first proposed in [63] and later adopted by several other papers, including that of the original CenterNet. This split ensures that there are no images from the same scene shared between training and validation sets. On the other hand, since no training and validation split proposals have been found in literature for the Ko-PER dataset, the first three parts of the main sequence have been used for training (totalling 3,626 images) and the fourth one has been used for validation (1,205 images).

The weights for the loss of each of the prediction heads has been manually assigned based on the magnitude of these. The magnitude of the loss of the size predictor is for instance around 60 times lower than that of the object center location predictor, and it therefore needs to be compensated with a much higher weight in order to have an influence on the training. The specific values for these weights employed in the training of the detection systems for the calibrated and uncalibrated cases can be found in Table III below.

TABLE III: Summary of weights applied to the loss of each prediction head during training.

Detection system	**Weights per loss type**			
	Location	**Size**	**Depth**	**Orientation**
Calibrated case detector	2	60	1	2
Uncalibrated case detector	1	-	-	1

TABLE IV: Summary of the hyperparameters considered in the random search for the re-training of the selected detection models after the experiments, showcasing the possible value ranges for each of them.

	Tested hyperparameters			
	Batch size	**Weight initialization**	**Optimizer**	**Learning rate**
Value ranges	2, 4, 8, 16	Xavier normal, He normal	Adam, SGD w/ Nesterov momentum	5e-4 to 1e-2

The general approach for training the different variations of each detector has been to first maintain the weights of the pre-trained encoder frozen while training the models for a maximum of 100 epochs. Despite this limit, early stopping with a patience of 7 epochs based on the validation loss is enforced to avoid proceeding with computations that are clearly suffering from overfitting. After this first training stage has reached its end, and having saved the model weights with the lowest validation loss, the encoder is unfrozen and the training is resumed with the stored weights. The same epoch limit and early stopping conditions are applied in this second stage.

For the experiments explained in Section IV-A, a fixed batch size of 8 and a learning rate of 1e-3 have been employed when initiating training. The learning rate is lowered to 1e-4 after unfreezing the encoder. Weights are in these tests initialized with a Xavier normal initializer, and Adam is used as an optimizer. In all cases, a mechanism to reduce the learning rate to 1/4 of its current value whenever the validation loss has not improved for more than 3 epochs is implemented. It is also worth mentioning that the learning capacity of the tested architectures for the tasks at hand is always tested beforehand by purposefully checking if the model is able to over-fit the training data.

After determining the best model configuration based on the insights from the proposed experiments for the calibrated case, an attempt is made to find the ideal re-training conditions to further improve this best system. This involves carrying out a random search to try to find the hyperparameters that lead to the best training results. The different hyperparameter types and possible values considered in this random search are detailed in Table IV. For the same model adapted to the uncalibrated case (through the removal of the size and depth prediction heads), the same random search is performed to better adjust the training conditions to the BEV-projected detection problem.

V. Results

This section presents the results obtained for the main experiments detailed in Section IV, providing an evaluation based on several detection accuracy metrics along with the final parameter counts and inference time of each configuration. The results for the best calibrated detector are also compared to those of the original CenterNet model. Next, a qualitative evaluation of the performance of the constructed calibrated and uncalibrated systems is provided along with comments regarding the potential usability of these in specific cases.

A. Evaluation metrics

Three different types of evaluation metrics are used to evaluate the detection performance of the calibrated detection system. First, average precision (AP) is used to determine how well the predicted 2D bounding boxes fit those provided in the ground truth. Next average orientation similarity (AOS) is employed, which as explained in [20], multiplies the AP for the 2D bounding box evaluation with the average cosine distance similarity for the orientation's azimuth. Finally, the AP for the rotated 2D bounding boxes seen from a BEV is also evaluated. These metrics involve 11 recalls (from 0 to 1 with increments of 0.1), with an intersection over union (IoU) threshold of 0.5.

As hinted at in Section III-A the evaluation proposed for the KITTI dataset is based on 3 difficulty levels, so the described metrics are provided for each of these. The "Easy" level involves the detection of fully visible objects with a minimum bounding box height of 40px and less than 15% of truncation. The "Medium" and "Hard" levels both include objects of at least 25px in height, the difference being that the former considers partly occluded objects with a maximum truncation of 30% and the latter heavily occluded objects with a maximum truncation of 50%. Note how this means cars that are very far from the camera or with more than half of their scope outside of the image are not taken into account in the evaluation.

For the detector designed for the uncalibrated case, considering how the detection problem is addressed as an evaluation of rotated 2D bounding boxes, only the BEV AP metric is employed. Due to how no records of performance scores for the Ko-PER dataset are found in existing literature, the results for the uncalibrated detector are not compared to similar systems.

B. Quantitative evaluation

Table V displays the results of the experiments regarding the testing of different encoder options and layer widths. Results are provided for training runs both with and without having used the previously mentioned data augmentation techniques of random horizontal flipping and brightness alterations.

TABLE V: Performance evaluation results of the calibrated detection system for different encoders and layer widths. Values in each cell are provided for the cases without and with data augmentation, separated by a forward slash. The reported inference times correspond to the average of all predictions for images in the validation set (using a NVIDIA GeForce RTX 3090 GPU).

Encoder model			AP			AOS			BEV AP			Parameters	Inference time (ms)
	α	k	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard		
MobileNetV2	0.35	0.5	72.0 / 72.8	52.3 / 52.9	46.3 / 46.3	70.2 / 71.6	51.1 / 51.3	43.5 / 44.5	28.4 / 31.3	19.3 / 23.8	15.7 / 19.8	1,657,450	28.5
		0.75	72.2 / 72.7	51.4 / 51.7	43.8 / 44.2	70.2 / 70.3	49.5 / 49.5	41.9 / 42.1	27.8 / 31.6	18.7 / 22.4	17.1 / 20.6	2,119,630	30.1
		1	68.9 / 72.9	46.2 / 50.4	40.0 / 43.4	66.2 / 69.0	44.1 / 47.3	37.9 / 40.5	23.5 / 28.2	15.9 / 21.2	12.3 / 17.5	2,697,010	31.7
	0.5	0.5	67.9 / 68.9	48.2 / 49.3	42.2 / 42.5	64.3 / 66.5	44.9 / 47.0	40.0 / 40.2	18.5 / 19.5	14.3 / 14.6	11.1 / 11.8	2,804,530	28.6
		0.75	70.6 / 72.4	49.1 / 49.0	41.8 / 41.8	68.3 / 70.1	47.2 / 48.9	39.9 / 39.7	31.5 / 31.4	21.9 / 23.4	20.5 / 19.5	3,736,426	29.5
		1	72.5 / 72.5	50.8 / 51.1	43.6 / 43.5	69.9 / 69.7	48.8 / 48.9	41.6 / 41.3	30.5 / 28.8	21.4 / 21.2	17.5 / 20.5	4,894,114	32.6
	1	0.5	75.6 / 76.3	57.8 / 59.0	44.2 / 46.1	70.1 / 74.0	53.1 / 56.4	42.8 / 44.0	27.4 / 31.4	19.7 / 22.9	19.0 / 21.1	7,816,346	35.1
		0.75	75.1 / 76.3	53.5 / 54.0	45.8 / 46.3	72.3 / 73.4	51.2 / 51.3	43.5 / 43.7	32.4 / 33.3	23.2 / 25.8	21.2 / 21.8	11,493,950	35.8
		1	74.9 / 76.0	54.8 / 54.6	46.4 / 46.7	72.8 / 73.4	53.0 / 52.4	44.7 / 44.6	27.4 / 34.4	19.3 / 23.6	17.6 / 21.6	16,070,114	36.6
EfficientNetB0	-	0.5	83.8 / 87.0	64.5 / 65.8	55.8 / 57.3	80.2 / 85.3	63.1 / 64.0	52.8 / 55.3	30.0 / 33.2	26.5 / 29.7	17.8 / 20.6	11,197,861	42.7
		0.75	81.7 / 82.2	63.0 / 63.2	54.5 / 54.5	79.3 / 79.3	60.5 / 60.2	51.8 / 51.6	23.9 / 28.7	18.1 / 22.7	16.4 / 20.8	16,275,829	43.2
		1	82.3 / 75.1	60.8 / 59.7	53.0 / 52.2	79.9 / 72.7	58.4 / 57.2	50.6 / 49.6	28.9 / 34.0	20.6 / 23.7	18.9 / 21.7	22,597,381	43.5

TABLE VI: Performance evaluation results for the selected calibrated detection system using nearest neighbors interpolation and bilinear interpolation as alternative up-sampling methods.

Up-sampling method	AP			AOS			BEV AP		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
Transposed convolutions	72.4	49.0	41.8	70.1	48.9	39.7	31.4	23.4	19.5
Nearest neighbor interp.	72.6	52.5	44.8	68.6	49.0	41.4	32.1	21.8	17.8
Bilinear interpolation	75.8	54.8	46.6	73.7	52.4	44.3	25.8	18.2	14.3

While there is a tendency for detection accuracy scores to improve as larger encoders are employed, these improvements are not very substantial when compared to the increase in the model parameter counts and inference times involved. Another observation can be made on how changes in the width of the layers that comprise the expanding part of the feature extractor, determined by the multiplier k, do not lead to any noticeable differences in the showcased metrics. This seems to indicate the feature extractor is just as capable of providing valuable information to the prediction heads using more reduced amounts of filters. In general, there is however a large gap between how the detection system performs on object instances with good visibility compared to more challenging ones. This is a common issue in all state of the art detectors, and even more so with ones relying only on monocular images. For handling heavy occlusions, the only truly effective way to improve the system is to feed it larger amounts of data. On the other hand, errors or missed 3D bounding box detections of truncated cars that only appear in the image frame partially could have to do with how 2D bounding box centers are being considered instead of the projection of the real 3D object centers.

The effects of the tested data augmentation mechanisms are also noticeable in the displayed results. While in all three kinds of metrics the cases that use data augmentation normally have higher scores, this improvement is the most striking when it comes to the BEV AP results. This appears to imply the main positive effect derived from the implementation of data augmentation is an improvement of the system's capability of interpreting depth cues.

Ideally several training runs for each case should be carried out to get more consistent and comparable results, but this was not possible here due to limitations in computational resources and time constraints. Since there is no good way of determining which of the tested model configurations is best, the option employing MobileNetV2 with $\alpha = 0.5$ and $k = 0.75$ is selected as a good middle ground between detection accuracy and system size/speed to carry out the next experiments, and data augmentation is always used.

The effects of using more simplistic forms of up-sampling as an alternative to transposed convolutions are evaluated in Table VI. Based on the results in this table, the use of nearest neighbor interpolation or bilinear interpolation does not appear to significantly affect the performance of the detection system in terms of accuracy metrics, despite how the learning capacity of the up-sampling layers is lost. The most noticeable phenomena is how bilinear interpolation ens up prioritizing 2D detection in exchange for worse BEV predictions. Nevertheless, transposed convolutions are still maintained in the final proposed system.

Table VII showcases the results when using leaky ReLU activation functions in the hidden layers of the expanding part of the model's feature extractor as compared to the default use of regular ReLUs. Again, this alternative setting does not appear to lead to any substantial improvement in the detection accuracy of the system, and actually worsens the BEV estimations. This should in principle indicate that the model is not suffering from a dying ReLU problem hindering its learning potential.

TABLE VII: Performance evaluation results for the selected calibrated detection system comparing the use of ReLU and leaky ReLU in the hidden layers of the feature extractor's expanding part.

Hidden layer activations	AP			AOS			BEV AP		
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard
Standard ReLU	72.4	49.0	41.8	70.1	48.9	39.7	31.4	23.4	19.5
Leaky ReLU	72.8	50.7	43.6	69.4	47.6	40.7	30.4	20.9	17.2

TABLE VIII: Comparative of performance evaluation results between the original CenterNet 3D detection model and the selected versions of our lightweight system. (*) refers to the system that locates the center of objects' 2D centers and regresses 2D size, while (**) refers to the version that directly locates the projected 3D object centers.

Detection system	AP			AOS			BEV AP			Parameters
	Easy	Medium	Hard	Easy	Medium	Hard	Easy	Medium	Hard	
Original CenterNet	90.2±1.2	80.4±1.4	71.1±1.6	85.3±1.7	75.0±1.6	66.2±1.8	31.4±3.7	26.5±1.6	23.8±2.9	20,615,323
Ours (*)	72.8±0.8	50.6±1.1	42.3±1.8	71.4±1.4	50.1±1.8	41.4±1.8	31.4±2.2	23.6±2.0	19.8±2.3	3,736,426
Ours (**)	32.1±1.6	20.2±1.8	18.9±1.8	31.17±1.6	20.2±1.8	18.9±1.9	31.6±2.5	23.5±2.2	19.5±2.6	3,476,832

Having selected a system configuration, the random search for the hyperparameters listed in Section IV-C applied to this model indicates the training is best when using a batch size of 8, He normal for weight initialization, Adam as an optimizer and an initial learning rate of 1.73e-3. The performance of the detection system re-trained under these conditions, which happen to be very similar to the ones used so far, is summarized in Table VIII. The performance of the model is also compared to the detection accuracy ratings of the original CenterNet model. A version of this system that shares its same architecture but predicts 3D object centers directly instead of 2D ones is also trained based on its own random search, and its results are included in the same table. The best hyperparameters in this case turn out to be a batch size of 4, He normal for weight initialization, Adam as an optimizer and an initial learning rate of 1.21e-3. As previously mentioned in the methodology section, it is important to remember this alternative approach does not regress 2D vehicle width and height, and instead determines 2D bounding boxes drawing the tightest possible rectangles than enclose their corresponding 3D bounding boxes projected in the image.

The original CenterNet clearly performs much better than the proposed systems in terms of predicting the 2D bounding boxes that enclose vehicles present in images, as the AP scores reflect. This large difference can probably be attributed to the fact our system is not regressing width and height parameters as effectively as the original CenterNet does, which could probably be improved a great deal by pre-training the 2D object center location and size prediction heads on an additional dataset with a large number of car instances, like for instance NuImages or COCO. In the case of the alternative version of our system that directly detects projected 3D bounding box centers and determines 2D bounding boxes based on the projection of the 3D ones, these AP values are much lower. This is to be expected considering how estimating 2D bounding boxes this way is extremely vulnerable to slight alterations in the predicted orientations and to the differences between the assumed average dimensions of cars compared to their real ones. Nevertheless, 2D detection is not the main focus of this study, so no further actions have been taken to try to improve AP scores.

In regard to AOS scores, while the original CenterNet's scores are once again much higher than those of the proposed detection systems, it is important to remember this metric is proportional to the discussed AP. If attention is instead placed on the part of the AOS that reflects the system's accuracy in terms of orientation estimation (referred to as orientation score or OS in [20]), which involves dividing it by its corresponding AP scores, the resulting metrics are slightly more favourable to our systems. This indicates the proposed 3D detection models perform somewhat better than the original CenterNet when it comes to estimating the orientation of cars, albeit not by much.

Finally, the BEV AP scores reflect how the CenterNet is generally a little bit superior than the proposed systems at locating the detected vehicle instances from a top view, at least when dealing with cases of medium or hard difficulty. This indicates our systems are slightly inferior in terms of estimating depth. All of these comparatives should be assessed while bearing in mind the size of the addressed systems, as there is a difference of more than 16M parameters between the original CenterNet and the detectors designed in this study.

Using the same model configuration as for the best calibrated detector, but limiting the prediction heads to object center location and orientation estimation, a version of the detector for the uncalibrated case is trained through yet another random search. The best hyperparameters this time are a batch size of 8, a weight initialization using Xavier normal, SGD with Nesterov momentum as an optimizer and an initial learning rate of 1.31e-3. The BEV AP results for this uncalibrated system on the part of the Ko-PER dataset dedicated to validation are reflected in Table 14 for different kinds of data augmentation.

TABLE IX: Performance evaluation results of the uncalibrated detector under different kinds of data augmentation.

BEV AP			
No data augmentation	**Random H flipping**	**Random V flippling**	**Random H & V flipping**
62.37	71.14	59.93	58.09

The resulting BEV AP scores appear to show how the use of random horizontal flipping appears to help increase the learning performance of the system. Vertical flipping, on the other hand, worsens the detection accuracy. While the reason behind this is not entirely clear, it is hypothesized that horizontally flipped images contain aspects such as shadows or projective distortions that are more alike those of the original footage compared to what is found on the vertically flipped images.

C. Qualitative evaluation

Figure 11 shows an example of some of the parameter maps predicted by the calibrated detector for an image from the validation split of the KITTI dataset. When it comes to the detection of the center of vehicles' 2D bounding boxes, it can be appreciated how the system is in most cases capable of accurately identifying these key-points with a good level of confidence. There are however exceptions where the system struggles and the predictions are more diffuse, e.g. in cases of heavily occluded cars or truncated ones that are only partially visible in the image frame. In terms of depth estimation, the plot clearly shows how the system is capable of associating higher absolute depth values to points that appear to be further down the street. The maps showcasing the predicted confidence values for each of the two bins considered when estimating local orientation are also shown, as they clearly illustrate how cars facing away from the camera are classified as belonging to bin 1, while ones facing towards the camera are classified as being part of bin 2.

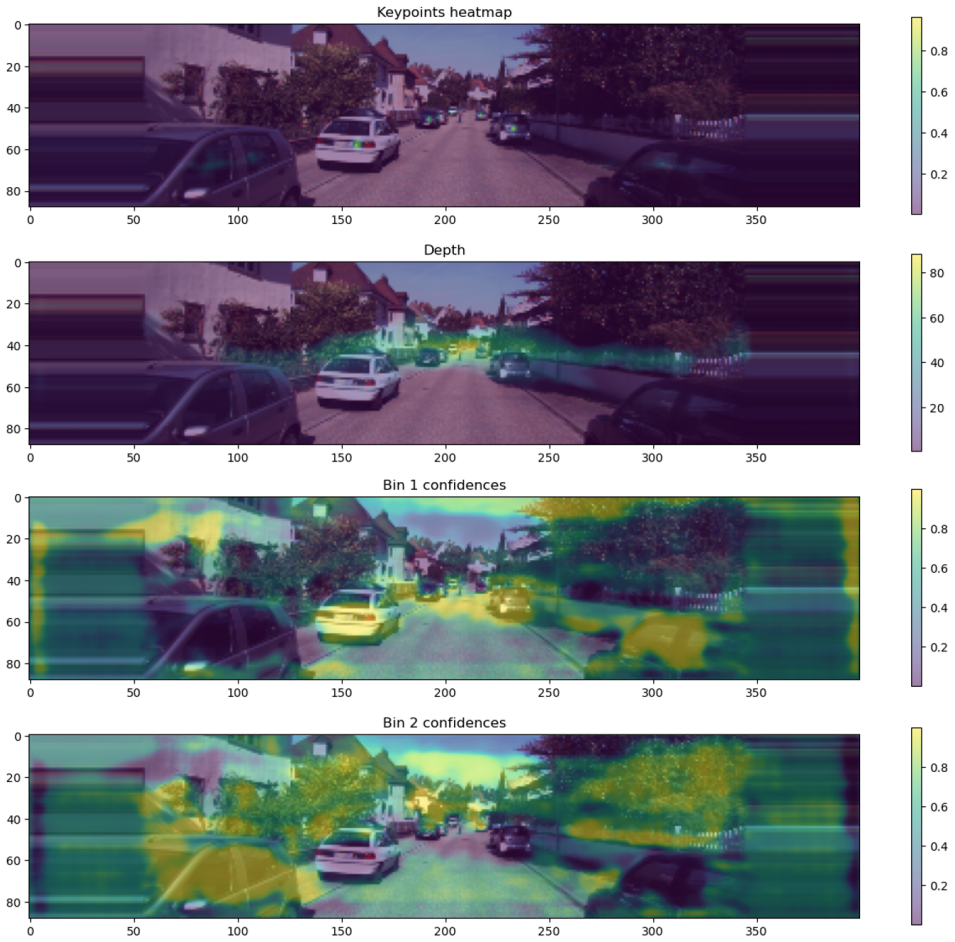


Fig. 11: Example of parameter maps for 2D object center location, depth, and orientation bin confidences for an image from the KITTI validation set. Note that the maps for the regression of 2D object size and local orientation are here not included.

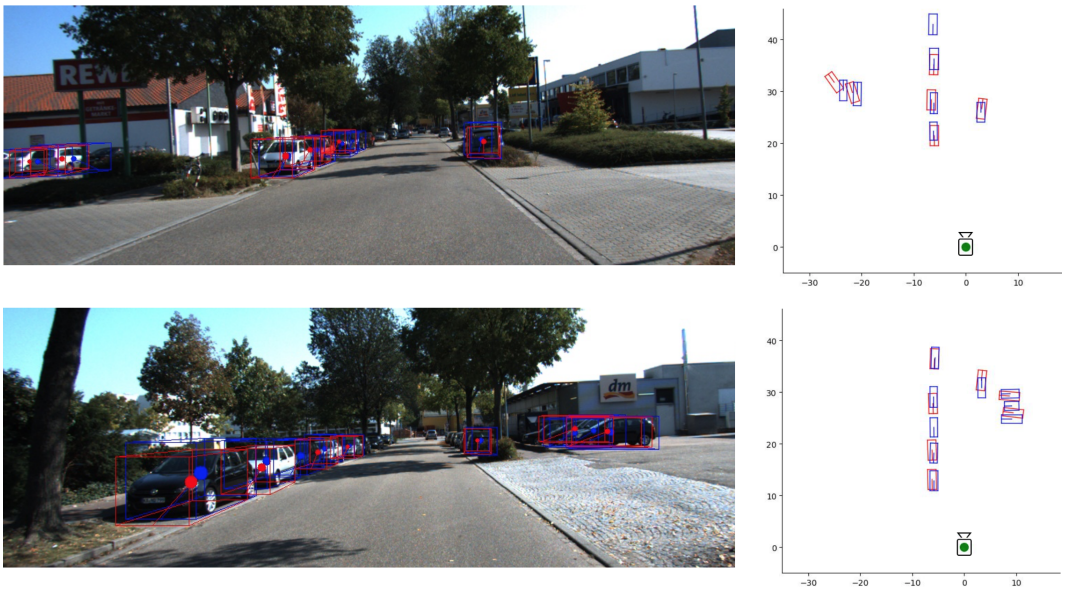


Fig. 12: Examples of 3D bounding box detections of the calibrated system for a couple of frames of the validation split of the KITTI dataset, alongside BEV perspectives. Ground truth boxes appear in blue, while the predicted ones are shown in red. The centers of the 3D bounding boxes are also drawn.

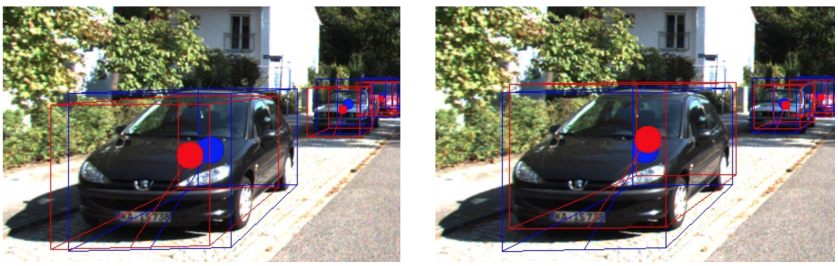


Fig. 13: Comparison between the 2D bounding box object centers predicted by the main proposed detection system (left) and the projected 3D bounding box object centers predicted by the alternative system (right). Lower errors between the ground truth centers and the predicted ones can be observed in the second case.

Figure 12 displays a couple of examples of 3D bounding box detections obtained after decoding predicted parameter maps like the ones showcased in the previous figure. Both the 3D bounding boxes projected on the image plane and their BEV versions are shown, and the ground truth boxes are also included for comparison purposes. The detected vehicles are generally well located in space when these are in good visibility conditions. Errors in this 3D localization are usually related to inaccuracies from the depth prediction head, which can cause the estimated object centers to be displaced in the camera's z axis. This is particularly common when the detected vehicles are 40+ meters away from the camera. Displacements between the projected ground truth 3D object centers and those estimated by the system are also obvious in some instances. This can most likely be attributed to the fact the centers of the 2D bounding boxes employed by the system are not in exactly the same position as the centers of the 3D bounding boxes projected on the image, as explained in [61].

Some errors in orientation can be observed too. While the overall direction of cars tends to be well interpreted by the detection system, the regression of the exact azimuth is at times off. These errors are usually low in vehicles facing forward or backwards relative to the camera, but can be more substantial in cases where they are more perpendicular to the camera or positioned far to one side. Finally, since the 3D dimensions of car instances are taken from the average values in the training dataset, an excessive length is associated to abnormally short cars.

When using the alternative version of the calibrated detector that locates projected 3D object centers instead of the centers of 2D bounding boxes, the mentioned displacements in 3D localization are slightly reduced. This can be observed in the example in Figure 13, where bounding box detections for the same car instance from the two versions of the detector are compared. The downside of this alternative approach is that due to the fact 2D bounding boxes are determined based on the scope of the 3D bounding boxes projected on the image, these 2D boxes are very sensitive to aspects like the accuracy of the orientation estimation or the divergence of the objects' 3D dimensions relative to the assumed average values for these.

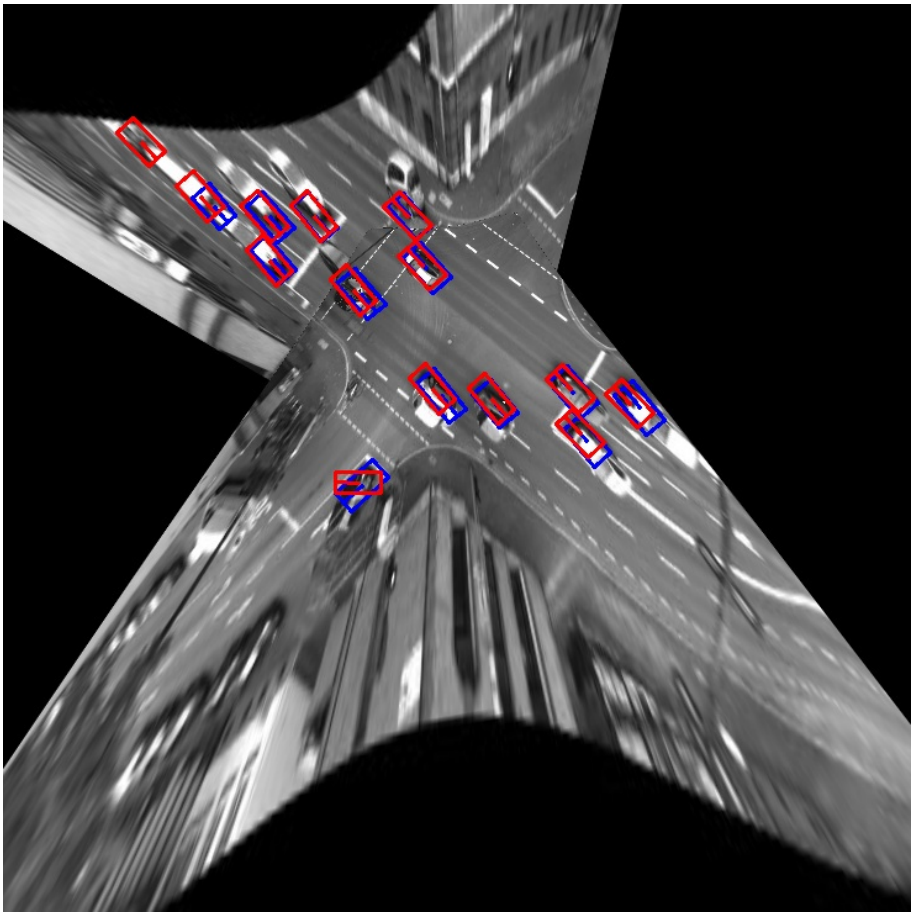


Fig. 14: Examples of rotated 2D bounding box detections of the uncalibrated system for a BEV-transformed frame of the validation split of the Ko-PER dataset. Ground truth boxes appear in blue, while the predicted ones are shown in red.

For the uncalibrated case, Figure 14 displays an example of the rotated bounding boxes predicted by the detection system on the BEV-projected images of a given timestamp within the validation set. Apart from some sporadic cases of cars not being picked up by the detector and the on and off presence of a few false positives (e.g. trucks being detected as cars), the detections appear to be quite accurate. Most of the predicted boxes fit those of the ground truth very well; some exceptions being displaced localizations, a number of car instances being detected as facing the opposite direction, alterations in azimuth and the same fixed length issue on small cars as discussed in the calibrated case. Despite these decent results, it appears the model is prone to learning to interpret orientation relying heavily on background information in the scene. When predicting the confidences for each bin and the angle increments during the multi-bin approach, the system tends to associate consistent values to parts of the streets regardless of the appearance of cars on it. This has to do with how the images on which the system is trained on always have the same static background, on which specific parts always have cars facing the same direction.

VI. Usability

Having tested and proven the potential of different kinds of lightweight detectors capable of determining the position and orientation of vehicles in space based only on monocular images, a reflection remains to be made regarding how suitable these kinds of systems can be in different situations. In the case of autonomous driving, calibrated detectors like the proposed one are the obvious choice. Since cameras are installed in cars with the task of real-time object detection as their purpose, calibration is taken for granted and the problem of generalization to other calibration parameters is not really an issue. 3D bounding box annotations can then be found in existing datasets like the one employed in this study, gathered in new scenes from camera footage and LiDAR point clouds, or synthetically created using tools like CARLA [64].

On the other hand, when dealing with cases of traffic monitoring, the options are more limited due to how the already installed cameras are rarely calibrated and how 3D bounding box annotations are not easy to access or generate. In order to use a calibrated 3D detection system like the one presented in this report, determining the matrix of the used camera/s would be essential. This could for instance be done using Zhang's method [65], which would require capturing a checkerboard-like

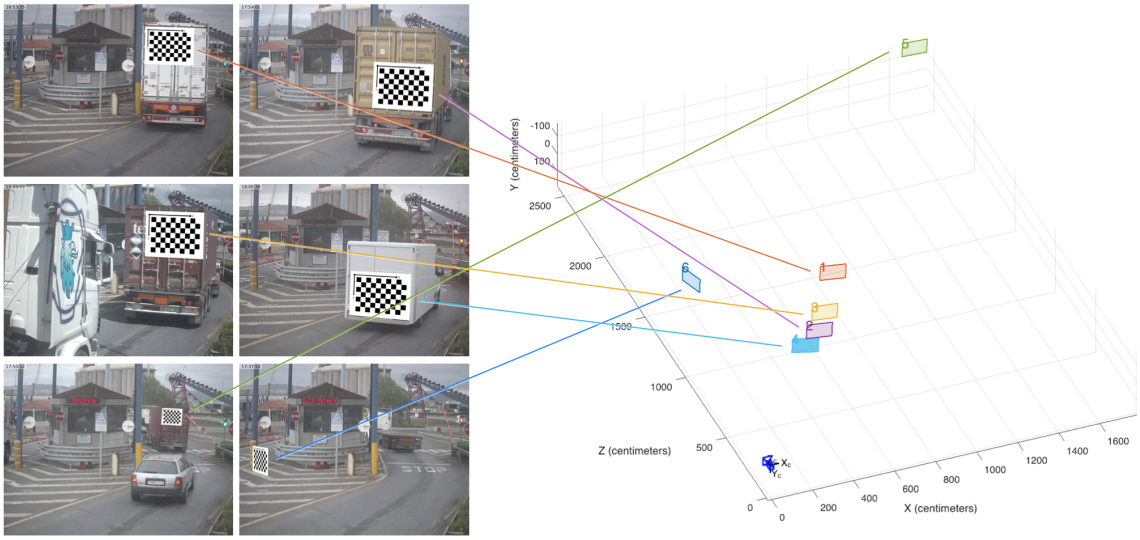


Fig. 15: Example of Zhang's method for the calibration of a security camera based on images where checkerboard patterns have been edited in onto elements with known dimensions (assuming no distortions). This is just meant as an illustrative demonstration of the kind of material needed for the process; physical patterns captured by the camera should be used to carry out the calibration in a reliable manner. Images property of AllRead Machine Learning Technologies S.L.

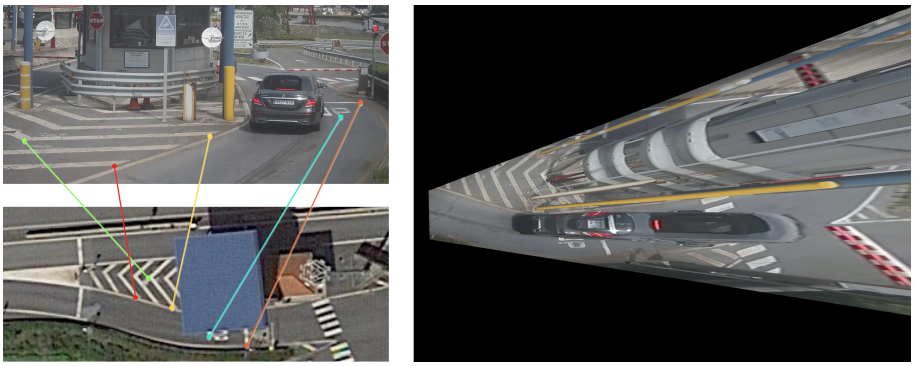


Fig. 16: Example of an image from an uncalibrated security camera transformed into a BEV version through its correspondence with a satellite view of the scene. The vehicle in the transformed image suffers from a strong projective distortion due to the low position of the camera. Images property of AllRead Machine Learning Technologies S.L. and Google LLC.

pattern with known dimensions on the camera/s in 2+ different orientations. This would make it possible to build and solve an homogeneous system of equations and carry out a non-linear refinement to eventually calculate the calibration matrix. Figure 15 portrays a visual example of how the relative position and orientation of calibration patterns present in images from an uncalibrated security camera (and in turn the calibration matrix of said camera) can be determined with the mentioned method.

Even if monitoring cameras can be calibrated, finding a way to access 3D bounding box annotations remains a problem. Unlike in autonomous driving, generating these kinds of annotations with the help of point clouds from LiDAR sensors or through synthetic data is in the vast majority of cases a non-viable option. This is where a system like the uncalibrated detector proposed in this study becomes an alternative to overcome these issues, at the cost of reducing the detection task to a 2D map. If images from the addressed footage can be related to any available map of the scene, such as a satellite image, these can be converted to BEV versions on which drawing 2D rotated bounding box annotations and training a detector can be feasible. This method has the potential to work best when cameras are in a high position relative to the ground, as the projective distortion of the vehicles is lower and a wider usable area of the targeted scene is captured. In a case like the one portrayed in Figure 16, while images from a security camera with a low positioning can be transformed to a BEV through point correspondences with a satellite view, the elements present in the resulting BEV images appear very distorted and can complicate and narrow the scope of the detection task. In addition, it remains to be tested how well a detector relying on this approach that is trained on footage from a specific scenario generalizes to other scenes.

VII. Conclusions

This study has presented an overview of the task of 3D bounding box vehicle detection in a variety of scenarios. After introducing the diverse kinds of data that existing systems are known to employ, the focus has been placed on monocular detection, leaving aside any potential additional inputs like stereo imaging or LiDAR point clouds. The complicated nature of monocular 3D detection has been emphasized, stressing the importance of having access to camera calibration information and 3D bounding box annotations that can help establish relationships between the image plane and the 3D space it represents.

Learning-based models have been highlighted as the way to go when building a system capable of relating appearance cues in images with 3D parameters linked to object localization, depth and orientation. The challenge of creating the simplest possible neural network-based detection system capable of learning these relationships and carrying out decent 3D bounding box estimations for vehicles in images has been proposed. The main idea behind the CenterNet has been adopted as the foundation for this purpose, which involves the representation of objects through their centers and a series of associated parameters that can provide information regarding aspects like their position and pose. The main motives for using a CenterNet-based model have been the end-to-end differentiability it enables, its inherent speed, and the fact it can easily be adapted to predict any kinds of relevant parameters.

Two different types of scenarios have been studied. The first has involved a situation where camera calibration information is known and 3D bounding box annotations are available for a set of images. This is usually the case in autonomous driving applications, but it can be rare to have access to these elements in other kinds of situations. When trying to implement 3D vehicle detection through existing traffic monitoring or security cameras, for instance, it is very unusual to have either camera matrices or 3D bounding box annotations available. This makes it necessary to approach the 3D detection task from a different angle, making certain assumptions that limit the kind of information one can obtain from the resulting system.

For the calibrated case, a detection system has been constructed from scratch to carry out 3D predictions on the KITTI dataset for autonomous driving. Similar to the original CenterNet, the system has been kitted with a feature extractor consisting of a U-Net-like backbone that outputs a map 4 times smaller than the input image and then feeds it to several independent prediction heads. These prediction heads are made responsible for the estimation of the localization of 2D bounding box object centers, 2D bounding box sizes, depth, and object azimuth orientations. The difference between local and global orientation is underlined, justifying how it is easier for the system to learn the former based on object appearance alone. Different kinds of lightweight encoders for the feature extractor are tested, as well as various widths for the hidden layers in the expanding part of the U-Net. The size and complexity of the encoders and the width of the hidden layers appear to be somewhat related to the detection accuracy of the resulting systems, but improvements are relatively minor when compared to the additional parameters added to the model. On the other hand, random horizontal flipping and brightness variations applied to images as data augmentation have proven to be an effective way of getting more information out of a limited amount of images and slightly improving the system's performance. Overall, the resulting detection system proves to be rather effective at detecting vehicles in decent visibility conditions, but has more trouble interpreting heavily occluded or truncated cases.

When it comes to the uncalibrated case, a flat-earth assumption is made and the 3D detection problem is treated as a 2D one. Having access to a map or a satellite view of the scene captured by the addressed camera/s, point correspondences are established between the targeted images and this top view. This enables the calculation of an homography relating both views, which in turn can be used to transform the camera images to BEV versions of themselves where dimensions are known. The task at hand is then treated as a 2D detection problem consisting of finding rotated bounding boxes, for which the same kind of CenterNet-based model is utilized.

While the two proposed lightweight detection systems have been proven to be functional, their usability in certain scenarios with limited annotated data for training is put into question. The generalization potential of the uncalibrated system remains to be tested through its application in different scenes, and the need for solutions that can help set up these kinds of detectors in new situations with as little human intervention as possible is made evident.

References

[1] S. Hoque, M. Y. Arafat, S. Xu, A. Maiti, and Y. Wei, "A comprehensive review on 3d object detection and 6d pose estimation with deep learning," *IEEE Access*, 2021.

[2] Hub-Tian, "Awesome-3d-detectors," https://github.com/Hub-Tian/Awesome-3D-Detectors, 2022.

[3] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *arXiv preprint arXiv:1904.07850*, 2019.

[4] Y. Liu, Y. Lu, Q. Shi, and J. Ding, "Optical flow based urban road vehicle tracking," in *2013 ninth international conference on computational intelligence and security*. IEEE, 2013, pp. 391–395.

[5] R. Manikandan and R. Ramakrishnan, "Video object extraction by using background subtraction techniques for sports applications," *Digital Image Processing*, vol. 5, no. 9, pp. 435–440, 2013.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[7] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.

[8] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.

[9] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[10] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[11] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[12] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[13] H. Law and J. Deng, "Cornernet: Detecting objects as paired keypoints," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 734–750.

[14] X. Zhou, J. Zhuo, and P. Krahenbuhl, "Bottom-up object detection by grouping extreme and center points," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 850–859.

[15] C. Sahin, G. Garcia-Hernando, J. Sock, and T.-K. Kim, "A review on object pose recovery: from 3d bounding box detectors to full 6d pose estimators," *Image and Vision Computing*, vol. 96, p. 103898, 2020.

[16] Z. Deng and L. Jan Latecki, "Amodal detection of 3d objects: Inferring 3d bounding boxes from 2d ones in rgb-depth images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 5762–5770.

[17] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o (n) solution to the pnp problem," *International journal of computer vision*, vol. 81, no. 2, pp. 155–166, 2009.

[18] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce, "3d object modeling and recognition using local affine-invariant image descriptors and multi-view spatial constraints," *International journal of computer vision*, vol. 66, no. 3, pp. 231–259, 2006.

[19] V. Ferrari, T. Tuytelaars, and L. Van Gool, "Simultaneous object recognition and segmentation from single or multiple model views," *International journal of computer vision*, vol. 67, no. 2, pp. 159–188, 2006.

[20] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, "3d bounding box estimation using deep learning and geometry," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 7074–7082.

[21] Z. Qin, J. Wang, and Y. Lu, "Monogrnet: A geometric reasoning network for monocular 3d object localization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8851–8858.

[22] Y. Chen, L. Tai, K. Sun, and M. M. Li, "Monocular 3d object detection using pairwise spatial relationships," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA*, 2020, pp. 14–19.

[23] T. Roddick, A. Kendall, and R. Cipolla, "Orthographic feature transform for monocular 3d object detection," *arXiv preprint arXiv:1811.08188*, 2018.

[24] F. Manhardt, W. Kehl, and A. Gaidon, "Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 2069–2078.

[25] M. Zhu, S. Zhang, Y. Zhong, P. Lu, H. Peng, and J. Lenneman, "Monocular 3d vehicle detection using uncalibrated traffic cameras through homography," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3814–3821.

[26] Y. Chen, F. Liu, and K. Pei, "Monocular vehicle 3d bounding box estimation using homograhy and geometry in traffic scene," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 1995–1999.

[27] S. Pillai, R. Ambruş, and A. Gaidon, "Superdepth: Self-supervised, super-resolved monocular depth estimation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 9250–9256.

[28] Y. Cai, B. Li, Z. Jiao, H. Li, X. Zeng, and X. Wang, "Monocular 3d object detection with decoupled structured polygon estimation and height-guided depth estimation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 10 478–10 485.

[29] M. Ding, Y. Huo, H. Yi, Z. Wang, J. Shi, Z. Lu, and P. Luo, "Learning depth-guided convolutions for monocular 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 1000–1001.

[30] B. Xu and Z. Chen, "Multi-level fusion based 3d object detection from monocular images," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2345–2353.

[31] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[32] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 2002–2011.

[33] A. Kundu, Y. Li, and J. M. Rehg, "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3559–3568.

[34] M. Zeeshan Zia, M. Stark, and K. Schindler, "Are cars just 3d boxes?-jointly estimating the 3d shape of multiple objects," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 3678–3685.

[35] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teuliere, and T. Chateau, "Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2040–2049.

[36] J. K. Murthy, G. S. Krishna, F. Chhaya, and K. M. Krishna, "Reconstructing vehicles from a single image: Shape priors for road scene understanding," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 724–731.

[37] T. He and S. Soatto, "Mono3d++: Monocular 3d vehicle detection with two-scale 3d hypotheses and task priors," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8409–8416.

[38] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, "3d object proposals for accurate object class detection," *Advances in neural information processing systems*, vol. 28, 2015.

[39] P. Li, X. Chen, and S. Shen, "Stereo r-cnn based 3d object detection for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7644–7652.

[40] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.

[41] C. R. Qi, W. Liu, C. Wu, H. Su, and L. J. Guibas, "Frustum pointnets for 3d object detection from rgb-d data," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 918–927.

[42] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 770–779.

[43] K. Shin, Y. P. Kwon, and M. Tomizuka, "Roarnet: A robust 3d object detection based on region approximation refinement," in *2019 IEEE intelligent vehicles symposium (IV)*. IEEE, 2019, pp. 2510–2515.

[44] M. Dubská, A. Herout, R. Juránek, and J. Sochor, "Fully automatic roadside camera calibration for traffic surveillance," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 3, pp. 1162–1171, 2014.

[45] M. Dubská, A. Herout, and J. Sochor, "Automatic camera calibration for traffic understanding." in *BMVC*, vol. 4, no. 6, 2014, p. 8.

[46] S. C. Lee and R. Nevatia, "Robust camera calibration tool for video surveillance camera in urban environment," in *CVPR 2011 WORKSHOPS*. IEEE, 2011, pp. 62–67.

[47] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *Advances in neural information processing systems*, vol. 27, 2014.

[48] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[49] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang, "The apolloscape dataset for autonomous driving," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2018, pp. 954–960.
[50] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.
[51] Q.-H. Pham, P. Sevestre, R. S. Pahwa, H. Zhan, C. H. Pang, Y. Chen, A. Mustafa, V. Chandrasekhar, and J. Lin, "A* 3d dataset: Towards autonomous driving in challenging environments," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 2267–2273.
[52] W. LLC, "Waymo open dataset: An autonomous driving dataset," 2019.
[53] N. Gählert, N. Jourdan, M. Cordts, U. Franke, and J. Denzler, "Cityscapes 3d: Dataset and benchmark for 9 dof vehicle detection," *arXiv preprint arXiv:2006.07864*, 2020.
[54] E. Strigel, D. Meissner, F. Seeliger, B. Wilking, and K. Dietmayer, "The ko-per intersection laserscanner and video dataset," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2014, pp. 1900–1901.
[55] Y. Xiang, R. Mottaghi, and S. Savarese, "Beyond pascal: A benchmark for 3d object detection in the wild," in *IEEE winter conference on applications of computer vision*. IEEE, 2014, pp. 75–82.
[56] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.
[57] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[58] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.
[59] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.
[60] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International conference on machine learning*. PMLR, 2019, pp. 6105–6114.
[61] X. Ma, Y. Zhang, D. Xu, D. Zhou, S. Yi, H. Li, and W. Ouyang, "Delving into localization errors for monocular 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4721–4730.
[62] B. Xu, N. Wang, T. Chen, and M. Li, "Empirical evaluation of rectified activations in convolutional network," *arXiv preprint arXiv:1505.00853*, 2015.
[63] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Data-driven 3d voxel patterns for object category recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1903–1911.
[64] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.
[65] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.