PlaneCalib: Automatic Camera Calibration by Multiple Observations of Rigid Objects on Plane

Vojtěch Bartl Roman Juránek Jakub Špaňhel Adam Herout
{ibartl, ijuranek, ispanhel, herout}@fit.vut.cz
Graph@FIT, Faculty of Information Technology, Brno University of Technology
Brno, Czech Republic

***Abstract*—In this work, we propose a novel method for automatic camera calibration, mainly for surveillance cameras. The calibration consists in observing objects on the ground plane of the scene; in our experiments, vehicles were used. However, any arbitrary rigid objects can be used instead, as verified by experiments with synthetic data. The calibration process uses convolutional neural network localisation of *landmarks* on the observed objects in the scene and the corresponding 3D positions of the localised landmarks — thus fine-grained classification of the detected vehicles in the image plane is done. The observation of the objects (detection, classification and landmark detection) enables to determine all typically used camera calibration parameters (focal length, rotation matrix, and translation vector). The experiments with real data show slightly better results in comparison with state-of-the-art work, however with an extreme speed-up. The calibration error decreased from $3.01\,\%$ to $2.72\,\%$ and $1\,223\times$ faster computation was achieved.**

I. Introduction

Camera calibration is an important task in many computer vision applications. A properly calibrated camera (including scale) enables to carry out measurements in real world units, such as metres, offering a broader usage of surveillance cameras, whose count is growing nowadays. One of the basic and widely used methods for camera calibration is the one proposed by Zhang [1] using a suitable pattern placed in front of the camera; in his case, this was a planar checkerboard, but arbitrary planar and non-planar ([2], [3]) objects have been used since, too. Despite their sufficient results in other cases, these methods are not very suitable for potentially unreachable surveillance cameras (*e.g.* camera stream from the other side of the world) or in the cases where the use of a pattern would be complicated (e.g. in the middle of the traffic lanes of a motorway).

Despite the fact that other methods for the calibration of surveillance cameras have been proposed — these will be discussed later in Section II — all of them either have some constraints, limitations, or do not reach a satisfactory accuracy. Our goal is to develop fully automatic calibration algorithms for surveillance, providing the intrinsic camera parameters, extrinsic parameters, as well as the scene's scale with regard to the ground plane. As we focus on traffic surveillance in this paper, we use vehicles as calibration objects, but the presented algorithm works with any other rigid objects available.

In general, the camera (following the pinhole model) projects 3D points $\mathbf{x}$ in world homogeneous coordinates $\mathbf{x} = (x, y, z, 1)^\top$ to their 2D position $\mathbf{x}' = (u, v, 1)^\top$ in the image plane:

$$\lambda\,\mathbf{x}' = \mathbf{K}\,[\mathbf{R}|\mathbf{t}]\,\mathbf{x}. \tag{1}$$

By calibration we mean the process of obtaining the intrinsic camera matrix $\mathbf{K}$ (3×3 matrix), the camera rotation matrix $\mathbf{R}$ (3×3 matrix) and its translation vector $\mathbf{t}$ (vector of dimension 3) that best model such a projection. Typically, some assumptions can be used with a surveillance camera — that the principal point is in the center of the image (surveillance cameras generally meet this assumption [4], [5]), the pixels are square and there is no skew, therefore $\mathbf{K}$ only contains one unknown parameter (focal length). The extrinsic parameters consist of rotation and translation $[\mathbf{R}|\mathbf{t}]$ and have six degrees of freedom. Our approach deals with a planar surface with moving objects (vehicles). As there is no given central point on the ground plane, the origin of the world coordinate system can be potentially everywhere. Since the first two elements of the translation vector $\mathbf{t}$ indicate the location of the world origin w.r.t. the principal point, we can set these values arbitrarily — further, zero values are used so that the world origin will be projected in the principal point. The third element of the translation vector $\mathbf{t}$ stands for the height of the camera and thus it is the only parameter of $\mathbf{t}$ which we will be interested in.

Our solution is based on localisation of landmarks on the selected object in the 2D image (vehicles in our case) and the corresponding 3D locations of these points in the object coordinate system (OCS). Further, these 2D-3D correspondences can be used for determining the rotation and translation of the object w.r.t. the camera. This task is usually solved using *PnP* (Perspective-n-Point) methods [6], [7], [8]. However, their algorithms require the knowledge of the focal length f, which is unknown in our task. While there are *PnPf* [9], [10], [11] methods which can recover the focal length, this is only possible based on one observation of a rigid structure. Moreover, the *PnPf* methods have a high error rate in focal length estimation when there are higher values of noise at the locations of the points, which is quite usual in automatically localised 2D landmarks as will be mentioned later. In our scenario, we want to use more observations of objects in the ground plane, which means that *PnP*/*PnPf* itself is not feasible as it relates to each object separately, without taking into account the relations between individual objects. In our case, the **relations between the observations of single objects are crucial** and carry necessary information.

In this paper, we propose a practical calibration method based on the usage of 2D-3D correspondences and transformation between the object, camera, and world coordinate systems

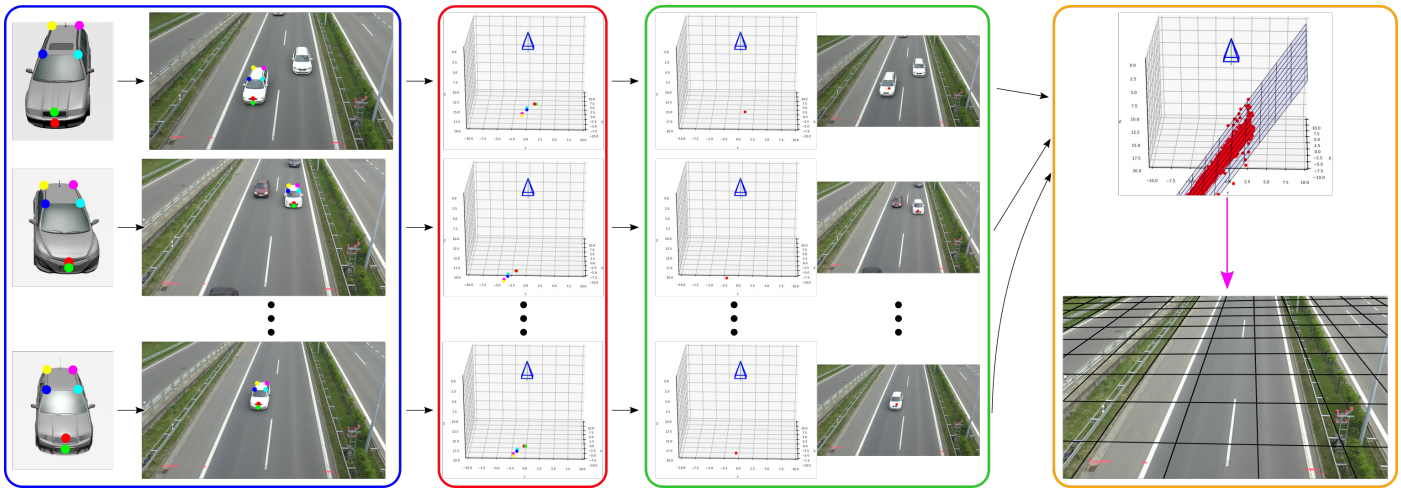

Fig. 1. Overview of the proposed *PlaneCalib* method. First of all, vehicles are detected, localised, classified in images (video frames), and landmarks are localised within these detections. Correspondences of the localised 2D and 3D landmarks (**blue**) are used to solve *PnP* to obtain the rotation and translation of the object with respect to the camera (**red**). For each object, the origin point which lies in the ground plane is transformed into the camera coordinate system (**green**). Finally, these potential ground plane points of all the objects observed are used to compute the calibration ground plane as the best fit to these origin points (**orange**) and the transformation of this ground plane to the world coordinate system is computed (**magenta**).

(see Figure 1 for an overview of the method). Our method mitigates most of the limitations typical for the other methods. However, certain requirements still must be fulfilled — the cars must move on a common ground plane, and there must be a certain number of observations of vehicles whose particular type (make & model) can be recognized by the system. Our method allows for car detection, fine-grained classification, and detection of landmarks on the detected cars.

II. Related Work

The existing methods for camera calibration applicable in traffic surveillance are mostly based on manual measurements [12], [13], markers [14], [15], [16], [17], vehicle movement [18], [4], [19], or other principles like optical flow, recognition of cars or license plates [20], [21], [22], [23].

With the exceptions of [20], [18], [19], traffic calibration solutions require us to know certain dimensions in the scene (*e.g.* width of lanes, length of dashed markings, height of the camera, *etc.*) and thus they cannot be used in a fully automatic manner. The calibration is often constrained to a limited range of viewpoints and it supports only straight motion of vehicles. Maduro *et al.* [13] assume a known angle of the camera and a known width of the traffic lanes. Marker-based methods either use special markers or horizontal road markings. Cathey and Dailey [14] detect the vanishing point of the lane marking with a known lane width. Grammatikopoulos *et al.* [16] assume a camera with zero roll and they detect the vanishing point of the road markings. He *et al.* [17] calibrate the camera according to a pattern formed by dashed line markings on the road. Do *et al.* [15] use an equilateral triangle with known dimensions drawn on the road.

Solutions based on vehicle movement typically detect vanishing points in the direction towards the vehicle motion (first VP) and in directions perpendicular to it (second and third VP). Schoepflin and Dailey [4] use a background model to detect lane boundaries in the activity map. The intersection of lanes is assumed to be the vanishing point of vehicle motion. The second vanishing point is detected from vehicle edges. One known length in the scene is required for full calibration.

Dubská *et al.* [18] propose a fully automatic method for calibration. It assumes straight movement of cars on the road; this condition is usually met on motorways, but it cannot be assumed in car parks, on roundabouts and in similar scenarios. Their method uses a particular form of cascaded Hough transform [24] to search for vanishing points and the scene scale is inferred from the mean size of the vehicles observed. Sochor *et al.* [19] extend this method by using a more accurate detection of vanishing points and scale inference. They use fine-grained recognition of vehicles and align the bounding box of a known 3D geometry to the observations in the image. The accuracy of this method is sufficient for speed measurement with a mean error of 1.1 kph.

Our method is to some extent similar to *AutoCalib* [20], which also uses observation of passing vehicles to perform the calibration. Their camera calibration is optimized by minimizing the re-projection error of known average 3D positions of detected landmarks on the vehicles from the rear side. However, *AutoCalib* works with a known focal length f and it is limited to a coherent view of the vehicles (roughly from the rear).

The proposed method builds upon a state-of-the-art work [25] where authors used similar conditions. Correspondences between 2D keypoints and 3D positions in the object coordinate system were used. The mentioned method consists of dual optimization — optimization of single vehicles' observations and optimization of all calibration parameters. For each setting of calibration parameters (which are optimized), the positions of all vehicles in the related world coordinate system are also optimized. The existing method reaches state of the art results in calibration accuracy; however, due to its dual optimization complexity it is extremely slow as we show in our experiments in Section V.

The first step of landmark localisation (either in video

or individual images) is detection of the vehicles in the 2D image. Nowadays, detection of objects is mostly performed using convolutional neural networks as these allow for high accuracy. The *feature extractor* is the backbone of detection networks; it is common for all the recent detectors' meta-architectures. Any of the available classification CNNs can be used (*e.g.* VGG-16 [26], Inception v2 [27], Inception v3 [28], ResNet-101 [29], MobileNet [30], *etc.*). In our case we use the Faster R-CNN detector [31] and ResNet-50 [29] for detection of cars in images.

Further, we need to classify the detected cars. Recently, a number of methods for fine-grained recognition of various objects have been published, even for vehicles in particular [32], [33], [34], [35], [36], [37]. A considerable group of existing fine-grained recognition algorithms are specialized in classification of vehicles. Works [38], [39] extract hand-crafted features from localised license plates or the vehicles' frontal masks which means that they are limited to frontal/rear images of vehicles only. State-of-the-art results in this field are achieved by methods based on CNNs. Liu *et al.* [34] propose to use *Deep Relative Distance* trained on the re-identification task to extract more discriminative feature vectors, and *Coupled Clusters Loss* function during training. On the other hand, Sochor *et al.* [37] improve the classification accuracy using an "unwrapped" version of the 3D bounding box of the vehicles detected to the 2D plane as the input of CNN for fine-grained recognition and other image modifications. Therefore, we use the recognizer [36], [37] which provides make & model & submodel & model year of the car and makes it possible to select and use proper 3D CAD model providing 3D positions of landmarks.

The last step is localisation of landmarks in the 2D image. It is based on a fully convolutional neural network proposed by Newell *et al.* [40]. This design of convolutional neural network was used by Wang *et al.* [41] for localising landmarks on vehicles (OIF, *Orientation Invariant Features*), which we use further.

III. PlaneCalib Method

The whole process of camera calibration is based on detection of landmarks on objects and their known 3D locations in object coordinate space (2D-3D correspondences).

A. Estimation of Extrinsic Camera Parameters

The process of extrinsic camera parameters estimation works with the known intrinsic matrix $\mathbf{K}$; it is thus considered to be known for the purpose of method description. The process of focal length estimation is described later in Section III-C. The video is transformed into a set of cars' observations:

$$\mathcal{C} = \{\mathbf{c}_1, \ldots, \mathbf{c}_N\} \tag{2}$$

and for each car $\mathbf{c}_i$, a set of 2D landmark locations detected by the neural network [41] is available:

$$\bar{\mathcal{K}}^{\mathbf{c}_i} = \left\{\bar{\mathbf{k}}_1^{\mathbf{c}_i}, \ldots, \bar{\mathbf{k}}_K^{\mathbf{c}_i}\right\}. \tag{3}$$

For each of the observed vehicles $\mathbf{c}_i$, the correct 3D positions in the vehicle's local coordinate system (OCS) are available:

$$\hat{\mathcal{K}}^{\mathbf{c}_i} = \left\{\hat{\mathbf{k}}_1^{\mathbf{c}_i}, \ldots, \hat{\mathbf{k}}_K^{\mathbf{c}_i}\right\}. \tag{4}$$

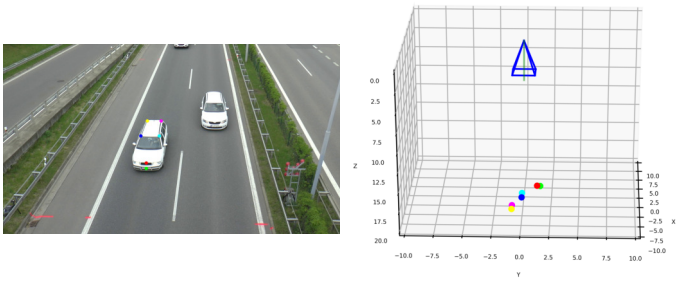

Fig. 2. *left:* Detected 2D landmarks in a single frame. *right:* 3D CCS with corresponding 3D positions of detected landmarks in the 2D frame. The *PnP* solution is computed to obtain parameters $[\mathbf{R}^{\mathbf{c}_i}|\mathbf{t}^{\mathbf{c}_i}]$.

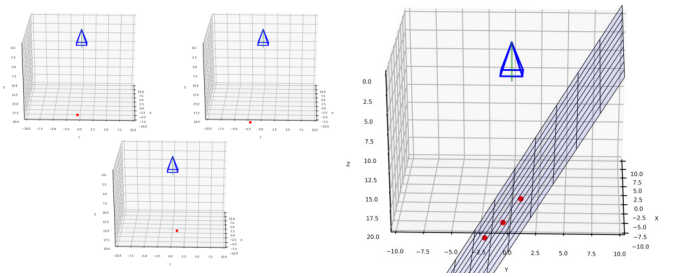

Fig. 3. Computing the plane in the camera coordinate system. *left*: Origins of three different objects transformed into the camera coordinate system. *right*: plane in the camera coordinate system localised by three (minimal case) origins of objects.

Detected 2D ($\bar{\mathcal{K}}^{\mathbf{c}_i}$) and 3D ($\hat{\mathcal{K}}^{\mathbf{c}_i}$) correspondences for each car $\mathbf{c}_i$ can be used to solve the *PnP* [7] problem — the solution provides extrinsic camera parameters (rotation matrix $\mathbf{R}^{\mathbf{c}_i}$ and translation vector $\mathbf{t}^{\mathbf{c}_i}$ for the transformation between the object coordinate system (OCS) and the camera coordinate system (CCS)).

Original 3D positions $\hat{\mathcal{K}}^{\mathbf{c}_i}$ in OCS can thus be transformed into 3D positions in CCS as:

$$\mathbf{k}_j^{\mathbf{c}_i} = [\mathbf{R}^{\mathbf{c}_i}|\mathbf{t}^{\mathbf{c}_i}] \begin{bmatrix} \hat{\mathbf{k}}_j^{\mathbf{c}_i} \\ 1 \end{bmatrix}. \tag{5}$$

An example of object points' positions in 3D CCS (after solving *PnP* and transformation by eq. (5) from OCS to CCS) with corresponding points in the 2D frame is depicted in Figure 2. For each car $\mathbf{c}_i$, parameters for transformation from OCS to CCS are available and the corresponding point describing the origin of OCS can be transformed into CCS. From at least three of these origin points, a plane in the CCS can be computed (an example can be seen in Figure 3).

a) CCS to WCS transformation: The extrinsic camera parameters describe the camera position in the WCS and the mutual transformation between WCS and CCS; the world ground plane is the same as the plane in the CCS after applying this transformation — this is the goal of the described method. The equation of the plane is:

$$ax + by + c = z. \tag{6}$$

The plane can be recovered in a least squares manner as follows:

$$\begin{bmatrix} x_0 & y_0 & 1 \\ x_1 & y_1 & 1 \\ & \vdots & \\ x_N & y_N & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} z_0 \\ z_1 \\ \vdots \\ z_N \end{bmatrix}, \tag{7}$$

where x_n, y_n, z_n are 3D coordinates of at least three car's ground points transformed to the camera coordinate system. This equation can be simplified to matrix notation as:

$$A\mathbf{x} = B, \tag{8}$$

where $\mathbf{x}$ are the plane parameters $[a, b, c]^\top$. These plane parameters can thus be computed as:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = A^+ B, \tag{9}$$

where $A^+ = (A^\top A)^{-1} A^\top$ is Moore-Penrose pseudoinverse of A.

Inference of the plane's normal vector (denoted as $\mathbf{n}$) from parameters $[a, b, c]^\top$ is straightforward. The rotation matrix $\mathbf{R}$ can be computed from the rotation axis $\mathbf{u}$ and angle θ. The axis is determined by the normal vector $\mathbf{n}$ and camera view vector $\mathbf{c} = [0, 0, 1]^\top$ (marked green in Figures 2 and 3) as $\mathbf{u} = \mathbf{c} \times \mathbf{n}$. The rotation angle θ is:

$$\theta = \frac{\arccos(\mathbf{c} \cdot \mathbf{n})}{|\mathbf{c}| \cdot |\mathbf{n}|}. \tag{10}$$

The final camera rotation matrix $\mathbf{R}$ is then obtained by *Rodrigues* formula:

$$\mathbf{R} = (cos\,\theta)I + (sin\,\theta)[\mathbf{u}]_\times + (1 - cos\,\theta)(\mathbf{u} \otimes \mathbf{u}), \tag{11}$$

where $[\mathbf{u}]_\times$ is a cross product matrix (skew-symmetric matrix):

$$[\mathbf{u}]_\times = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{12}$$

and $\mathbf{u} \otimes \mathbf{u}$ is a tensor product of the vectors:

$$\mathbf{u} \otimes \mathbf{u} = \mathbf{u}\mathbf{u}^\top = \begin{bmatrix} u_x^2 & u_x u_y & u_x u_z \\ u_x u_y & u_y^2 & u_y u_z \\ u_x u_z & u_y u_z & u_z^2 \end{bmatrix}. \tag{13}$$

Assuming the world origin in the principal point, the first two elements of the vector $\mathbf{t}$ are set to zero and the Z-coordinate of the camera is derived from equation (6) with x and y set to zero (world origin projected into the principal point — no translation within x or y axis). The translation vector $\mathbf{t}$ is then set to $[0, 0, c]^\top$ where c is one of the plane parameters from (9). An example of final calibration plane is shown in Figure 4.

B. Suppression of Outlier Samples

Although the localisation of the landmarks works well enough (according to [41], 88.8 % of landmarks are correctly predicted within 3 pixels), it fails significantly in some vehicle instances (occluded ones, weird angles, unusual painting, custom modifications, ...). These outliers bring noise into the calibration process and can affect the plane fitting computation

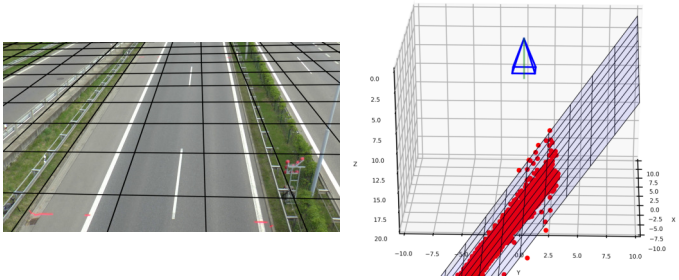

Fig. 4. Final calibration plane with the use of all available cars in $\mathcal{C}$. *left:* 2D frame with the final calibration in WCS. *right:* 3D CCS with potential ground points and fitted plane.

(9) and thereby the final calibration. For this reason, we propose to compute the re-projection error for each car $\mathbf{c}_i$ and transform it into weight. For each observed car, *PnP* [7] is computed to obtain extrinsic parameters. The 3D points from $\hat{\mathcal{K}}^{\mathbf{c}_i}$ are projected to the image plane by (1), yielding:

$$\tilde{\mathcal{K}}^{\mathbf{c}_i} = \left\{ \tilde{\mathbf{k}}_1^{\mathbf{c}_i}, \ldots, \tilde{\mathbf{k}}_K^{\mathbf{c}_i} \right\}. \tag{14}$$

The particular vehicle instance $\mathbf{c}_i$ is then assigned its individual normalized re-projection error:

$$\epsilon(\mathbf{c}_i) = \frac{\sum_{j=1}^{K} \left| \tilde{\mathbf{k}}_j^{\mathbf{c}_i}, \bar{\mathbf{k}}_j^{\mathbf{c}_i} \right|}{\sum_{j=1}^{K} \left| \tilde{\mathbf{k}}_j^{\mathbf{c}_i}, \mathbb{K}^{\mathbf{c}_i} \right|}, \tag{15}$$

where $\mathbb{K}^{\mathbf{c}_i}$ is the mean of all the points $\bar{\mathbf{k}}^{\mathbf{c}_i}$. The fraction normalizes the re-projection error so that vehicle instances of different sizes are mutually comparable.

For each vehicle $\mathbf{c}_i$, its re-projection error $\epsilon(\mathbf{c}_i)$ is thus available, which serves for setting of the *weight*:

$$\mathbf{w}^{\mathbf{c}_i} = \left(\frac{1}{\epsilon(\mathbf{c}_i)} \right). \tag{16}$$

Equation (9) for fitting a plane to the set of points must be slightly modified for the use of weights. It is convenient to denote the weights as a matrix W with $\mathbf{w}^{\mathbf{c}_i}$ on its diagonal. Then, the plane fitting equation can be modified as:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = (A^\top W A)^{-1} A^\top W B. \tag{17}$$

The calibration ground plane is thus computed by the equation (17) instead of eq. (9), and the rest of the calibration parameters inference is identical to the process described in Section III-A.

C. Focal Length Estimation

As mentioned earlier, our approach assumes the principal point in the middle of the frame, square pixels, and zero skew. The only unknown intrinsic parameter is thus the *focal length*. Many previous methods as well as the *AutoCalib* [20] method assume the focal length to be known — we consider this assumption too limiting as the goal is to calibrate a general

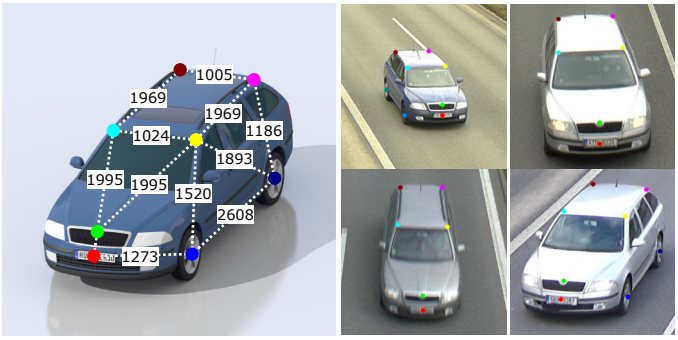

Fig. 5. *left:* Distances $\hat{\delta}$ (in millimeters) in the 3D model of *Skoda Octavia mk2* car. Only a small subset of all the distances in the set $\hat{\mathcal{K}}^{\mathbf{c}_i}$ is shown. *right:* Examples of detected landmarks on actual vehicles of the same type.

camera without any previous knowledge. We thus propose to use focal length estimation.

The process of focal length estimation uses the detected 2D landmarks ($\bar{\mathcal{K}}^{\mathbf{c}_i}$) and precise 3D positions ($\hat{\mathcal{K}}^{\mathbf{c}_i}$) located on the cars' models. We define the pairwise distance between points a and b as:

$$\hat{\delta}(\mathbf{c}_i, a, b) = \left| \hat{\mathbf{k}}_a^{\mathbf{c}_i}, \hat{\mathbf{k}}_b^{\mathbf{c}_i} \right|. \tag{18}$$

An example of these distances $\hat{\delta}(\mathbf{c}_i, a, b)$ and a few examples of the detected 2D landmarks $\bar{\mathcal{K}}^{\mathbf{c}_i}$ are shown in Figure 5.

The whole process of focal length estimation consists of computing the known distances $\hat{\delta}(\mathbf{c}_i, a, b)$ and optimization of the proposed error function describing the fit of these distances based on calibration parameters. For this reason, it is necessary to recompute each 3D point's position in WCS based on its 2D landmark's projection $\bar{\mathbf{k}}_j^{\mathbf{c}_i}$ and some calibration parameters $\mathbf{K}$, $\mathbf{R}$ and $\mathbf{t}$ (denoted together as ϕ). This 3D position computation is based on the known height above the ground plane (the Z-coordinate of $\hat{\mathbf{k}}_j^{\mathbf{c}_i}$ in the 3D model) and the assumption that the vehicles are moving in a single ground plane. With given calibration parameters ϕ, the 3D position can be reconstructed; it will be named $\mathbf{k}_j^{\mathbf{c}_i}(\phi)$ as it is a function of the calibration parameters ϕ (forming the intrinsic matrix $\mathbf{K}_\phi$, rotation matrix $\mathbf{R}_\phi$ and translation vector $\mathbf{t}_\phi$).

Starting with the camera projection (1), in our notation:

$$\lambda \begin{bmatrix} \bar{\mathbf{k}}_j^{\mathbf{c}_i} \\ 1 \end{bmatrix} = \mathbf{K}_\phi \left(\mathbf{R}_\phi \mathbf{k}_j^{\mathbf{c}_i}(\phi) + \mathbf{t}_\phi \right), \tag{19}$$

which can be rearranged to:

$$\mathbf{R}_\phi^{-1} \mathbf{K}_\phi^{-1} \lambda \begin{bmatrix} \bar{\mathbf{k}}_j^{\mathbf{c}_i} \\ 1 \end{bmatrix} = \mathbf{k}_j^{\mathbf{c}_i}(\phi) + \mathbf{R}_\phi^{-1} \mathbf{t}_\phi \tag{20}$$

and further:

$$\mathbf{k}_j^{\mathbf{c}_i}(\phi) = \mathbf{R}_\phi^{-1} \left(\mathbf{K}_\phi^{-1} \lambda \begin{bmatrix} \bar{\mathbf{k}}_j^{\mathbf{c}_i} \\ 1 \end{bmatrix} - \mathbf{t}_\phi \right). \tag{21}$$

The projective scale λ can be computed from eq. (20) by using the Z-coordinate determined from the CAD model $\left[\hat{\mathbf{k}}_j^{\mathbf{c}_i} \right]_3$. For computing the λ, only the third component of all the column vectors is used from (20) (operator $[\mathbf{x}]_3$ symbolizes extraction of the third member of a vector):

$$\lambda = \frac{\left[\hat{\mathbf{k}}_j^{\mathbf{c}_i} \right]_3 + \left[\mathbf{R}_\phi^{-1} \mathbf{t}_\phi \right]_3}{\left[\mathbf{R}_\phi^{-1} \mathbf{K}_\phi^{-1} \begin{bmatrix} \bar{\mathbf{k}}_j^{\mathbf{c}_i} \\ 1 \end{bmatrix} \right]_3} \tag{22}$$

For each car $\mathbf{c}_i$ and each pair of visible landmarks a, b and their position in the world coordinate system $\mathbf{k}_j^{\mathbf{c}_i}(\phi)$, their 3D distance can then be computed as:

$$\delta(\mathbf{c}_i, a, b, \phi) = \left| \mathbf{k}_a^{\mathbf{c}_i}(\phi), \mathbf{k}_b^{\mathbf{c}_i}(\phi) \right|. \tag{23}$$

The error of the distance between a pair of visible landmarks given some calibration parameters ϕ can be expressed as follows:

$$\varepsilon(a, b, \phi) = \frac{\delta(\mathbf{c}_i, a, b, \phi) - \hat{\delta}(\mathbf{c}_i, a, b)}{\hat{\delta}(\mathbf{c}_i, a, b)}. \tag{24}$$

The total error/cost of all the observations in the scene can be computed as follows:

$$\mathcal{E}(\phi) = \sum_{\mathbf{c}_i \in \mathcal{C}} \sum_{a,b} \frac{\mathbf{w}^{\mathbf{c}_i} \varepsilon(a, b, \phi)}{\mathbf{w}^{\mathbf{c}_i}}, \tag{25}$$

where $\mathbf{w}^{\mathbf{c}_i}$ is the weight defined by (16). The process of focal length estimation consists of localising such a focal length f that minimizes this error (25). Once a focal length value f is determined, the rest of the calibration parameters ($\mathbf{R}_\phi$ and $\mathbf{t}_\phi$) are determined by the method described in Section III-A.

As can be seen in Figure 6, the proposed error function is convex and a single minimal value can be found. In our experiments, the optimal focal length f is found by *Brent method*.

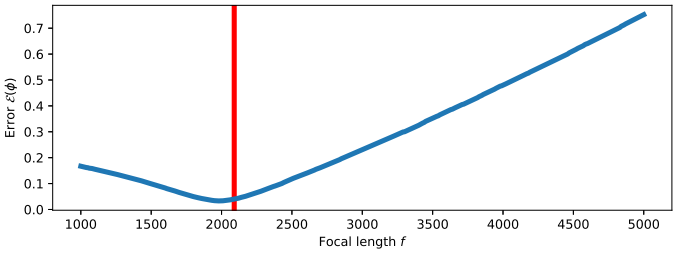

Fig. 6. Error function $\mathcal{E}(\phi)$ on single scene of real world dataset (see Section IV). Ground truth focal length value is marked red.

D. Summary of Calibration Process

The main part of the whole calibration is the optimization of focal length f. In each step of the focal length optimization, *weights* (as defined by (16)) are computed as these can change by the different settings of parameter f. Afterwards, extrinsic camera parameters are obtained by the process described in Section III-A. With all known camera parameters ϕ, the error function (25) can be evaluated. Optimization reaches the best possible focal length value f together with the final extrinsic camera calibration parameters.

Fig. 7. Sample images from the *BrnoCompSpeed* dataset [42] with resulting calibration.

IV. Datasets

Two datasets were used for evaluation of the proposed method: A synthetic dataset which simulates passing of objects in front of the camera, and a real-world dataset *BrnoCompSpeed* [42], which contains recordings of motorways.

a) BrnoCompSpeed: Published by Sochor *et al.* [42]; its purpose was speed measurement of vehicles by a single monocular camera. The dataset contains motorway video recordings captured from multiple bridges over the motorway and it mimics typical traffic surveillance scenarios. Ground truth measurements on the ground plane and 20,865 vehicles with ground truth speed are available (see Figure 7 for a few samples). The dataset was shot during approximately one hour at seven different locations with three cameras at each location, making ≈ 21 hours of recordings in total.

b) Synthetic Dataset: For the purpose of evaluation of the calibration, similarly as in the state-of-the-art work [25], a synthetic dataset simulating objects passing in front of the camera (similar to common motorway surveillance cameras) was used. The cars' landmarks (same as those located within camera calibration in Section III-A), as well as different simple models (cube, block, table, *etc.*) to face diverse possibilities and prove method stability and usability with other types of rigid objects and viewpoints are generated within the scenes — 40 different scenes are generated, each containing randomly 500 – 4000 passages of objects, making 93,652 objects in total. The calibration parameters are set randomly for each scene in ranges which follow typical surveillance camera settings. This dataset is made public with the publication of this paper.[1]

A. Ground Truth for Evaluation

Both of the datasets are equipped with ground-truth measurements in the ground plane and corresponding 2D positions in the image plane which form distances in the ground plane; these distance measurements can be used for the calibration precision evaluation. The *BrnoCompSpeed* dataset is equipped with 4 – 10 measurements for each of the 21 cameras, typically in the direction of the vehicles' movement and in the direction perpendicular to it. For the synthetic dataset, 20 point pairs (measurements) were randomly generated in the image plane with corresponding distances between these points in the ground plane.

[1] https://medusa.fit.vutbr.cz/traffic

V. Experiments

Each video frame in the real *BrnoCompSpeed* dataset is processed by a sequence of neural networks performing individual steps in the whole process of detection, classification, and localisation of landmarks. First, a neural network for detection and localisation of vehicles is used (Faster R-CNN [31], in our case trained on the COD20k dataset published by Juránek *et al.* [43]). Then, the detected vehicle instances are classified into fine-grained classes (make & model & submodel & model year) by employing an existing vehicle classifier [37]. For further processing, we use 9 most frequent models in the data for which we have precise 3D CAD models, and for these, landmarks are localised in the vehicles' images by another neural network [41]. For calibration, we use a set of 12 most stable landmarks out of 20 detected by the model.

a) Experimental Results: As mentioned in Section IV-A, for each of the testing scenes, measurements in the world coordinate system are available with distances between these points in the real world:

$$\hat{\mathcal{D}} = \left\{\hat{d}_1, \ldots, \hat{d}_D\right\}. \tag{26}$$

For each measurement, 2D endpoints in the image plane are also available which refer to this distance measurement. These 2D points can be re-projected to the ground plane with the knowledge of the calibration parameters obtained by an arbitrary method, by using the same technique as described in Section III-C, eq. (21) with Z-coordinate set to the value of 0. The measurements between the re-projected points are the following:

$$\mathcal{D} = \{d_1, \ldots, d_D\} \tag{27}$$

and these can be compared against the ground truth $\hat{\mathcal{D}}$ by measuring the relative root mean square error:

$$\mathrm{RMSE} = \sqrt{\frac{1}{D}\sum_{i=1}^{D}\left(\frac{d_i - \hat{d}_i}{\hat{d}_i}\right)^2}. \tag{28}$$

We compared our proposed method to *OptInOpt* [25] — the state-of-the-art alternative solution — see Figure 8. The mean RMSE across all scenes is 3.65 % for the *PlaneCalib* method and 3.01 % for the *OptInOpt* method (median values 3.23 % and 2.43 % for *PlaneCalib* and *OptInOpt* respectively). Several examples of calibration on the *BrnoCompSpeed* dataset can be seen in Figure 7.

As mentioned previously in Section II, the computation complexity of the *OptInOpt* method is much higher than that of the proposed *PlaneCalib* method. The comparison of computation speed is depicted in Figure 9. The *OptInOpt* method computation takes days, while the computation time of the proposed *PlaneCalib* method only takes minutes with a comparable level of accuracy. That means that on average, the new method offers 1 223 × faster computation than the previous one.

b) Noise in Landmarks Detection: The influence of the inaccuracies in the detection of the landmarks was tested on the synthetic dataset (Section IV). Random values from Gaussian distribution were added to all detected 2D landmarks, which simulates the error in the localisation of 2D landmarks. The

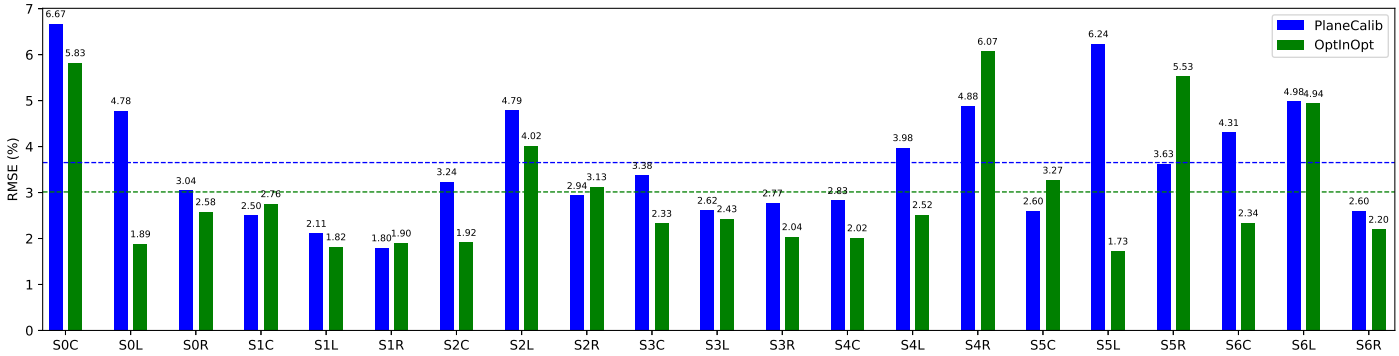

Fig. 8. Comparison of accuracy for the proposed *PlaneCalib* method and the *OptInOpt* method. Average results are as follows: *PlaneCalib* 3.65 %, *OptInOpt* 3.01 %. These results are measured when calibrating from all the input samples; further, a mechanism for selection of a more suitable subset is discussed.

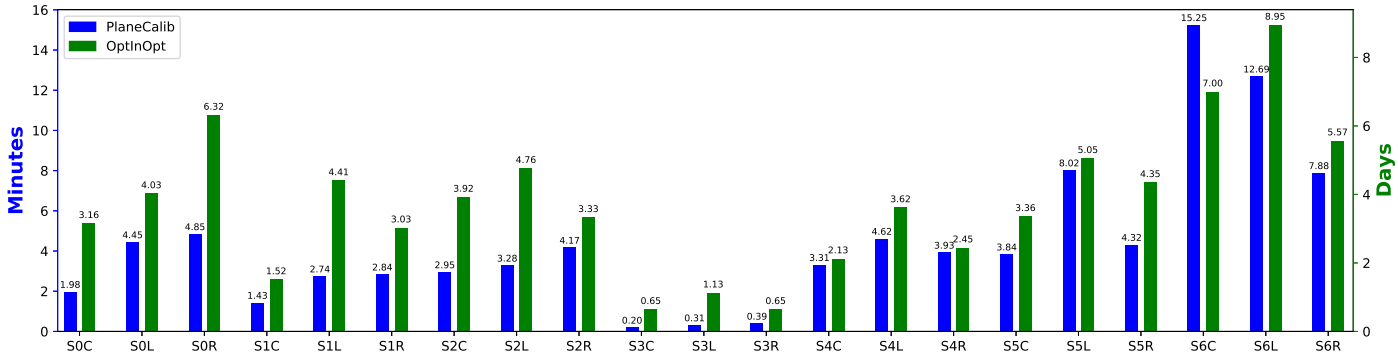

Fig. 9. Comparison of computation speed for the proposed *PlaneCalib* method and the *OptInOpt* method. Average results are as follows: *PlaneCalib* 4.44 *minutes*, *OptInOpt* 3.77 *days* — two y-axis scales are used.

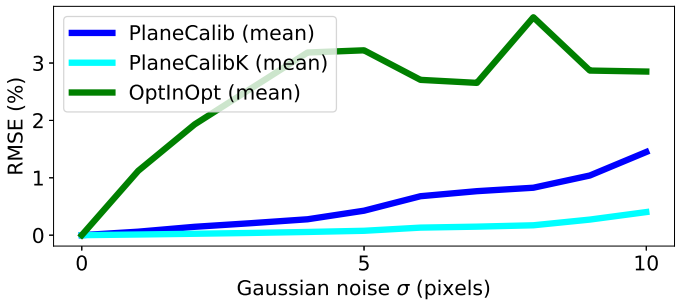

Fig. 10. Comparison of the calibration error by the proposed *PlaneCalib* method and the *OptInOpt* method on synthetic dataset w.r.t. varying noise levels in landmarks' 2D positions. Random Gaussian shift is added to all localised landmarks — noise in landmarks localisation.

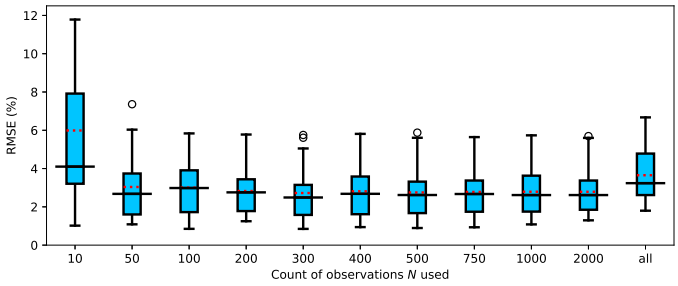

Fig. 11. Calibration error on all scenes from *BrnoCompSpeed* dataset with varying count of observations used.

results of calibration accuracy w.r.t. varying noise levels can be seen in Figure 10. Similarly to the evaluation on the real dataset, RMSE error (28) of the known distances in the scene was computed.

The *PlaneCalib* method uses focal length estimation (Section III-C), *PlaneCalibK* counts with a known intrinsic matrix $\mathbf{K}$. The *PlaneCalib* method seems to be more resistant to noise in the landmarks' positions than the *OptInOpt* method. The results presented are the mean values of 100 trials carried out using random scenes from the synthetic dataset.

c) Computation Speed-up and Improvement: To speed up and improve the calibration process, an approach which does not use all the observations was considered. As was mentioned in Section III-D, in each optimization step, the weights of individual observations are computed. To make the calibration process faster, after each re-computation of the weights, only N best observations (those with the highest weights) are used for further computation. The influence of value N selection can be seen in Figure 11. In cases with high N, some noisy observations can be present during calibration and these can affect the result. On the other hand, cases with low N can become problematic due to insufficient observation coverage of the ground plane. During our experiments, all observations in the scene were used for fair comparison with *OptInOpt*. But with the proper selection of value N, the resulting accuracy of 2.72 % error can be reached, which outperforms the previous method.

VI. Conclusion

This article presents a novel method for surveillance camera calibration. Although our experiments were performed on traffic surveillance cameras, the method itself can be, without any changes, applied to any static camera observing any scene in which arbitrary rigid objects are moving in a single plane. The method is based on localisation of landmarks on rigid objects and transformations between the object, camera, and world coordinate systems. In comparison with a state-of-the-art solution solving a similar task, our method reaches comparable results, with an extreme computation speed-up. Our algorithm was evaluated on a publicly available dataset *BrnoCompSpeed* and the error was reduced from $3.01\,\%$ to $2.72\,\%$ with a $1\,223\times$ computation speed-up.

Acknowledgments

This work was supported by TACR project "SMART-CarPark", TH03010529, as well as by the VRASSEO project (VI20172020068, Tools and methods for video and image processing to improve effectivity of rescue and security services operations) of the Czech Ministry of the Interior.

References

[1] Z. Zhang, "A flexible new technique for camera calibration," *TPAMI*, 2000.

[2] X. Meng and Z. Hu, "A new easy camera calibration technique based on circular points," 2003.

[3] Z. Zhang, "Camera calibration with one-dimensional objects," *TPAMI*, 2004.

[4] T. N. Schoepflin and D. J. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *TITS*, 2003.

[5] K.-T. Song and J.-C. Tai, "Dynamic calibration of Pan–Tilt–Zoom cameras for traffic monitoring," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2006.

[6] L. Kneip, H. Li, and Y. Seo, "Upnp: An optimal o(n) solution to the absolute pose problem with universal applicability," in *ECCV*, 2014.

[7] V. Lepetit, F. Moreno-Noguer, and P. Fua, "Epnp: An accurate o(n) solution to the pnp problem," *IJCV*, 2008.

[8] Y. Zheng, Y. Kuang, S. Sugimoto, K. Åström, and M. Okutomi, "Revisiting the pnp problem: A fast, general and optimal solution," in *ICCV*, 2013.

[9] A. Penate-Sanchez, J. Andrade-Cetto, and F. Moreno-Noguer, "Exhaustive linearization for robust camera pose and focal length estimation," *TPAMI*, 2013.

[10] Y. Zheng and L. Kneip, "A direct least-squares solution to the PnP problem with unknown focal length," in *CVPR*, 2016.

[11] Y. Zheng, S. Sugimoto, I. Sato, and M. Okutomi, "A general and simple method for camera pose and focal length determination," in *CVPR*, 2014.

[12] D. C. Luvizon, B. T. Nassu, and R. Minetto, "Vehicle speed estimation by license plate detection and tracking," in *ICASSP*, 2014.

[13] C. Maduro, K. Batista, P. Peixoto, and J. Batista, "Estimation of vehicle velocity and traffic intensity using rectified images," in *IEEE-ICIP*, 2008.

[14] F. Cathey and D. J. Dailey, "A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras," in *Intelligent Vehicles Symposium*, 2005.

[15] V.-H. Do, L.-H. Nghiem, N. P. Thi, and N. P. Ngoc, "A simple camera calibration method for vehicle velocity estimation," in *ECTI-CON*, 2015.

[16] L. Grammatikopoulos, G. Karras, and E. Petsa, "Automatic estimation of vehicle speed from uncalibrated video sequences," in *International Symposium on Modern Technologies, Educationand Profeesional Practice in Geodesy and Related Fields*, 2005.

[17] X. C. He and N. H. C. Yung, "A novel algorithm for estimating vehicle speed from two consecutive images," in *WACV*, 2007.

[18] M. Dubská, J. Sochor, and A. Herout, "Automatic camera calibration for traffic understanding," in *BMVC*, 2014.

[19] J. Sochor, R. Juránek, and A. Herout, "Traffic surveillance camera calibration by 3D model bounding box alignment for accurate vehicle speed measurement," *CVIU*, 2017.

[20] R. Bhardwaj, G. K. Tummala, G. Ramalingam, R. Ramjee, and P. Sinha, "AutoCalib: Automatic traffic camera calibration at scale," in *BuildSys*, 2017.

[21] P. Filipiak, B. Golenko, and C. Dolega, "NSGA-II based auto-calibration of automatic number plate recognition camera for vehicle speed measurement," in *EvoApplications*, 2016, pp. 803–818.

[22] J. Lan, J. Li, G. Hu, B. Ran, and L. Wang, "Vehicle speed measurement based on gray constraint optical flow algorithm," *Optik*, 2014.

[23] D. F. Llorca, C. Salinas, M. Jimenez, I. Parra, A. G. Morcillo, R. Izquierdo, J. Lorenzo, and M. A. Sotelo, "Two-camera based accurate vehicle speed measurement using average speed at a fixed point," in *ITSC*, 2016.

[24] M. Dubská and A. Herout, "Real projective plane mapping for detection of orthogonal vanishing points," in *BMVC*, 2013.

[25] V. Bartl and A. Herout, "Optinopt: Dual optimization for automatic camera calibration by multi-target observations," in *AVSS*, 2019.

[26] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *CoRR*, 2014.

[27] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[28] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *CVPR*, 2016.

[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[30] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[31] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *NIPS*, 2015.

[32] M. Biglari, A. Soleimani, and H. Hassanpour, "A cascaded part-based system for fine-grained vehicle classification," *TITS*, 2018.

[33] X. Li, L. Yu, D. Chang, Z. Ma, and J. Cao, "Dual cross-entropy loss for small-sample fine-grained vehicle classification," *TVT*, 2019.

[34] H. Liu, Y. Tian, Y. Yang, L. Pang, and T. Huang, "Deep relative distance learning: Tell the difference between similar vehicles," in *CVPR*, 2016.

[35] Z. Ma, D. Chang, J. Xie, Y. Ding, S. Wen, X. Li, Z. Si, and J. Guo, "Fine-grained vehicle classification with channel max pooling modified cnns," *TVT*, 2019.

[36] J. Sochor, A. Herout, and J. Havel, "BoxCars: 3D boxes as CNN input for improved fine-grained vehicle recognition," in *CVPR*, 2016.

[37] J. Sochor, J. Špaňhel, and A. Herout, "Boxcars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance," *TITS*, 2019.

[38] G. Pearce and N. Pears, "Automatic make and model recognition from frontal images of cars," in *AVSS*, 2011.

[39] B. Zhang, "Classification and identification of vehicle type and make by cortex-like image descriptor HMAX," *IJCVR*, 2014.

[40] A. Newell, K. Yang, and J. Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*, 2016.

[41] Z. Wang, L. Tang, X. Liu, Z. Yao, S. Yi, J. Shao, J. Yan, S. Wang, H. Li, and X. Wang, "Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification," in *ICCV*, 2017.

[42] J. Sochor, R. Juránek, J. Špaňhel, L. Maršík, A. Široký, A. Herout, and P. Zemčík, "Comprehensive data set for automatic single camera visual speed measurement," *TITS*, 2018.

[43] R. Juránek, A. Herout, M. Dubská, and P. Zemčík, "Real-time pose estimation piggybacked on object detection," in *ICCV*, 2015.