Fully Automatic Roadside Camera Calibration for Traffic Surveillance

Markéta Dubská, Adam Herout, Roman Juránek, and Jakub Sochor

***Abstract*—This paper deals with automatic calibration of roadside surveillance cameras. We focus on parameters necessary for measurements in traffic-surveillance applications. Contrary to the existing solutions, our approach requires no *a priori* knowledge, and it works with a very wide variety of road settings (number of lanes, occlusion, quality of ground marking), as well as with practically unlimited viewing angles. The main contribution is that our solution works fully automatically—without any per-camera or per-video manual settings or input whatsoever—and it is computationally inexpensive. Our approach uses tracking of local feature points and analyzes the trajectories in a manner based on cascaded Hough transform and parallel coordinates. An important assumption for the vehicle movement is that at least a part of the vehicle motion is approximately straight—we discuss the impact of this assumption on the applicability of our approach and show experimentally that this assumption does not limit the usability of our approach severely. We efficiently and robustly detect vanishing points, which define the ground plane and vehicle movement, except for the scene scale. Our algorithm also computes parameters for radial distortion compensation. Experiments show that the obtained camera parameters allow for measurements of relative lengths (and potentially speed) with $\sim$2% mean accuracy. The processing is performed easily in real time, and typically, a 2-min-long video is sufficient for stable calibration.**

***Index Terms*—Camera calibration, camera distortion correction, camera surveillance, diamond space, ground plane estimation, Hough transform, orthogonal vanishing points, PClines, speed estimation, vanishing points (VPs).**

I. Introduction

THE number of Internet-connected cameras is quickly increasing, and a notable amount of them are used in traffic monitoring. At the moment, many of these are used primarily for simply transferring the image to a human operator, and they lack automatic processing. This is mainly because machine-vision-based data mining algorithms require manual configuration and maintenance, involving a considerable effort of skilled personnel and, in many cases, measurements and actions taken in the actual real-life scene [1]–[6]. The goal of our research is to provide fully automatic (no manual input whatsoever) traffic processing algorithms—leading toward vehicle classification and counting, speed measurement, congestion detection, etc. Different applications can be fostered by compensation of local projection [7].

In this paper, we are dealing with the problem of fully automatic camera calibration in a common traffic-monitoring scenario. We automatically determine radial distortion compensation parameters and solve the calibration of internal camera parameters and external parameters (camera orientation and position up to scale with respect to the dominant motion of the vehicles). The solution is fully automatic in the sense that there are no inputs or configuration parameters specific to a particular traffic setting, camera type, etc. In addition, we are not assuming any appearance model of the vehicles, which would differ for different countries, time periods, and the kind of traffic on the particular road. Moreover, we assume no *a priori* knowledge about the road itself (number of lanes, appearance of the marking, presence of specific features, etc.). The only assumption is approximately straight shape of the road. Experiments show that normal curves on highways are still "straight enough" to meet our assumption. Sharp corners cannot be directly handled by our approach and will be subject of further study.

Because of the lack of information that can be obtained from road data, the majority of methods assume the principal point to be at the center of the image [8], [9]. A popular assumption also is a horizontal horizon line, i.e., zero roll of the camera [8], [10], [11]—which turns out to be severely limiting, because it is difficult to find an actual roadside camera meeting this assumption accurately enough.

Some works require user input in the form of annotation of the lane marking with known lane width [2], [6] or marking dimensions [3], camera position [2], [12], and average vehicle size [13] or average vehicle speed [8]. A common feature of virtually all methods is detection of the vanishing point (VP) corresponding to the direction of moving vehicles. A popular approach to obtaining this VP is to use road lines [5], [9], [14] or lanes [9], [11], more or less automatically extracted from the image. These approaches typically require a high number of traffic lanes and a consistent and well-visible lane marking.

Another class of methods disregards the line marking on the road (because of its instability and impossible automatic detection) and observes the motion of the vehicles, assuming straight and parallel trajectories in a dominant part of the view. Schoepflin and Dailey [8] constructed an activity map of the road with multiple lanes and segmented out individual lanes. Again, this approach relies on observing a high number of traffic lanes—high-capacity motorways and similar settings. Other researchers detect vehicles by a boosted detector and observe

Manuscript received January 28, 2014; revised June 11, 2014; accepted August 19, 2014. Date of publication September 24, 2014; date of current version May 29, 2015. This work was supported by TACR project V3C, TE01020415, and by the IT4Innovations Centre of Excellence, ED1.1.00/02.0070. The Associate Editor for this paper was K. Wang.

M. Dubská, R. Juránek, and J. Sochor are with the Graph@FIT, Brno University of Technology, 612 66 Brno, Czech Republic.

A. Herout is with the Graph@FIT, Brno University of Technology, 612 66 Brno, Czech Republic, and also with Angelcam, Inc., Minden, NV 89423-8607 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2014.2352854

their movement [15] or analyze edges present on the vehicles [16]. Beymer *et al.* [4] accumulated tracked feature points, but also required the user to provide defined lines by manual input. Kanhere and Birchfield [17] took a specific approach for cameras placed low above the street level. Once the calibration and scene scale is available, the road plane can be rectified, and various applications such as speed measurement can be done in a straightforward manner [3], [6], [10], [12], [18].

We assume that the shape of the road is (approximately) straight, and therefore, the majority of vehicles move in straight mutually parallel trajectories. Our experiments verify that our approach is tolerant to a high rate of outliers from this assumption. We are using a variant of the Hough transform, which is very tolerant to random outliers, and these do not affect the found global maximum. Due to this, vehicles changing lanes, passing one another, or avoiding a small obstacle do not harm the algorithm, and our method is easily and reliably applicable on a vast majority of real traffic-surveillance videos (of approximately straight roads). The calibration of internal and external parameters of the camera is achieved by first computing three orthogonal VPs [19]–[21], which define the vehicle motion.

Our calibration is provided up to scale, which is generally impossible to determine (from just an image, one can never tell a matchbox model from real vehicles). The scale can be provided manually [2], [12] or recognized by detecting known objects [13].

We harness our finite and linear parameterization of the real projective plane published recently [22]. In this previous work, we streamlined the cascaded Hough Transform by stacking two stages and skipping one intermediate accumulation step. This approach allows for detection of VPs (and triplets of orthogonal VPs) from noisy linelet data. These input data can be coming online and be accumulated to a fixed-size accumulation space—which is suitable in online video processing. Instead of using road marking or lane border edges [3], [5], [8], we accumulate fractions of trajectories of detected feature points on moving objects and relevant edges.

Contrary to previous works, our approach provides the camera calibration fully automatically for very versatile camera views and road contexts (coming and going vehicles, from the side, from the top, small roads and highways, close-up and fisheye lenses, and so on). The method can be applied on virtually any roadside camera without any user input—the experiments presented in this paper were done without any per-camera or per-video settings.

We collected a set of video recordings of traffic on roads of different scales, with various cameras, in different weather conditions. The MATLAB source codes and the video data set are publicly available for comparison and future work.[1]

II. Automatic Roadside Camera Calibration

Let us consider a road scene with a single principal direction of the vehicle movement. The position of the ground plane and the direction of the vehicle movement relative to the camera can be defined and described by three VPs [19], [20].

[1]http://medusa.fit.vutbr.cz/pclines

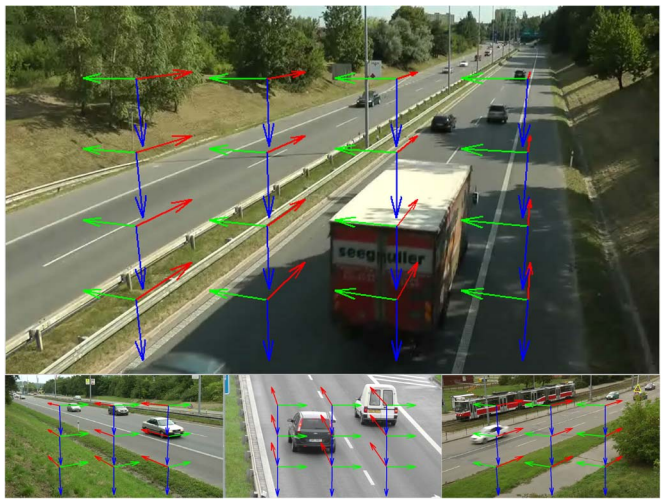

Fig. 1. We propose a fully automatic approach to camera calibration by recognizing the three dominant VPs, which characterize the vehicles and their motion. (Top) Three recognized VPs. (Bottom) Various scenes in which the algorithm automatically calibrates the camera.

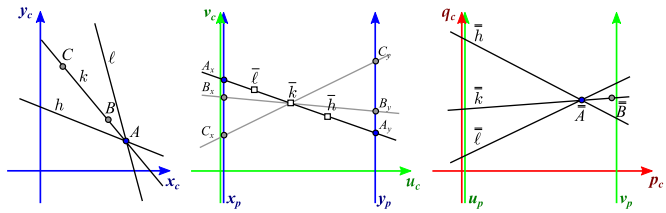

Fig. 2. Two cascaded PClines transformations. (Left) Original image space with points and lines. (Middle) Same objects in parallel coordinates. A new Cartesian coordinate system is defined (green, $u_c; v_c$). (Right) Second transformation to parallel coordinates $u_p; v_p$.

Fig. 1 shows the VPs and the color notation and VP visualization to be used throughout this paper: *red*: *first* VP—in the direction of the car motion; *green*: *second* VP—in the ground plane, perpendicular to the vehicle motion; *blue*: *third* VP—perpendicular to the ground plane. We find it intuitive to use dense markers pointing toward the three VPs. The positions of the markers are irrelevant—they are distributed in a regular grid.

We assume cameras without skew and with equal scale in horizontal and vertical directions (aspect ratio $= 1$, square pixels). From our experience, these assumptions are perfectly met for all practically used surveillance cameras. In addition, following most previous works in this field [8]–[11], we assume the camera's principal point to be at the image center. This assumption is met approximately, not exactly (contrary to the previous ones). However, for the target applications of our algorithms—distance/speed measurement, reprojection of observed objects into the 3-D space, etc.—the error caused by this assumption is tolerable (see measurements in Section III).

Although we can afford to consider this much simplified camera model, radial distortion—usually perceived as a more advanced camera parameter—cannot be omitted for some cameras. Practically used cameras, even the expensive and sophisticated ones, tend to exhibit a high degree of radial distortion (see Fig. 12 for sample images). Radial distortion compensation is therefore dealt with in Section II-E.

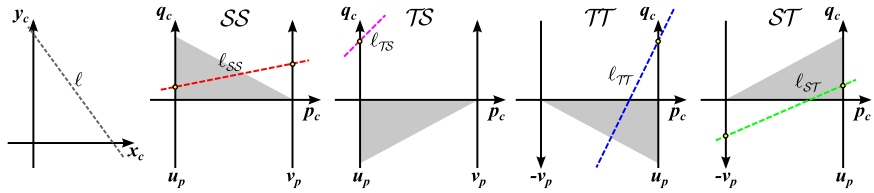

Fig. 3. Representation of a line in the different transformations. Only darker triangular subspaces are parts of the final diamond space; each one is an image of one quadrant from the original projective plane.

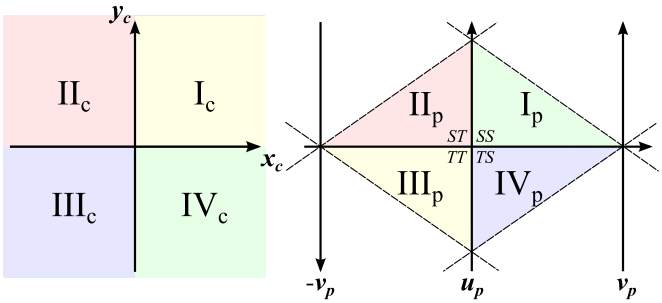

Fig. 4. Correspondence between quadrants in Cartesian coordinates and triangle subspaces of the space of parallel coordinates. (Left) Quadrants of the original infinite Cartesian space. (Right) Quadrants of the parallel spaces attached to each other.

A. Diamond Space: Detection of VPs by Hough Transform

The detection of VPs is done by using the *diamond space* accumulation scheme based on the Hough transform [22]. In order to make this paper self-contained, we describe this detection scheme here; for details, please refer to the original article [22] or the Ph.D. thesis by Dubská [23].

This method maps the whole 2-D projective plane into a finite space referred to as the diamond space by a piecewise linear mapping of lines using parallel coordinates—a coordinate system, where the axes are mutually parallel. In parallel coordinates, a 2-D point $[x, y]$ is represented by a line intersecting the parallel coordinate axis in values x or y. Fig. 2 shows the cascaded transformation of the points and lines—each point is transformed to a line and then again to a point (similarly for line–point–line transformation).

In the situation presented in Fig. 2, both systems of parallel coordinates have their axes oriented toward the same direction. Transformation to the parallel coordinates with this property is denoted by $\mathcal{S}$ (straight). If the axes in parallel coordinate system are oriented in different directions, the transformation is denoted by $\mathcal{T}$ (twisted) [24]. As a result, four different composite mappings can be constructed: $\mathcal{S}\circ\mathcal{S}, \mathcal{T}\circ\mathcal{S}, \mathcal{T}\circ\mathcal{T}, \mathcal{S}\circ\mathcal{T}$. Fig. 3 shows a line and its image in these four different transformations.

Each of the aforementioned mappings is a transformation of one infinite space to another. However, application of the mappings, each on one quadrant of the real projective plane, leads to a finite domain of the transformation. The dark triangular parts in Fig. 3 are images of IV, III, I, and II quadrant, respectively (see Fig. 4).

These four finite triangular subspaces can be attached together and form a diamond space (see Fig. 4). A line with homogeneous coordinates (a, b, c) from 2-D projective plane is represented by a polyline in the diamond space. The number of its segments corresponds to the number of quadrants the line originally passes. The endpoints of the polyline can be in general defined by

$$\begin{aligned} &\alpha = \operatorname{sgn}(ab) \quad \beta = \operatorname{sgn}(bc) \quad \gamma = \operatorname{sgn}(ac) \\ &(a,b,c) \rightarrow \quad \left[\frac{\alpha a}{c+\gamma a}, \frac{-\alpha c}{c+\gamma a}\right], \left[\frac{b}{c+\beta b}, 0\right], \\ &\left[0, \frac{b}{a+\alpha b}\right], \left[\frac{-\alpha a}{c+\gamma a}, \frac{\alpha c}{c+\gamma a}\right] \end{aligned} \tag{1}$$

where $\operatorname{sgn}(x)$ is the nonzero signum function. When a line passes through just two quadrants (vertical lines, horizontal lines, lines through the origin), one segment always degenerates to a point.

Such a transformation maps both the ideal and regular points to regular points and is used as a parameterization in the Hough transform scheme: lines in the image domain are transformed to the diamond space, and the resulting polyline is accumulated there—each pixel of the polyline is incremented. The diamond space is then searched for a global maximum corresponding to the VP—the point where majority of lines pass through. The point with homogeneous coordinates $[p, q, 1]$ from the diamond space is backprojected to the original 2-D projective plane, i.e.,

$$[p, q, 1] \rightarrow [q, \operatorname{sgn}(p)p + \operatorname{sgn}(q)q - 1, p]\,. \tag{2}$$

In this paper, we collect evidence accumulated to the diamond space over time—more observed cars mean more evidence and better signal-to-noise ratio. Contrary to the original use of the diamond space (detection of VPs of edges in single images [22]), in this work, we can accumulate as much evidence as necessary for a sufficiently stable VP. This convergence in time is measured and discussed in Section III-B (see Fig. 14).

B. First VP Extraction

The VP of direction parallel to the movement of the vehicles is considered to be the first VP. For its detection, we propose to use the Hough transform based on the parallel coordinates described in Section II-A.

In each video frame, feature points are detected (minimum eigenvalue algorithm [25] is used in the experiments) and tracked by Kanade–Lucas–Tomasi tracker [26] in the subsequent frame. Successfully detected and tracked points exhibiting a significant movement are treated as fragments of vehicle trajectories. These fragments of trajectories are extended to infinite lines, assuming that they pass through the first VP. All these lines vote in the diamond space accumulator. The most voted point is considered to be the first VP.

Fig. 5. Illustration of tracked points. Points marked by green exhibit a significant movement, and they are accumulated. Points marked by yellow are stable points and do not vote. The accumulated diamond space is in the top left corner.

Fig. 6. Tracked points and their classification based on the first VP position. Points marked by red/green move toward/from the first VP. Points marked by yellow are moving elsewhere.

It should be noted that the whole process is not very sensitive to the selection of the feature detector and tracker. The tracking needs to be robust/accurate to some degree (acceptable *precision*) so that the accumulator space is not polluted with noise. However, the Hough transform in this case proves to be fairly resistant to noise; thus, the algorithm does not need to be perfect. At the same time, the algorithms do not need to be very sensitive (no high *recall* is required), because if the number of found/tracked features is low, a longer period of observation is needed in order to accumulate a sufficient amount of evidence about the VPs, but the algorithm is not harmed. In general, better feature detection/tracking helps, but basically any existing fast method will suffice, because the algorithm is not very sensitive to this selection.

The diamond space turns out to be robust to noise, and it provides reliable estimates of the most distinctive VP in the frame. Experiments show that, in most cases, observing only two or three vehicles suffices to find a good estimate of the first VP. Later, with more observation and accumulation, the first VP is very stable and accurate—we have not seen a single video where the first VP was not established correctly or was unstable. Vehicles that violate the assumption of the straight motion do not seem to be able to influence the proper solution by adding their noisy motion. One advantage of the Hough transform is that the found global maximum is in no way influenced by the outliers, unless the outliers are too numerous and too unified so that the global maximum is found elsewhere. This does not happen in traffic scenes, where vehicles are passing or changing lanes more or less randomly and their overall dominant motion is very clear.

Fig. 5 shows the tracked points accumulated to the diamond space. Once the first VP is determined, moving points can be discerned whether they move toward the VP or from it or whether they are moving in a completely different direction (see Fig. 6).

C. Second VP

The second VP corresponds to the direction parallel to the road (or the ground plane) and perpendicular to the first direction. Again, the diamond space (see Section II-A and [22]) is used for its detection. Many edges on the vehicles coincide with the second VP, and thus, we let them vote in the accumulation space.

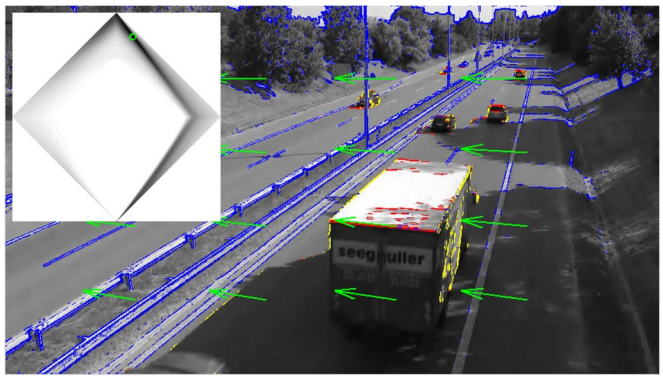

Fig. 7. Accumulation of the second VP. Blue edges belong to the background (see Fig. 8). Yellow edges are omitted from voting because of their vertical direction or direction toward the first VP. Red edges are accumulated to the diamond space (in the corner; green circle marks the maximum).

We use an edge background model in order to select only edges on moving objects—probable vehicles. The model is updated by each frame to deal with shadows and other slow changes. The edge background model stores for each pixel the confidence score of occurrence of an oriented edge. For each pixel (i, j) in an incoming frame I_t of size $w \times h$, its gradient magnitude m and orientation is obtained by using responses of horizontal and vertical difference filters, and the orientation is discretized to eight bins. Image H_t of size $w \times h \times 8$, representing the gradients, is created. Each plane corresponds to one orientation bin, where $H_t(i, j, k)$ is set to the magnitude m if the discretized orientation of pixel (i, j) is k; otherwise, it is zero. At the beginning, we initialize the background model as $B_1 = H_1$. Model B_t is updated by moving average, i.e.,

$$B_t = \alpha B_{t-1} + (1 - \alpha) H_t \tag{3}$$

where α is a smoothing coefficient close to 1 ($\alpha = 0.95$ is used in the experiments).

For every edge pixel (i.e., pixel with gradient magnitude higher than some low threshold τ_1) of an incoming frame I_t, a background test is performed. An edge passes the test if the difference between its magnitude and the value in the corresponding bin in the model B_t is higher than a threshold τ_2 (see Fig. 8).

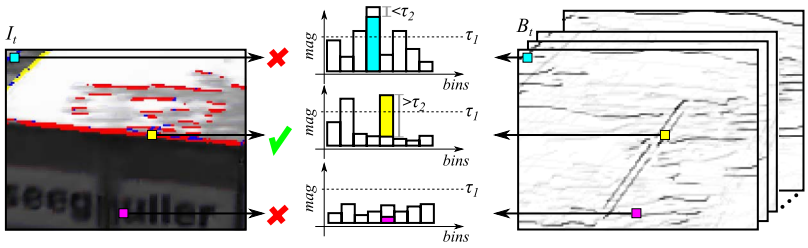

Fig. 8. Model of background edges. Significant edges are further processed (see Fig. 7) only if recognized as foreground (*yellow*). Background edges (with difference $> \tau_2$, *cyan*) and insignificant edges (with magnitude $< \tau_1$, *magenta*) are omitted.

Edges passing the background test are further processed and filtered. Edges (extended to infinite lines) passing close to the first VP, known from the previous step, are excluded from further processing. Edges with approximately vertical direction are also excluded, based on the assumption of the scene horizon being approximately horizontal (with a high tolerance $\pm 45^\circ$). This condition can be disregarded when the first VP is detected to be close to infinity horizontally. In such a case, the edges supporting the second VP are allowed to have vertical direction. However, these cases are not frequent, because traffic-surveillance cameras are rarely observing the road exactly from profile. Edges passing this filtering stage vote in the diamond space in order to find the second VP. Fig. 7 shows the edge background model and omitted and accumulated edges together with the diamond space. Fig. 8 shows the model of the background edges.

D. Third VP, Principal Point, and Focal Length

The third VP corresponds to the direction perpendicular to the ground plane. Unfortunately, in majority of the roadside views, there seems to be minimal amount of edges supporting the third VP. Some works [14], [16] derive this VP from standing pedestrians in the scene; however, the majority of scenes we are dealing with do not contain a sufficient number of pedestrians by far (much of them do not contain pedestrians at all because access to motorways, overpasses, tunnels, etc., is restricted). Instead of finding the third VP, we calculate its position by using the first two VPs (U and V) and the assumption that the principal point P is in the middle of the image. Two VPs and the position of the principal point are sufficient for computing focal length f, i.e.,

$$\begin{aligned} &U = (u_x, u_y) \quad V = (v_x, v_y) \quad P = (p_x, p_y) \\ &f = \sqrt{-(U - P) \cdot (V - P)}. \end{aligned} \tag{4}$$

With a known f, the third VP, denoted by W, can be calculated as

$$\begin{aligned} U' &= (u_x, u_y, f) \quad V' = (u_x, u_y, f) \\ P' &= (p_x, p_y, 0) \\ W' &= (U' - P') \times (V' - P') \end{aligned} \tag{5}$$

where U', V', W', P' stand for the world coordinates of the points, and U, V, P stand for the coordinates in the image plane.

When one of the first two VPs is in infinity, the focal length f and the third VP cannot be calculated. However, for some applications, e.g., the distance/speed measurement, knowing only the first two VPs is enough. In these cases, the VPs in infinity are handled gracefully due to the use of homogeneous coordinates and because the diamond space used for their detection represents the whole projective plane, including the ideal points (points in infinity).

E. Radial Distortion

The previous sections discussed the core contribution of our paper—calibration of the road scene. This section extends the contribution by one more step, which is optional. In practice, some real-life cameras exhibit a large degree of radial distortion; however, some cameras are practically free from it—depending on the used optics. Depending on the particular camera, application of radial distortion compensation might not be necessary. That is why we have not discussed this issue until now, although this phase of radial distortion compensation precedes the calibration in a practical implementation. It should be noted that, apart from radial distortion, practically observed cameras are equipped with reasonable optics. Therefore, the assumption of the principal point being at the image center (or close to it), absence of skew, and equal scale in the x- and y-directions ("square pixels") are kept, and the presented algorithm is general.

Provided the assumption of the road being straight, the tracked trajectories can be used to compensate for the camera's radial distortion [27]. It should be noted that the requirement for straight road is much less strict in the case of the calibration itself (refer to Fig. 11). The corrected position $(\overline{x}, \overline{y})$ of input point (x, y) can be modeled by the polynomial radial distortion model [28] defined by

$$\begin{aligned} \overline{x} &= (x - x_c)(1 + k_1 r^2 + k_2 r^4 + \cdots) \\ \overline{y} &= (y - y_c)(1 + k_1 r^2 + k_2 r^4 + \cdots) \\ r &= \sqrt{(x - x_c)^2 + (y - y_c)^2} \end{aligned} \tag{6}$$

where (x_c, y_c) define the position of the distortion center, and coefficients k_i are unknown distortion parameters.

In order to find $K = \{k_1, k_2, \ldots\}$, the extracted trajectories T are used. Each point is tracked until the tracker is lost. Each trajectory represented by a sequence of points $\tau = \{a_1, a_2, \ldots\} \in T$ is projected by (6) to their transformed versions $\overline{\tau}_K$. Optimal parameters K^* are found by minimization of the sum of square differences of all points in all trajectories to their best fitting lines $\ell_{\overline{\tau}_K}$, i.e.,

$$K^* = \arg\min_K \sum_{\overline{\tau} \in T} \sum_{\overline{a} \in \overline{\tau}} (\ell_{\overline{\tau}_K} \cdot \overline{a})^2. \tag{7}$$

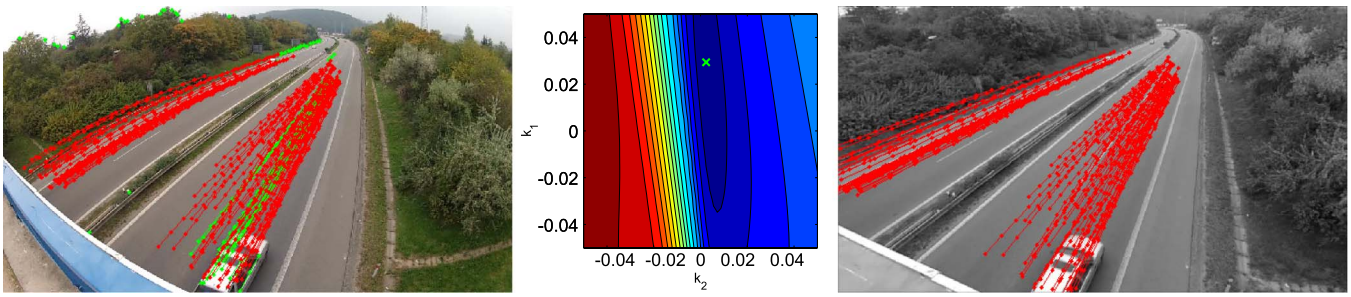

Fig. 9. Radial distortion compensation. Although for the calibration itself only a part of the road has to be approximately straight, for radial distortion compensation, the road has to be straight along a major part of the vehicle movement. In this example, trajectories end before the turn because motion of the cars is small there and the tracker got naturally lost. (Left) Original image with trajectories. (Middle) Parameter space with value calculated from trajectories using (7). Green cross stands for the optimal parameter combination found by the evolution algorithm. The color gradient shows the error for each combination of the parameters k_1, k_2. (Right) Undistorted image using the optimal coefficients.

We use $(1+\lambda)$-ES evolutionary strategy (with $\lambda = 8$) [29] to search for the first two coefficients. The optimization was done online. When new trajectories were tracked, one iteration of the optimization was executed. The whole radial distortion compensation process is shown in Fig. 9.

In practice, the radial distortion compensation is computed first. Then, accumulation of the VPs is performed on the undistorted tracked features, i.e., the tracked features and edges are transformed by (6), and the algorithms in Sections II-B and C are working on $(\bar{x}, \bar{y})$ pairs. Once the radial distortion is compensated for, accumulation of the two VPs can happen simultaneously.

F. Camera Calibration From the VPs

The problem of obtaining camera projection matrix $\mathbf{P}$ from detected VPs has been already discussed several times. Therefore, here, we provide only a brief overview; more information can be found elsewhere [11], [14], [16]. Projection matrix $\mathbf{P}$ transforms a world point $[x_w, y_w, z_w, 1]'$ into point $[x_p, y_p, 1]'$ in the image plane, i.e.,

$$\lambda_p [x_p, y_p, 1]' = \mathbf{P}[x_w, y_w, z_w, 1]'. \tag{8}$$

Projection matrix $\mathbf{P}$ can be decomposed into three matrices, i.e., one with intrinsic parameters of the camera $\mathbf{K}$ and two with extrinsic parameters, namely, rotation matrix $\mathbf{R}$ and translation vector $\mathbf{T}$, i.e.,

$$\mathbf{P} = \mathbf{K}[\mathbf{R}\ \mathbf{T}]. \tag{9}$$

With the assumption of zero skew, location of the principal point at the center of image plane, and unit aspect ratio, the only parameter to be found in matrix $\mathbf{K}$ is the focal length f derived before (4). The rotation matrix $\mathbf{R}$ is fully defined by the positions of the three orthogonal VPs. For the translation vector $\mathbf{T}$, additional information has to be known; it can be derived from the height of the camera from the ground—as discussed by Zhang *et al.* [16]—or a known distance of two projected points.

III. Experimental Results

We evaluated our approach on five groups of videos, each containing five to eight videos. Videos in a common group share the same camera intrinsic parameters (no changes of camera settings were done between shots) but have different extrinsic parameters and capture different scenes or scenes from a different view. Examples of real-life videos are presented in Figs. 10–13.

First, we logically intended to evaluate the calibration accuracy by comparing the obtained camera parameters with calibration parameters obtained by checkerboard-based calibration [30]. In many cases, the focal length of the cameras was high (cameras zoomed in), and the checkerboard calibration was inaccurate itself (different individual calibrations ended in dramatically different results). In cases of some cameras, it could have been caused by the camera automatically refocusing to the calibration pattern (close to the cameras) and back to the traffic scene. That is why we had to select a different approach to evaluation—based on the intended application tasks: distance/speed measurement. The same approach has been already used in the literature [16], which allows for fair comparison.

In order to evaluate the accuracy of the VPs detection, we compute the precision of length measurements in videos similarly to Zhang *et al.* [16]. From each video, 15–25 pairs of feature points are tracked in 21 subsequent frames. These points are projected with the matrix obtained from the VPs, and we evaluate the stability of their distance d. Error of the ith pair in the jth sequence is calculated as

$$e_{ji} = \left| 1 - \frac{d_{ij}}{\overline{d}_j} \right| \tag{10}$$

where $\overline{d}_j$ is the mean distance in the jth sequence. For each video, two errors are computed from e_{ji}—the worst error, i.e., e_w^v, and the mean error, i.e., e_m^v. The same is computed for each group of videos (e_w^g, e_m^g).

Table I shows the worst and mean errors for the groups and the computed focal lengths. The focal length f is taken from the video with the lowest e_m^v in the group. We mention it here in order to illustrate the differences in the camera settings. Larger f leads to smaller length-measurement error due to smaller perspective distortion and consequent smaller dependence on the point tracker accuracy.

As a particular example, Table II shows the mean and worst errors and the computed focal lengths for all videos from group g1 (one video from the group is shown in Fig. 5).

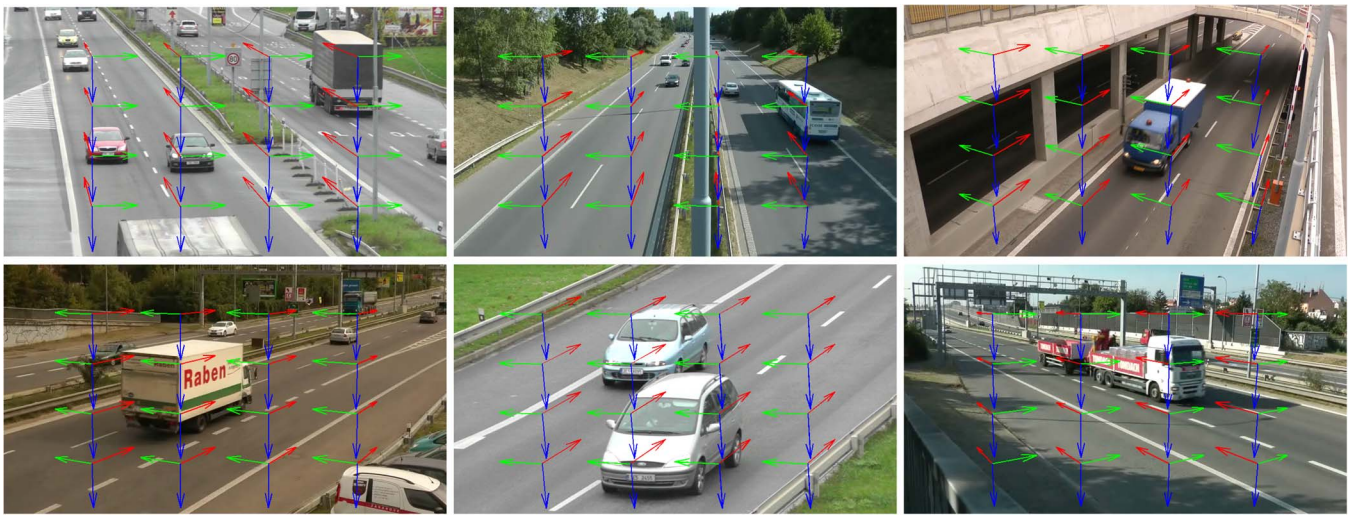

Fig. 10. Examples of real-life videos. Automatic detection of three VPs. Presented here is a small selection of traffic scenes; more samples can be found in the supplementary video. Three VPs are robustly found regardless of camera f (zoom), shadows, lighting conditions, and camera position with respect to the road (for reasonable traffic-surveillance scenarios).

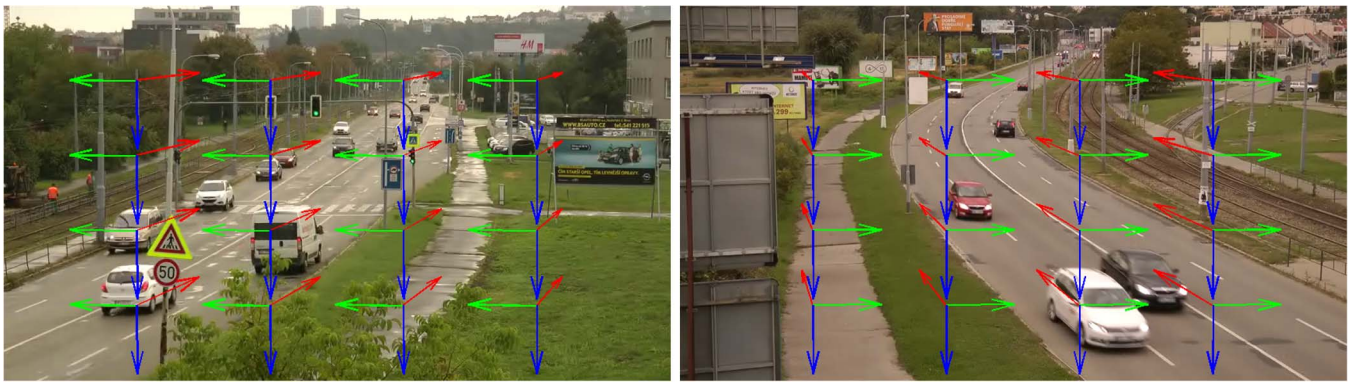

Fig. 11. Examples of real-life videos. Successful camera calibration/orientation in scenes with bent roads. These are to illustrate that the assumption of *straight* vehicle motion is not very strict. However, radial distortion would be more difficult to compensate in these scenes.

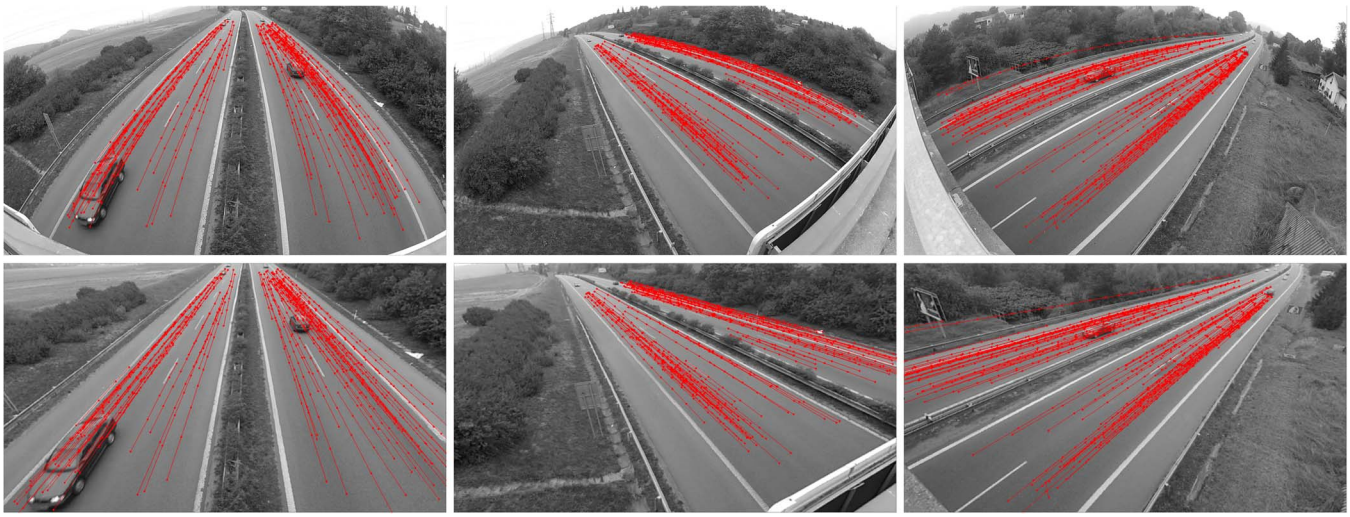

Fig. 12. Examples of real-life videos. Radial distortion estimation and compensation. (Top) Original videos (converted to grayscale). (Bottom) Processed videos with radial distortion compensated. (Red polylines) Tracks of features used in the computation.

The extracted focal length in Table II differs. Its error can be caused by some videos having the second VP near infinity and by the camera refocusing automatically (because the viewpoint changed). However, an error in estimation of camera f does not prevent the measurement (the desired traffic-surveillance application) to be precise enough in all cases. This is because

Fig. 13. Examples of real-life videos. Ghost images (composition of two images of different moments in time by blending) and measured equal distances. These lines are observations of identical object measures at different times (consecutive frames). These distances are used in the evaluation (see Tables I and II).

TABLE I
MEAN AND WORST LENGTH-MEASUREMENT ERRORS FOR GROUPS OF VIDEOS IN PERCENT AND THE COMPUTED FOCAL LENGTHS

group	g1	g2	g3	g4	g5
e_w^g (%)	6.5	1.8	10.1	5.3	4.0
e_m^g (%)	1.2	0.2	1.3	0.8	0.7
f	705.7	7163.7	674.6	769.6	2465.1

TABLE II
MEAN AND WORST LENGTH-MEASUREMENT ERRORS FOR INDIVIDUAL VIDEOS WITHIN GROUP G1 AND COMPUTED FOCAL LENGTHS

video	v1	v2	v3	v4	v5	v6
e_w^v (%)	5.5	6.0	5.2	4.7	5.3	6.4
e_m^v (%)	1.0	1.3	1.1	0.8	1.3	1.6
f	742.0	688.2	685.0	705.7	803.8	830.0

the VP near infinity does not increase the measurement error (although it spoils f). Zhang *et al.* [16] reported similar measurements (single scene, 28 point pairs, six sequences); their mean, worst, and second worst errors appear to be 6%, 19%, and 13%, respectively.

A. Evaluation of Processing Speed

Our algorithm is fairly efficient—capable of processing the video stream in real time. In order to demonstrate and evaluate this, we created an efficient implementation in C++ and processed the input videos used in the evaluation above. Table III shows the speed measurement results: *VP1*—first VP accumulation (see Section II-B) and *VP2*—second VP accumulation (see Section II-C). The times are averaged from all measured videos (approximately 3 min each) in two groups (*traffic intensity*), which differ slightly in their quantity of observed cars.

TABLE III
PROCESSING SPEED EVALUATION

resolution	traffic intensity	VP1 (*ms*)	VP2 (*ms*)
854 × 480	high	19	14
	medium	19	11
1920 × 1080	high	98	69
	medium	97	57

It should be noted that our algorithm processes only ∼5 frames per second (FPS) so that the movement of tracked points is measurable and stable. Therefore, the measured times in all cases [including the full high-definition (HD) video processing] allow for comfortable real-time processing. The measurements were done on a machine with Intel dual-core i5 1.8 GHz and 8-GB DDR3 random access memory, and the frame rates are reported for pure computation (video decompression, etc., are omitted).

Our C++ implementation of first VP detection uses only a limited number of tracked feature points; therefore, the frame rates are invariant to the traffic intensity. However, the frame rate of the second VP detection differs with respect to the traffic intensity. The variance is caused particularly by the necessity to handle edge points of passing cars and therefore accumulate more lines corresponding to the edge points into the diamond space [22]. The second VP detection frame rate also depends on the motion of other objects (people, moving trees, etc.) in the video stream.

B. Accumulation Convergence

The convergence of the accumulation of the first and second VPs is shown in Fig. 14. For the vast majority of the videos,

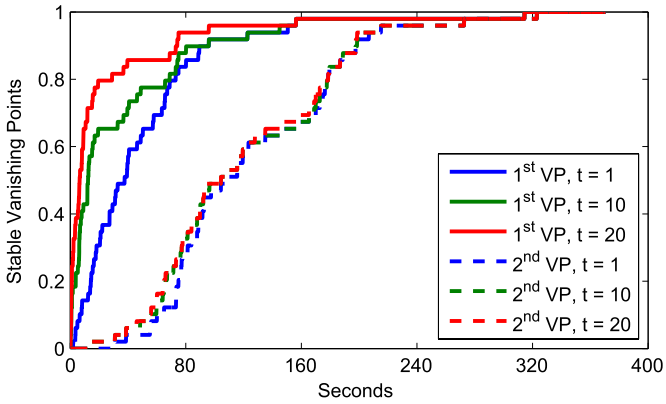

Fig. 14. Convergence of VP1 and VP2 computation. We computed the pixel distance between the VP detected at the given time and the final detection. For a threshold t, the time after which the distance is lower then t is evaluated. The graph shows the fraction of videos reaching the threshold condition within the given number of seconds. It should be noted that only every fourth frame in a 50-FPS video was processed for the same reason as in Table III.

the first VP remains totally stable since 160 s of 50-FPS video, and the second VP is stable after processing 250 s of video (Zhang *et al.* [16] mentioned processing of 2 h of recording).

IV. Conclusion

We have presented a method for fully automatic calibration of roadside cameras (up to scale). It requires no user input and assumes very little about the scene itself. We experimented with videos taken at roads of different classes (from small streets with little traffic to busy highways) by various cameras (handycam, GoPro, Internet Protocol cameras, high-end and inexpensive ones) and lenses (from close-up views to nearly fisheye). The results are stable, reliable, and usable by various applications without any per-video or per-camera settings. The efficient implementation is fast, and the concept is thus ready for real-time implementation on low-end hardware, even on full HD video sources.

The solution consists of a method for compensation of radial distortion and a way of obtaining three orthogonal VPs related to the motion of the vehicles. These three orthogonal VPs define intrinsic and extrinsic camera parameters. Virtually any roadside scene captured by a static camera can be fully automatically calibrated up to scale. Our method assumes approximately straight motion of the vehicles at least along a large portion of their motion. Bent roads and sharp corners followed/preceded by a straight stretch of the road are easily dealt with.

The main contribution and advantage is that we strictly avoid any real-life measurement in the scene and/or any manual input. Our algorithms open space for fully automatic processing of almost any traffic-surveillance video footage. We collected a set of evaluation videos (see supplementary video for examples) accompanied by ground truth calibration parameters. This data set, together with the MATLAB sources, is made available for comparison and future work.

References

[1] N. Kanhere and S. Birchfield, "A taxonomy and analysis of camera calibration methods for traffic monitoring applications," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 441–452, Jun. 2010.
[2] K. Wang, H. Huang, Y. Li, and F.-Y. Wang, "Research on lane-marking line based camera calibration," in *Proc. IEEE ICVES*, 2007, pp. 1–6.
[3] F. Cathey and D. Dailey, "A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras," in *Proc. IEEE Intell. Veh. Symp.*, 2005, pp. 777–782.
[4] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik, "A real-time computer vision system for measuring traffic parameters," in *Proc. IEEE Conf. CVPR*, 1997, pp. 495–501.
[5] R. Dong, B. Li, and Q.-M. Chen, "An automatic calibration method for PTZ camera in expressway monitoring system," in *Proc. World Congr. Comput. Sci. Inf. Eng.*, 2009, pp. 636–640.
[6] D. Dawson and S. Birchfield, "An energy minimization approach to automatic traffic camera calibration," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1095–1108, Sep. 2013.
[7] L. Liu, J. Xing, G. Duan, and H. Ai, "Scene transformation for detector adaptation," *Pattern Recognit. Lett.*, vol. 36, pp. 154–160, Jan. 2013.
[8] T. Schoepflin and D. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 90–98, Jun. 2003.
[9] K.-T. Song and J.-C. Tai, "Dynamic calibration of pan–tilt–zoom cameras for traffic monitoring," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 36, no. 5, pp. 1091–1103, Oct. 2006.
[10] X. C. He and N. H. C. Yung, "A novel algorithm for estimating vehicle speed from two consecutive images," in *Proc. IEEE WACV*, 2007, pp. 1–6.
[11] G. S. K. Fung, N. H. C. Yung, and G. K. H. Pang, "Camera calibration from road lane markings," *Opt. Eng.*, vol. 42, no. 10, pp. 2967–2977, Oct. 2003.
[12] T.-W. Pai, W.-J. Juang, and L.-J. Wang, "An adaptive windowing prediction algorithm for vehicle speed estimation," in *Proc. IEEE Intell. Transp. Syst.*, 2001, pp. 901–906.
[13] D. Dailey, F. Cathey, and S. Pumrin, "An algorithm to estimate mean traffic speed using uncalibrated cameras," *IEEE Trans. Intell. Transp. Syst.*, vol. 1, no. 2, pp. 98–107, Jun. 2000.
[14] Y. Zheng and S. Peng, "A practical roadside camera calibration method based on least squares optimization," *IEEE Trans. Intell. Transp. Syst.*, vol. 15, no. 2, pp. 831–843, Apr. 2014.
[15] N. K. Kanhere, S. T. Birchfield, and W. A. Sarasua, "Automatic camera calibration using pattern detection for vision-based speed sensing," *J. Transp. Res. Board*, vol. 2086, no. 1, pp. 30–39, 2008.
[16] Z. Zhang, T. Tan, K. Huang, and Y. Wang, "Practical camera calibration from moving objects for traffic scene surveillance," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 3, pp. 518–533, Mar. 2013.
[17] N. K. Kanhere and S. T. Birchfield, "Real-time incremental segmentation and tracking of vehicles at low camera angles using stable features," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 148–160, Mar. 2008.
[18] F. Cathey and D. Dailey, "Mathematical theory of image straightening with applications to camera calibration," in *Proc. IEEE Intell. Transp. Syst. Conf.*, 2006, pp. 1336–1369.
[19] R. Cipolla, T. Drummond, and D. P. Robertson, "Camera calibration from vanishing points in image of architectural scenes," in *Proc. BMVC*, 1999, pp. 38.1–38.10.
[20] B. Caprile and V. Torre, "Using vanishing points for camera calibration," *Int. J. Comput. Vis.*, vol. 4, no. 2, pp. 127–139, Mar. 1990.
[21] J. Deutscher, M. Isard, and J. MacCormick, "Automatic camera calibration from a single manhattan image," in *Proc. ECCV*, 2002, pp. 175–188.
[22] M. Dubská and A. Herout, "Real projective plane mapping for detection of orthogonal vanishing points," in *Proc. BMVC*, 2013, pp. 90.1–90.10.
[23] M. Dubská, "Point and line parameterizations using parallel coordinates for Hough transform," Ph.D. dissertation, Brno University of Technology, Brno, Czech Republic, 2014.
[24] M. Dubská, A. Herout, and J. Havel, "PClines—Line detection using parallel coordinates," in *Proc. IEEE CVPR*, 2011, pp. 1489–1494.
[25] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Conf. CVPR*, 1994, pp. 593–600.
[26] C. Tomasi and T. Kanade, "Detection and tracking of point features," School Comput. Sci., Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU-CS-91-132, 1991.
[27] F. Devernay and O. Faugeras, "Straight lines have to be straight," *Mach. Vis. Appl.*, vol. 13, no. 1, pp. 14–24, Aug. 2001.
[28] D. C. Brown, "Close-range camera calibration," *Photogramm. Eng.*, vol. 37, no. 8, pp. 855–866, 1971.
[29] H.-G. Beyer and H.-P. Schwefel, *Evolution strategies—A comprehensive introduction*, vol. 1, no. 1, pp. 3–52, May 2002.
[30] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.

Markéta Dubská received the Ph.D. degree from Brno University of Technology (BUT), Brno, Czech Republic, in 2014.

She is a Postdoctoral Fellow with the Department of Computer Graphics and Multimedia, Faculty of Information Technology, BUT. Her research interests include computer vision, geometry, and computation using parallel coordinates.

Adam Herout received the Ph.D. degree from Brno University of Technology (BUT), Brno, Czech Republic.

He is an Associate Professor with BUT, where he also leads the Graph@FIT Research Group. His research interests include fast algorithms and hardware acceleration in computer vision. He is also a Cofounder of angelcam.com, which provides web streaming from network cameras and real-time computer vision in the cloud.

Roman Juránek received the Ph.D. degree from Brno University of Technology (BUT), Brno, Czech Republic, in 2012.

He is currently a Postdoctoral Fellow with the Graph@FIT Research Group, Department of Computer Graphics and Multimedia, Faculty of Information Technology, BUT. His professional interests include computer vision, machine learning, and pattern recognition.

Jakub Sochor received the M.S. degree from Brno University of Technology (BUT), Brno, Czech Republic. He is currently working toward the Ph.D. degree in the Department of Computer Graphics and Multimedia, Faculty of Information Technology, BUT.

His research focuses on computer vision, particularly traffic surveillance.