

Road Curb Detection and Localization with Monocular Forward-view Vehicle Camera

Stanislav Panev, Francisco Vicente, Fernando De la Torre and Véronique Prinnet

Abstract—We propose a robust method for estimating road curb 3D parameters (*size, location, orientation*) using a calibrated monocular camera equipped with a fisheye lens. Automatic curb detection and localization is particularly important in the context of Advanced Driver Assistance System (ADAS), i.e. to prevent possible collision and damage of the vehicle’s bumper during perpendicular and diagonal parking maneuvers. Combining 3D geometric reasoning with advanced vision-based detection methods, our approach is able to estimate the vehicle to curb distance in real time with mean accuracy of more than 90%, as well as its orientation, height and depth.

Our approach consists of two distinct components – curb detection in each individual video frame and temporal analysis. The first part comprises of sophisticated curb edges extraction and parametrized 3D curb template fitting. Using a few assumptions regarding the real world geometry, we can thus retrieve the curb’s height and its relative position w.r.t. the moving vehicle on which the camera is mounted. Support Vector Machine (SVM) classifier fed with Histograms of Oriented Gradients (HOG) is used for appearance-based filtering out outliers. In the second part, the detected curb regions are tracked in the temporal domain, so as to perform a second pass of false positives rejection.

We have validated our approach on a newly collected database of 11 videos under different conditions. We have used point-wise LIDAR measurements and manual exhaustive labels as a ground truth.

Index Terms—curb detection, parking assistance, monocular camera, HOG, SVM, template fitting, tracking.

I. INTRODUCTION

OVER the last few years, the automotive industry has been focused on developing autonomous driving vehicles to reduce accidents and increase independence. As an intermediate step toward fully autonomous vehicles, the importance of active safety technologies, such as adaptive cruise control, blind spots warning, and automatic park system, has increased. Those features rely for most on sensor-based technologies, that try to understand the host vehicle’s surrounding, i.e. to detect dynamic and static obstacles within a certain range. For example, moving objects, such as pedestrians and vehicles, can be detected to warn drivers to be cautious. Automatic detection of road signs can be used to control or adjust vehicles speed.

S. Panev, F. Vicente, and F. De la Torre are with the Carnegie Mellon University, Pittsburgh, PA 15213-3815 USA (e-mail: spanev@cmu.edu; franciscovicencar@gmail.com; ftorre@cs.cmu.edu).

V. Prinnet is currently with The Hebrew University of Jerusalem, Israel. This work was initiated while she was at General Motors. (e-mail: vprinnet@gmail.com).

Manuscript received December 15, 2017; revised June 30, 2018 and September 15, 2018; accepted September 20, 2018. Date of publication November 12, 2018; date of current version August 27, 2019.

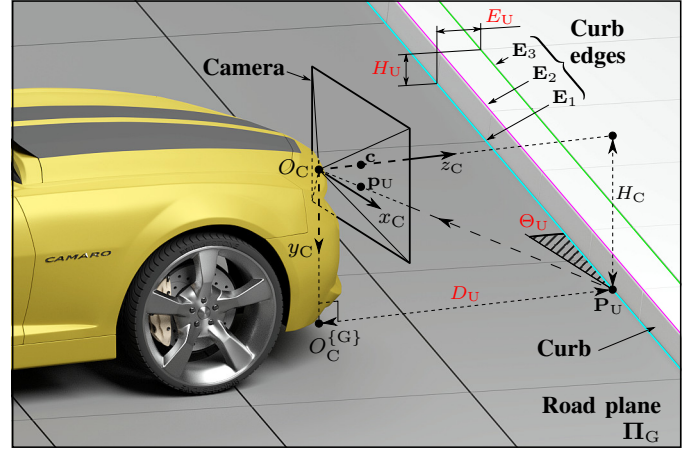


Fig. 1. Our system estimates the relative position, orientation and size of a curb w.r.t. a host vehicle, by the means of a monocular forward-viewing fisheye camera, advanced geometrical reasoning, temporal analysis and machine learning.

Curbs or sidewalks (in addition to road surface markings) are clues that are exploited in positioning systems. Often they indicate the boundary of parking areas. Technologies that can accurately detect and estimate curb location and height are embedded in any assist/autonomous parking systems: they enable to predict the vehicle-to-curb distance, hence to avoid a potential collision between the curb and the vehicle, cause of damages on tires and bumpers. The constraints put on such systems are very high: near-zero false negative detection, distance and height estimation with centimeter accuracy.

The challenges to build such systems are at least two-fold. First, curbs are objects of small size. This compels to use sensors of very high resolution to capture data where the object of interest covers a sufficiently large region. Second, curb shapes and appearance textures can vary drastically (depending on weather condition, pavement material, painting, etc.). This requires the development of advanced recognition techniques, capable to robustly classify an object as a curb or not.

Most common techniques in the literature tackle mainly the first issue, i.e. 3D detection, using active sensors (LIDAR, laser range finders, etc.) or camera stereo-vision systems. Those approaches assume that the curb’s shape is in itself a discriminative and robust feature. They are likely to fail in poor SNR conditions (e.g. bad weather) or when facing damaged curbs. To our knowledge, few work attempted to couple 3D geometry with vision-based appearance models, so

as to improve robustness and accuracy.

In this paper, we describe a system intended to assist the drivers and reduce the risk of running over obstacles, such as the curbs, in the everyday parking activities. Thereby, preventing unwanted damages and accidents. A single monocular wide-angle camera is used as an input device, which is one of the most economically efficient solutions nowadays. The cost of such sensors sometimes can be several magnitudes lower than the price of the more complex devices, such as LIDARs and stereo cameras. Our system works in real time, while maintaining high detection rate and curb parameters estimation accuracy. It relies on geometric reasoning, simple hand-crafted features (image edges and *Histograms of Oriented Gradients* – HOG), model-based machine learning techniques (*Support Vector Machines* – SVM) and temporal filtering. Our *Inverse Perspective-compressing Mapping* (IPcM) technique approaches the curb edge detection in a sophisticated scale-invariant manner, without the need of maintaining large multi-scale image space in order to reliably handle the broad variations of the curb size in the image. All that allowed us to come up with an CPU-based implementation giving our system high level of flexibility. For example, the current setup consists of a single front-mounted fisheye camera, which is applicable to perpendicular and diagonal forward parking scenarios. Just by attaching few more cameras to the system that cover the lateral and rear perimeters around the vehicle, will extend the system applicability to perpendicular reverse and in-line parking. The aforementioned advantages also would let our system to operate on a single computing platform in conjunction with algorithms designed to solve more challenging tasks based on deep models, such as motion planning and control. Thus, each subsystem occupies separate processing unit - CPU and GPU.

II. RELATED WORK

The research of the curb detection algorithms can be figuratively organized according to the types of the sensors employed. Undoubtedly, the LIDARs are among the most popular active sensors. They provide 3D point cloud data based on laser scanning and range measurements. In [1], for example, the authors voxelize the LIDAR data for computational efficiency and detect those containing ground points, based on the elevation information and plane fitting. The candidate curb points are selected using three spatial cues. Employing short-term memory technique along with a parabolic curb model and RANSAC they remove the false positives. For temporal curb tracking a particle filter is used. In [2] the curb is modeled as parabola and Integral Laser Points features are used for speed up. Instead of temporal filters and spline fitting methods, in [3] a robust regression method to deal with occluding scenes, called Least Trimmed Squares (LTS), is used in combination with Monte Carlo localization. In [4] instead of extracting features, the LIDAR scan lines are processed directly. Initial curb point candidates are determined by Hough transform and then iterative Gaussian Process Regression is used to represent the curb models. In [5] the parabola model is employed as well, but the tracking technique is based on Kalman filter in combination with GPS/IMU data.

Another active sensor which provides 3D point cloud data is the Time-of-Flight (ToF) camera. It extracts the depth information from a 3D scene based on the phase shifts of light signals caused by the different times they travel in space to bounce off the objects and return back to the camera. In [6] the authors take advantage of the ToF camera's high frame rate to improve the results by space-time data accumulation using grid based approach. For estimating ego-motion parameters they employ Kalman filter. In [7] CC-RANSAC method is used for improved plane fitting into the raw point cloud data.

The laser range finders (LRF) are active sensors from the LIDARs family, but instead of providing 3D point cloud data, they usually scan just a single line and estimate the distances of each measurement point along it. The curb detection algorithm in [8] is accomplished in two steps. Firstly, the authors detect the potential curb positions in the LRF data, then they refine the results by employing Particle filter. In [9], LRF data captured sequentially is used to build local Digital Elevation Maps (DEM) and Gaussian process regression is at the final curb detection stage. A set of 3 LRF sensors is used in [10]. Peak detection is accomplished on the results of derivative-based distance method described there and then they merge the data from the individual sensors.

A popular passive sensor is the stereo camera. Similar to the LIDARs, it provides 3D point cloud data which has higher resolution, but usually contains more noise. DEMs are often used for efficient representing the 3D data in the area of curb detection. In [11], edge detection is applied to the DEM data to highlight the height variations. The noise from the stereo data is reduced significantly by creating multi-frame persistent map. Hough accumulator for lines is built with the persistent edge points. Each curb segment is refined using the RANSAC approach to fit optimally the 3D data of the curb. In [12], a 3D environment model is utilized. It consists of various primal entities, such as road, sidewalk, etc. The 3D data points are assigned to the different part of the model using temporally integrated Conditional Random Fields (CRF). In [13], temporal integration of the DEM data is also used, but in combination with least squares cubic spline fitting. The algorithm described in [14] presents an interesting idea of combining the 3D point cloud data with the intensity information from the stereo camera. First, they extract model-based curb features from the 3D point cloud and validate them by using the intensity image data. The curbs are presented as 3D polynomial chains. The approach described in [15] is based on the curvature of the 3D point cloud data. It is estimated by applying the nearest neighbor paradigm. The method's performance is evaluated by applying it to both – stereo camera and LIDAR data.

All the sensor types used in the algorithms above directly provide some kind of 3D information, either point cloud or line-wise. However, monocular cameras data lacks completely from depth information. Thus, extracting curbs using them is a challenging task, usually founded on preliminary constraints and assumptions. In [16] the image is divided in regular grid of cells. Then 3D reconstruction is applied by pixel-wise image labeling based on CRFs. Besides the camera, the vehicles CAN-bus data is employed as well in [17]. Then two

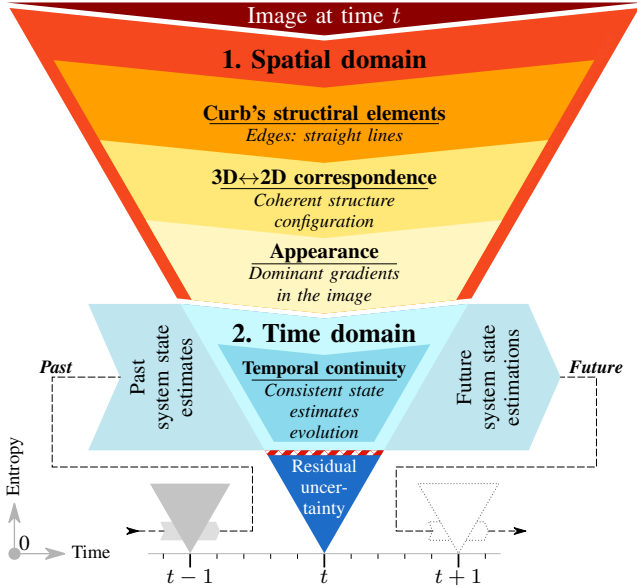


Fig. 2. System paradigm: the intersection between the spatial and temporal information cues is used to minimize the uncertainty of curb parameters estimation.

complementing methods are applied: localizing borders using texture based area classification with local binary patterns (LBP) features and Harris features tracking using Lucas-Kanade tracker to extract 3D information. The curb detection system described in [18] is closely related to our approach. It also involves use of a fisheye camera and incorporates Histogram of Oriented Gradients (HOG) features. Unlike our approach, they preserve the original camera image. Hence, their curb model is polynomial. The temporal filtering is based on Kalman filter.

III. SYSTEM DESCRIPTION

A. Algorithm summary

The objective of our system is to acknowledge the presence of a curb close to the vehicle, along its forward motion path, and to help identifying if it is an immediate threat to the vehicle's integrity by estimating its position, orientation and size relative to the vehicle. These parameters constitute the system state vector \mathbf{x} , described later in Section III-B. The system tracks just one curb at a time, as only the closest to the vehicle one is significant.

Fig. 2 illustrates graphically the paradigm our system is founded on. Its shape likens inverse pyramid, situated in the *Time/Entropy* plain. Pyramid tip points to the moment of time t which corresponds to the last captured camera frame (image). The width of its layers (and their coloring) depicts overall entropy rate in terms of the state vector \mathbf{x} estimation. The paradigm is inspired by the idea of the attentional cascade presented in [19], but instead of boosted classifiers we use various filtering techniques. The direction of processing flow is from the top to the bottom and each stage is purposed to reduce system's state entropy until the uncertainty is low enough that a reasonable inference for the values of curb parameters can be made.

The cascade consists of two major layers which handle the information from space and time perspectives. Immediately after a new image is delivered by the camera at time t , it is fed to the “**Spatial domain**” pyramid layer, which consists of three sub-layers. The first one searches for individual *curb's primitive structural elements* in the image. As such, we engage curb's edges, since they are 3D lines. Projecting them onto the camera's image plane won't take away their straightness, because of the linearity of the perspective transform. Therefore, curb's edges can be detected just by performing line detection in the image. This part of our algorithm is described in Section III-E1.

Next, we raise the level of generalization up by utilizing curb's geometry itself, i.e. the configuration of its primitive structural elements (points, edges, faces) that constitute its 3D structure. Here, our algorithm relies on the prior knowledge of curb's geometry and the *3D to 2D correspondence* in order to estimate which compositions of image lines could probably represent a projection of a 3D curb-like body. All the successful guesses mold the initial hypothesis set of *curb candidates*. Its outlying members are meant to be rejected by the next layers of our paradigm pyramid. Detailed description is presented in Section III-E2.

The last operation in the spatial domain shifts the focus from curb's geometry to its *appearance* in the image. Here we perform object detection to validate every curb candidate by the means of sophisticated Machine Learning techniques. More information can be found in Section III-E3.

The purpose of our system is to estimate curb parameters while vehicle is in motion. Luckily, it is a considerably massive object with predictable kinematics. Hence, the evolution of curb's parameters (system state) in the “**Time domain**” follow smooth trajectories, with no abrupt discontinuities. In other words, our system deals with environment which obeys *temporal continuity*. Thus, from all the curb candidates we can select only those which comply with it and also make reasonable predictions for system's future states. Our Curb tracking technique is described in Section III-F.

The most bottom layer of the pyramid represents the residual uncertainty which our system, as a non-ideal one, cannot resolve and bring the entropy to the theoretical value of 0, i.e. 100% confidence about curb parameters estimates. Our goal is to minimize it and in Section IV we present results which demonstrate the promising performance of our system.

B. Geometrical considerations and definitions

Here we describe the fundamental assumptions and constraints our algorithm is based on.

1) *Road*: We consider the road as a perfectly flat structure. Although, the real roadways are not ideally planar due to technological slopes or deformations caused by exploitation, their surface curvature is smooth enough to allow us make such an assumption considering the size of our system's working area (*Curb Detection Domain*, defined below in III-B4) and the amount deviation from the perfect plane within it. The road plane is denoted as $\Pi_G \in \mathbb{R}^3$ and its projection in camera image as $\pi_G \in \mathbb{R}^2$ (Fig. 1).

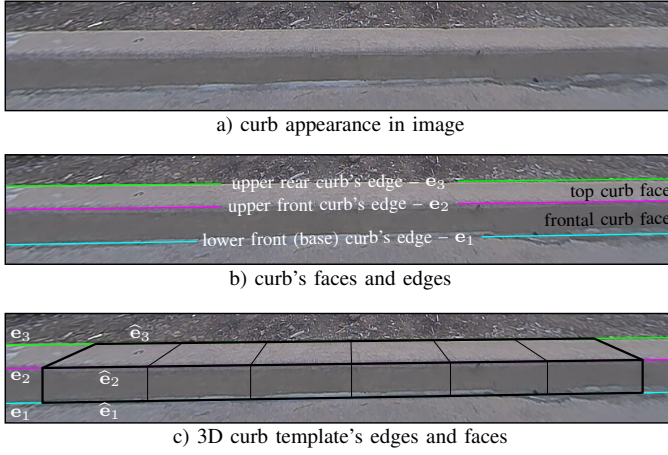


Fig. 3. Curb elements, shape and definitions.

2) *Camera*: The camera is mounted in the middle of vehicle's front and points upon vehicles forward movement direction. Camera's projection center (focal point) O_C is elevated above the road at a fixed distance H_C and this is where camera's coordinate frame $O_C x_C y_C z_C$ originates (Fig. 1). During parking vehicle's speed is relatively low, therefore we can neglect the actuation caused by vehicle's suspension system. Furthermore, for the sake of simplicity and computational efficiency we define that camera's plane $x_C z_C$ is parallel to the road plane Π_G (Fig. 1).

3) *Curb*: We define the curbs as rectangular prismoidal rigid structures, which determine road plane borderlines. They are usually significantly elevated above the road surface and often specify the boundaries between the road and sidewalk, for example. We also assume that the curbs have negligibly small fillets (roundings) of the corners, which results in clear and abrupt brightness transitions (edges) in the camera images.

Our algorithm uses curb's edges as primal features. They are straight, easily detectable and can help us to drastically reduce curb detection time. We would like also to note that if a curb appears in the image, three of its horizontal edges and two faces defined by them will always be presented in it (Fig. 3a and b). Let \mathbf{E}_1 , \mathbf{E}_2 and $\mathbf{E}_3 \in \mathbb{R}^3$ be the three lines depicting curb's lower front (base), upper front and rear edges, respectively (Fig. 1). Accordingly, \mathbf{e}_1 , \mathbf{e}_2 and \mathbf{e}_3 are their \mathbb{R}^2 projections in the image (Fig. 3b).

Curb detection and localization aim to the estimation of four basic curb parameters: D_U – the distance between the curb and camera, Θ_U – the rotation angle of the curb about vertical axis, H_U and E_U – curb's height and depth, respectively. To precise the definition of D_U we introduce the notion of curb's reference point \mathbf{P}_U (Fig. 1), which is the intersection between the plane $y_C z_C$ and \mathbf{E}_1 – *curb's base edge*. Then D_U can be described as the distance between \mathbf{P}_U and the orthogonal projection of the origin O_C on the road plane $O_C^{\{G\}}$. As a consequence of our system's simplified geometrical configuration (see above), D_U is equal to the z -coordinate of \mathbf{P}_U in camera's frame $O_C x_C y_C z_C$. Θ_U can be defined as the angle between camera's axis x_C and the edges $\mathbf{E}_{1,2,3}$, H_U is the distance between \mathbf{E}_1 and \mathbf{E}_2 , respectively, E_U is the

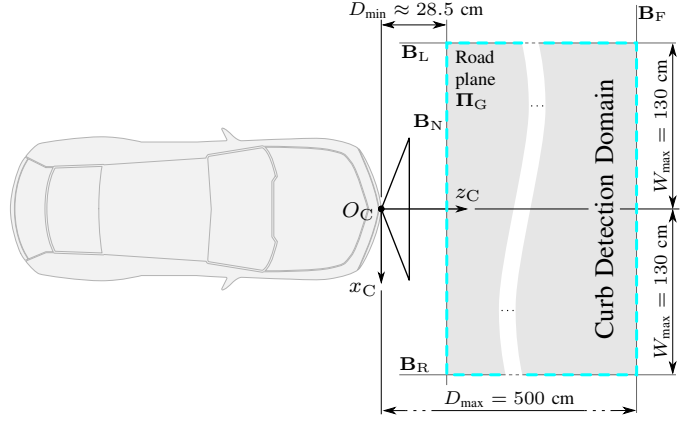


Fig. 4. System's *Curb Detection Domain* (CDD) (cyan) – shape and dimensions.

distance between \mathbf{E}_2 and \mathbf{E}_3 (Fig. 1). Semantically, we split curb's parameters in two groups – essential and secondary. D_U , Θ_U and H_U are considered as essential, because they provide enough information to determine the safe-clearance between the curb and vehicle. E_U is considered secondary and it is used for an additional information cue to improve system's reliability during operation.

We define system's state vector as follows

$$\mathbf{x} = [D_U, \Theta_U, H_U, E_U]^T, \quad (1)$$

and it holds all the curb parameters. Their estimation is accomplished by the means of a 3D parametric template (shown in Fig. 3c). Similar to the curb, it consists of two orthogonal rectangular faces and three edges $\hat{\mathbf{E}}_1$, $\hat{\mathbf{E}}_2$ and $\hat{\mathbf{E}}_3$ which correspond to the curb's ones. Consequently, their projections in the image plane are $\hat{\mathbf{e}}_1$, $\hat{\mathbf{e}}_2$ and $\hat{\mathbf{e}}_3$, respectively. The template has the same set of parameters as the curb and they are organized in the template's state vector

$$\hat{\mathbf{x}} = [\hat{D}_U, \hat{\Theta}_U, \hat{H}_U, \hat{E}_U]^T. \quad (2)$$

Detailed description of the fitting procedure is presented in Section III-E2.

A curb detection in the image frame at time t is considered as successful, if at least its essential parameters are estimated correctly. Therefore, we need to determine at least the position and orientation of \mathbf{E}_1 and \mathbf{E}_2 w.r.t. the camera from their projections \mathbf{e}_1 and \mathbf{e}_2 , respectively.

4) *Curb detection domain and image's curb searching region*: From a practical point of view, we define the rectangular area of the road plane directly in front of the vehicle as *Curb Detection Domain* (CDD) (Fig. 4). In essence, CDD determines system's domain of definition (or operation) w.r.t. $D_U \in [D_{\min}, D_{\max}]$. D_{\max} is chosen to be 500 cm, whereas D_{\min} is calculated from camera's vertical *Field of View* (FoV) bottom boundary. In our setup its value is ≈ 28.7 cm. The CDD's side limit W_{\max} is derived from FoV as well and is rounded to 130 cm. The total area covered by the our system's CCD is ≈ 12.25 m². The four sides of CDD are defined by the lines \mathbf{B}_N , \mathbf{B}_F , \mathbf{B}_L and \mathbf{B}_R . Their projections in the image are respectively \mathbf{b}_N , \mathbf{b}_F , \mathbf{b}_L and \mathbf{b}_R .

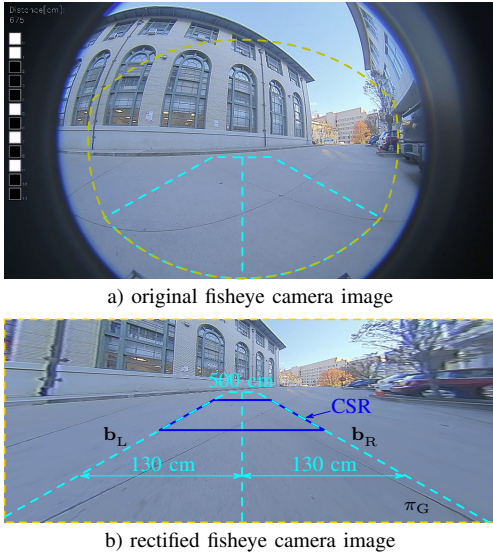


Fig. 5. The original (a) and rectified (b) versions of an image from the forward-view vehicle fisheye camera. *cyan dashed line* – Curb Detection Domain (CDD), *orange dashed line* – boundaries of the rectified camera image, *blue solid line* – exemplar position of Curb Searching Region (CSR).

In order to reduce the amount of data being processed for each frame and eliminate the influence of outliers, we introduce the notion of *Curb Searching Region* (CSR) in the image (Fig. 5b). It defines the image area used for extracting curb features. Its size and position are variable and determined by the expected curb location at the time of current camera frame. Essentially, it represents the projection in the image of a CDD subregion which has the same width, but significantly shorter length. Its initial position is set at the far end of CDD and in *Curb tracking* mode (see Section III-F) the system updates its position accordingly.

C. System calibration

All images from the camera are rectified before any further processing to eliminate the radial distortions introduced by the fisheye lens (Fig. 5). Otherwise, curb detection would be much more complex, involving second or higher order curves detection and fitting. We employ the fisheye camera model described in [20] to estimate camera’s intrinsic parameters in an offline calibration procedure using a planar target. In the rest of this paper, by the notion “image” we refer to the rectified version of the original image, unless anything else is explicitly stated.

The next step is, assuring that the camera extrinsic parameters follow the geometric definitions above (Section III-B). We don’t estimate those parameters through a calibration procedure. Instead, we set some of them manually. For instance, camera tilt angle should be set to zero. To achieve that, we employ a simple four-steps calibration procedure, illustrated in Fig. 6-top. A point target (marker) mounted on a stand (tripod), whose height is adjustable, is used. Repeating the these steps 3-4 time ensures that camera orientation will easily converge to the desired state. Not only the correct orientation of the camera is set during this process, but we also get an accurate

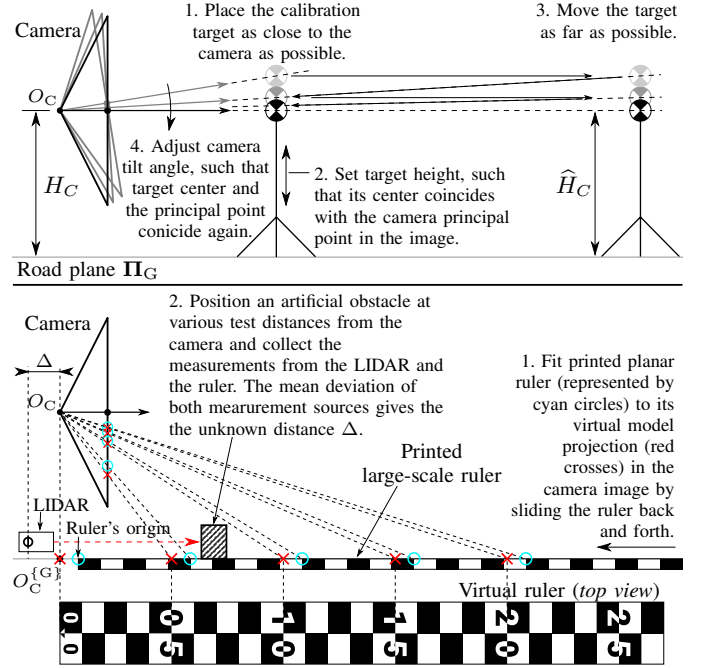


Fig. 6. System geometry calibration: (*top*) Assure that camera optical axis is parallel to the road plane by the means of point calibration target with adjustable height. This procedure give us also an accurate estimate of camera elevation H_C . (*bottom*) Estimate the distance offset Δ between the camera and LIDAR measurements.

estimate of the distance $H_C \rightarrow \hat{H}_C$, which is the distance from target center to the ground and can be measured with a ruler.

Fig. 6-bottom illustrates our technique to estimate the horizontal offset between camera and LIDAR origins – Δ , which is needed when evaluating system distance measurement capabilities. The peculiarity here is that the position of camera coordinate frame origin O_C is hard to be measured. Therefore, we came up with a technique to estimate its position implicitly. More precisely, we do not measure its actual position, but the position of its projection on the road plane $O_C^{\{G\}}$. Thanks to our geometric setup this is sufficient to estimate Δ . We use a 3D model of a planar large-scale ruler with a regular grid (10 cm per division). First, we create a virtual 3D model of it, which is coplanar with the road plane Π_G , its origin coincides with $O_C^{\{G\}}$ and its measurement axis is parallel to z_C . Then we project this virtual model of the ruler to camera plane and overlay its projection in the image. Next, we take a printed version of the same ruler model with the same scale, lay it in front of the camera and fit it to the overlaid projection of its virtual analog. When finished, we know that the origin of the printed ruler corresponds to the origin of the virtual one and, consequently, to $O_C^{\{G\}}$. Afterwards, we use a test obstacle to initiate measurements from the LIDAR and the camera, through the ruler (see Section III-D). By averaging the differences of those pairs of measurements we can calculate Δ .

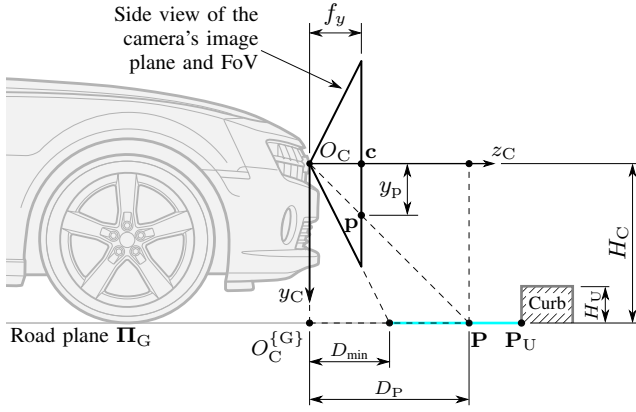


Fig. 7. Employing a monocular camera to measure the distance between a point from the road plane and the camera.

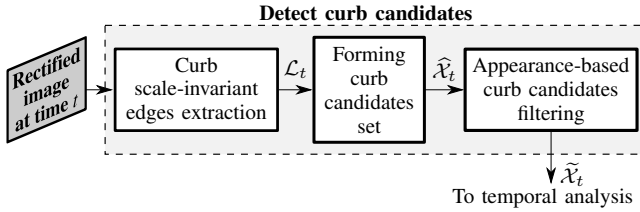


Fig. 8. Curb candidates detection in a single frame.

D. Measuring distances with a monocular camera

The plane $y_C z_C$ of camera's coordinate frame is illustrated in Fig. 7. Let $\mathbf{P} \in \mathbb{R}^3$ is a point from the road plane with coordinates $\mathbf{P} = [x_P, y_P, z_P]^\top = [x_P, H_C, D_P]^\top$ in camera's coordinate frame. Its projection onto the image plane is the point $\mathbf{p} = [x_p, y_p, 1]^\top \in \mathbb{P}^2$ w.r.t. the frame originating at the principal point \mathbf{c} . The relation between their coordinates is

$$\mathbf{p} = \frac{K_c \mathbf{P}}{D_P}, \quad (3)$$

where $K_c = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$, f_x and f_y are camera's focal lengths along its x_C and y_C axes, respectively. From (3) we get

$y_p = \frac{f_y H_C}{D_P}$, which means that y_p depends only on D_P , since f_y and H_C are practically constants. Furthermore, if we invert the equation, we can calculate the distance D_P of every point \mathbf{P} from the road plane just by using the vertical coordinate y_p of its projection \mathbf{p} from the image. Hence, for curb's reference point \mathbf{P}_U (Fig. 1) can be rewritten

$$D_U = \frac{f_y H_C}{y_U}, \quad (4)$$

where y_U is the vertical coordinate of \mathbf{P}_U 's projection in the image.

E. Detect curb candidates in an image

Here we describe our approach for detecting a curb in the image. The algorithm is inspired by the paradigm for boosted attentional cascade presented in [19], but instead of using a cascade of boosted classifiers with gradually increasing complexity, our cascade consists of various filtering techniques.

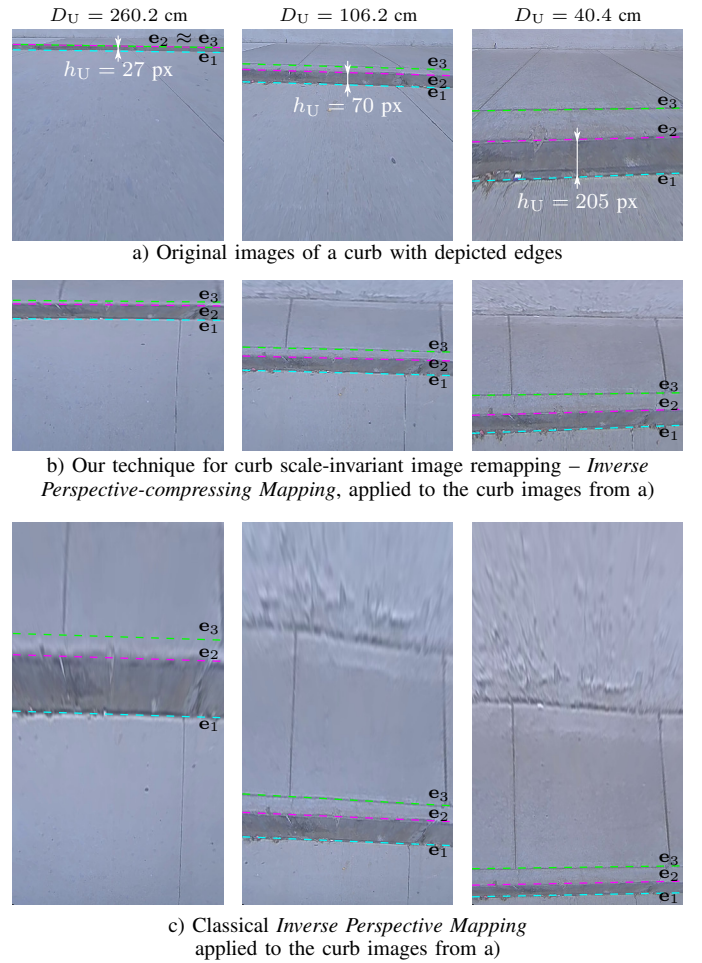


Fig. 9. A set of three curb images taken at three different distances D_U from the same video sequence. They demonstrate the issues related to edge detection and our approach to solve them.

Thus, the amount of data being processed is greatly reduced by rejecting image regions which do not contain curb features. As result, only positively classified curb candidates are left for further temporal analysis.

Fig. 8 illustrates our curb detection pipeline, which consists of three consecutive operations:

1) *Curb scale-invariant edges extraction*: As we have already mentioned earlier, the primary features which we exploit are the curb's edges (Fig. 3). The straight lines in the image are extracted by the well known combination of *Hough transform* (HT) [21] and the *Canny edge detector* [22] applied to the camera images. Also, as we have already described in Section III-B3, the least sufficient condition to perform successful curb detection in the image at time t is that both curb's edges projections e_1 and e_2 are correctly detected. They define curb's frontal face projection in the image on which we are going to emphasize here.

Let R_U is curb's frontal face vertical spatial sampling rate in the image

$$R_U = \frac{h_U}{H_U} = \frac{f_y}{D_U} \left(\frac{\text{px}}{\text{cm}} \right), \quad (5)$$

where h_U is the curb's frontal face projection height in the

image. Essentially, R_U provides information about how many sampling locations (pixels) are used by the camera to represent every centimeter of a vertical line from curb's frontal face, which is located at the distance D_U from the camera. As a consequence from (5), R_U is a non-constant function with respect to D_U , which is demonstrated graphically in Fig. 9a. The figure shows three sample images of the same curb taken at 3 different distances D_U . It is obvious that h_U (marked in the figure) varies significantly. In particular, based on our system's setup the ratio

$$\frac{R_U(D_U = D_{\min})}{R_U(D_U = D_{\max})} = \frac{D_{\max}}{D_{\min}} \approx 17.5, \quad (6)$$

i.e. the projection of a curb taken at the distance $D_U = D_{\min}$ will contain about 17.5 times more pixel information than the projection of the same curb taken at a distance $D_U = D_{\max}$.

Our system relies on detecting curb's edges. Principally, edge detection aims in finding the points of discontinuity in the image brightness by incorporating either its first- or second-order derivatives. The *Canny edge detector*, for example, uses first-order operators, such as Sobel [23], [24], Scharr [25], Prewitt [26] or Roberts cross [27] to estimate brightness gradient magnitude and direction. Having such a significant variation of R_U , though, results in volatile gradient magnitude over curb's edges projections, thus inconsistent curb detection. In order to overcome this problem, we have derived an image remapping technique – *Inverse Perspective-compressing Mapping* (IPcM), which aims to equalize the spatial sampling rate of the curb in the image (Fig. 9b). After warping the image, the detection of curb's edges tends to be much more steady and robust through the entire CDD. Also, the original trapezoidal shapes of CDD and CSR in the image are transformed into rectangles. Detailed explanation about the derivation of our technique can be found in Appendix A.

The classical *Inverse perspective mapping* (IPM) normalizes the spatial sampling rate of the road and any other parallel to it plane (Fig. 9c), but not for the orthogonal frontal curb's face, as can be seen on the figure. Notably, h_U is still dependent on D_U and varies considerably. The difference is that after the transformation their relation is proportional. Moreover, the IPM transform introduces an additional issue, which can be easily acknowledged from the figure. The output image suffers from gradually increasing interpolation smoothing, mainly along its vertical axis. It is caused by the irregular density of camera pixels sampling locations in the 3D scene (the density of the points \mathbf{P}_i in Fig. 21 is variable).

Let $\mathcal{L} = \{\mathbf{l}_i\}_1^{N_{\mathcal{L}}}$ be the set of straight lines in the image at time t , detected by applying edge detection to the IPcM remapped image and then transforming the lines back to the original image space. $N_{\mathcal{L}}$ is the size of the set and $\mathbf{l}_i = [a_i, b_i]^T \in \mathbb{R}^2$ are vectors representing the individual lines from the set by their parametric form: a_i – slope and b_i – intercept. This procedure is expected to produce significant amount of outliers. That's exactly what we are aiming at. The purpose of the following processing blocks of the flow diagram (Fig. 8) is rejection of everything, which is not associated with the curb. As the curb edges are expected to be among the longest ones in the CSR (extending through its full width), we

can reduce the processing time by using just the six lines with the highest voting scores from HT ($N_{\mathcal{L}} \in [0, 6]$). The other types of edges from different geometric shapes, which could possibly be presented in the image (from sidewalk pavement, tiles, etc.), usually have much shorter length, hence lower voting scores.

Finally, we examine the detected lines as a set of points in their parametric space and try to find clusters. We consider every cluster as a noisy representation of a single edge. Thus, all the members in a cluster are replaced by its mean.

2) *Forming curb candidates set:* We construct two new sets – \mathcal{G}_2 and \mathcal{G}_3 , which contain all possible combinations of 2- and 3-tuples, respectively, of non-intersecting lines \mathbf{l}_i from \mathcal{L} , i.e.

$$\begin{aligned} \mathcal{G}_2 = \left\{ \left(\mathbf{g}_{21}^{(j)}, \mathbf{g}_{22}^{(j)} \right)_j \right\} = \left\{ (\mathbf{l}_p, \mathbf{l}_q)_j \right\} : \\ p, q = 1 \dots N_{\mathcal{L}}, p \neq q, b_p > b_q \\ \mathbf{l}_p^{(h)} \times \mathbf{l}_q^{(h)} \notin I_t,^1 \\ j = 1 \dots N_{\mathcal{G}_2}, N_{\mathcal{G}_2} \leq \binom{N_{\mathcal{L}}}{2} \end{aligned} \quad (7)$$

and

$$\begin{aligned} \mathcal{G}_3 = \left\{ \left(\mathbf{g}_{31}^{(k)}, \mathbf{g}_{32}^{(k)}, \mathbf{g}_{33}^{(k)} \right)_k \right\} = \left\{ (\mathbf{l}_p, \mathbf{l}_q, \mathbf{l}_r)_k \right\} : \\ p, q, r = 1 \dots N_{\mathcal{L}}, p \neq q \neq r, b_p > b_q > b_r \\ \mathbf{l}_p^{(h)} \times \mathbf{l}_q^{(h)} \notin I_t,^1 \\ \mathbf{l}_p^{(h)} \times \mathbf{l}_r^{(h)} \notin I_t,^1 \\ \mathbf{l}_q^{(h)} \times \mathbf{l}_r^{(h)} \notin I_t,^1 \\ k = 1 \dots N_{\mathcal{G}_3}, N_{\mathcal{G}_3} \leq \binom{N_{\mathcal{L}}}{3}, \end{aligned} \quad (8)$$

where $\mathbf{l}_{p,q,r}^{(h)}$ are the homogeneous representations of the lines $\mathbf{l}_{p,q,r}$, I_t is the camera's image at time t and $N_{\mathcal{G}_{2,3}}$ are the sizes of the two sets. The lines in every tuple are sorted in ascending order of their vertical placement in the image, i.e. in descending order of lines' intercept parameter b_i^2 .

A tuple of three lines is sufficient to estimate all of the four curb template parameters (2), rather than the 2-lines case, where we omit E_U in favor of estimating the other three more important parameters: \widehat{D}_U , $\widehat{\Theta}_U$ and \widehat{H}_U . In order to shorten our presentation we are going to describe the triplet case only, because it is more general, complex and \mathcal{G}_2 could be considered as a special case subset of \mathcal{G}_3 .

The next step is fitting the curb template's projection in the image to each individual triplet in \mathcal{G}_3 . Every line from k -th triplet is explicitly related to a specific curb's edge, because they will always appear in the image in the same vertical order. Therefore, $\mathbf{g}_{31}^{(k)}$ represents \mathbf{e}_1 , $\mathbf{g}_{32}^{(k)} \rightarrow \mathbf{e}_2$ and $\mathbf{g}_{33}^{(k)} \rightarrow \mathbf{e}_3$. The objective of the fitting is to estimate the optimal values of the parameters $\widehat{\mathbf{x}}_k = \left[\widehat{D}_U^{(k)}, \widehat{\Theta}_U^{(k)}, \widehat{H}_U^{(k)}, \widehat{E}_U^{(k)} \right]^T$, such that curb's

¹Due to the properties of the perspective projection, the relative position and orientation of the camera, the road plane and the curb and their shapes, it is impossible that any of the lines \mathbf{e}_1 , \mathbf{e}_2 and \mathbf{e}_3 have common points in the image I_t . Therefore, all the 2- and 3-tuples members of \mathcal{G}_2 and \mathcal{G}_3 , respectively, which contain intersecting lines are rejected.

²Note that the vertical image axis v points downwards.

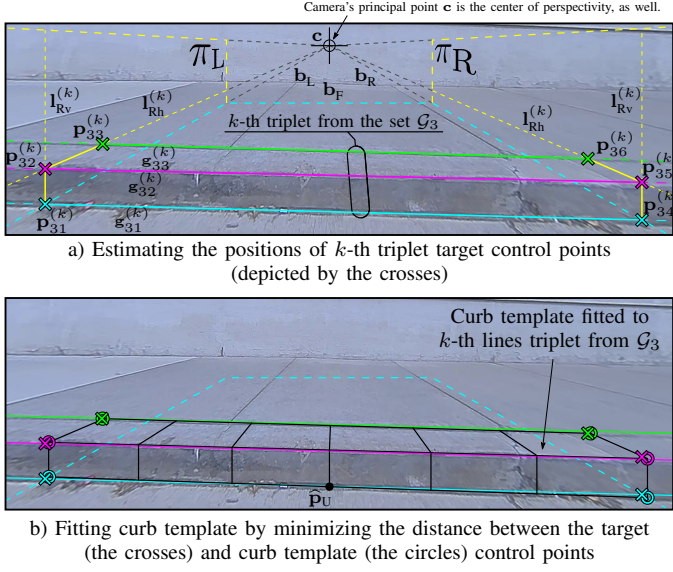


Fig. 10. Curb template fitting. First, the position of the target control points for k -th line triplet from \mathcal{G}_3 (depicted by the crosses) are estimated by following a reasoning based on the principles of the perspective transform and the simplified curb geometry defined in Section III-B3. Second, 3D curb template is fitted to every line tuple by minimizing the distance between its control points (depicted by circles) and the target control points through adjusting templates parameters $\hat{\mathbf{x}}$.

template edges projections in the image $\mathbf{e}_{1,2,3}$ are aligned to their corresponding lines from the triplet $\mathbf{g}_{31,32,33}$ in the best possible way.

We evaluate the similarity of two \mathbb{R}^2 lines by calculating the Euclidean distance between two pairs of corresponding points laying on each of them, which we call *control points*. Hence, to fit the template to a triplet of lines we need to use six pairs of control points. In order to define their position, we need to introduce the planes Π_L and Π_R and their projections in the image π_L and π_R (Fig. 10a). They intersect Π_G in \mathbf{B}_L and \mathbf{B}_R and $\Pi_{L,R} \perp \Pi_G$. Now curb template's control points are defined by intersecting its edge lines $\hat{\mathbf{E}}_{1,2,3}$ with Π_L and Π_R , which results in two \mathbb{R}^3 point triplets: $(\hat{\mathbf{P}}_1, \hat{\mathbf{P}}_2, \hat{\mathbf{P}}_3)$ and $(\hat{\mathbf{P}}_4, \hat{\mathbf{P}}_5, \hat{\mathbf{P}}_6)$ – Fig. 11. The calculation of their coordinates in camera's coordinate frame is significantly simplified due to systems geometrical setup, namely

$$\begin{aligned} \hat{\mathbf{P}}_1 &= \begin{bmatrix} -W_{\max}, & H_C, & \hat{D}_U + \Delta D_1 \end{bmatrix}^\top \\ \hat{\mathbf{P}}_2 &= \begin{bmatrix} -W_{\max}, & H_C - \hat{H}_U, & \hat{D}_U + \Delta D_1 \end{bmatrix}^\top \\ \hat{\mathbf{P}}_3 &= \begin{bmatrix} -W_{\max}, & H_C - \hat{H}_U, & \hat{D}_U + \Delta D_1 + \Delta D_2 \end{bmatrix}^\top \\ \hat{\mathbf{P}}_4 &= \begin{bmatrix} W_{\max}, & H_C, & \hat{D}_U - \Delta D_1 \end{bmatrix}^\top, \\ \hat{\mathbf{P}}_5 &= \begin{bmatrix} W_{\max}, & H_C - \hat{H}_U, & \hat{D}_U - \Delta D_1 \end{bmatrix}^\top \\ \hat{\mathbf{P}}_6 &= \begin{bmatrix} W_{\max}, & H_C - \hat{H}_U, & \hat{D}_U - \Delta D_1 + \Delta D_2 \end{bmatrix}^\top \end{aligned} \quad (9)$$

where

$$\begin{aligned} \Delta D_1 &= W_{\max} \tan \hat{\Theta}_U \\ \Delta D_2 &= \frac{\hat{E}_U}{\cos \hat{\Theta}_U} \end{aligned} \quad (10)$$

Afterwards, we can estimate their projections in the image as

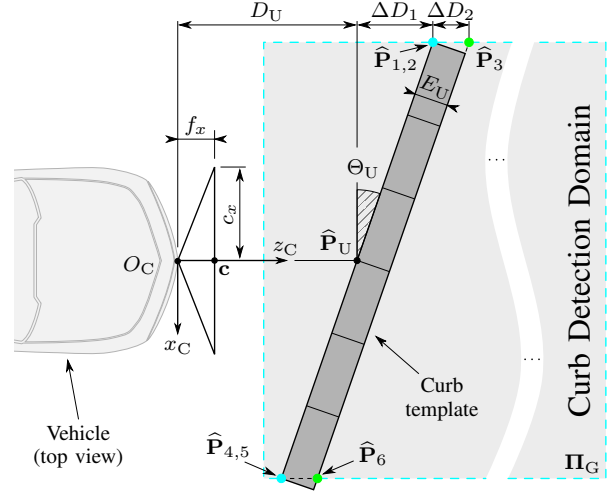


Fig. 11. Top view of the vehicle, camera and curb template, showing the location of the control points $\hat{\mathbf{P}}_m : m = 1, \dots, 6$.

follows

$$\hat{\mathbf{p}}_m(\hat{\mathbf{x}}) = K \hat{\mathbf{P}}_m, \quad (11)$$

where $m = 1 \dots 6$ and $K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$ is the camera

matrix, c_x and c_y are camera's principal point \mathbf{c} coordinates along the horizontal and vertical axes, respectively. It should be noted that since K , w_{\max} and H_C are constants, $\hat{\mathbf{p}}_m$ is function only of the curb parameters vector $\hat{\mathbf{x}}$.

The estimation of k -th triplet control points³ positions in the image is demonstrated in Fig. 10a. It follows the presumptions of template's control points location in the image and incorporating the properties of perspective projection. Firstly, we intersect $\mathbf{g}_{31}^{(k)}$ with $\mathbf{b}_{L,R}$ and thus estimate the control points $\mathbf{p}_{31}^{(k)}$ and $\mathbf{p}_{34}^{(k)}$ (depicted with cyan crosses in the figure).

$$\mathbf{p}_{31}^{(k)} = \mathbf{b}_L \times \mathbf{g}_{31}^{(k)} \quad \text{and} \quad \mathbf{p}_{34}^{(k)} = \mathbf{b}_R \times \mathbf{g}_{31}^{(k)}. \quad (12)$$

We know that curb's front face is vertical, i.e. parallel to camera's image plane. Therefore, we find $\mathbf{p}_{32}^{(k)}$ and $\mathbf{p}_{35}^{(k)}$ by intersecting $\mathbf{g}_{32}^{(k)}$ with the two vertical lines $\mathbf{l}_{L_V}^{(k)}$ and $\mathbf{l}_{R_V}^{(k)}$ from the figure which pass through $\mathbf{p}_{31}^{(k)}$ and $\mathbf{p}_{34}^{(k)}$, i.e.

$$\mathbf{p}_{32}^{(k)} = \mathbf{l}_{L_V}^{(k)} \times \mathbf{g}_{32}^{(k)} \quad \text{and} \quad \mathbf{p}_{35}^{(k)} = \mathbf{l}_{R_V}^{(k)} \times \mathbf{g}_{32}^{(k)}. \quad (13)$$

In Fig. 10 these control points are depicted by magenta crosses.

Camera's principal point \mathbf{c} is the vanishing point (center of perspective), where the projections in the image of all \mathbb{R}^3 lines parallel to z_C converge. Hence, the lines $\mathbf{l}_{L_h}^{(k)}$ and $\mathbf{l}_{R_h}^{(k)}$ in Fig. 10a that constitute the projections of the intersections of curb's top face and the planes $\Pi_{L,R}$ will pass through it and we can estimate the positions of the last two target control points $\mathbf{p}_{33}^{(k)}$ and $\mathbf{p}_{36}^{(k)}$ as follows

$$\mathbf{l}_{L_h}^{(k)} = \mathbf{c} \times \mathbf{p}_{32}^{(k)} \quad \text{and} \quad \mathbf{l}_{R_h}^{(k)} = \mathbf{c} \times \mathbf{p}_{35}^{(k)}, \quad (14)$$

$$\mathbf{p}_{33}^{(k)} = \mathbf{l}_{L_h}^{(k)} \times \mathbf{g}_{33}^{(k)} \quad \text{and} \quad \mathbf{p}_{36}^{(k)} = \mathbf{l}_{R_h}^{(k)} \times \mathbf{g}_{33}^{(k)}. \quad (15)$$

³We will call them *target control points*.

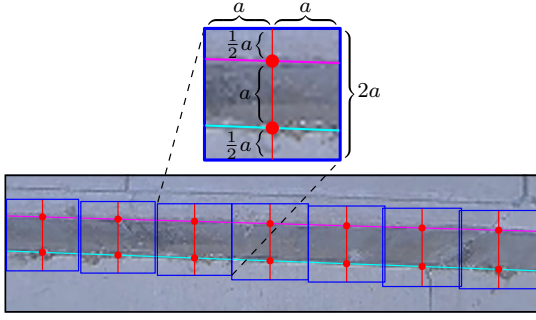


Fig. 12. Sampling square uniformly distributed windows along the frontal face projection of a curb template in the IPcM image to extract HOG features vector.

They are depicted by the green crosses in the figure.

After we have derived the equations of all control points in the image, we can define the objective function, which we are going to optimize in order to fit the curb template's projection in the image to the lines of k -th triplet from \mathcal{G}_3 . First, let the distance $r_{3m}^{(k)}$ between two corresponding control points in the image be the L^2 -norm for their difference, i.e.

$$r_{3m}^{(k)}(\hat{\mathbf{x}}) = \left\| \mathbf{p}_{3m}^{(k)} - \hat{\mathbf{p}}_{3m}(\hat{\mathbf{x}}) \right\|_2, \quad (16)$$

where $m = 1 \dots 6$. Then, we construct the error vector

$$\mathbf{r}_3^{(k)}(\hat{\mathbf{x}}) = \left[r_{31}^{(k)}, r_{32}^{(k)}, \dots, r_{36}^{(k)} \right]^\top \quad (17)$$

and the curb template's parameters that produce the best fit to the k -th triplet are determined as follows

$$\begin{aligned} \hat{\mathbf{x}}_3^{(k)} &= \arg \min_{\hat{\mathbf{x}}} \alpha_D \left\| \mathbf{r}_3^{(k)}(\hat{\mathbf{x}}) \right\|_2 \\ &\equiv \arg \min_{\hat{\mathbf{x}}} \alpha_D \sum_{m=1}^6 \left[r_{3m}^{(k)}(\hat{\mathbf{x}}) \right]^2, \end{aligned} \quad (18)$$

where $\alpha_D = \frac{\hat{D}_U}{D_{\max}}$ is a normalization term, which regularizes the dependence between the template's re-projection error in the image and the distance \hat{D}_U . As the minimization of L^2 -norm of a vector is equivalent to minimizing the sum of its squared elements and we have closed form differentiable solution for $\mathbf{r}_3^{(k)}$, we can incorporate Levenberg-Marquardt optimization algorithm [28], [29]. Fig. 10 illustrates an example of a successfully optimized (fitted) curb template.

After fitting the curb template to all the line tuples in \mathcal{G}_2 and \mathcal{G}_3 , we build the curb candidates set

$$\hat{\mathcal{X}} = \left\{ \hat{\mathbf{x}}_2^{(1)}, \hat{\mathbf{x}}_2^{(2)}, \dots, \hat{\mathbf{x}}_2^{(N_{\mathcal{G}_2})}, \hat{\mathbf{x}}_3^{(1)}, \hat{\mathbf{x}}_3^{(2)}, \dots, \hat{\mathbf{x}}_3^{(N_{\mathcal{G}_3})} \right\}. \quad (19)$$

The next task is rejecting the outliers from $\hat{\mathcal{X}}$. The first level of filtering is described in the next section, where curb candidates are rejected based on their appearance in the image at time t . Afterwards, among the "survivals" only the ones, whose parameters follow the prediction, based on temporal analysis of the previous frames, are selected. This procedure is described in Section III-F.

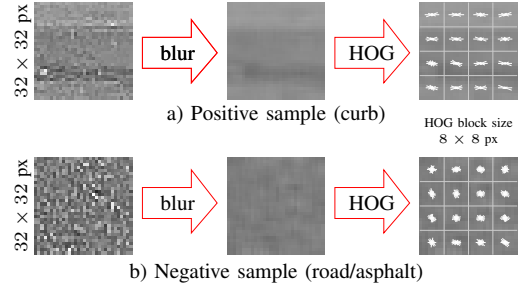


Fig. 13. Classification samples and their HOG representations.

3) *Appearance-based curb candidates filtering*: It is very unlikely that a trustworthy curb detection could be accomplished by employing only the straight curb's edges from the image, since they are not informative enough. I.e. relying only on the curb's geometry won't bring down the entropy to levels that the algorithm can reliably discriminate between curb and non-curb shapes. At this stage we try to reject the outliers in $\hat{\mathcal{X}}$ by exploiting curb's appearance in the image. Thus, we have built an object detector based on a *Support Vector Machine* (SVM) and *Histograms of Oriented Gradients* (HOG) features [30].

Curbs occupy areas in the image, which have the shape of thin, mostly horizontal, stripes. They are characterized by a couple of distinctive transitions of pixels' brightness along the vertical axis, caused by the differences in reflecting/scattering properties of the individual curb's surfaces. In the case of curb detection in images the HOG is a suitable descriptor, because it accounts the direction of that transitions and a machine learning algorithm can be trained to discriminate among curb and non-curb image patterns. Moreover, HOG is also popular with its computational efficiency.

We extract the HOG features from the IPcM image, because the vertical size of the curb is invariant to D_U . Fig. 12 illustrates our sampling approach. Seven uniformly distributed square windows are sampled along curb template frontal face. Only the two frontal curb edges are needed to accomplish that operation. The size of the patches is determined by the distance between the two edge lines in vertical direction (on the figure, depicted by a). Each window is scaled down to a fixed size of 32×32 px – Fig. 13. In order to eliminate the influence of the high-frequency components, we apply a smoothing Gaussian filter to the windows before calculating their HOG features. In the end, the pixel information of each of them is converted to 288-dimensional HOG feature vector, which is fed to the pre-trained linear SVM classifier [31]. Fig. 13 illustrates examples of a positive (a) and a negative (b) windows. Even an unexperienced human eye can easily notice the considerable difference between the gradient histograms of the two samples.

We define our classification problem in a *Multiple Instance Learning* manner. The set of the seven feature vectors sampled from the same curb template from $\hat{\mathcal{X}}$ form a bag of instances \mathcal{B} . We define two types of bags – positive (\mathcal{B}^+) and negative (\mathcal{B}^-). If the majority of the instances in a bags are positive, then that bag is considered positive. And respectively, if the

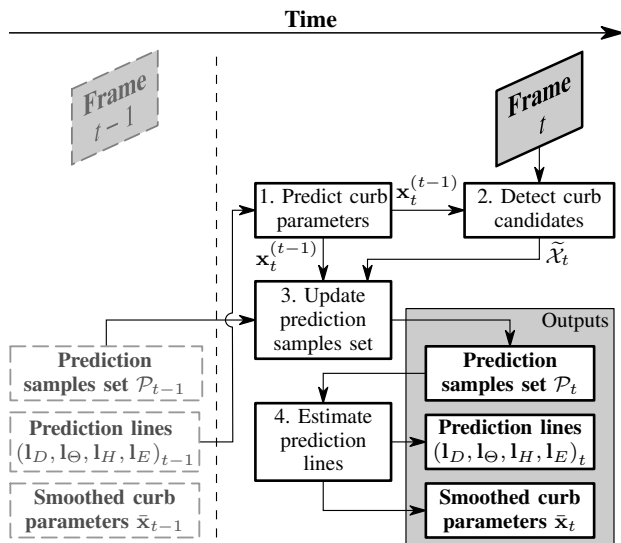


Fig. 14. Flow diagram of the “Curb tracking” mode.

majority of the instances are negative, then the bag is also negative. Thus, our classifier will be more robust against outliers in the bags. In other words, instances, which represent a curb, but are sampled from a regions which contain vertical cracks or joints, is expected to be placed far from the other instances of the same class in the feature space. The curb candidates from $\tilde{\mathcal{X}}$ that are approved by this classification procedure form the final in-frame curb candidates set

$$\tilde{\mathcal{X}} = \{\tilde{\mathbf{x}}_1, \tilde{\mathbf{x}}_2, \dots, \tilde{\mathbf{x}}_N\}, \quad (20)$$

where $N \leq N_{\mathcal{G}_2} + N_{\mathcal{G}_3}$ is its size. Note that if $\tilde{\mathcal{X}} = \emptyset$, the curb detection is unsuccessful and thus, we accept that such frames does not contain curbs.

E. Tracking the curb through the time

Here we present our scheme for curb tracking in the time domain at frame-to-frame basis. The reasoning here is based on the assumption that curb false candidates are result of faulty lines detection that occur, because of the noisy output from the Canny operator and false positive classification by the HOG+SVM. To the first reason contributes the significant local contrast in the small details produced by the HDR camera we use. Therefore, we can assume that those faulty detections don’t follow a predictable and smooth pattern in the time domain.

We have prior knowledge regarding the nature of curb parameters evolution (1). Namely, we know that H_U and E_U are constants and D_U and Θ_U evolve smoothly over time, since the vehicle is a physical object whose motion is continuous function of time. Thus, we model them as autoregressive processes.

The temporal filtering we apply has two alternating modes:

- *Collecting initial prediction set*
- *Curb tracking*

During the first one, the successful curb candidates set $\tilde{\mathcal{X}}_t$ of the current frame at time t is appended to a finite length

buffer \mathcal{C} . When the critical minimum for tracking is reached (5 consecutive successful frames), curb parameters prediction lines are estimated and then the mode is switched to “Curb tracking”. I.e. $\mathcal{C} = \{\tilde{\mathcal{X}}_t, \tilde{\mathcal{X}}_{t-1} \dots \tilde{\mathcal{X}}_{t-4}\}$, where $\tilde{\mathcal{X}}_{t-n} \neq \emptyset$: $n = 0, \dots, 4$.

The prediction lines are used to predict the future system state and to smooth the measurements of the current state, by taking into account the previous system states. We assume that within a short span of time (for example 7 frames) the evolution curves of the curb parameters have mainly linear character. Each curb parameter has its individual prediction line $-(l_D, l_\Theta, l_H, l_E)_t$. Their parameters are estimated by linear regression applied to the corresponding elements of the curb candidate vectors $\tilde{\mathbf{x}}$ of every possible combination of 5 candidates $(\tilde{\mathbf{x}}_i^{(t)}, \tilde{\mathbf{x}}_j^{(t-1)}, \tilde{\mathbf{x}}_k^{(t-2)}, \tilde{\mathbf{x}}_l^{(t-3)}, \tilde{\mathbf{x}}_m^{(t-4)})$: $\tilde{\mathbf{x}}_i^{(t-n)} \in \tilde{\mathcal{X}}_{t-n}$, $n = 0, \dots, 4$, $i = 1, \dots, N^{(t)}$, $j = 1, \dots, N^{(t-1)}$, $k = 1, \dots, N^{(t-2)}$, $l = 1, \dots, N^{(t-3)}$, $m = 1, \dots, N^{(t-4)}$, sampled from \mathcal{C} . The combination, which has minimal fitting error, is chosen to be the prediction samples set \mathcal{P}_t at time t .

Now, let’s assume that system processing mode at the current time t is “Curb tracking”. The flow chart diagram from Fig. 14, presents it graphically. The first operation is predicting system state at time t

$$\tilde{\mathbf{x}}_t = [D_U^{(t)}, \Theta_U^{(t)}, H_U^{(t)}, E_U^{(t)}]^\top, \quad (21)$$

based only on the information from the previous frames $-(l_D, l_\Theta, l_H, l_E)_{t-1}$. This gives us information for the approximate position of the curb in the current frame and length and position of CSR can be updated accordingly before detecting the curb candidates in the second operation.

Then the current frame’s prediction set \mathcal{P}_t is obtained by selecting the closest to the prediction $\tilde{\mathbf{x}}_t$ curb candidate from $\tilde{\mathcal{X}}_t$ and appending it to \mathcal{P}_{t-1} . The last step is estimating the prediction lines set for the current frame $(l_D, l_\Theta, l_H, l_E)_t$ and the smoothed (filtered) version of the curb state $\tilde{\mathbf{x}}_t$, which supposedly contains much less noise than the individual measurements.

IV. EXPERIMENTAL RESULTS

A. Video dataset

We have collected a dataset, which consists of 11 videos captured with a monocular forward-view fisheye HDR camera in typical forward perpendicular parking situations during the bright part of the day in a natural lighting environment – Table I. All but one videos contain a single sidewalk curb. Only video sequence 8 has two perpendicular curbs presented in the CCD. Two distinct weather conditions were presented at the time of data collection – clear/sunny, which is characterized with sharp deep shadows and bright highlights creating unreal edges in the image, and shadow (overcast), which is characterized with soft shadows that smoothly grade to highlights. In sequences 2, 3, 8, 11 the front bottom curb edge is fully or partially obstructed by tree leaves and different kind of debris. Camera frame rate is approximately 21 fps and the original resolution is 1920×1080 pixels.

TABLE I
DATASET DETAILS

| Vid. seq. # | Curb height (cm) | Curb depth (cm) | Weather conditions | Curb/road physical properties | Frames count [‡] |
|-------------|------------------|-----------------|--------------------|-------------------------------|---------------------------|
| 1 | 11.1 | 20.6 | Clear | Co./As.* | 702/374 |
| 2 | 13.3 | 20.6 | Clear | Co./As.* | 665/344 |
| 3 | 10.6 | 20.6 | Clear | Co./As.* | 626/378 |
| 4 | 16.2 | 15.9 | Shadow | Co./As.* | 519/332 |
| 5 | 14.6 | 16.4 | Shadow | Co./As.* | 497/321 |
| 6 | 10.5 | 20.6 | Clear | Co./As.* | 580/345 |
| 7 | 10.8 | 20.3 | Shadow | Co./As.* | 545/318 |
| 8 | 9.8 | 21.6 | Shadow | Co./As.* | 521/341 |
| 9 | 11.4 | 20.8 | Shadow | Pa./St. [†] | 486/291 |
| 10 | 9.8 | 20.3 | Shadow | Pa./St. [†] | 412/308 |
| 11 | 13.7 | 20.8 | Clear | Co./As.* | 555/360 |

* Concrete/Asphalt

[†] Painted/Strained

[‡] Total number of frames in the sequence/Number of the frames with curb presented in the CCD

The distance D_U ground truth (GT) data is collected by the means of a point-wise LIDAR sensor. The height H_U and the depth E_U of the curb are measured manually with a ruler. No direct ground truth measurements are made of curb's rotation angle Θ_U . It has been implicitly estimated through manually fitted template to the curb appearances in the video frames. The maximum labeling distance is 5 m.

B. Performance

Our curb detector has been implemented as a Python-language program. Most of the image processing procedures are accomplished by open source libraries, such as *OpenCV* [32] and *LibLinear* [31]. The overall processing performance is evaluated at average 11 – 12 fps for Full HD images and by considering the fact that no graphical processor (GPU) optimization is used. In other words, there are opportunities for further significant improvements of the execution speed.

Figures 15 and 16 present a visual example of our system's processing procedures in real situation. More specifically, the time of execution t equals to frame number 223 of video sequence 4 from our dataset. The vehicle is approaching a side-walk curb and the system has already collected sufficient temporal data and the current working mode is "Curb tracking". Both figures correspond to the intra-frame (spatial) and inter-frame (temporal) curb detection, described in Sections III-E and III-F, respectively.

As described in Fig. 14, first of all the system predicts curb parameters $\hat{\mathbf{x}}_t$ at time t . That process is visualized in Fig. 16, where the prediction lines from frame $t - 1$ are used to extrapolate all elements of the vector $\hat{\mathbf{x}}_t$ (21) independently. Knowing the approximate location of the curb, the system determines CSR's size and position, such that curb's frontal face is centered in CSR IPcM remapped region (Fig. 15a and b). The next step is extracting the straight lines from it (Fig. 15c) supposing that most of them are going to represent curb edges. The lines are inversely transformed from IPcM remapped image to original space and the lines set \mathcal{L}_t is constructed (Fig. 15d). As can be observed in the figure, the system has detected 4 lines, 3 of which represent the three

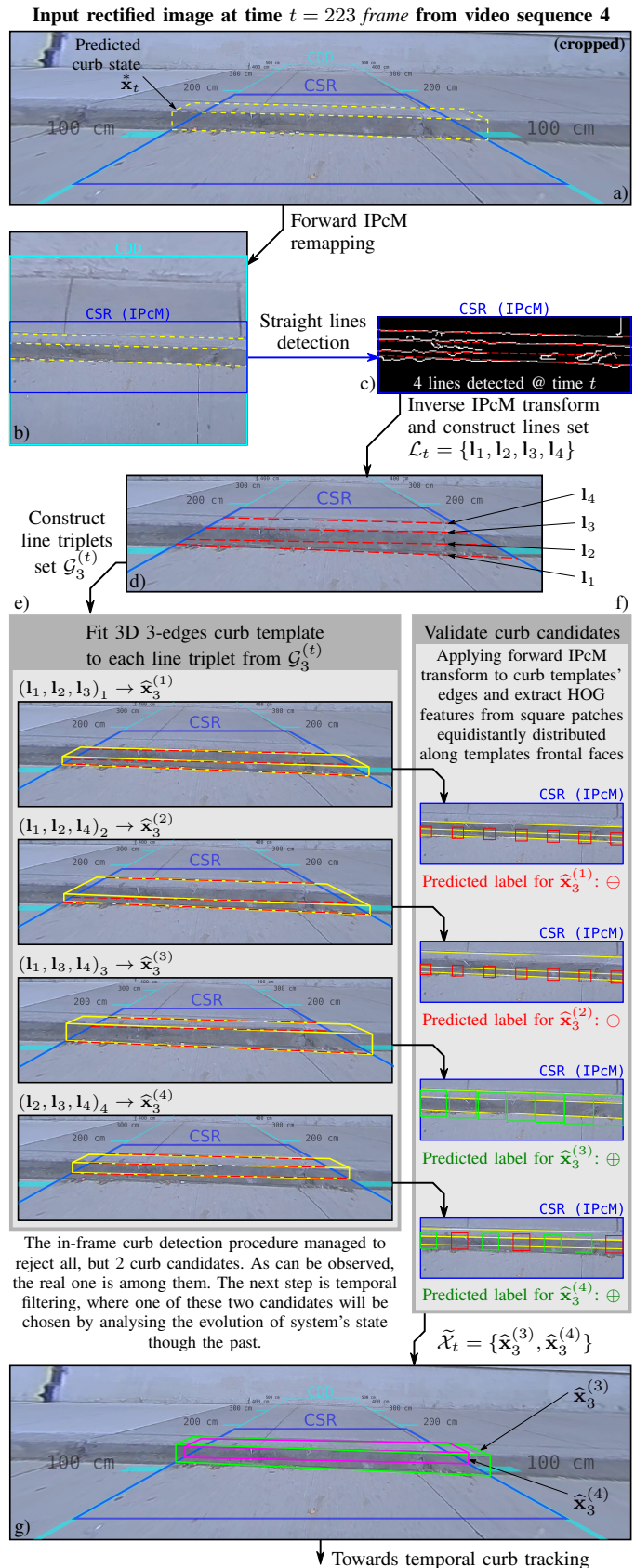


Fig. 15. Demonstration of system's spatial domain processing procedure for detecting 3-edges curb candidates in frame #223 from video sequence 4 of our dataset.

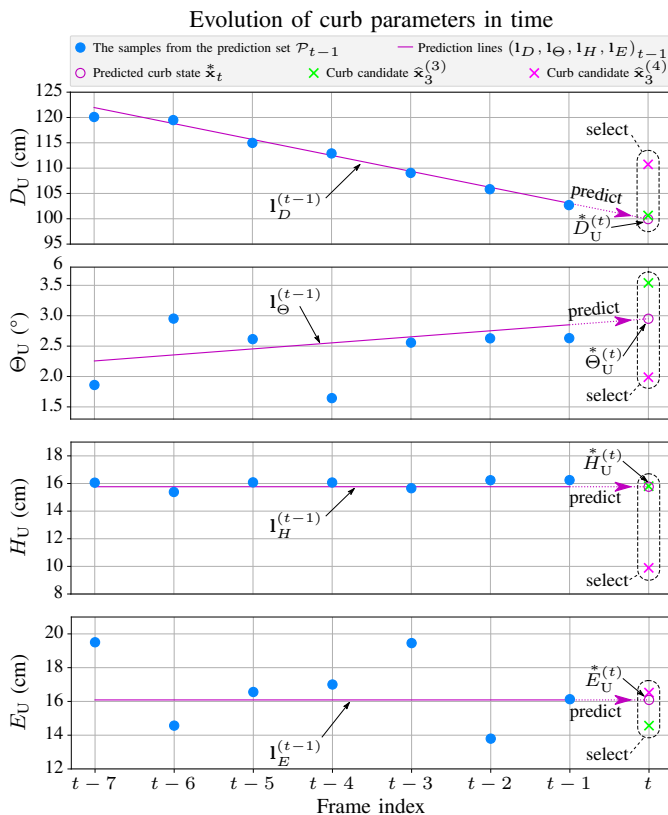


Fig. 16. Demonstration of system’s temporal domain processing procedure for detecting 3-edges curb candidates in frame 223 of video sequence 4.

curb edges – I_1 , I_3 and I_4 . The line I_2 is an outlier, which is going to be rejected during the following processing stages. Next, the set $\mathcal{G}_3^{(t)}$ is constructed from all possible combinations of \mathcal{L}_t line triplets⁴. In the current example they are four (Fig. 15e). Into each of them 3D curb templates are fitted and their state vectors $\hat{\mathbf{x}}_3^{(i)}$, $i = 1 \dots 4$ constitute 3-lines curb candidates set $\hat{\mathcal{X}}_t$. Three of these four candidates are false – caused by I_2 . In the following validation procedure the seven uniformly distributed square windows are sampled along each template’s frontal face. HOG features are extracted from them and grouped into bags: $\mathcal{B}_1^{(t)}$, $\mathcal{B}_2^{(t)}$, $\mathcal{B}_3^{(t)}$, $\mathcal{B}_4^{(t)}$. Each bag contains the HOG feature vectors sampled from the same template. In Fig. 15f with green and red colors are depicted the positive and negative labels, respectively, assigned by the classifier. From the figure, it is obvious that bags $\mathcal{B}_1^{(t)}$, $\mathcal{B}_2^{(t)}$ and $\mathcal{B}_3^{(t)}$ have very convincing labels, whereas the classifier is “uncertain” about $\mathcal{B}_4^{(t)}$ and the assigned label is falsely positive, since the majority of bags members have positive labels. At the end of this procedure, the in-frame curb candidates set $\tilde{\mathcal{X}}_t$ consists of two members (Fig. 15g), which are going to be a subject to temporal filtering – Fig. 16.

Among all curb candidates in $\tilde{\mathcal{X}}_t$, the one closest of the prediction $\hat{\mathbf{x}}_t$ is chosen. As curb parameters are heterogeneous⁵, the system compares each one independently, in application-

⁴ Current example is related to 3-edges curb candidates detection for brevity. The procedure for 2-edges ones (\mathcal{G}_2) is similar and simpler, as curb candidates depth estimations are skipped.

⁵Their nature, magnitude and range are different.

TABLE II
CURB DETECTION LEAVE-ONE-VIDEO-OUT CLASSIFICATION RATE.

| Video sequence # | Accuracy | F_1 score |
|------------------|--------------|--------------|
| 1 | 99.7% | 0.997 |
| 2 | 99.1% | 0.989 |
| 3 | 93.6% | 0.926 |
| 4 | 97.4% | 0.986 |
| 5 | 97.4% | 0.819 |
| 6 | 83.9% | 0.901 |
| 7 | 96.2% | 0.974 |
| 8 | 81.7% | 0.871 |
| 9 | 90.5% | 0.940 |
| 10 | 91.6% | 0.956 |
| 11 | 81.7% | 0.798 |
| Average: | 91.4% | 0.923 |

wise importance ascending order⁶. In Fig. 16 we can see that $\hat{\mathbf{x}}_3^{(3)}$ is much closer to $\hat{D}_U^{(t)}$, than $\hat{\mathbf{x}}_3^{(4)}$ is. Moreover, $\hat{\mathbf{x}}_3^{(4)}$ is undeniable outlier, since the distance to $\hat{D}_U^{(t)}$ is way longer than the width of a 99% confidence interval, which proves that it has not been drawn from the same distribution as the prediction samples in \mathcal{P}_{t-1} and $\hat{\mathbf{x}}_3^{(3)}$. Even if we propagate the temporal analysis deeper to the next most important parameter – H_U , we can clearly see the $\hat{\mathbf{x}}_3^{(3)}$ almost coincides with $H_U^{(t)}$ and $\hat{\mathbf{x}}_3^{(4)}$ is quite far away. The situation with the rotation angle Θ_U is similar. The only exclusion is the observation for the curb’s depth E_U (Fig 16). $\hat{\mathbf{x}}_3^{(4)}$ is evidently closet to the prediction $\hat{E}_U^{(t)}$ than $\hat{\mathbf{x}}_3^{(3)}$ is, but the system cannot make strong inference, because both candidates are deviated less than $\pm 3\sigma_{E_U}$ from the prediction line. We can conclude that the only inlier is $\hat{\mathbf{x}}_3^{(3)}$ at time t . Then, \mathcal{P}_t is created by updating \mathcal{P}_{t-1} though appending $\hat{\mathbf{x}}_3^{(3)}$ and removing the “oldest” sample in order to preserve its fixed length.

Demonstration videos of our system can be found in [33].

C. Detection rate evaluation

The initial operation on every newly captured camera frame is *curb candidates detection*. In this section we evaluate the capabilities of our system to correctly detect curbs in the image, before estimating its parameters. The classification accuracy (ACC) and F_1 score are measured by incorporating *Leave-one-video-out* cross validation technique for each individual video sequence in our dataset. The results are summarized in Table II.

ACC is calculated according to the following equation

$$\text{ACC} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}, \quad (22)$$

where TP, TN, FP and FN are true positives, true negatives, false positives and false negatives, respectively. The F_1 score is given by

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}, \quad (23)$$

⁶ The main purpose of the current system is to avoid collisions between the vehicle and obstacles, such as curbs. Therefore, we can assign figurative importance level to each of curb parameters, from application point of view. We consider D_U as the most important parameter, because if the curb is far enough, we know that a collision won’t occur, regardless the values of the other curb parameters. If we cannot rely on D_U to infer whether or not the vehicle will collide, we should take into account H_U . Therefore, it is considered as the second most important parameter, etc.

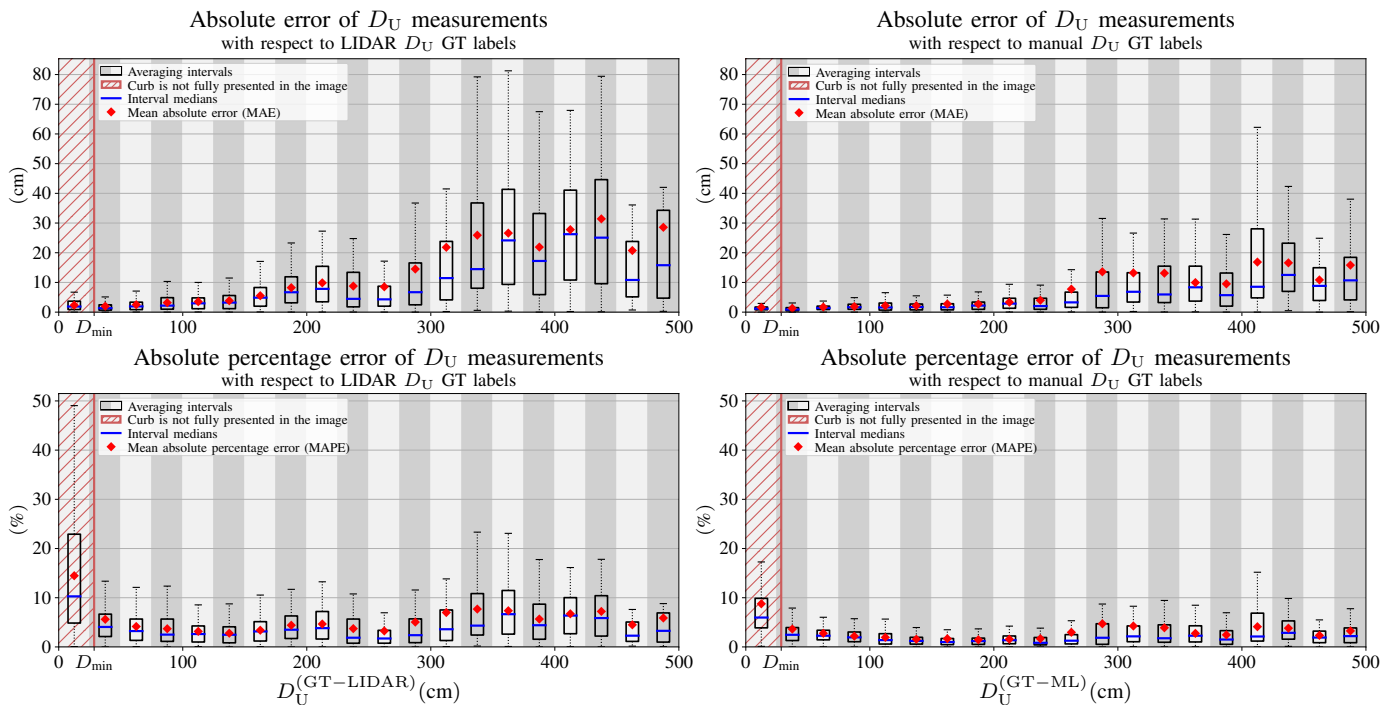


Fig. 17. Absolute (*top*) and relative (*bottom*) curb distance D_U measurement errors with respect to LIDAR ground truth (GT) data (*left*) and manual labels GT (*right*).

where

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}, \text{ recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \quad (24)$$

As can be observed in Table II, the majority of the videos have accuracy greater than 90% and the average is about 91.40%. Also F_1 scores tend to maintain very high values as well.

D. Curb parameters estimation accuracy

In this section we evaluate system’s capabilities as a curb parameters measuring device. We use two metrics: an absolute – *Mean Absolute Error* (MAE) and a relative – *Mean Absolute Percentage Error* (MAPE). We show how these errors change with respect to the GT distance D_U . Therefore, we divide the length of CDD in averaging intervals of 25 cm each. The graphs shown here summarize the results of processing all video sequences from the dataset.

Fig. 17 illustrates box plots of the absolute error of \hat{D}_U estimates for our dataset. D_U is the only parameter, which has two sources of GT data – a point-wise LIDAR and manual labels. Thus, the figure has information for both of them. The LIDAR labels should be considered as the more accurate baseline measurements, because they are realized through an independent device, not by the camera. Hence, the error with respect to them is greater than the one with respect to the manual labels. Expectedly, the MAE errors and variances are proportional to the distance between the vehicle and the curb. On the other hand, MAPE errors tend to maintain relatively uniform character through the entire length of CDD – below 8-9%. The only exception is the closest averaging interval – $D_U \in [0, D_{\min}]$, where the curb is partially presented in the camera frame and the system approximates its parameters

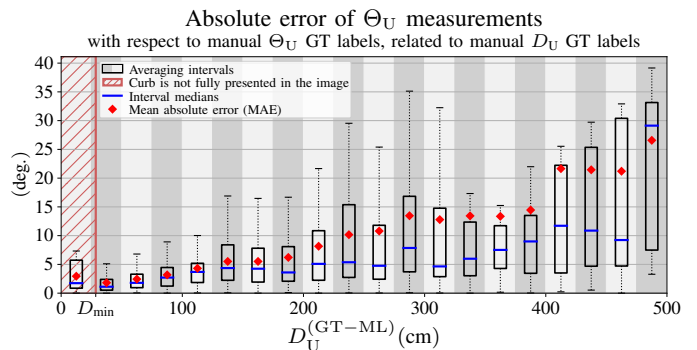


Fig. 18. Absolute curb rotation angle Θ_U measurement errors with respect to the manual labels GT.

using a reduced set of edges (one or two). This can be considered as completely anticipated result.

Fig. 18 illustrates the absolute errors of rotation angle measurements with respect to the manually labeled curb parameters. Similar to the previous parameter’s measurement statistics, the MAE and error variance of Θ_U are proportional to D_U . Relatively high errors (especially at longer distances) can be explained by the nature of perspective projection and relative camera and curb positions and orientations. As the camera is relatively close to the road plane, its optical axis is parallel to it, its field of view is very large (fisheye), large changes in curb rotation are going to have minimal effect in its appearance in the image.

In Fig. 19 and Fig. 20 the absolute measurement errors of H_U and E_U can be seen. The characters of both MAE errors seem to be relatively uniform, unlike the one of D_U and Θ_U . This demonstrates that these two parameters are less (or not)

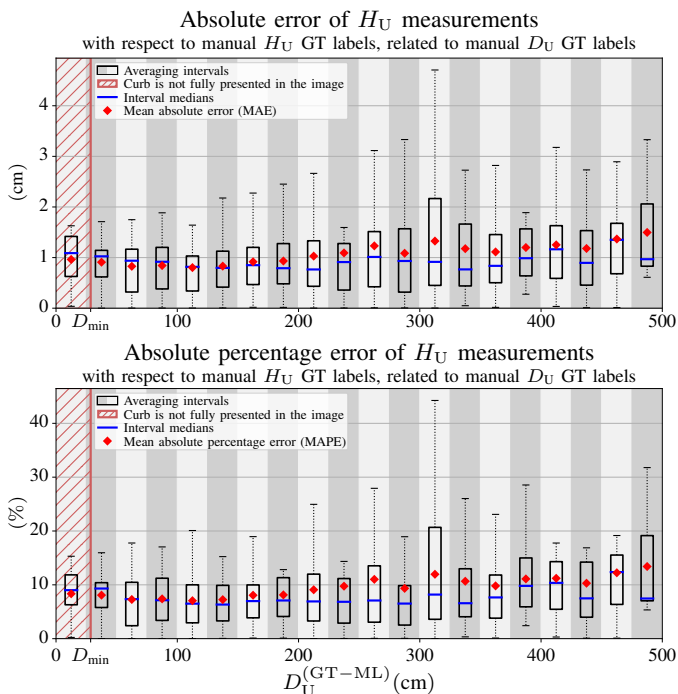


Fig. 19. Absolute (*top*) and relative (*bottom*) curb height H_U measurement errors with respect to manual labels GT.

dependent on the measurement distance D_U .

V. CONCLUSION

In this paper we presented a system for vision-based curb detection and its parameters estimation. We managed to achieve high detection and curb parameters measurement accuracies just by using a single fisheye camera and CPU computations. Our algorithm is capable of successfully detecting and tracking curbs at a distance of more than 4 m with mean absolute error less than 9% for the curb distance estimates in the *Curb Detection Domain* and not more than 1.5 cm mean deviation for the curb height estimates. Robust tracking and high detection rate are accomplished by implementing online temporal analysis.

Here we would like also to discuss the aspects of our future activities, which should be considered as a part of the efforts to improve our system capabilities. One of the main shortcomings is the amount of diversity in the dataset currently used. Along with the already available sunny and cloudy daylight conditions, there should be considered also the situations of rain, snow, twilight, night, etc. Another parameter, which should be improved, is the variability of curb types presented in the data. For example, parking curbs or curbs, which consist of rounded edges, may cause problems due to their specific geometry and it is important to test the system performance against it. As our curb detection system relies on detecting edges in the image, it is also relevant testing the system on various types of the road and sidewalk. For instance, introducing types, which include tiles or pavements. That will introduce edges, which may confuse the detector. All these consideration are going to be taken into account when collecting the next datasets.

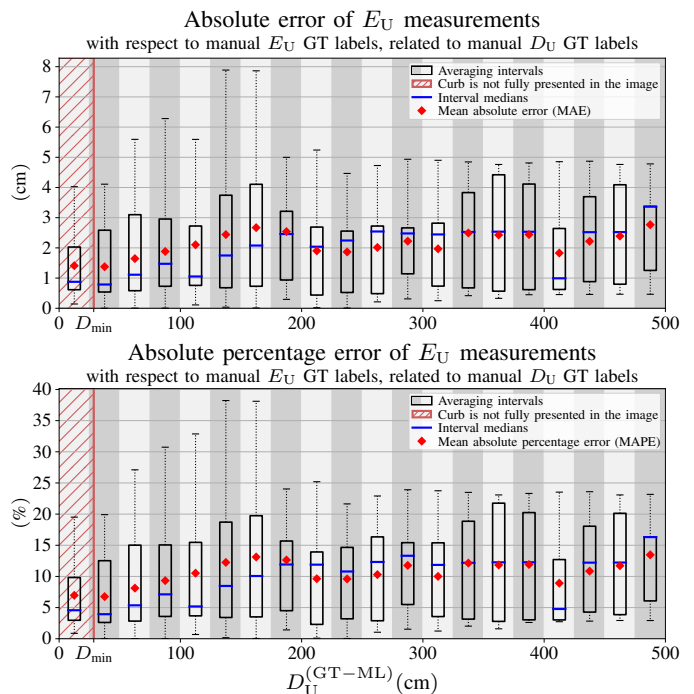


Fig. 20. Absolute (*top*) and relative (*bottom*) curb depth E_U measurement errors with respect to manual labels GT.

The current system uses a single camera as an input device and we have proved that it can perform considerably well. Other types of sensors are often used as well. A common practice is the employment of stereo-pairs, which provide depth information as an additional cue. Another source of 3D data is the LIDARs. Their main drawback is the cost, as it is significantly higher than the one of the cameras. On the other hand, they are active sensors, thus they do not rely on external light sources, such as the sun, street lamps or head lights. Having multiple sources of information requires an appropriate methods to deal with them. A common approach towards making decision in that cases is by fusing those streams of information. In [34] is presented a technique for multi-modal data fusion, which can serve as a method to combine the visual and 3D data for more reliable curb detection.

APPENDIX

INVERSE PERSPECTIVE-COMPRESSING MAPPING FOR CURB'S SCALE-INVARIANT EDGE DETECTION

In this section we are presenting the derivation of our method for *Inverse Perspective-compressing Mapping* (IPcM), whose purpose and application are described in Section III-E1 and illustrated in Fig. 9b. Its objective is minimizing (or even completely eliminating) the dependence of curb's spatial sampling rate R_U from the distance D_U , through its entire definition interval, i.e. $R_U(D_U) \xrightarrow{\text{remapping}} \hat{R}_U(D_U) = \text{const} : D_U \in [D_{\min}, D_{\max}]$. The condition to avoid the interpolation is

$$\hat{R}_U \leq \min [R_U(D_U)] : D_U \in [D_{\min}, D_{\max}]. \quad (25)$$

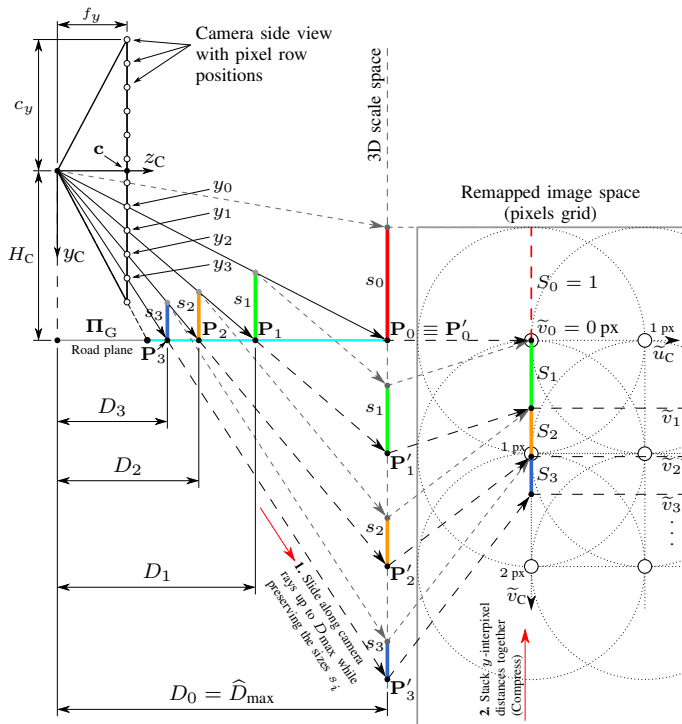


Fig. 21. Curb scale-invariant image remapping: Forward transform – (vertical) y -interpixel distances scaling.

In order to preserve the largest amount of details in the image we take into account the equation (5) and choose the highest possible value $\hat{R}_U = R_U(D_{\max})$.

The derivation of the remapping functions presented below consists of two logical parts – forward and backward (inverse) transforms

$$\underbrace{(u, v)}_{\text{original image space}} \xrightarrow[\text{transform}]{f - \text{forward}} \underbrace{(\tilde{u}, \tilde{v})}_{\text{remapped image space}} \xrightarrow[\text{transform}]{\hat{f} - \text{inverse}} \underbrace{(\hat{u}, \hat{v})}_{\text{(pseudo) original image space}} \quad (26)$$

A. Forward transform (remapping)

This transformation defines the mapping of the image point (pixel) coordinates from the original image to the new one. It is important to note that we consider pixels as sampling locations, i.e. points from the camera image plane with no designated areas, contrary to the common understanding that they are small regions. Thus, the remapping procedure in essence is scaling of the distances between the pixels in an appropriate way. Moreover, the camera’s pixels grid is regular and square, i.e. all the inter-pixel distances are equal.

Fig. 21 depicts the theoretical foundations of our approach. Every pixel from image plane defines a ray which originates in O_C – camera’s coordinate frame origin, and passes through its location on the image plane. We will refer to these rays as *sampling rays* and their intersection points with the road plane – *sampling points*. Furthermore, due to our system’s geometrical setup, all sampling points corresponding to the pixels sharing the same row from the image will have the identical z_C coordinates, i.e. they will be equidistant w.r.t. the camera’s coordinate plane $x_C y_C$. Thus, by “a sampling ray”

or “a sampling point” we will refer to all the rays and points corresponding to the pixels from the same image row.

Let y_i be a positive vertical coordinate of the pixels located on the v_i -th image row, i.e. $v_i > c_y$, where $v_i = y_i + c_y$ and i is row index. Hence, y_i will correspond to a sampling ray which intersects the road plane in the sampling point $\mathbf{P}_i \in \mathbb{R}^3$ at a distance D_i that can be calculated by the equation (4). Let $D = \{D_i\}$ be a set, whose members are all the possible distances D_i defined by the camera. Also we define the longest sampling distance \hat{D}_{\max} , where $\hat{D}_{\max} = \sup \{D_j \in D : D_j \leq D_{\max}\}$. For the sake of simplicity we set the index $i = 0$ for the image pixel row which corresponds to the distance \hat{D}_{\max} , i.e. $y_0 \leftrightarrow \mathbf{P}_0 \leftrightarrow D_0 = \hat{D}_{\max}$.

Let s_i be the distance between the two sampling rays corresponding to the pixel rows y_i and y_{i-1} along vertical direction, measured at the sampling point \mathbf{P}_i (Fig. 21). Then we get

$$s_i = \frac{D_i}{f_y} = \frac{1}{R_U(D_i)}, \quad (27)$$

which shows that $s_i \propto D_i$. Hence, normalizing R_U for all sampling positions D_i , which is our goal, is equivalent to normalizing s_i and we do it by using the distance s_0 at D_0 as a reference and define the scaling factors S_i , such that

$$S_i = \frac{s_i}{s_0} = \frac{D_i}{D_0} = \frac{y_0}{y_i}. \quad (28)$$

On Fig. 21 we can see the geometrical interpretation of the scaling process. Let’s imagine that a curb with height H_U slides along the axis z_C on the ground. Hence, the height of its projection in the image is going to change inversely proportional to the distance D_U (as we have already seen on Fig. 9a), i.e. the farther the curb is, the smaller its projection in the camera will be. We want to resolve that problem by making curbs size in the image fixed for all the distances $D_U \in [D_{\min}, D_{\max}]$. Our approach to achieve it is virtually “dragging” the curb along the sampling ray corresponding to the point \mathbf{P}_i from its location at the distance $D_U = D_i$ to a plane which is parallel to $x_C y_C$ and passes through the point \mathbf{P}_0 . Thus, the curb basically will always be at the fixed distance $D_U = D_0$ w.r.t. the camera, thereby its size in the image will be constant, and at the same time it will still be projected in the image at the vertical position y_i which corresponds to the point of its real location D_i .

Since we do not use any of the image information outside its RoI, there is no need to spend processing time on remapping it. That’s why, we choose that the y_0 is mapped to the first row of the resulting image, i.e. $\tilde{v}_0 = f_v(y_0) = 0$ px, where f_v is the remapping function if the vertical point coordinates only. Based on Fig. 21 for f_v we can write

$$\tilde{v}_n = f_v(y_n) = \sum_{i=1}^n S_i = \sum_{i=1}^n \frac{y_0}{y_i} = y_0 \sum_{i=1}^n \frac{1}{y_0 + i}. \quad (29)$$

Obviously, f_v has an opened form, which is inconvenient to estimate its inverse. Fortunately, the sum term has the pattern of harmonic series, with the difference that the sum is not

infinite. However, it can be presented by a difference of two harmonic series partial sums

$$f_v(y_n) \approx y_0 \left(\sum_{k=1}^{\tilde{y}_n} \frac{1}{k} - \sum_{k=1}^{\tilde{y}_0} \frac{1}{k} \right) \approx y_0 (H_{y_n} - H_{y_0}), \quad (30)$$

where

$$H_q = \ln(q) + \gamma + \frac{1}{2q} - \frac{1}{12q^2} + \frac{1}{120q^4} - \varepsilon_q \quad (31)$$

is the q -th harmonic number determined by expanding some of the terms in the *Hurwitz zeta function*, γ is the *Euler-Mascheroni constant* and $\varepsilon_q \in \left(0, \frac{1}{252q^6}\right)$. \tilde{y}_n and \tilde{y}_0 are the floored values of y_n and y_0 , respectively. In the general case, c_x and c_y – the coordinates of the camera’s principal point \mathbf{c} , are real numbers. Hence, y_n and y_0 will also be real which cannot be used as a limit of the sum operators in (30). Therefore, we floor them by taking into account that the produced error will be negligible. After the subtraction of H_{y_n} and H_{y_0} , γ is eliminated and the influence of the last 3 terms in (31) from practical point of view is imperceptible. The term $\frac{1}{2q}$ is very well approximated by $\ln\left(\frac{q}{q-\frac{1}{2}}\right)$, estimated by the *Laurent series* expansion. The estimated total approximation error is less than $2 \cdot 10^{-3}$ pixels and we get the final simplified closed form expression for f_v

$$\tilde{v} = f_v(v) = y_0 \ln \left[\frac{(v - c_y)^2}{y_0(v - c_y - \frac{1}{2})} \right] - \frac{1}{2} \quad (32)$$

Every line of the image is scaled symmetrically with the same scale factor S_i , i.e. the mapping function for the horizontal direction has the following form

$$\tilde{u} = f_u(u, v) = y_0 \frac{u - c_x}{v - c_y} + c_x \quad (33)$$

B. Inverse transform

To derive the inverse transform we just inverse equations (32) and (33) and we get

$$\begin{aligned} \hat{v} = g_v(\tilde{v}) &= \frac{m + \sqrt{m(m-2)}}{2} + c_y, \\ \hat{u} = g_u(\tilde{u}, \hat{v}) &= \frac{(\tilde{u} - c_x)(\hat{v} - c_y)}{y_0} + c_x, \end{aligned} \quad (34)$$

where

$$m = y_0 \exp \left(\frac{\tilde{v} + \frac{1}{2}}{y_0} \right). \quad (35)$$

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their valuable comments, which allowed us to improve the presentation of our paper and resolve various imperfections. This work was partially supported by General Motors Research. Stanislav Panev’s scholarship in CMU was funded by the Bulgarian-American Commission of Education Exchange “Fulbright” with a grant number 15-21-04 from June 3, 2015.

REFERENCES

- [1] G. Zhao and J. Yuan, “Curb detection and tracking using 3d-LIDAR scanner,” in *2012 19th IEEE International Conference on Image Processing*, Sep. 2012, pp. 437–440.
- [2] W. Yao, Z. Deng, and L. Zhou, “Road curb detection using 3d LIDAR and integral laser points for intelligent vehicles,” in *The 6th International Conference on Soft Computing and Intelligent Systems, and The 13th International Symposium on Advanced Intelligence Systems*, Nov. 2012, pp. 100–105.
- [3] A. Y. Hata, F. S. Osorio, and D. F. Wolf, “Robust curb detection and vehicle localization in urban environments,” in *2014 IEEE Intelligent Vehicles Symposium Proceedings*, Jun. 2014, pp. 1257–1262.
- [4] T. Chen, B. Dai, D. Liu, J. Song, and Z. Liu, “Velodyne-based curb detection up to 50 meters away,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2015, pp. 241–248.
- [5] Y. Zhang, J. Wang, X. Wang, C. Li, and L. Wang, “A real-time curb detection and tracking method for UGVs by using a 3d-LIDAR sensor,” in *2015 IEEE Conference on Control Applications (CCA)*, Sep. 2015, pp. 1020–1025.
- [6] U. Scheunert, B. Fardi, N. Mattern, G. Wanielik, and N. Keppeler, “Free space determination for parking slots using a 3d PMD sensor,” in *2007 IEEE Intelligent Vehicles Symposium*, Jun. 2007, pp. 154–159.
- [7] O. Gallo, R. Manduchi, and A. Rafii, “Robust curb and ramp detection for safe parking using the Canesta TOF camera,” in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, Jun. 2008, pp. 1–8.
- [8] J. Byun, J. Sung, M. C. Roh, and S. H. Kim, “Autonomous driving through Curb detection and tracking,” in *2011 8th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)*, Nov. 2011, pp. 273–277.
- [9] Z. Liu, D. Liu, T. Chen, and C. Wei, “Curb Detection Using 2d Range Data in a Campus Environment,” in *2013 Seventh International Conference on Image and Graphics*, Jul. 2013, pp. 291–296.
- [10] E. Pollard, J. Perez, and F. Nashashibi, “Step and curb detection for autonomous vehicles with an algebraic derivative-based approach applied on laser rangefinder data,” in *2013 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2013, pp. 684–689.
- [11] F. Oniga, S. Nedeveschi, and M. M. Meinecke, “Curb Detection Based on a Multi-Frame Persistence Map for Urban Driving Scenarios,” in *2008 11th International IEEE Conference on Intelligent Transportation Systems*, Oct. 2008, pp. 67–72.
- [12] J. Siegemund, U. Franke, and W. Frstner, “A temporal filter approach for detection and reconstruction of curbs and road surfaces based on Conditional Random Fields,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2011, pp. 637–642.
- [13] F. Oniga and S. Nedeveschi, “Curb detection for driving assistance systems: A cubic spline-based approach,” in *2011 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2011, pp. 945–950.
- [14] M. Kellner, U. Hofmann, M. E. Bouzouraa, and N. Stephan, “Multi-cue, Model-Based Detection and Mapping of Road Curb Features Using Stereo Vision,” in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, Sep. 2015, pp. 1221–1228.
- [15] C. Fernandez, D. F. Llorca, C. Stiller, and M. A. Sotelo, “Curvature-based curb detection method in urban environments using stereo and laser,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2015, pp. 579–584.
- [16] V. Haltakov, H. Belzner, and S. Ilic, “Scene understanding from a moving camera for object detection and free space estimation,” in *2012 IEEE Intelligent Vehicles Symposium*, Jun. 2012, pp. 105–110.
- [17] A. Seibert, M. Hhnel, A. Tewes, and R. Rojas, “Camera based detection and classification of soft shoulders, curbs and guardrails,” in *2013 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2013, pp. 853–858.
- [18] V. Prinnet, J. Wang, J. Lee, and D. Wettergreen, “3d road curb extraction from image sequence for automobile parking assist system,” in *2016 IEEE International Conference on Image Processing (ICIP)*, Sep. 2016, pp. 3847–3851.
- [19] P. Viola and M. Jones, “Rapid object detection using a boosted cascade of simple features,” in *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1, 2001, pp. I–511–I–518 vol.1.
- [20] J. Kannala and S. S. Brandt, “A generic camera model and calibration method for conventional, wide-angle, and fish-eye lenses,” *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 28, no. 8, pp. 1335–1340, Aug 2006.
- [21] R. O. Duda and P. E. Hart, “Use of the hough transformation to detect lines and curves in pictures,” *Commun. ACM*, vol. 15, no. 1, pp. 11–15, Jan. 1972.

- [22] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, no. 6, pp. 679–698, Jun. 1986.
- [23] I. Sobel and G. Feldman, "A 3x3 Isotropic Gradient Operator for Image Processing," 1968, never published but presented at a talk at the Stanford Artificial Project.
- [24] R. Duda and P. Hart, *Pattern Classification and Scene Analysis*. Wiley, 1973, pp. 271–272.
- [25] B. Jähne, H. Scharf, and S. Körkel, "Principles of filter design," in *B. Jähne, H. Haussecker and P. Geissler, Handbook of Computer Vision and Applications*. Academic Press, 1999, no. v. 2, pp. 125–152.
- [26] J. Prewitt, "Object enhancement and extraction," in *B.S. Lipkin and A. Rosenfeld, editors, Picture Processing and Psychopictorics*. Academic Press, 1970, ch. 1, pp. 75–150.
- [27] L. G. Roberts, *Machine Perception of Three-Dimensional Solids*, ser. Outstanding Dissertations in the Computer Sciences. Garland Publishing, New York, 1963.
- [28] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of Applied Mathematics*, vol. 2, no. 2, pp. 164–168, 1944.
- [29] D. W. Marquardt, "An algorithm for least-squares estimation of non-linear parameters," *Journal of the Society for Industrial and Applied Mathematics*, vol. 11, no. 2, pp. 431–441, 1963.
- [30] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 886–893 vol. 1.
- [31] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A library for large linear classification," *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008. [Online]. Available: <http://www.csie.ntu.edu.tw/~cjlin/liblinear>
- [32] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.
- [33] "Curb detection system demonstration videos," <https://goo.gl/NdMJFY>, accessed: 2017-04-24.
- [34] H. Liu, Y. Wu, F. Sun, B. Fang, and D. Guo, "Weakly paired multimodal fusion for object recognition," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 2, pp. 784–795, April 2018.



Stanislav Panev is currently a postdoctoral fellow at the Robotics Institute of Carnegie Mellon University. He received BSc and MSc degrees in telecommunications, as well as PhD degrees in computer science from the Technical University of Sofia, Bulgaria in 2005, 2008 and 2014, respectively. His research topics include, but are not limited to: computer vision, machine learning, computer graphics, artificial intelligence.



Francisco Vicente received a B.Sc. degree in tele communications and a M.Sc. degree in telematics from Universidad Politecnica de Cartagena, Cartagena, Spain, in 2007 and 2008, respectively. In addition, he received a M.S. degree in Robotics at Carnegie Mellon University, Pittsburgh, PA, USA. in 2015.

Since 2015, he has been a Scientific Specialist with the Robotics Institute, Carnegie Mellon University. His research interests are in the fields of computer vision and machine learning.



Fernando De la Torre received the BSc degree in telecommunications, as well as the MSc and PhD degrees in electronic engineering from the La Salle School of Engineering at Ramon Llull University, Barcelona, Spain in 1994, 1996, and 2002, respectively. He is an associate research professor in the Robotics Institute at Carnegie Mellon University. His research interests are in the fields of computer vision and machine learning. Currently, he is directing the Component Analysis Laboratory (<http://ca.cs.cmu.edu>) and the Human Sensing Laboratory (<http://humansensing.cs.cmu.edu>) at Carnegie Mellon University. He has more than 150 publications in referred journals and conferences and is an associate editor at IEEE TPAMI. He has organized and co-organized several workshops and has given tutorials at international conferences on component analysis.



Véronique Prinnet is currently a Visiting Professor at The Hebrew University of Jerusalem, Israel. She received a Ph.D. in Computer Science from INRIA, in collaboration with University Paris-11 Orsay and CNES (French National Center for Spatial Studies). In 1999, she was a post-doctoral research fellow at the Institute of Automation, Chinese Academy of Sciences (CASIA), Beijing. She subsequently hold an Associate Professor position in the same institution from 2000 to 2012. She worked as a senior researcher at General Motors Advanced Technical

Center, Israel, from 2014 to 2016. Her research interests are in data-driven approaches to image processing and computer vision.