IEEE copyright notice

Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redis- tribution to servers or lists, or to reuse any copyrighted com- ponent of this work in other works must be obtained from the IEEE. Contact: Manager, Copyrights and Permissions / IEEE Service Center / 445 Hoes Lane / P.O. Box 1331 / Piscataway, NJ 08855-1331, USA. Telephone: + Intl. 908-562-3966.

Motion Compensation for Obstacle Detection Based on Homography and Odometric Data with Virtual Camera Perspectives

Michael Miksch and Bin Yang
Chair of System Theory and Signal Processing
University of Stuttgart, Germany
{michael.miksch,
bin.yang}@lss.uni-stuttgart.de

Klaus Zimmermann
European Technology Center (EuTEC)
Sony Deutschland GmbH
D-70327 Stuttgart, Germany
klaus.zimmermann@sony.de

***Abstract*— In this paper we present a method to compensate the image motion of a monocular camera on a moving vehicle in order to detect obstacles. Due to the camera motion, the road surface induces a characteristic image motion between two camera shots. The motion of the camera is determined by the use of odometric data received from the CAN-bus, and the position and orientation of the road is continuously estimated with camera self-calibration. This all leads to a motion field which is predicted based on homography. To prevent the drawbacks of the real camera perspective, different virtual camera perspectives are presented in combination with motion compensation. Possible virtual perspectives are the bird's eye view and image rectification. In addition, a non-linear camera model is used which does not limit the range of obstacle detection to a certain distance and efficiently uses the available image information.**

I. Introduction

Video cameras are becoming more and more important in automotive applications. A variety of video-based *Advanced Driver Assistance Systems* (ADAS) are already in use. Characteristic is the use of a single video sensor for different applications. A side view camera, for example, assists the driver during a parking maneuver, is part of an *Around View Monitor*, prevents side collisions and warns the driver of the presence of obstacles in the blind spot.

In this paper, we address the detection of obstacles in the adjacent lane of the vehicle. Possible applications are the *Lane Change Assistant* or the *Blind Spot Assistant*. If the driver intends to change lanes in the presence of an obstacle, he is warned in order to prevent a collision. For both applications it is important to detect objects regardless whether it is a vehicle, motorcycle, bus or something else. We make use of a monocular camera in combination with odometric data (wheel speed and yaw rate). The basic idea is to exploit the image motion for the detection of obstacles. The road surface induces a specific image motion between two successive frames if the camera has moved. Fortunately, the image motion of static or moving objects differs from the image motion on the road. Therefore, localizing the differences in the image motion leads directly to the identification of possible threats.

One could imagine determining the image motion for each image position, resulting in a motion field. This is well known as both motion estimation and optical flow. When the motion in the image is determined, it can be compared to the image motion on the road. This idea is used in [1]–[4]. Logically, the following question arises: What is the image motion on the road? There are two possible solutions: either the motion vectors are predicted based on a known camera motion and road plane, or a model of motion vectors is estimated which fits best to the measured optical flow. The second solution is proposed in [5], [6]. However, the calculation of the optical flow is difficult for automotive applications, since the road has almost no surface structure to reliably assign corresponding image points.

Motion estimation is in general difficult and computationally expensive. Therefore, the motion between two consecutive images is often not explicitly calculated but the current image is warped using the predicted optical flow. The transformed image corresponds to the previous image for points on the road, whereas image points located on static or moving objects often significantly differ from each other. The so-called *Inverse Perspective Mapping* (IPM) is the basic idea to compensate the motion on the road. Generally, the conversion of 2D image coordinates to 3D world coordinates is ambiguous, since the depth is unknown. In the presence of a road, the mapping becomes unique. Furthermore, the obtained 3D world points can be projected onto the second camera perspective and the mapping between the previous and current images is established. If the road is assumed to be flat, the transformation can be expressed easily by a projective transformation, also called homography or collineation. The motion compensation, for example, is applied in [7]–[9] for obstacle detection.

The real perspective of the camera has the following disadvantages for motion compensation and obstacle detection: There is no prediction for image points above the horizon. The *Region of Interest* (ROI), namely the adjacent lane, is only a subset of the image and the size of an object in the image increases if the object is approaching the camera. This complicates the processing of the image. For this reason, the image is transformed to a top view or side view perspective in [10]–[12] to simplify the image processing. This kind of transformation is called *Virtual Camera Perspective* (VCP).

In this paper, we combine the motion compensation with different VCPs. We predict the motion on the road as mentioned before. The camera motion is computed based on odometric data available from the CAN-bus. The road is assumed to be flat. Both the camera motion and the road plane require the exact position and orientation of the camera, also known as the extrinsic parameters of the camera. Recently, we presented a method for the calibration of the extrinsic parameters in [13]. This method continuously

calibrates the camera parameters online. This enables a reliable operation of the system even if the orientation of the camera changes. Up until now, most of the video-based ADAS rely on an offline calibration process with a fixed extrinsic parameter set. To describe the motion on the road, we determine a homography matrix. Besides a top view perspective we also make use of image rectification, which is well known in stereo vision. Additionally, we introduce a non-linear camera model, which does not limit the range of obstacle detection to a certain distance. Furthermore, it does not replicate information like the pinhole camera model but efficiently uses the available image information.

The outline of this paper is as follows: In Sec. II we introduce the fundamentals. The estimation of the motion of the camera is described in Sec. III-A. The calculation of the motion field including motion compensation, and the different virtual camera perspectives are presented in Sec. IV. Finally, a conclusion is given in Sec. V.

II. Basics

A. Camera projection

The underlying camera model is the pinhole camera. It describes the projection of a 3D object point $\mathbf{M}_v = [X, Y, Z]^T$ onto its image point $\mathbf{m}_p = [u_p, v_p]^T$ by

$$\widetilde{\mathbf{m}}_p = \lambda \begin{bmatrix} \mathbf{m}_p \\ 1 \end{bmatrix} = \mathbf{A} \, [\mathbf{R}_e \ \mathbf{t}_e] \begin{bmatrix} \mathbf{M}_v \\ 1 \end{bmatrix} \tag{1}$$

where λ is a scalar factor for the normalization, see (4), $[\mathbf{R}_e \ \mathbf{t}_e]$ are the extrinsic parameters, and $\mathbf{A}$ is the intrinsic matrix. The intermediate step

$$\widetilde{\mathbf{m}}_c = \lambda \begin{bmatrix} \mathbf{m}_c \\ 1 \end{bmatrix} = \mathbf{M}_c = \mathbf{R}_e \, \mathbf{M}_v + \mathbf{t}_e \tag{2}$$

transforms the point $\mathbf{M}_v$ from the *vehicle coordinate system* VCS to the point $\mathbf{M}_c$ in the *camera coordinate system* CCS by an Euclidean transform. $\mathbf{R}_e$ is a rotation matrix with $\mathbf{R}_e^{-1} = \mathbf{R}_e^T$ and $\mathbf{t}_e$ is a translation vector, resulting in normalized coordinates $\mathbf{m}_c = [u_c, v_c]^T$. The intrinsic transform between the normalized and the image coordinates is defined by

$$\widetilde{\mathbf{m}}_p = \mathbf{A} \, \widetilde{\mathbf{m}}_c \quad \text{and} \quad \widetilde{\mathbf{m}}_c = \mathbf{A}^{-1} \, \widetilde{\mathbf{m}}_p. \tag{3}$$

Here, "$\sim$" represents homogeneous coordinates. The relationship between Cartesian coordinates $\mathbf{m}$ and homogeneous coordinates $\widetilde{\mathbf{m}}$ is

$$\lambda \in \Re \backslash \{0\} : \lambda \begin{bmatrix} \mathbf{m} \\ 1 \end{bmatrix} = \widetilde{\mathbf{m}}. \tag{4}$$

Therefore, homogeneous coordinates can be scaled arbitrarily while maintaining the representation of the same point.

B. Vehicle coordinate system VCS

Without loss of generality, let us consider a VCS whose origin is located on the road surface and below the camera's *center of projection* (COP). The z-axis of the VCS is pointing vertically towards the camera, whereas the x-axis is pointing parallel to the lateral profile of the vehicle into the direction of travel. As a result, the x- and y-axis are on the road surface.

C. Extrinsic parameters

The extrinsic parameters $[\mathbf{R}_e \ \mathbf{t}_e]$ were already defined in (2). The Euclidean transform can be inverted as follows

$$\mathbf{M}_v = \mathbf{R}_e^T \, \mathbf{M}_c + \mathbf{t}_h \quad \text{with} \quad \mathbf{t}_h = -\mathbf{R}_e^T \, \mathbf{t}_e. \tag{5}$$

Based on the assumption in II-B, the COP has the coordinates $\mathbf{M}_c = 0$ in the CCS and $\mathbf{M}_v = \mathbf{t}_h = [0, 0, h_c]^T$ in the VCS, where h_c denotes the camera height. Since

$$\mathbf{t}_e = -\mathbf{R}_e \, \mathbf{t}_h, \tag{6}$$

the extrinsic parameters can alternatively be expressed by the rotation matrix $\mathbf{R}_e$ and the camera height h_c. The extrinsic parameters and the VCS are illustrated in Fig. 1.

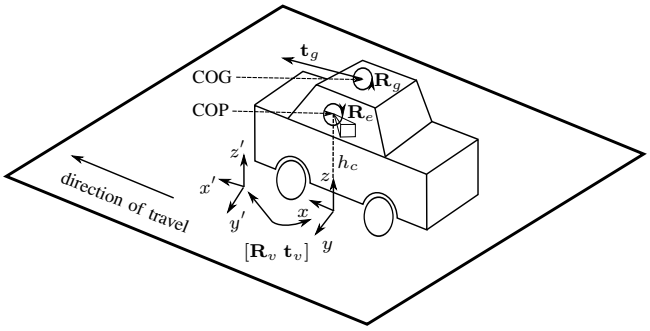

Fig. 1. Definition of the extrinsic parameters $\mathbf{R}_e$ and h_c with respect to the VCS, and the relationship between the motion in the VCS and the motion of the COG defined by $[\mathbf{R}_v \ \mathbf{t}_v]$ and $[\mathbf{R}_g \ \mathbf{t}_g]$, respectively

D. Relationship between the motion of the center of gravity, the motion of the vehicle and the motion of the camera

The *center of gravity* (COG) of the vehicle is often used as the reference point to model the motion of the vehicle. The position of the COG with respect to the VCS is described by the vector $\mathbf{t}_s$, which is typically known when the camera is mounted to the vehicle. The motion of the vehicle can be described by a translation vector $\mathbf{t}_g$ and the rotation matrix matrix $\mathbf{R}_g$ (see Fig. 1). The resulting motion in the VCS can be expressed by an Euclidean transform as follows

$$\mathbf{M}_v^{'} = \mathbf{R}_v \, \mathbf{M}_v + \mathbf{t}_v \tag{7}$$

where $\mathbf{M}_v^{'}$ represents the same 3D point as $\mathbf{M}_v$ after the vehicle has moved. See Appendix I for a detailed derivation of $\mathbf{R}_v = \mathbf{R}_g^T$ and $\mathbf{t}_v = \mathbf{t}_s - \mathbf{R}_g^T \, \mathbf{t}_s - \mathbf{R}_g^T \, \mathbf{t}_g$. In the following, (2), (5) and (7) are combined to determine the motion of the camera. If T is the elapsed time between two camera shots, the motion of the camera in the CCS is

$$\mathbf{M}_c^{'} = \underbrace{\mathbf{R}_e \mathbf{R}_v \mathbf{R}_e^T}_{\mathbf{R}_c} \mathbf{M}_c + (\underbrace{-\mathbf{R}_e \mathbf{R}_v \mathbf{R}_e^T \mathbf{t}_e + \mathbf{R}_e \mathbf{t}_v + \mathbf{t}_e}_{\mathbf{t}_c}). \tag{8}$$

The relationship between different coordinates of the same object point in the VCS and the CCS before and after the motion is illustrated in Fig. 2. The extrinsic parameters are assumed to be identical for both camera shots.

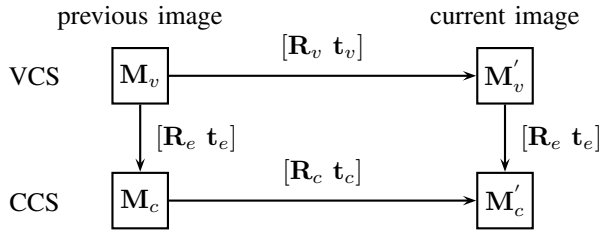

Fig. 2. Relationship between different coordinates

III. Motion of the camera between two frames

A. Motion of the vehicle

Modern vehicles are equipped with sensors, like the yaw rate sensor, steering wheel sensor and wheel speed sensor, which allow the estimation of the motion of the vehicle, namely the rotation $\mathbf{R}_g$ and translation $\mathbf{t}_g$. The motion is defined with respect to the COG, because many sensors are calibrated in reference to the COG. Similar to [14], we use the velocity and the yaw rate of the vehicle to estimate the trajectory of the vehicle. The velocity v and the yaw rate $\dot{\phi}$ of the vehicle are assumed to be constant for the short time interval T. Consequently, the trajectory has the shape of an ideal circle

$$\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} = \int_0^T v \begin{bmatrix}\cos(\dot{\phi}\,t)\\ \sin(\dot{\phi}\,t)\end{bmatrix} dt = r \begin{bmatrix}\sin(\dot{\phi}\,T)\\ 1-\cos(\dot{\phi}\,T)\end{bmatrix} \tag{9}$$

where the radius of the circle is $r = v/\dot{\phi}$. The yaw of the vehicle is defined by the angle $\phi = \dot{\phi}\,T$. The motion of the vehicle is assumed to be planar. This means that the vehicle rotates by the angle ϕ about the z-axis and the rotation matrix $\mathbf{R}_g$ has the form

$$\mathbf{R}_g = \begin{bmatrix}\cos(\phi) & -\sin(\phi) & 0\\ \sin(\phi) & \cos(\phi) & 0\\ 0 & 0 & 1\end{bmatrix}. \tag{10}$$

The angle ϕ is called the yaw angle. The roll and pitch of the vehicle are neglected. The translation vector $\mathbf{t}_g$ is defined by

$$\mathbf{t}_g = [\Delta x, \Delta y, 0]. \tag{11}$$

The last component of $\mathbf{t}_g$ is zero, since the motion is assumed to be planar. Δx is the movement into the direction of travel, whereas Δy is the lateral motion of the vehicle.

The radius in (9) is undefined if the yaw rate is zero. Nevertheless, the translation is $\mathbf{t}_g = [v\,T, 0, 0]$. To avoid the case discussion, the radius can be reformulated by

$$r = \frac{v\,T}{2\sin(\frac{\dot{\phi}\,T}{2})} \approx \frac{v\,T}{2\frac{\dot{\phi}\,T}{2}} \quad \text{for} \quad |\dot{\phi}\,T| \ll \pi \tag{12}$$

if the distance $\Delta s = v\,T$ of the circular motion is assumed to be identical to the distance in a straight line. This simplification is illustrated in Fig. 3.

The smaller the yaw angle ϕ is, the better the approximation is. We will discuss in Sec. III-B that the assumption of a small yaw angle holds for our purpose. Using $\sin(2\,\dot{\phi}\,T) = 2\sin(\dot{\phi}\,T)\cos(\dot{\phi}\,T)$ and $\cos(2\,\dot{\phi}\,T) = 1 - 2\sin^2(\dot{\phi}\,T)$, the radius of (12) substiuted into (9) leads to

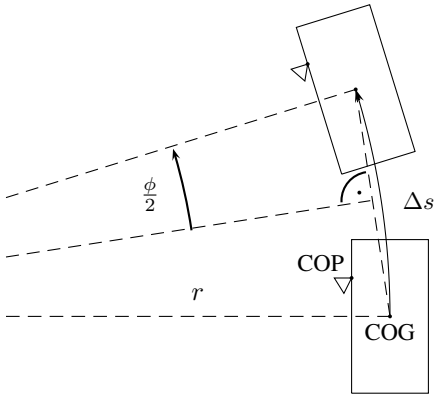

Fig. 3. The distance $\Delta s = v\,T$ of the circular motion versus the identical distance in a straight line

$$\begin{bmatrix}\Delta x\\ \Delta y\end{bmatrix} = \Delta s \begin{bmatrix}\cos(\frac{\dot{\phi}\,T}{2})\\ \sin(\frac{\dot{\phi}\,T}{2})\end{bmatrix} \quad \text{with} \quad \Delta s = v\,T. \tag{13}$$

The simplification not only prevents numerical problems for $\dot{\phi} = 0$ but also enables easy adaption of the formula if the acceleration of the vehicle is additionally considered. The solution of the differential equation in (9) is more complex if the velocity is modeled by $v(t) = v + \dot{v}\,t$. Combined with the approximation in (13), the vehicle's acceleration $\dot{v}$ leads directly to $\Delta s = v\,T + \frac{1}{2}\dot{v}\,T^2$. The model of the yaw angle can be extended similarly.

One could imagine that the yaw rate depends on the velocity and steering angle of the vehicle. This is true and the relationship is established in the area of vehicle dynamics. Unfortunately, all derivations need vehicle-specific quantities such as the wheel base etc. Additionally, they fail to predict the yaw angle in the presence of crosswind and on sloping roads. In comparison, a yaw rate sensor measures the real rotation of the vehicle where the camera is attached, an advantage that enables the direct determination of the yaw angle without the knowledge of vehicle-specific parameters. These are the main reasons to make use of the velocity and yaw rate alone. It is worthwhile mentioning that further sensory data is successfully integrated in systems like the *Electronic Stability Control* (ESC) to determine the driver-intended trajectory based on the steering angle etc. However, the main contribution of this work is the use of odometric data in video-based ADAS. We will show that our approach is suitable for that purpose.

B. Access to odometric data

We obtain the velocity and yaw rate via the *Controller Area Network* (CAN-bus) of the vehicle. The velocity sensor, actually the wheel speed sensors, and the yaw rate sensor have a temporal resolution of 20 ms. The quantization is $0.01\,\mathrm{km/h}$ and $0.04\,^\circ/\mathrm{s}$, respectively. Both values might be different for other vehicles or sensors. The yaw rate and velocity are depicted in Fig. 4 for a part of a typical drive. The statistical analysis of more than $20,000$ km measured

data reveals that the yaw rate is less than $0.65°/s$ for 50% of the time, less than $1.07°/s$ for 75% of the time, less than $1.65°/s$ for 90% of the time and greater than $2.19°/s$ for only 5% of the time. A camera in an automotive application normally operates with frame rates in the range of $10-50$ fps. This confirms the assumption $|\dot{\phi}T| \ll \pi$ from the previous section.

C. Filtering and synchronizing the sensor data

The yaw rate in Fig. 4 is obviously much more affected by noise than the velocity. The Kalman filter has been proven to reliably estimate a parameter set (state variables) based on a known physical model (state space model) if measurements are affected by noise. The transition state model describes the fact that the velocity remains constant or slightly changes if the vehicle accelerates between two measurements. This has the advantage that the acceleration of the vehicle can be directly integrated in the estimation process. The measurement noise of the sensor data is also part of the estimation. The Kalman filter is known to be used for this purpose [15] and will not be covered in this paper.

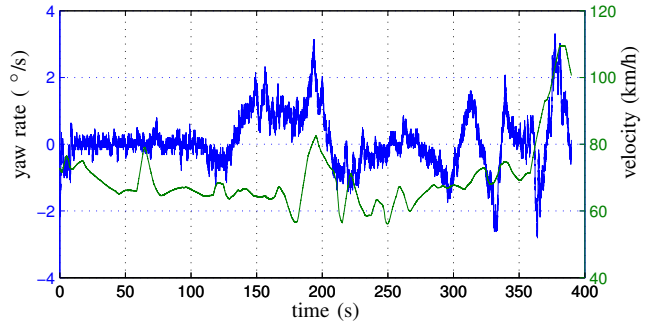

Fig. 4. Noisy yaw rate (top) and velocity (bottom) over time (400 seconds)

The temporal discretization of the transition state model depends on the measurements. Unfortunately, the video camera normally operates at a different temporal resolution than the other sensors. For a proper operation, the required velocity and yaw rate for the estimation of the trajectory have to be in sync with the video data. There are different solutions to this problem: either the required parameters are interpolated or extrapolated from adjacent estimates of the Kalman filter, or an asynchronous Kalman filter is applied. Note that the latency of the sensors and the CAN-bus have to be considered as well and the timestamps of the camera should respect the shutter time.

IV. Obstacle Detection

A. Prediction of the image motion on the road

The motion on the road can be described by a homography matrix $\mathbf{H}_c$, if the road is assumed to be flat. The motion of a point in image and normalized coordinates, respectively, is defined by

$$\widetilde{\mathbf{m}}_p^{'} = \mathbf{A}\,\mathbf{H}_c\,\mathbf{A}^{-1}\,\widetilde{\mathbf{m}}_p \quad \text{and} \quad \widetilde{\mathbf{m}}_c^{'} = \mathbf{H}_c\,\widetilde{\mathbf{m}}_c \tag{14}$$

if the point is on the road. It is known in literature [16] that the homography matrix $\mathbf{H}_c$ for a plane Π_c is defined by

$$\mathbf{H}_c = \mathbf{R}_c + \frac{\mathbf{t}_c\,\mathbf{n}_c^T}{d_c}. \tag{15}$$

where the plane $\Pi_c : \mathbf{n}_c^T\mathbf{M}_c - d_c = 0$ is defined with respect to the first view. $\mathbf{R}_c$ and $\mathbf{t}_c$ can be calculated using (8), if the trajectory of the vehicle is estimated according to Sec. III-A. With the definition of the VCS from Sec. II-B, the road surface is defined by $\mathbf{n}_v = [0,0,1]^T$ and $d_v = 0$. This plane is transformed to a plane in the CCS by

$$\Pi_c : \underbrace{\mathbf{n}_v^T\mathbf{R}_e^T}_{\mathbf{n}_c^T}\mathbf{M}_c - \underbrace{(d_v + \mathbf{n}_v^T\mathbf{R}_e^T\mathbf{t}_e)}_{d_c} = 0. \tag{16}$$

The form of $\mathbf{R}_v$ and the definition of the extrinsic parameters in (6) simplifies the translation vector $\mathbf{t}_c$ and the distance d_c. The relevant parameters required in (15) are finally:

$$\begin{aligned}
\mathbf{R}_c &= \mathbf{R}_e\mathbf{R}_v\mathbf{R}_e^T, \\
\mathbf{t}_c &= \mathbf{R}_e(\mathbf{R}_v\mathbf{t}_h - \mathbf{t}_h + \mathbf{t}_v) &&= \mathbf{R}_e\mathbf{t}_v, \\
\mathbf{n}_c &= \mathbf{R}_e\,\mathbf{n}_v, \\
d_c &= -\mathbf{n}_v^T\mathbf{R}_e^T\mathbf{R}_e\mathbf{t}_h &&= -h_c.
\end{aligned}$$

This leads to a homography between two camera shots which can be expressed by the matrix

$$\mathbf{H}_c = \mathbf{R}_e\left(\mathbf{R}_v + \frac{\mathbf{t}_v\,\mathbf{n}_v^T}{-h_c}\right)\mathbf{R}_e^T. \tag{17}$$

Hence, the automatic calibration of the extrinsic parameters proposed in [13] and the use of odometric data enables the calculation of the homography matrix $\mathbf{H}_c$.

B. Motion compensation

In the following, we assume that the homography between two images is known. The intrinsic parameters of the camera are calibrated offline, according to [17], and the lens distortion is considered as well although it is not mentioned explicitly. This allows the resampling of one of the images in order to compensate for the motion on the road. In the so-called forward transformation, we compute

$$\mathbf{I}_m([u,v]^T) = \mathbf{I}_c(\mathbf{A}\,\mathbf{H}_c\,\mathbf{A}^{-1}\,[u,v,1]^T) \tag{18}$$

according to (14), where $\mathbf{I}_m$ is the motion compensated image and $\mathbf{I}_c$ the current image, see Fig. 5. Additionally, a subset of the resampling coordinates is depicted by the blue points. Note that an interpolation is necessary because the coordinates have non-integer values; we use bilinear interpolation. The previous image is not resampled, but points above the horizon are rejected - this is also indicated by the blue points. After the compensation, $\mathbf{I}_m$ is compared with the previous image $\mathbf{I}_p$ by subtracting the intensity values for each pixel as follows

$$\mathbf{I}_d([u,v]^T) = |\mathbf{I}_m([u,v]^T) - \mathbf{I}_p([u,v]^T)|. \tag{19}$$

Fig. 5 (e) and (f) show this difference image and the result of a simple pixel-wise threshold segmentation $\mathbf{I}_d([u,v]^T) > t_h$. Obviously, only moving and static objects are segmented as obstacles. Note that vehicles in the adjacent lane can be detected independently of their velocity. This means that they are detected even if they are traveling at the same speed as the vehicle equipped with the camera. Unfortunately, the car in the adjacent lane is only partially segmented as an object, because it is almost homogeneous in color. Homogeneous

regions appear similar although they have shifted. However, the lane markings, for example, overlay each other almost perfectly. This demonstrates the accuracy of the underlying motion field. We do not claim that a pixel-wise threshold segmentation is the best method for obstacle detection in this case, but it shows how simple it could be to implement this. More advanced methods are presented in [**?**], [**?**], [11].

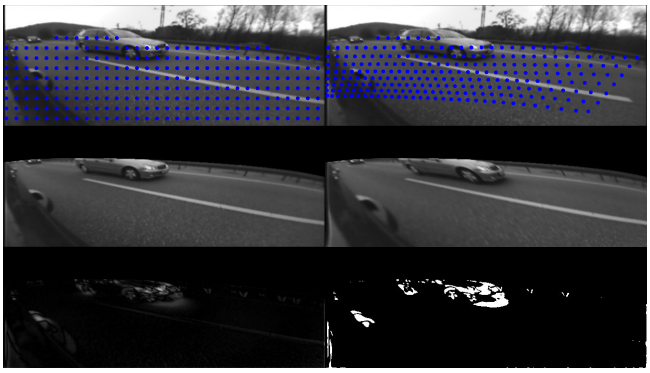

Fig. 5. From top left to bottom right: (a) previous image, (b) current image, (c) previous image and horizon cropped, (d) current image motion compensated and horizon cropped, (e) difference image, (f) pixel-wise threshold segmentation

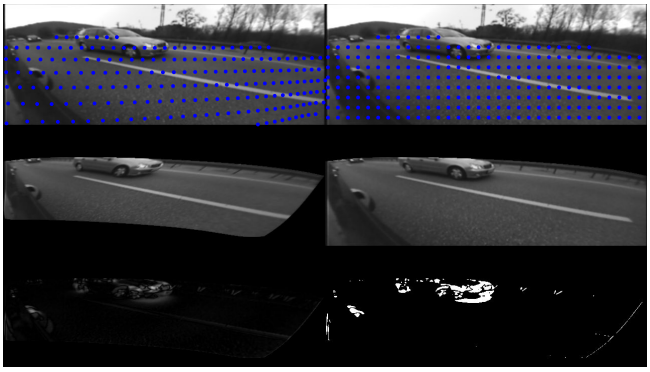

Fig. 6. From top left to bottom right: (a) previous image, (b) current image, (c) previous image motion compensated and horizon cropped, (d) current image and horizon cropped, (e) difference image and undefined region cropped, (f) pixel-wise threshold segmentation

In the backward transformation, the previous image is motion compensated by using $\mathbf{H}_c^{-1}$ instead of $\mathbf{H}_c$. The original and the transformed version of the previous image is illustrated in Fig. 6. However, the backward transformation has the problem that those image regions which are not visible in the previous image remain undefined. This is difficult to handle in practice, since the size of these regions depends on the motion of the camera. Both the forward and backward transformation have additional restrictions:

- The complete image is resampled but only a subset of the image is required, namely the adjacent lane.
- No reasonable motion compensation is possible for image points above the horizon.
- Cars in the adjacent lane appear with a different size although their real size is almost identical. This makes the segmentation difficult.

C. Virtual Camera Perspective (VCP)

To overcome the restrictions of the real camera perspective mentioned in Sec. IV-B, an additional resampling of the image data is performed. The resampling process can be interpreted as the introduction of a virtual camera perspective. There are two classes: either the centers of projection of the virtual and real cameras are, or are not identical. The VCP is realistic only if the COP remains the same. Otherwise, an arbitrary resampling or the idea of an inverse perspective mapping IPM can be applied. The image rectification in stereo vision is a method of the first class. The bird's eye view, also known as top view, is a representative of the second class and well known to be used as a VCP.

D. Virtual bird's eye view

The advantages of a top view are demonstrated in [10]. Actually, the top view can be generated similar to Sec. IV-A by a homography

$$\mathbf{T}_t = \mathbf{A}\left[\mathbf{r}_{e,1}, \mathbf{r}_{e,2}, \mathbf{t}_e\right]\mathbf{V}^{-1} \tag{20}$$

except that the homography matrix only depends on the intrinsic matrix $\mathbf{A}$ and extrinsic parameters $[\mathbf{R}_e\ \mathbf{t}_e]$. Note that $\mathbf{R}_e = [\mathbf{r}_{e,1}, \mathbf{r}_{e,2}, \mathbf{r}_{e,3}]$. The derivation can be found in [17]. The matrix $\mathbf{V}$ has the structure

$$\mathbf{V} = \begin{bmatrix} \alpha_x & 0 & o_x \\ 0 & \alpha_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{21}$$

and is therefore similar to the intrinsic matrix of an ideal pinhole camera, with the difference that the road is sampled instead of the image sensor. The sampling frequency and an offset can be applied by adjusting $[\alpha_x, \alpha_y]$ and $[o_x, o_y]$, respectively. In Fig. 7, the top view is combined with the motion compensation. The range is limited to a distance of $\approx 20\,\mathrm{m}$. Even though the sampling on the road is equidistant, the resampling coordinates in the image domain are not. The advantage of the top view is that the real proportions of the road are recovered. However, only the road is correctly mapped, whereas other objects are unrealistically deformed. Unfortunately, the resampled image contains only a small fraction of the information available in the original image.

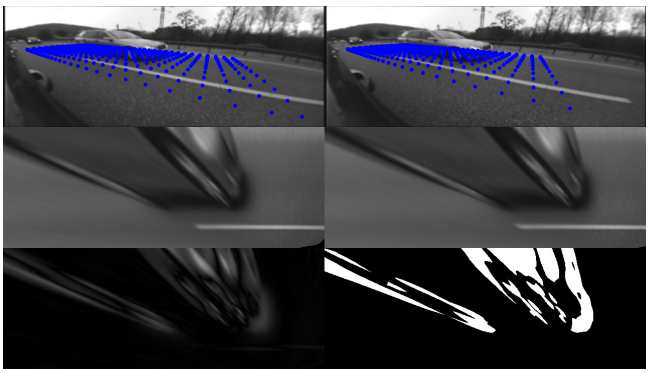

Fig. 7. From top left to bottom right: (a) previous image, (b) current image, (c) previous image with top view, (d) current image motion compensated with top view, (e) difference image, (f) pixel-wise threshold segmentation

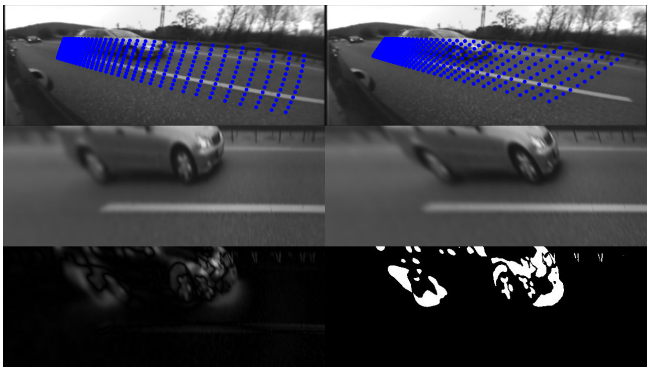

Fig. 8. From top left to bottom right: (a) previous image, (b) current image, (c) previous previous rectified, (d) current image motion compensated and rectified, (e) difference image, (f) pixel-wise threshold segmentation

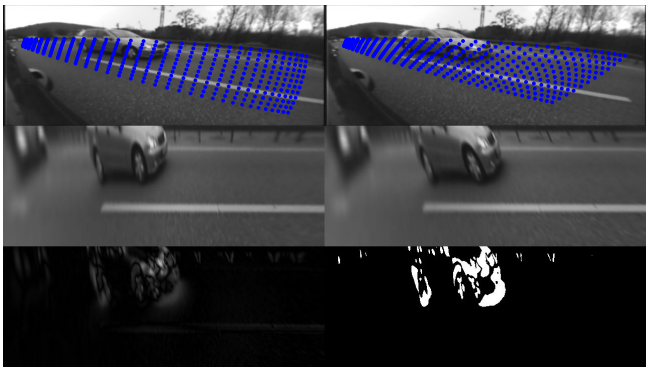

Fig. 9. From top left to bottom right: (a) previous image, (b) current image, (c) previous image non-linear rectified, (d) current image motion compensated and non-linear rectified, (e) difference image and undefined region cropped, (f) pixel-wise threshold segmentation

E. Image rectification

Image rectification is a transformation process in stereo vision. It can be considered to be a rotation around the center of projection. After rectification, the correspondence problem is reduced to one dimension, namely the rows of the image. We recommend the excellent work of [16] for further information. The transformation process also uses homography. Here, it is necessary to specify a homography matrix for the transformation of the previous image $\mathbf{T}_p$ and current image $\mathbf{T}_a$:

$$\mathbf{T}_p = \mathbf{A}\,\mathbf{R}_p\,\mathbf{V}^{-1} \quad \text{and} \quad \mathbf{T}_a = \mathbf{A}\,\mathbf{R}_a\,\mathbf{V}^{-1}. \tag{22}$$

$\mathbf{R}_p$ and $\mathbf{R}_a$ are the corresponding rotation matrices. They are calculated by making use of $[\mathbf{R}_c\,\mathbf{t}_c]$ as follows

$$\mathbf{R}_p = [\frac{-1}{a_x}(\mathbf{R}_c^{-1}\mathbf{t}_c), \frac{1}{a_y}(\mathbf{e}_z \times \mathbf{R}_{p,1}), \frac{1}{a_z}(\mathbf{R}_{p,1} \times \mathbf{R}_{p,2})] \tag{23}$$

and

$$\mathbf{R}_a = [\frac{-1}{a_x}\mathbf{t}_c, \frac{1}{a_y}(\mathbf{R}_c\,\mathbf{e}_z \times \mathbf{R}_{a,1}), \frac{1}{a_z}(\mathbf{R}_{a,1} \times \mathbf{R}_{a,2})]. \tag{24}$$

As mentioned before, they rotate the original perspective to the new axially parallel alignment. The parameters a_x, a_y and a_z normalize the column vectors of the rotation matrix. Note that a rotation matrix is orthogonal. The rotation is based on the idea that the first column vector of the matrix rotates the x-axis of the new perspective to the baseline between the two centers of projection, the second column vector is chosen to be perpendicular to this baseline and the original z-axis ($\mathbf{e}_z = [0, 0, 1]^T$), and the third column vector is computed as the cross product of the first and second column vector.

Fig. 8 shows the result of the rectification in combination with motion compensation. As expected, the resampled images are realistic versions of an ideal pinhole camera. However, the range is limited and the resampling is not beneficial in terms of information redundancy.

F. Non-linear camera model

Until now, the virtual perspectives were defined by the matrix $\mathbf{V}$, which is derived from an ideal pinhole camera model. We propose to use a non-linear camera model instead of the linear pinhole camera model to improve the resampling and to increase the range of obstacle detection. In principle, the transformation $\mathbf{V}^{-1}[u, v, 1]^T$ is replaced by $[f_x(u, v), f_y(u, v), 1]^T$ where $f_x(u, v)$ and $f_y(u, v)$ are non-linear functions. Fig. 9 demonstrates the advantages of a non-linear camera model in combination with image rectification and motion compensation. The range of obstacle detection is increased without reducing the image information in the foreground. Note that in Fig. 7 and Fig. 8 the vehicle in the background is not visible at all.

Actually, a non-linear function is used for $f_x(u, v)$ so that the resampling coordinates are almost uniformly distributed. In general, if $f_y(u, v)$ is independent of u, the epipolar constraint remains unaffected. We use $f_y(u, v) = \frac{1}{\alpha_y}v - \frac{o_y}{\alpha_y}$. Theoretically, the range need not to be limited since the vanishing point is represented by the epipole. The problem is that there is no information remaining in the vertical direction of the resampled image. In practice, it is reasonable to limit the range and to apply a beneficial non-linear resampling.

A series of segmentations over time are presented in Fig. 10. The camera operates with a resolution of 640x240 pixels and a frame rate of 30 fps. For demonstration, only every tenth image pair has been selected. It is easy to track the segmentation, although the motion increases from left to right.

V. Conclusion

We presented a method to detect obstacles in the adjacent lane of a vehicle based on the image motion on the road. The motion field is precisely predicted with the yaw rate and velocity, if the extrinsic parameters of the camera are known. The motion field is then used to compensate the motion in the image. We perform a simple threshold segmentation of the difference image, which is the subtraction of the motion compensated image and the corresponding reference image, in order to demonstrate the accuracy of the underlying motion field. To simplify the image processing, the bird's eye view or image rectification is also used. However, the range of obstacle detection is limited to a certain distance and the resampling is not beneficial in terms of information redundancy. We successfully applied a non-linear camera model to overcome these drawbacks.

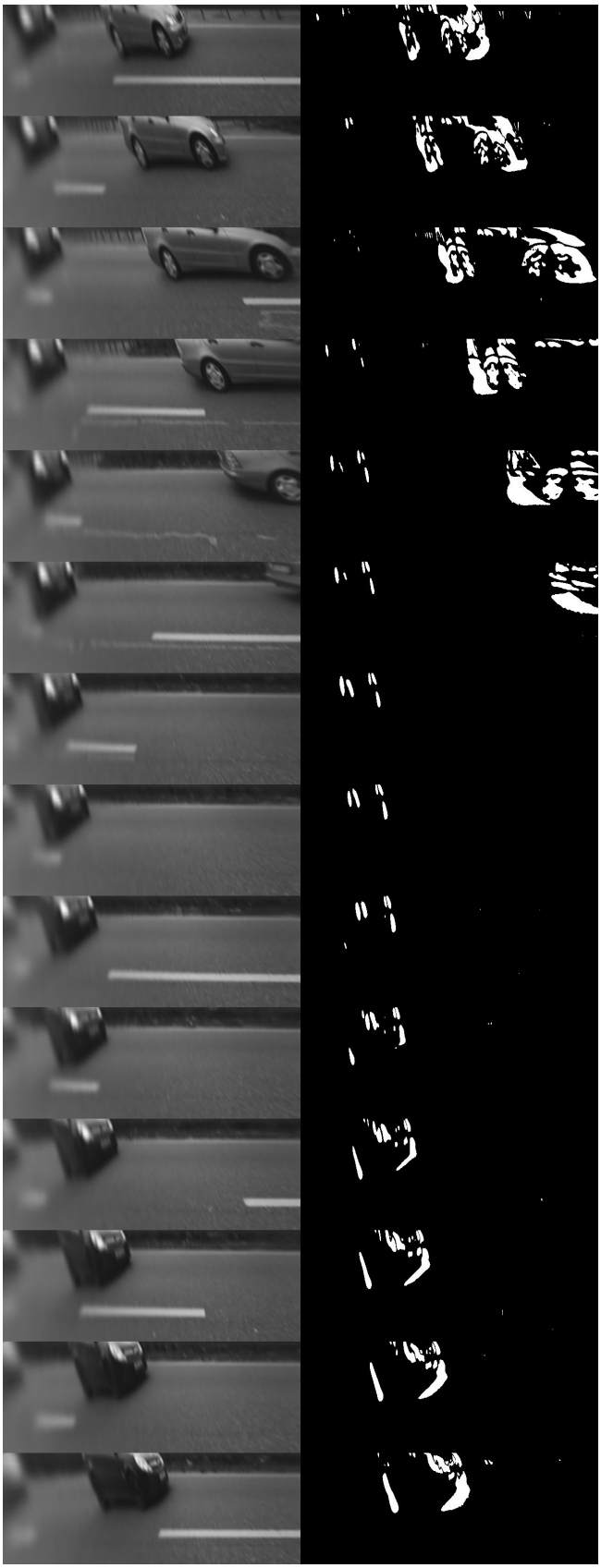

Fig. 10. Segmentation results over time, every tenth image is selected, current image motion compensated and non-linear rectified on the left side, pixel-wise threshold segmentation on the right side

APPENDIX I

Assume a coordinate system with the same orientation as the VCS but with the COG as the origin. The relation between a point $\mathbf{M}_g$ in the new coordinate system and a point $\mathbf{M}_v$ in the VCS is expressed by

$$\mathbf{M}_v = \mathbf{M}_g + \mathbf{t}_s \quad \text{and} \quad \mathbf{M}_g = \mathbf{M}_v - \mathbf{t}_s. \tag{25}$$

Remember that $\mathbf{t}_s$ is the position of the COG with respect to the VCS. The motion of the vehicle defined in Sec. III-A can be summarized as follows

$$\mathbf{M}_g = \mathbf{R}_g \, \mathbf{M}_g^{'} + \mathbf{t}_g \quad \text{and} \quad \mathbf{M}_g^{'} = \mathbf{R}_g^T \, \mathbf{M}_g - \mathbf{R}_g^T \, \mathbf{t}_g. \tag{26}$$

The combination of the previous equations leads finally to

$$\mathbf{M}_v^{'} = \underbrace{\mathbf{R}_g^T}_{\mathbf{R}_v} \, \mathbf{M}_v + \underbrace{\mathbf{t}_s - \mathbf{R}_g^T \, \mathbf{t}_s - \mathbf{R}_g^T \, \mathbf{t}_g}_{\mathbf{t}_v}. \tag{27}$$

REFERENCES

[1] W. Enkelmann, "Obstacle detection by evaluation of optical flow fields from image sequences," in *European Conference on Computer Vision*, 1990, pp. 134–138.

[2] W. Enkelmann, V. Gengenbach, W. Krüger, S. Rössle, and W. Tolle, "Obstacle detection by real-time optical flow evaluation," in *IEEE Proc. Intelligent Vehicles Symposium*, 1994, pp. 97–102.

[3] W. Krüger, W. Enkelmann, S. Rössle, and W. Tolle, "Real-time estimation and tracking of optical flow vectors for obstacle detection," in *IEEE Proc. Intelligent Vehicles Symposium*, 1995, pp. 304–309.

[4] N. Ancona, "A fast obstacle detection method based on optical flow," in *European Conference on Computer Vision*, 1992, pp. 267–271.

[5] G. Lefaix, T. Marchand, and P. Bouthemy, "Motion-based obstacle detection and tracking for car driving assistance," in *IEEE Proc. International Conference on Pattern Recognition*, vol. 4, 2002, pp. 74–77.

[6] A. Giachetti, M. Campani, and V. Torre, "The use of optical flow for road navigation," *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 34–48, 1998.

[7] S. P. Batavia, D. A. Pomerleau, and C. E. Thorpe, "Overtaking vehicle detection using implicit optical flow," in *IEEE Proc. International Conference on Intelligent Transportation Systems*, 1997, pp. 729–734.

[8] W. Krüger, "Robust real-time ground plane motion compensation from a moving vehicle," *Machine Vision and Applications*, vol. 11, no. 4, pp. 203–212, 1999.

[9] C. Braillon, C. Pradalier, J. L. Crowley, and C. Laugier, "Real-time moving obstacle detection using optical flow models," in *IEEE Proc. Intelligent Vehicles Symposium*, 2006, pp. 466–471.

[10] H. A. Mallot, H. H. Bülthoff, J. J. Little, and S. Bohrer, "Inverse perspective mapping simplifies optical flow computation and obstacle detection," *Biological Cybernetics 64*, pp. 177–185, 1991.

[11] M. Bertozzi and A. Broggi, "Gold: A parallel real-time stereo vision system for generic obstacle and lane detection," *IEEE Transactions on Image Processing*, vol. 7, no. 1, pp. 62–81, 1998.

[12] S. Tan, J. Dale, A. Anderson, and A. Johnston, "Inverse perspective mapping and optic flow: A calibration method and a quantitative analysis," *Image and Vision Computing 24*, p. 153165, 2006.

[13] M. Miksch, B. Yang, and K. Zimmermann, "Homography-based extrinsic self-calibration for cameras in automotive applications," in *Workshop on Intelligent Transportation*, 2010, pp. 1–6.

[14] M. Woock, F. Pagel, M. Grinberg, and D. Willersinn, "Odometry-based structure from motion," in *IEEE Proc. Intelligent Vehicles Symposium*, 2007, pp. 1112–1117.

[15] M. Bühren and B. Yang, "On motion models for target tracking in automotive applications," in *Workshop on Intelligent Transportation*, Hamburg, Germany, 2007.

[16] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.

[17] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 1330–1334, 1998.

[18] O. D. Faugeras, *Three-Dimensional Computer Vision: A Geometric Viewpoint*. MIT Press, 1993.