Robust Image Processing for an Omnidirectional Camera-based Smart Car Door

Christian Scharfenberger, Samarjit Chakraborty and Georg Färber
Inst. for Real-Time Computer Systems
Technische Universität München, Germany
{christian.scharfenberger, samarjit.chakraborty, georg.faerber}@rcs.ei.tum.de

***Abstract*—Over the last one decade there has been an increasing emphasis on driver-assistance systems for the automotive domain. In this paper we report our work on designing a camera-based surveillance system embedded in a "smart" car door. Such a camera is used to monitor the ambient environment outside the car – e.g., the presence of obstacles such as approaching cars or cyclists who might collide with the car door if opened – and automatically control the car door operations. This is an enhancement to the currently available side-view mirrors which the driver/passenger checks before opening the car door. The focus of this paper is on fast and robust image processing algorithms specifically targeting such a smart car door system. The requirement is to quickly detect traffic objects of interest from gray-scale images captured by omnidirectional cameras. Whereas known algorithms for object extraction from the image processing literature rely on color information and are sensitive to shadows and illumination changes, our proposed algorithms are highly robust, can operate on gray-scale images (color images are not available in our setup) and output results in real-time. To illustrate these, we present a number of experimental results based on image sequences captured from real-life traffic scenarios.**

I. Introduction

Driver assistance systems are increasingly gaining importance in high-end cars. Examples of these include Lane Departure Warning System (LDW), Adaptive Cruise Control (ACC), Forward Collision Warning (FCW) and Blind-spot detection (BSD) [1]. While there are many safety-oriented driver-assistance systems that function when the car is moving, a number of accidents happen while the car is stationary and one of its doors is being opened. A standard practice is to check the side-view mirrors of the car before opening the door. However, it is still fairly common for approaching cyclists to hit suddenly-opened car doors.

In this paper we report our work on designing a smart car door, that is equipped with one omnidirectional camera on each side of the car. These cameras monitor the ambient environment outside the car and warn passengers (or car door users) about obstacles like approaching cars, bicycles or pedestrians. Collision avoidance systems use this information to control, stop or lock car door operations in order to avoid potential accidents. Figure 1 gives a high-level overview of our smart car door system. In [2], we presented a generic control system for intelligent, actuated car doors with arbitrary degrees of freedom. The focus of this paper was the mechanical design and the control of the door. However, an important component

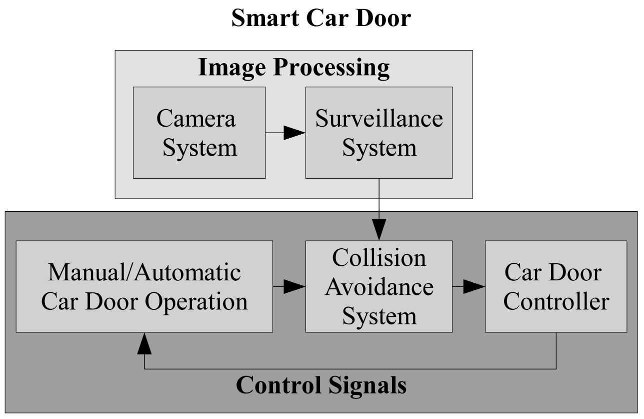


Fig. 1. Smart Car Door System

of such a smart door is the image capturing and processing subsystem, whose output serves as an input to the control subsystem. In this paper we focus on the camera subsystem and on robust algorithms for object extraction from image sequences captured by the camera. The cameras in question are omnidirectional vision sensors consisting of a perspective camera focused on a cone-like hyperboloidal mirror (see Figure 2). Figure 3 shows such an omnidirectional camera system integrated with the side-view mirror of a car for monitoring the external environment. Given the large field-of-view of the vision sensors, the camera is able to monitor the side of the car door in its entirety (see Figure 4) and the associated image processing algorithms enable early-detection of impending obstacles. While opening a car door there is usually a small time interval between parking and the door operation. It is important to detect approaching traffic before any car door operation is performed. Approaching cars or cyclists must be identified even if they are relatively far away from the car. This allows us to formulate certain preconditions under which our object detection and extraction algorithms may and should operate: (i) We may assume a static camera for a short time interval (parking). This interval may be used to learn the environment around the door. (ii) Objects that are further away from the camera occupy less real estate on each video frame (i.e., they occupy fewer pixels) and are hence not easy to differentiate from the background. (iii) Due to the large field-of-view of the

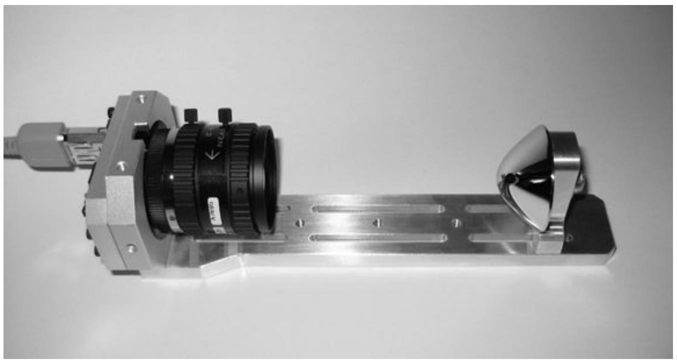


Fig. 2. An omnidirectional camera system – a perspective camera focuses on a hyperboloidal mirror and takes pictures with a field-of-view of 360°.

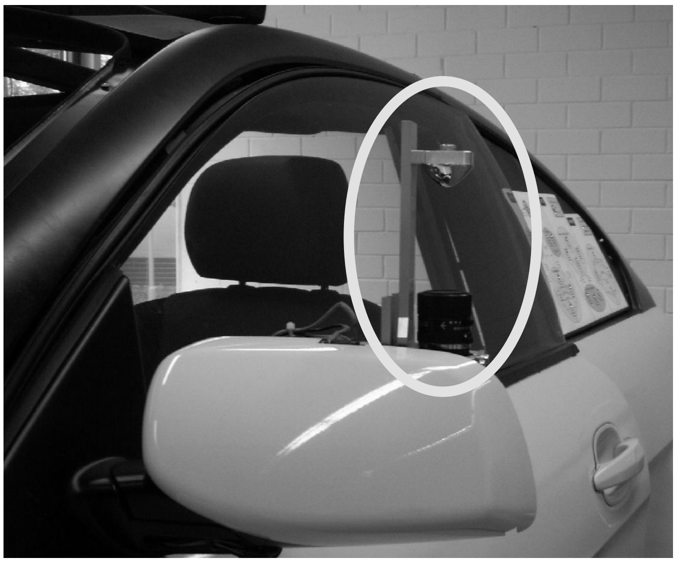


Fig. 3. Our camera system integrated with the side-view mirror of a car.

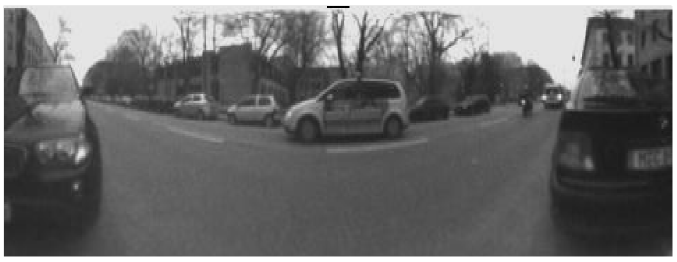


Fig. 4. Panoramic image of the environment around a car door.

cameras used, there is a different view of same object (front, side and back) as it moves (see Figure 5). (iv) Algorithms to detect approaching traffic and obstacles must operate in real-time. Cars driving at $13.89m/s$ (about $50km/h$) move approximately $0.5m$ between two frames tracked with cameras having frame rates of 30 frames/second. Hence, fast-moving cars have to be detected within a maximum of two or three frames.

A number of techniques for object detection and extraction

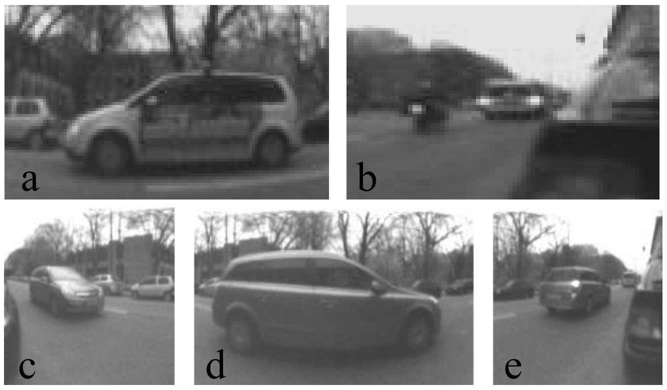


Fig. 5. a) Large and b) very small road objects. c) - e) Different views of a driving car.

exist in the image processing literature, viz., Hidden Markov Models [3], Template Matching [4], feature-based detection methods, optical flow-based methods [5],[6] and background separating methods. Table I gives a brief overview of possible methods and compares them in terms of the required computation time, parallelizability for fast implementation, and the ability to handle perspective changes. Background estimation and optical flow are not sensitive to perspective changes and object size. Furthermore, they do not need prior knowledge of the object's property, e.g., color, shape and geometry. One disadvantage of optical flow is that it can not detect static or very slow moving objects. Although background estimation solves this problem, it needs a small time interval to learn the background. Fortunately, in our setting such an interval is available, viz., the time interval between parking and door operation. In this paper we focus on real-time moving

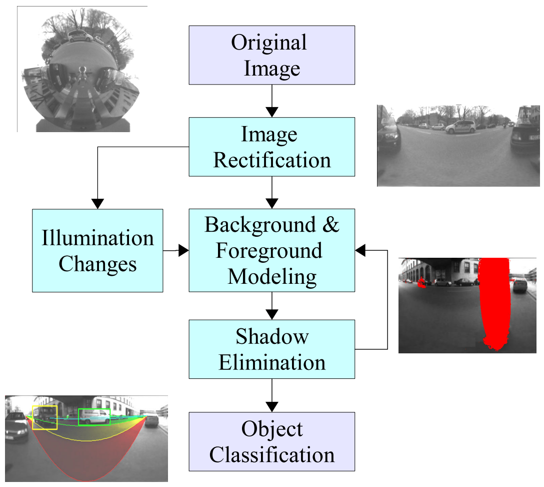


Fig. 6. Block diagram for object detection using background estimation, shadow compensation and handling of illumination changes.

object extraction algorithms using background estimation-based techniques. Figure 6 gives a high-level overview of our smart car door system. In particular, we present extensions to background estimation (e.g., illumination compensation

	Initia-lisation	Prior Knowl.	Static Objects	Obj. Size Changes	Perspective Changes	Paralleliza-bility	Exec. Time
Template Matching	no	needed	ind	bad	bad	middle	slow
HMM	no	needed	ind	bad	bad	middle	slow
Feature Based	no	needed	ind	bad	bad	good	fast
Optical Flow	no	no	bad	ind	ind	good	middle
Background Estimation	needed	no	ind	ind	ind	good	fast

TABLE I
COMPARISON OF DIFFERENT OBJECT DETECTION METHODS FOR ENVIRONMENT SURVEILLANCE. *Background estimation* SEEMS TO YIELD THE BEST RESULTS IN OUR SETTING. (*ind: independent*)

and shadow elimination for gray-scale images) which are specifically tuned to our setting of a smart car door equipped with omnidirectional cameras. The details of our algorithm for object extraction are described in what follows.

A. Related Work and Our Contributions

The problem of extracting objects from a video sequence has been widely studied in surveillance [7], traffic monitoring [8] and vehicle guidance. In most applications, separating the foreground from the background is the first step for object tracking. Background subtraction and foreground modeling are powerful methods whose advantages are feature-independent segmentation (e.g., textures, direction of move, speed). Some common techniques for background subtraction include Kalman filtering [9], kernel density estimation [10], hidden Markov models [11], mixture of Gaussians [12], [13] and the use of color-based intensity independent features [14]. Most of these algorithms represent each background pixel using a probability density function (PDF) and classify pixels from new images as background depending on the description of pixels by their density functions. As an alternative, Bhaskar *et al.* [15] developed a foreground detection algorithm using cluster density estimation based on a Gaussian mixture model. This algorithm is suitable for handling illumination changes as well as dynamic backgrounds. Similar work was done in [16] using Kalman filtering to iteratively estimate the dynamic background texture and the regions of foreground objects. Kalman filtering was also used by Karman *et al.* [9] to model the background dynamics of each pixel by choosing two different gains, thereby allowing fast adaptation of background changes and slow adaptation of foreground pixels. Ridder *et al.* [17] improved this approach and presented a shadow detection method assuming only small differences between overshadowed and non-overshadowed background.

Although many background subtraction techniques have been proposed, the majority of the algorithms address shadow detection and illumination compensation by exploiting color information (see [10], [16]). In scenarios where monochromatic video cameras are used – such as ours – the existing methods are no longer suitable. Furthermore, it is difficult to differentiate between small illumination changes caused by shadows or by small, valid foreground objects in gray scale images. These problems, along with the accuracy of background subtraction, the handling of sudden illumination changes and the possibility of parallelizing the algorithms are the underlying motivations of our work.

Inspired by the background estimator of Ridder *et. al.* [17] and by the shadow detector proposed by Jacques Jr. *et al.* [18], we develop robust background estimation and foreground detection algorithms for gray scale images. Shadow borders are identified in [17], but once detected as foreground it is impossible to differentiate between a shadow and a foreground. A good shadow detector for gray scale images was introduced by Jacques *et al.* [18] using normalized cross correlation (NCC). The detector assumes shadow pixels as scaled versions (darker) of the corresponding background pixels, so that the NCC in a neighboring region is close to unity. On the other hand, this shadow detector misclassifies valid foreground pixels with small differences as shadow pixels. To overcome these limitations, we combine and modify these different methods to design a powerful background subtractor. We also extend the shadow detector with zero means cross correlation (ZNCC) in order to distinguish between shadows and valid foreground pixels. Our proposed algorithm detects illumination changes using local search windows and updates the background to compensate for slow or sudden illumination changes.

Our experiments in complex outdoor and indoor environments under various lightning conditions demonstrate promising results. We also evaluated and compared our approach with a background estimator based on Gaussian Mixtures, with the approach of Jacques Jr. *et al.* [18], as well as the approach of Ridder *et al.* [17]. We also evaluated our algorithms for their parallelizability potential and compared sequential and parallel implementations (on an AMD Quad-Core CPU). Our results indicate that they work in real-time and are suitable for implementation on embedded platforms.

The rest of the paper is organized as follows. We describe the image rectification techniques in Section II, the background estimator, the shadow detector, as well as our handling of illumination changes in Section III. The results obtained are discussed in Section IV. Finally, we conclude by briefly outlining some possibilities for future work.

II. CALIBRATION AND IMAGE RECTIFICATION

In this section we provide the technical details related to the omnidirectional camera subsystem. Original images from omnidirectional vision sensors are distorted and are not easy to handle for conventional image processing algorithms. The main problem is in extracting geometric and perspective relations like size and position of objects. To overcome this limitation, original images are transformed into panoramic (rectified) images. But in this case the camera model and the calibration must be precisely known. We designed an omnidirectional vision system based on the well-known single point of view (SPOV) theorem of Nayar and Baker [19]. SPOV is also known as the projection center of the mirror onto which the perspective camera should focuses. This is a prerequisite for geometrically correct panoramic image transformation. Our omnidirectional camera uses a mirror whose surface is based on a hyperboloidal equation. Using such mirrors, the SPOV constraint is only valid for an accurate alignment of the mirror and the camera. However, this is difficult to realize and hence the camera system must be calibrated to compensate for misalignments as well as to obtain a precise relation between 3d world point coordinates and the camera sensor coordinates.

A. Calibration

To determine the 3d position of object points projected on the sensor plane, the function $f(\vec{p})$ (see Eq. (1)) describing a relation between $\vec{p}$ in world coordinates x_P, y_P and z_p and the camera coordinates u_P and v_P has to be found (see Eq. 1).

$$\vec{P} = \begin{bmatrix} u_P \\ v_P \end{bmatrix} = f(\vec{p}) \text{ with } \vec{p} = \lambda \cdot \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}, \lambda > 0 \qquad (1)$$

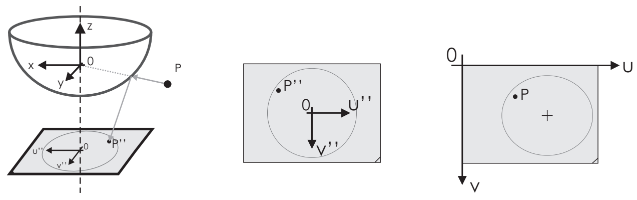
Different techniques are known for determining the function $f(\vec{p})$ [20], [21]. We use the calibration method developed by Scaramuzza *et al.* [22]. All points on a vector $\vec{p}$ in world coordinates (see Figure 7(a)) are projected to the corresponding point P'' on the virtual plane E'' (see Figure 7(b)) and are then transformed to the real sensor plane which concerns to misalignments of the camera sensor (see Figure 7(c)). Scaramuzza *et al.* propose the relation between world and virtual sensor plane as

$$\begin{aligned} \vec{p} &= \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix} = \lambda \cdot \begin{bmatrix} u_{P''} \\ v_{P''} \\ f(\rho) \end{bmatrix} \\ &= \begin{bmatrix} x_p \\ y_p \\ a_0 + a_2\rho^2 + \ldots + a_N\rho^N \end{bmatrix} \end{aligned} \qquad (2)$$

with $\lambda = 1$ and $\rho = \sqrt{x_{\vec{p}}^2 + y_{\vec{p}}^2}$. Furthermore, they approximate the component z of $f(\vec{p})$ depending on the curvature as a polynomial function. The relation between the real sensor plane and the virtual or ideal sensor plane is given as an affine transformation (see Eq. 3).

$$\begin{aligned} \vec{P''} &= \mathbf{A} \cdot \vec{P} + \vec{t} \quad \text{with} \quad \mathbf{A} = \begin{bmatrix} c & d \\ d & 1 \end{bmatrix} \\ \vec{P} &= \begin{bmatrix} u_P \\ v_P \end{bmatrix} \quad \text{and} \quad \vec{t} = \begin{bmatrix} u_{center} \\ v_{center} \end{bmatrix} \end{aligned} \qquad (3)$$

The parameters $a_0 \ldots a_n$, $\mathbf{A}$ and $\vec{t}$ are the calibration parameters.



(a) World to virtual sensor plane E'' (b) Point P'' on the virtual sensor plane (c) Real sensor plane

Fig. 7. The camera model (a) used in this paper. The world point P is mapped on a virtual sensor plane E'' (b) and the projection transformed to the real sensor plane (c) using affine transformations.

B. Image Rectification

Using the camera model and the calibration data, the corresponding 3d object points of all pixels on the sensor plane can be calculated and vice versa. This allows us to determine a projection area based on individual projection parameters like width M, height N as well as a region of interest (ROI) for image rectification as a first step. Secondly, each pixel $[m, n]^T$ of the projection area is stored in a $M \times N \times 3$ dimensional matrix $\mathbf{F}$ containing its world coordinates $\mathbf{X}(m, n)$, $\mathbf{Y}(m, n)$ and $\mathbf{Z}(m, n)$. Lastly, the corresponding pixel position of each point on the projection area is calculated and stored in a look up table (LUT). Using this information in the LUT, every original image from the camera can be transformed into a panoramic images. Figure 8 illustrates this flow.

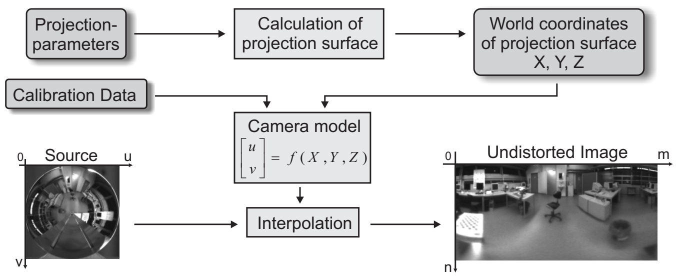

Projection-parameters
Calculation of projection surface
World coordinates of projection surface X, Y, Z
Calibration Data
Camera model
$\begin{bmatrix} u \\ v \end{bmatrix} = f(X, Y, Z)$
0 Source u
v
Interpolation
0 Undistorted Image m
n

Fig. 8. Proposed image rectification process.

III. Background Model

As discussed in Section I, we used background estimation for extracting objects of interest from the captured images. Our background model is based on the approach of Karman *et al.* [9] and Ridder *et al.* [17]. It is extended to provide better shadow detection and to be more robust against illumination changes for our application. We present the mathematical details of the background model along with the shadow detector and a method to account for illumination changes. We describe the background model based on Kalman filtering proposed by Karman and Ridder to model the dynamics of each background pixel in Section III-A. Secondly, we classify pixels as background or possible foreground pixels using thresholding. Possible foreground pixels are then classified as valid foreground or shadow pixels using the NCC and the

ZNCC (see Section III-B and III-C). Finally, we present a method to account for global illumination changes in Section III-D.

A. Kalman Background Estimation

In this section, we describe the background model on which our approach is based. The intensity of a pixel at position (x,y) at time t is given by $I_{x,y}(t)$. The estimated system state of the background model is denoted as $\hat{I}_{x,y}(t)$ and its derivative as $\hat{\hat{I}}_{x,y}(t)$. The estimation on the background is

$$\begin{bmatrix} \hat{I}_{x,y}(t) \\ \hat{\hat{I}}_{x,y}(t) \end{bmatrix} = \begin{bmatrix} \tilde{I}_{x,y}(t) \\ \tilde{\tilde{I}}_{x,y}(t) \end{bmatrix} + K_{x,y}(t) \cdot \left(I_{x,y}(t) - H \cdot \begin{bmatrix} \tilde{I}_{x,y}(t) \\ \tilde{\tilde{I}}_{x,y}(t) \end{bmatrix} \right) \tag{4}$$

Following Eq. 5, the prediction $\tilde{I}_{x,y}(t)$ of the system state $\hat{I}_{x,y}(t)$ and its derivative $\tilde{\tilde{I}}_{x,y}(t)$ at time t is given by:

$$\begin{bmatrix} \tilde{I}_{x,y}(t) \\ \tilde{\tilde{I}}_{x,y}(t) \end{bmatrix} = S \cdot \begin{bmatrix} \hat{I}_{x,y}(t-1) \\ \hat{\hat{I}}_{x,y}(t-1) \end{bmatrix} \tag{5}$$

With the system matrix S, the measurement matrix H and the Kalman gain K is:

$$S = \begin{bmatrix} 1 & s_{1,2} \\ 0 & s_{2,2} \end{bmatrix}, H = \begin{bmatrix} 1 & 0 \end{bmatrix} \\ \text{and} \quad K_{x,y}(t) = \begin{bmatrix} k1_{x,y}(t) \\ k2_{x,y}(t) \end{bmatrix} \tag{6}$$

In [9], $s_{1,2} = s_{2,2} = 0.7$ was used for modeling the background dynamics. Because the camera returns only the intensities $I_{x,y}(t)$, the measurement matrix H is a constant. The Kalman gain was chosen depending on detected foreground or background using a pre-estimation of the next system state (see Eq. 7 and Eq. 8).

$$m_{x,y}(t) = \begin{cases} 1, & if \quad \left[d'_{x,y}(t) \geq th_{bg}\right] \vee \\ & \quad \left[(d'_{x,y}(t) < th_{bg}) \wedge \right. \\ & \quad \left. (d''_{x,y}(t) \geq th_{bg})\right] \\ 0, & if \quad \left[d'_{x,y}(t) < th_{bg}\right] \wedge \\ & \quad \left[d''_{x,y}(t) < th_{bg}\right] \end{cases} \tag{7}$$

$$\begin{aligned} d'_{x,y}(t) &= |I_{x,y}(t) - \tilde{I}_{x,y}(t)| \\ d''_{x,y}(t) &= |I_{x,y}(t) - \hat{I}'_{x,y}(t)| \\ \text{with} \quad \hat{I}'_{x,y}(t) &= \tilde{I}_{x,y}(t) + \beta \cdot \left[I_{x,y}(t) - \tilde{I}_{x,y}(t)\right] \end{aligned} \tag{8}$$

Pixels whose differences of the intensity to the system state are smaller than a fixed threshold ($d' < th_{bg}$), do not necessarily indicate background. To identify such pixels, a pre-estimation $\hat{I}'_{x,y}(t)$ of the next system state is calculated assuming that these pixels belong to background. If the pre-estimated value d'' is greater than th_{bg} this pixel nevertheless belongs to foreground. A binary map $m_{x,y}(t)$ represents the segmentation of pixels (1 for foreground and 0 for background), and the Kalman gain $k1,2_{x,y}(t) = \alpha$ or $k1,2_{x,y}(t) = \beta$ is chosen depending on the binary map $m_{x,y}(t)$ (see Eq. 9).

$$k1,2_{x,y}(t) = \begin{cases} \alpha, & if \quad m_{x,y}(t) = 1 \\ \beta, & if \quad m_{x,y}(t) = 0 \end{cases} \tag{9}$$

If an image only contains background objects, the initialization of the background can be done using one image. The pixel information of this image will be the initial background state. The system state can be given by one image following Eq. 10.

$$\begin{bmatrix} \hat{I}(x,y,t_0) \\ \hat{\hat{I}}(x,y,t_0) \end{bmatrix} = \begin{bmatrix} I(x,y,t_0) \\ 0 \end{bmatrix} \tag{10}$$

Otherwise, if images contain background along with moving objects, the method proposed in [18] extract background information from an image sequence and can be used to learn the initial background.

B. Shadow Detection

For a safe door operation, it is important to predict the risk of possible collisions in advance. This prediction is based on the object's positions and on recognizing if there are objects with dangerous trajectories. Shadows could cause an inaccuracy in determining the position that could lead to a wrong prediction of possible collisions. Hence, shadow pixels must be detected and suppressed. We use the normalized cross correlation (NCC, [18]) as an initial step for shadow detection and refined it using zero means normalized cross correlation (ZNCC) to handle foreground pixels with small differences with the background. Let $\tilde{I}_{x,y}(t)$ be the estimated background image and $I_{x,y}(t)$ an image given by the camera system. For each foreground pixel, we generate a template $T_{xy}(n,m)$ such that $T_{xy}(n,m) = I_{x+n,y+m}(t)$ for $-N \leq (n,m) < N$ where $\bar{t}$ is the mean of the template $T_{xy}(n,m)$. Furthermore, let $B_{xy}(n,m)$ be the template of the background such that $B_{xy}(n,m) = \hat{I}_{x+n,y+m}(t)$ where $\bar{b}$ is the mean of template $B_{xy}(n,m)$. The ZNCC as well as the NCC ($\bar{t} = 0$, $\bar{b} = 0$) between $T_{xy}(n,m)$ and $B_{xy}(n,m)$ at pixel (x,y) can be calculated using Eq. 15:

$$ZNCC_{x,y} = \frac{EZR_{x,y}}{EZB_{x,y} \cdot EZT_{x,y}} \tag{11}$$

with

$$\begin{aligned} EZR_{x,y} &= \sum_{n=-N}^{N} \sum_{m=-N}^{N} |(B_{xy}(n,m) - \bar{b})||(T_{xy}(n,m) - \bar{t})| \\ EZB_{x,y} &= \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} (B_{xy}(n,m) - \bar{b})^2} \\ EZT_{x,y} &= \sqrt{\sum_{n=-N}^{N} \sum_{m=-N}^{N} (T_{xy}(n,m) - \bar{t})^2} \end{aligned} \tag{12}$$

where $EZT_{x,y}$ considers the energy of the image template and $EZB_{x,y}$ considers the energy of the estimated background. A pixel may potentially be classified as shadow if its NCC in the neighborhood is close to unity and its energy $ET_{x,y}$ is lower than the energy of the background $EB_{x,y}$ (see Eq. 13).

$$NCC_{x,y} \geq th_{NCC} \text{ and } EB_{x,y} > ET_{x,y} \tag{13}$$

$EB_{x,y}$ as well as $ET_{x,y}$ can be determined by calculating $EZB_{x,y}(\bar{b} = 0)$ and $EZT_{x,y}(\bar{t} = 0)$.

C. Shadow Refinement

Depending on the chosen threshold th_{NCC} with ($th_{NCC} < 1.0$), many foreground pixels with small differences to background pixels may be misclassified as shadow pixels. To overcome this limitation, we refined the classification of shadow- and nonshadow-pixels using the ZNCC. The advantage of ZNCC is light invariance, so that only differences in texture cause significant changes in its value. The refinement stage verifies if there are significant changes through textures and not through illumination. Although the ZNCC is light invariant, image noise (texture changes) influence the ZNCC and cause an offset. This offset θ can be determined while learning the background model or can be accounted for by the threshold th_{ZNCC}. Similar to the NCC, a pixel is a shadow candidate if the ZNCC in the neighborhood is close to the learned initial value and the energy $ET_{x,y}$ of the template is lower than the energy of the background $EB_{x,y}$. But contrary to the NCC, $EZT_{x,y}$ and $EZB_{x,y}$ represent only the energy of the textures from the background and the template. Hence, the energy of texture from valid foreground pixels can be lower than the energy of texture from background which is the case for large homogenous objects like cars. So, a pixel must then be a shadow candidate if the energy $EZT_{x,y}$ is approximately as equal as the energy $EZB_{x,y}$. Eq. 15 expresses this relation.

$$\begin{aligned} |ZNCC_{x,y} - (1.0 - \theta)| &\leq th_{ZNCC}\ and \\ |EZB_{x,y} - EZT_{x,y}| &\leq th_{comp}\ and \\ ET_{x,y} &< EB_{x,y} \end{aligned} \tag{14}$$

D. Active Light Adaptation

Background models based on Kalman filtering can follow slow illumination changes in the background. However, when foreground objects cover the background, illumination changes in the background cannot be detected. Furthermore, sudden illumination changes cannot be respected by the background because depending on the chosen threshold sudden illumination changes cannot be classified as valid foreground or illumination changes. So there is a need to modify the background, taking illumination changes into account. Therefore, we subdivide every new image into m subimages at each position $p_{x(m)}$ and $p_{y(m)}$ fitting the hole image and calculate their mean gray values (see Eq. 15).

$$\mu(m,t) = \frac{1}{J \cdot I} \sum_{j=-J/2}^{J/2} \sum_{i=-I/2}^{I/2} I(p_{x(m)} + j, p_{y(m)} + i, t) \tag{15}$$

Here, J and I are the subimage sizes. The global illumination change $\Delta(t)$ can now be detected by calculating the median of all local illumination changes $\delta(m,t)$:

th_{bg}	FG: α	BG: β	Size NCC/ZNCC
7	0.00004	0.004	5×5

th_{NCC}	th_{ZNCC}	th_{Δ}	NoW: Size(J $\times$ I)
0.992	0.2	2.0	90 (30 $\times$ 18)

TABLE II
OVERVIEW OF OPTIMIZED PARAMETERS.

$$\Delta(t) = \underset{m}{median}\,\delta(m,t) \tag{16}$$

with

$$\delta(m,t) = \mu(m,t) - \mu(m,t-1) \tag{17}$$

Because small illumination changes are adapted by the background model, we decided to use simple thresholding in order to avoid modifying the background model too frequently.

$$\Delta(t) = \begin{cases} 0, & if \quad \Delta(t) < th_{\Delta} \\ \Delta(t), & if \quad \Delta(t) \geq th_{\Delta} \end{cases} \tag{18}$$

Finally, Eq. 5 for background estimation is modified to overcome illumination changes.

$$\begin{bmatrix} \tilde{I}_{x,y}(t) \\ \tilde{\tilde{I}}_{x,y}(t) \end{bmatrix} = S \cdot \begin{bmatrix} \hat{I}_{x,y}(t-1) \\ \hat{\hat{I}}_{x,y}(t-1) \end{bmatrix} + \begin{bmatrix} \Delta(t) \\ 0 \end{bmatrix} \tag{19}$$

Using this approach, the background model can account for slow as well as for sudden illumination changes.

E. Parallelization

For all our proposed techniques, it may be seen that different pixels may be processed in parallel. In other words, image rectification, the background estimator, the shadow detector as well as the illumination compensation can all be run in parallel on a multi-core CPU. As mentioned before, our algorithm have to work in real-time and hence such parallelization is highly desirable. This is useful for speeding up the system for time critical task like detection of traffic participants. The original image returned by the camera subsystem is divided into n subimages and the same number of threads is generated to run on a multi-core platform. After processing each image the results from all threads must be merged and interpolated (e.g., when an object being detected is split across two or more subimages) for further object detection and classification algorithms. Figure 9 illustrates our parallelization technique.

IV. Experimental Results

To verify and evaluate our approach, we conducted experiments in complex environments containing weak and strong shadows as well as small differences between foreground and background using an omnidirectional vision system (ODVS). Image rectification was used to transform the captured images into panoramic images of size 480×204 pixels. Images from the ODVS were used to test the algorithm under various conditions (dark and light regions, image noise and different resolutions due to image rectification and interpolation). Table II gives an overview of the optimized parameters.

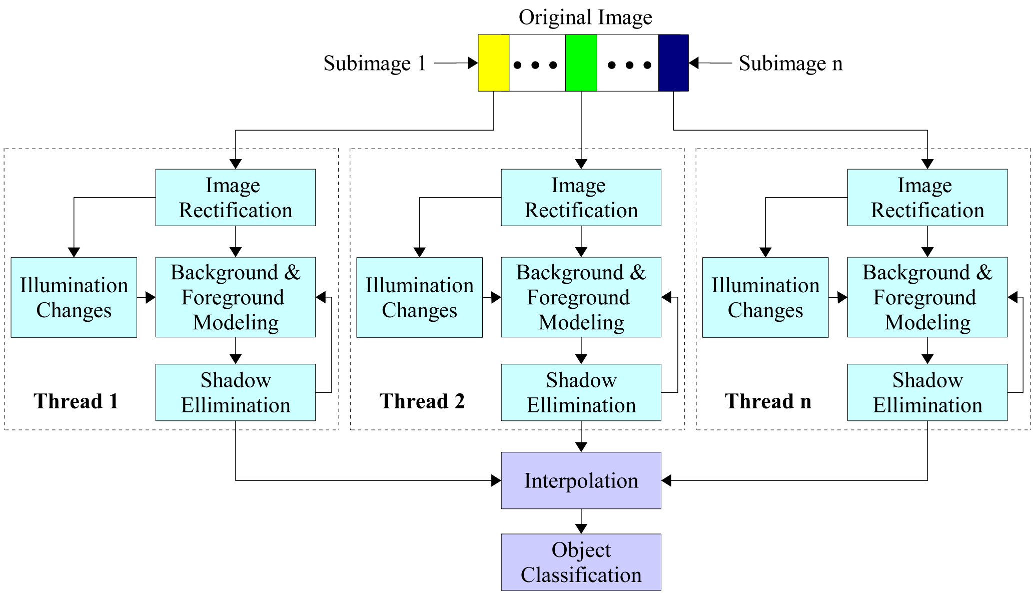


Fig. 9. Parallelization of the object detecting algorithm.

A. *Detection of Shadow Pixel Candidates*

To detect small differences in intensity between foreground and background objects, the threshold th_{bg} (see Eq. (7)) must be low. An experimentally obtained value was $th_{bg} \geq 7$ which allows the detection of small intensity differences, but still many shadow pixels and noise are detected. Figure 10 illustrates the result of foreground detection (BG/FG) for one pixel over time using NCC and ZNCC. NCC was useful to pre-estimate shadow pixels, but valid foreground pixels are often misclassified as shadow pixels which can be seen on the noisy characteristic for the foreground (*BG/FG NCC*). ZNCC overcomes these limitations by taking textural changes into account, so that foreground pixels are not misclassified as background. The result is a smoother characteristic of the foreground / background characteristic (*BG/FG ZNCC*).

B. *Illumination Changes and Background Adaptation*

Figure 11 illustrates our experiments with various illumination changes. The reference characteristic of the background is presented in Figure 11(a), various characteristics of the background disturbed by illumination changes in Figure 11(b) and Figure 11(c). The background model accounts for slow illumination changes even if the background was covered for a short time interval and illumination changes were not to large (see Figure 11(c), frames $0 - 100$). Sudden illumination changes, which are larger than th_{bg}, cause wrong foreground information (see frames (280 - 310) and (380 - 400)). Lastly, figure (11(d)) demonstrates, that detected illumination changes can successfully be compensated if they are accounted for by the background model (see Eq. (19)). We also conducted experiments to find the optimal number of search windows (NoW) for detecting global illumination changes. The number of search windows must be chosen so that influence of illumination changes caused by foreground objects is minimized (see Eq. (16)). We generated a test profile of illumination changes (IC) and tracked it by detecting illumination changes with different numbers of search windows. Experiments showed that at least 60 search windows were necessary to track the light profile sufficiently. The main problem of less then 60 search windows is the high influence of lighting changes

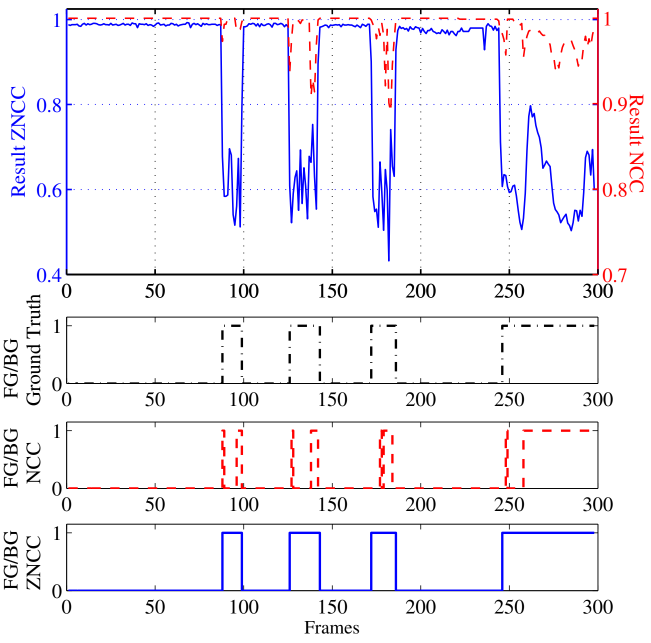
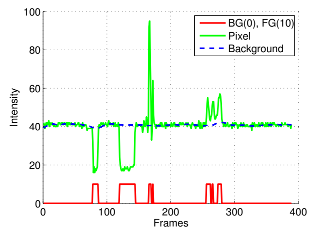
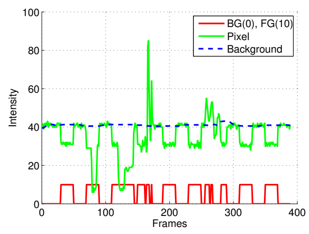


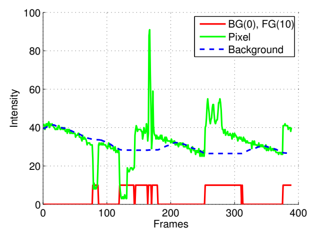
Fig. 10. Top: Characteristic of NCC and ZNCC from an image pixel. Middle and Down: Classification of foreground ($FG/BG = 1/0$) using NCC and refined by ZNCC. ZNCC can better distinguish valid foreground from shadow.



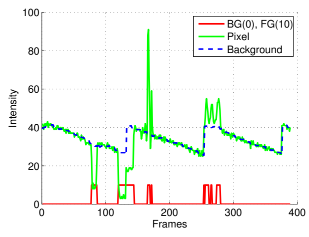
(a) No illumination changes (IC)



(b) Detect. result by fast IC.

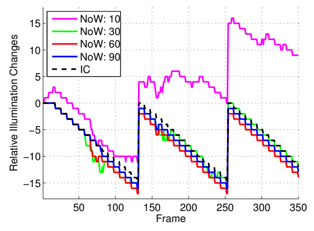


(c) Detect. result by slow IC.

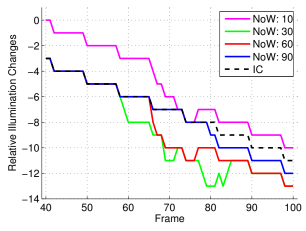


(d) Compensation of IC.

Fig. 11. (a) One pixel and the detected foreground over time (reference). (b) Misclassification of foreground pixels caused by fast illuminations changes. (c) Misclassification of foreground pixels caused by slow illuminations changes. (d) Adaptation of fast and slow illumination changes.



(a) Illumination changes



(b) Frames 40 - 100

Fig. 12. The more number of search windows (NoW) can be used for detecting illumination changes(IC), the better is the detection result. Good results give a NoW of 60 - 90.

caused by foreground objects. The influence of foreground objects is almost suppressed using 90 NoW. Our measurements of tracking the test profile using different numbers of search windows illustrates Figure 12. We also derived from our experiments that one search window should not be smaller than (15×15) pixels due to the increased influence of image noise from smaller window size. An offset less than 2 (less than $th_{\triangle}$) in tracking the profile was tolerable and automatically adapted by the background model.

C. Validation of Foreground Pixels

Not all detected foreground pixels need to be valid (true positives = t.p.), i.e., there might also be false positives. For example, shadow pixels are often misclassified as valid foreground (false positives = f.p.). On the other hand, pixels having small differences to background can falsely be classified as background pixels (f.n., false negatives). Figure 13 gives an example of a typical road scenario containing both true and false positives as well as false negatives. We evaluated

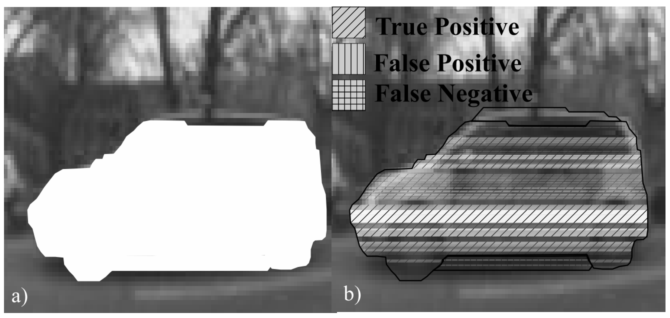


Fig. 13. a) Detected pixels of a foreground object. b) Some pixels are not detected as background or shadow pixels and highlighted as true positives. There are also valid foreground pixels which are not highlighted as foreground (false negatives).

our algorithm under various conditions like diffused light, direct sunlight and indoor conditions. We compared the results obtained with a perfect detection. These results are shown in Table III, where the percentages were computed based on 200 test images. Good detection rates were achieved for large

Scenario	Obj. Size	t.p.	f.n.	f.p.
Diffuse light	small	85%	15%	2%
	large	95%	5%	3%
Sunlight	small	76%	24%	7%
	large	93%	7%	10%
Indoor cond.	small	90%	10%	1%
	large	97%	3%	4%

TABLE III
OVERVIEW OF OUR VALIDATION RESULTS.

foreground objects in all tested scenarios. Clearly, having a large fraction of misclassified pixels results in an object not being detected. Furthermore, shadow pixels in images with sunlit scenarios can easily be misclassified as valid foreground pixels.

D. Computation Time and Parallelization

We chose a complex indoor environment with three walking people, shadow effects and some illumination changes (switching light on/off) to measure the execution time of the proposed algorithm. We use about 400 test images of this data set and calculated the mean execution time as well as the standard deviation (Std. Dev.). Table IV demonstrates the execution times for rectification, background modeling as well as shadow detection and illumination changes using a single CPU of a 2.3 GHz AMD Phenom 9650 quad-core CPU. As discussed in Section III-E, we parallelized our object detection algorithm using multithreading on the quad-core CPU and measured the execution times for the same test data set. The image was divided into n subimages, each of which was processed by a different concurrent thread. The result of all threads is then merged using small thread called interpolation. Clearly, as the total computation time will be decreased with increasing

	Mean Time	Std. Dev.	Image Size
Rectification	≈ 3.6 ms	0.7 ms	640 × 480
Background	≈ 30.1 ms	1.2 ms	480 × 204
Shadow Dect	≈ 24.1 ms	2.3 ms	”
Ill. Comp	≈ 10.4 ms	1.6 ms	”
Total Time	68.2 ms		

TABLE IV
THIS TABLE ILLUSTRATES THE COMPUTATION TIME ON A 2.3 GHZ AMD PHENOM 9650 QUAD-CORE FOR THE NON PARALLELIZED ALGORITHM.

number of threads, the time for merging and interpolation increased. Table V gives an overview of the measured times.

	1 CPU	2 CPU	4 CPU
Rectification	≈ 3.6 ms	≈ 1.9 ms	≈ 1.0 ms
Background	≈ 30.1 ms	≈ 15.1 ms	≈ 7.5 ms
Shadow Dect	≈ 24.1 ms	≈ 12.2 ms	≈ 6.3 ms
Ill. Comp	≈ 10.4 ms	≈ 5.6 ms	≈ 2.8 ms
Interpolation	0.0 ms	≈ 2.0 ms	≈ 2.2 ms
Total Time	68.2 ms	36.8 ms	19.8 ms

TABLE V
THIS TABLE ILLUSTRATES THE COMPUTATION TIME ON A 2.3 GHZ AMD PHENOM 9650 QUAD-CORE FOR THE PARALLELIZED ALGORITHM.

Figure 14 illustrates the single computation time for each processing step as well as the overall computation time for more than 4 CPUs. This estimation is based on our measurements for 1, 2 and 4 CPUs and is extrapolated up to 16 CPUs. Due to the increasing time for merging and interpolation the results of all threads there is no appreciable speed up using more than 16 CPUs.

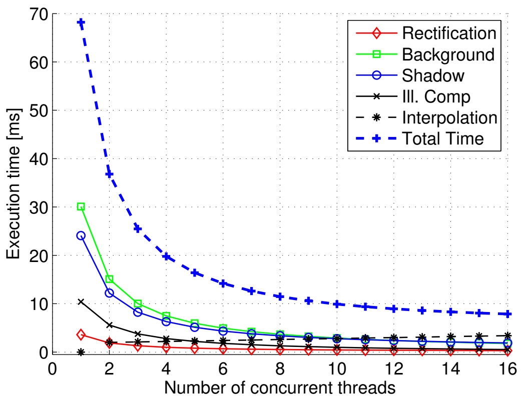
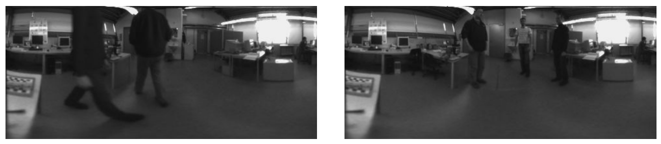
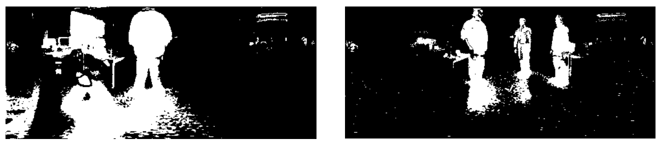


Fig. 14. Estimation of computation time using more than 4 CPUs.

Finally, we evaluated our proposed method using various scenarios and compare our approach with other well known algorithm described in Section I-A. Figure 15 illustrates one of our evaluated scenarios containing up to 500 test frames. Difficulties of this scenario are a less textured environment as well as weak and strong shadows induced by different lightning sources. While the approach in [17] modeled the background well, shadow detection failed in some cases. Similarly, while the shadow detector in [18] performed well, the background model has some limitations. One limitation is that once the background is learned the background is not updated, which results in many noisy foreground pixels caused by illumination changes etc. The combination of both algorithms and the modification of the shadow detector as well as the light compensation led in a powerful background estimator which resulted in better foreground detection on grayscale images, when compared with state of the art techniques (see Figure 15).

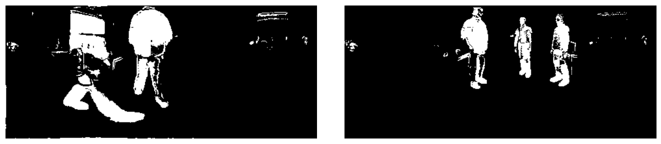


Original image sequence.

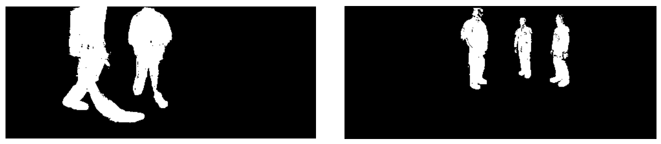


Foreground detection using Mixture of Gaussian and low thresholding results misclassification (shadows, noise).

Less noise in background modeling based on [17], but still detected shadows pixels.



There is a good shadow detection in [18], but still image noise.



Our approach with proposed shadow and foreground detection. Noise as well as shadows can be better suppressed.

Fig. 15. Evaluation of our approach with different background models and shadow detection algorithms.

After successful and precise extracting of traffic participants (foreground) object classification and recognition would be the next step for consequent task. Figure 16 demonstrates detected road users and the prediction of possible collision using the object's image positions as well as their trajectories.

V. CONCLUDING REMARKS

We proposed algorithms for robust background estimation and foreground detection in gray scale images captured by omnidirectional cameras embedded inside a smart car door. Our

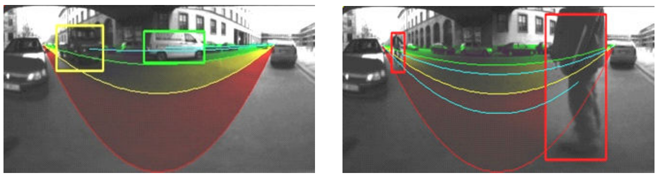


Fig. 16. Detected traffic participants and road user.

algorithms segment the foreground based on Kalman-filtering as a first step. Shadow pixel candidates are determined using NCC, and a refinement is carried out using ZNCC to distinguish between foreground pixels with small differences from background or shadow pixels. In order to reduce the influence of illumination changes that cause foreground detection to fail, illumination changes were detected by local search windows and compensated for by updating the background model. The proposed algorithms were successfully implemented to track objects like walking humans, motorbikes, bicycles and cars. They were parallelized to obtain attractive speedups on multi-core processors. Finally, they were evaluated against other state-of-the-art methods and showed enhanced foreground detection in our setting. We believe that further studies on optimally selecting the different parameters associated with the algorithms are necessary to make them work better in a wide variety of traffic scenarios and lighting conditions. Another interesting direction to explore would be to implement our algorithms on small form-factor embedded platforms (e.g., ones made up of a heterogeneous collection of reconfigurable and general-purpose processors) which blend well with a standard automotive electronics setup. Such platforms might also lead to more efficient/fast implementations and therefore better reaction times for the smart door.

Acknowledgment

The authors would like to thank Michelle Karg and the reviewers for their comments which have greatly improved the paper.

References

[1] European Commission, "Europeans information society, driver-assistance systems," *t*, 2009. [Online]. Available: http://ec.europa.eu/information_society/activities/intelligentcar/technologies/das/index_en.htm

[2] M. Strolz, Q. Mühlbauer, C. Scharfenberger, G. Färber and M. Buss, "Towards a generic control system for actuated car doors with arbitrary degrees of freedom," *In Proceedings of IEEE Intelligent Vehicles Symposium 2008, Eindhoven, The Netherlands*, pp. 391–397, 2008.

[3] R. Lawrence, Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition." *Proceedings of the IEEE*, vol. 77, no. 2, p. 257286, 1989.

[4] R. Brunelli, "Template matching techniques in computer vision: Theory and practice," *Wiley, ISBN 978-0-470-51706-2 ([1] TM book)*, 2009.

[5] O. Achler, M.M. Trivedi, "Real-time traffic flow analysis using omnidirectional video network and flatplane transformation," *Computer Vision and Robotics Research Laboratory - University of California, San Diego*, 2002.

[6] T. Ghandi, M.M. Trivedi, "Motion based vehicle surround analysis using an omni-directional camera," *In IEEE Intelligent Vehicle Symposium*, 2004.

[7] I. Haritaoglu, D. Harwood, L.S. Davis, "W4: Who? when? where? what? a real-time system for detecting and tracking people." *In Proceedings of the third IEEE International Conference on Automatic Face and Gesture Recognition.*, pp. 222–227, 1998.

[8] N. Friedman, S. Russel, "Image segmentation in video sequences: A probalistic approach." *In Proceedings of the 13th Conference on Uncertainity in Artificial Intelligence*, 1997.

[9] K.-P. Karman, A. Brandt von, "Moving object recognition using an adaptive background memory," *In V. Cappellini (ed.), Time-varying Image Processing and Moving Object Recognition, 2, Elsevier Publishers B.V., Amsterdam, The Netherlands*, pp. 297–307, 1990.

[10] A. Elgammal, R. Duraiswami, D. Harwood, L. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proceedings of the IEEE*, vol. 90, no. 7, pp. 1151–1163, 2002.

[11] D. Stenger, V. Ramesh, N. Paragios, F. Coetzec, J.M. Buhmann, "Topology free hidden markov models: application to background modeling." *In Proceedings of the International Conference on Computer Vision*, 2001.

[12] Z. Zivkovic, F.V. Heijden, "Efficient adaptive density estimation per image pixel for task of background subtraction." *In Pattern Recognition Letters.*, pp. 773–780, 2006.

[13] Z. Tang, Z. Miao, "Fast background subtraction and shadow elimination using improved gaussian mixture model," *Proceedings of the IEEE International Workshop on Haptic, Audio and Visual Environments and Games, HAVE 2007*, pp. 38–41, 2007.

[14] H. Ardoe, R. Berthilsson, "Adaptive background estimation using intensity independent features," *In Proceedings of the International Conference on the 17th British Machine Vision Conference*, 2006.

[15] H. Bhaskar, L. Mihaylova, S. Maskell, "Background modeling using adaptive cluster density estimation for automatic human detection," *In Lecture Notes in Informatics, GI Jahrestagung (2)*, pp. 130–134, 2007.

[16] J. Zhong, S. Sclaroff, "Segmenting foreground objects from a dynamic textured background via a robust kalman filter," *In Proceedings of the International Conference on Computer Vision, October*, 2003.

[17] C.Ridder, O. Munkelt, H. Kirchner, "Adaptive background estimation and foreground detection using kalman-filtering," *In Proceedings on International Conference on ICRAM*, pp. 193–199, 1995.

[18] J. Jacques, C. Jung, S. Musse, "Background subtraction and shadow detection in grayscale video sequences," *In Proceedings of the 18th Brazilian Symposium on Computer Graphics and Image*, pp. 189–196, 2005.

[19] S. Baker, S. K. Nayar, "A theory of single-viewpoint catadioptric image formation," *International Journal of Computer Vision*, vol. 35, no. 2, pp. 175–196, 1999.

[20] S. Baker, S. Nayar, "Single viewpoint catadioptric cameras," *Monographs in Computer Science*, pp. 39–72, 2001. [Online]. Available: Internet,http://citeseer.ist.psu.edu/baker01single.html

[21] D. Scaramuzza, A. Martinelli, R. Siegwart, "A flexible technique for accurate omnidirectional omnidirectional camera calibration and structure from motion," *In Proceedings of the Fourth IEEE Conference on Computer Vision Systems*, 2006.

[22] A. Martinelli, D. Scaramuzza, R. Siegwart, "A toolbox for easily calibration omnidirectional cameras," *In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2006.