Efficient detection of crossing pedestrians from a moving vehicle with an array of cameras

Gianni Allebosch,* **David Van Hamme**, **Peter Veelaert**, and **Wilfried Philips**
imec - Ghent University, TELIN-IPI, Department of Telecommunications and Information Processing, Ghent, Belgium

Abstract. We describe a method for detecting crossing pedestrians and, in general, any object that is moving perpendicular to the driving direction of the vehicle. This is achieved by combining video snapshots from multiple cameras that are placed in a linear configuration and from multiple time instances. We demonstrate that the proposed array configuration imposes tight constraints on the expected disparity of static objects in a certain image region for a given camera pair. These regions are distinct for different camera pairs. In that manner, static regions can generally be distinguished from moving targets throughout the entire field of view when analyzing enough pairs, requiring only straightforward image processing techniques. On a self-captured dataset with crossing pedestrians, our proposed method reaches an F_1 detection score of 83.66% and a mean average precision (MAP) of 84.79% on an overlap test when used stand-alone, being processed at 59 frames per second without GPU acceleration. When combining it with the Yolo V4 object detector in cooperative fusion, the proposed method boosts the maximal F_1 scores of this detector on this same dataset from 87.86% to 92.68% and the MAP from 90.85% to 94.30%. Furthermore, combining it with the lower power Yolo-Tiny V4 detector in the same way yields F_1 and MAP increases from 68.57% to 81.16% and 72.32% to 85.25%, respectively. [DOI: 10.1117/1.OE.62.3.031210]

Keywords: autonomous vehicles; pedestrian detection; multicamera; motion segmentation; epipolar geometry.

Paper 20221080SS received Sep. 19, 2022; accepted for publication Nov. 28, 2022; published online Dec. 16, 2022.

1 Introduction

Crossing pedestrians pose a potential risk for any vehicle, notably in dense, urban environments. Automatically detecting these people quickly and robustly and thereby avoiding collisions is paramount for autonomous driving assistance systems (ADAS) and the further development of autonomous vehicles. Broadly, there are two ways to detect moving pedestrians with a camera setup. The first is to detect pedestrians in each single camera frame separately. This is an object class-oriented approach, i.e., distinguishing image regions corresponding to the "pedestrian" class from other object classes. The second approach is to compare multiple frames, taken at different time instances. Local changes between these frames often correspond to moving targets. These changes are thus relevant features for motion segmentation. In this paper, we describe a novel and very efficient method to tackle this specific motion segmentation problem, and we demonstrate that fusing both ways yields significant performance benefits over using a single detector.

For a video sequence that is produced by a static camera, motion segmentation (also called foreground-background segmentation) is a relatively easy challenge. Because the background typically remains static as well, a motion segmentation algorithm can build a detailed background model for each individual pixel or each pixel group by observing a relatively short sequence of video frames. By incorporating more sophisticated statistical techniques, motion

*Address all correspondence to Gianni Allebosch, gianni.allebosch@ugent.be

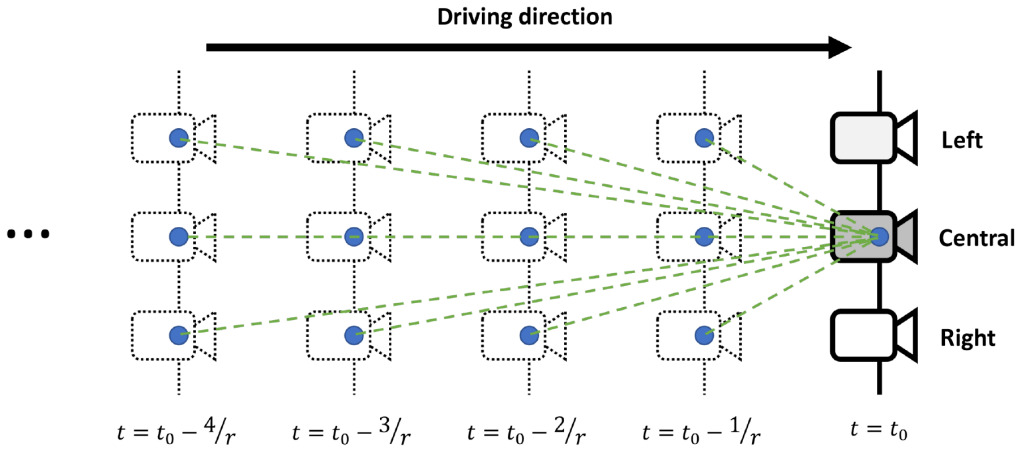


Fig. 1 Proposed camera array setup, example with three cameras. The cameras are placed in a linear configuration on a moving vehicle. Multiple virtual camera pairs can be generated w.r.t. a chosen reference camera (gray) by selecting different cameras at different time instances. r is the frame rate in frames per second.

segmentation algorithms can also handle small amounts of camera jitter, changing illumination, and continuous changes in the background such as flowing water or waving tree leaves.

Motion segmentation from a moving camera remains a challenging problem, however. Objects that are stationary with respect to the scene do appear to be moving in the video sequence and at different rates depending on the distance. This phenomenon is called parallax. To build a useful background model, the parallax due to the camera motion has to be compensated for. Accurate parallax compensation requires information about the camera's motion and the scene structure, which has to be inferred from the video frames at an impractically high computational cost (similar to high accuracy depth from stereo) or acquired directly through nontrivial fusion with additional sensors (e.g., lidar). Strategies for simplified approximate motion compensation exist, but even small inaccuracies in the compensation quickly degrade the background model, resulting in many inaccurate moving object detections.

Although the traditional motion segmentation methods can also be applied to dynamic cameras, the results are not satisfactory. Acceptable results have only been obtained when (i) the camera moves very slowly and (ii) the motion of the camera is restricted to pan or tilt while its projection center remains fixed, e.g., on the PTZ sequences of the ChangeDetection.NET 2014 dataset.[1] Although we can compensate relatively easy for pan/tilt, it is much more difficult to compensate for the disparity caused by parallax because the inherent dependency on object depth typically causes a wide variety of observed disparities within a given scene. Clearly, compensating for the disparity due to parallax will be much easier when the disparity is small for the entire field of view, i.e., on the order of a few pixels.

Alternatively, over the last few years, deep learning-based approaches have proven their potential to vastly outperform classical techniques in many fields of computer vision, with object-of-interest detection and classification being notable examples. Modern neural network-based detectors are dominating the leader-boards on well-known benchmarks such as KITTI[2] or NUSCENES.[3] However, researchers have realized that a state-of-the-art performance on public datasets does not directly translate to a well-generalizable solution, e.g., in the case of pedestrian detection.[4] There is significant potential for over-fitting, which can be problematic in scenarios with, e.g., lighting/scenic differences or partial occlusion.

Our proposed solution is fundamentally different from existing methods. Instead of applying sophisticated models to compensate for the parallax due to ego motion, we propose arranging and combining the cameras in such a way that the parallax becomes small and manageable in well-defined subregions of the image. The key to our solution is the combination of image pairs from both different time instances as well as different physical cameras, which are placed in a (linear) array configuration, to obtain these subregions. Figure 1 shows a high-level overview of our proposed setup.

With this setup, the effect of parallax is easier to compensate for, regardless of which specific motion segmentation method is used. Because the parallax from motion is reduced, motion compensated background models become much more robust and facilitate a better distinction between static and dynamic feature points using epipolar and structural constraints. In fact, because the parallax is small, a simple motion segmentation method with very limited motion compensation is already valuable, requiring less processing power and/or cheaper hardware cameras than existing solutions. Regular (RGB) cameras are both ubiquitous and generally cheaper than other sensors such as lidar scanners and short-wave infrared or long-wave infrared thermal cameras. Moreover, in its most basic form with one stereo camera pair, the necessary recording hardware is already present in many modern vehicles. However, we note that this does not exclude the proposed low parallax method being efficiently combined with more computationally demanding systems in which the additional hardware and/or processing cost is not a limitation.

In this paper, we specifically demonstrate a novel approach for detecting moving objects from a moving vehicle with an array of cameras. We further develop, extend, and validate the ideas coined by Veelaert et al.[5] The main contributions of this paper are as follows:

i. We present a theoretical framework for obtaining regions of low disparity between camera pairs, given a known camera configuration, the vehicle speed, and (potentially) the distance to the closest object in front of the vehicle.
ii. We demonstrate a very efficient algorithm for detecting moving objects that can be implemented with only simple vector/matrix type calculations, such as pixel shifts, absolute differences, summation, and thresholding.
iii. We experimentally verify the performance of our techniques on a self-recorded dataset, and we demonstrate the potential as both a stand-alone detector and in cooperative fusion with a state-of-the-art neural network-based pedestrian detector, Yolo V4.[6] The stand-alone detector is shown to outperform a lightweight version of Yolo (coined Yolo-Tiny) on a bounding box overlap test, while still running entirely on an 11th generation Intel i7 CPU at 59 frames per second. There was no GPU acceleration for the camera array detector. The fused method is demonstrated to boost the detection scores of the Yolo V4 and Yolo-Tiny V4 significantly.

The paper is structured as follows. In Sec. 2, we present a brief overview of techniques in the literature that attempt to estimate moving objects, mostly focusing on non-static cameras. Next, in Sec. 3, we describe our main idea: how to exploit a multicamera configuration to obtain these regions of low disparity. This insight leads to our proposed motion detection algorithm, which is described in Sec. 4. In Sec. 5, the method is thoroughly tested on a traffic sequence that was recorded in an urban environment. We conclude the paper and discuss future prospects and remaining challenges in Sec. 6.

2 Related Work

The goal of vision-based motion segmentation (also called foreground/background segmentation) is to find the silhouettes of all moving objects in a video sequence. A comprehensive survey of the traditional and recent methods for motion segmentation can be found in the survey of Bouwmans.[7] Below, we provide a more limited overview, highlighting the most relevant techniques within the scope of this paper.

2.1 *Motion Segmentation with Stationary Cameras*

The simplest way to detect motion is to subtract two consecutive image frames. The reasoning behind this method is that, because the color of background pixels will typically not change, only foreground pixels will have a large value in the difference image. This naive approach rarely is sufficient because of illumination changes, camera noise, camera jitter, and changes in the background. The statistical mixture of Gaussians (MoG) method was one of the first methods that could handle small periodic changes in the background, such as waving trees and flags, flowing water, or smoke plumes.[8] This method models the color of each pixel as a mixture of Gaussian

distributions. Static background regions correspond to a small number of narrow Gaussian components, whereas foreground pixels are less predictable and correspond to wider Gaussian distributions. Extensive overviews of recent versions of MOG and related statistical background models can be found in the survey of Bouwmans et al.[9]

Another improvement came in the form of adaptive background maintenance. The basic idea is to update the model more rapidly and to raise the detection threshold automatically in dynamic regions.[10,11] These regions are typically characterized by a larger deviation between the input and the background model or from the detection of "blinky" or isolated foreground pixels. The aforementioned strategies can also be successfully applied to PTZ cameras; the background model can be accurately transformed during camera rotation with a specific form of the homography matrix.[12] However, when the camera also physically moves to another location during the recording time, additional strategies are needed, as is discussed below.

2.2 *Motion Segmentation with Moving Cameras*

The main approaches for vision-based motion segmentation from a moving camera can be classified into three distinct categories. All approaches described in this section assume that the images are acquired by a monocular system, i.e., one moving camera.

2.2.1 *Foreground/background segmentation with compensation for generic motion*

These methods analyze the optical flow between consecutive image frames to derive the optical flow of the background specifically. To avoid contamination of the background model, several schemes, such as two-step optical flow analysis[13] and using a dual mode single Gaussian model,[14,15] have been proposed. The advantage of these methods is that the camera motion may be arbitrary. An important disadvantage is that the background model will still be contaminated in complex environments. When the moving objects occupy a large part of the images or when the camera motion is rapid relative to the frame rate, accurate feature correspondences for fine-grained flow analysis become difficult to obtain.

2.2.2 *Motion clustering*

These methods track features across video frames and attempt to cluster them into different motion patterns, with the dominant pattern assumed to represent background motion (i.e., due to the motion of the camera itself) and any minority patterns to represent moving objects within the scene. Some methods apply high-level clustering techniques (e.g., spectral clustering) to find groups of features that may belong to the same moving object. Multibody structure-from-motion (MBSfM) techniques assume that all moving objects are rigid and group trajectories according to how well they match the motion of a rigid body.[16] Because MBSfM assumes rigidity, moving pedestrians and cyclists cannot be reliably detected.

2.2.3 *Geometric constraint-based methods*

These methods are based on the observation that the motion of each feature point that belongs to the background has to satisfy an epipolar constraint. Specifically, points in the image corresponding to static objects always appear to move on a well-defined line, defined by the point itself and another specific point, the epipole, that is determined by geometric properties. Any feature point that does not satisfy the epipolar constraint must therefore be part of a moving object. Unfortunately, the epipolar constraint by itself cannot detect all moving objects. An important degenerative case is for objects moving parallel to the vehicle from which the recording is made, e.g., when the recording vehicle is following another vehicle. In that case, the motion of any feature point on the vehicle ahead will also lie on an epipolar line. To cope with such degenerative cases, additional constraints have been added: flow vector bounds,[17–19] modified structure consistency,[20] or algebraic three-view constraints. Another important drawback of these methods is their vast computation time. We note that in the proposed method we use similar

geometric constraints; however, we are able to avoid many costly operations in a multi-camera setup. This idea is further fleshed out from Sec. 3 onward.

2.3 *Moving Object Detection Without Motion Segmentation*

By far the most popular approach in recent years, which avoids geometric reasoning and works for non-rigid objects, is detection-by-recognition, typically using neural networks. In this case, certain types of objects (pedestrians, bicycles, and cars) are first detected and then tracked while they move. Notable examples of modern, state-of-the-art object detectors are Faster RCNN,[21] Yolo V4,[22] Scaled Yolo V4,[23] and EfficientDet.[24]

In spite of its popularity, this approach has several disadvantages. Only a limited class of known objects can be detected, i.e., those that have been "learned" by the system. Similarly, the fact that these networks are trained from examples means that they are vulnerable to errors related to occlusion, poor illumination conditions, difficult weather, etc. if these cases are under-represented in the training dataset. Furthermore, high-performing networks typically require high-end GPU or TPU support for (close to) real-time applications and/or have nontrivial memory requirements. Hence low-power solutions were developed (e.g., Yolo-Tiny[25] and Multibox SSD[26]), but the achieved detection performance of these networks is significantly inferior to the full-grown detectors. Furthermore, tracking an object only becomes reliable when the tracking is done over several frames, which is highly undesirable for fast detection in autonomous driving applications. Finally, learning the object models requires large datasets and, as mentioned previously in the introduction, the models have poor carryover to circumstances outside those of the dataset on which they were trained. This degrades detection performance, e.g. in strongly different lighting conditions or when objects are only partially visible.

In conclusion, there is currently no optimal solution for detecting moving objects from a moving platform that satisfies all of the requirements for satisfactory application in autonomous driving. There is still room for improvement, notably on the more challenging scenarios. In this paper, we propose an efficient method that tackles many of the current limitations of the state-of-the-art listed above, specifically for detecting crossing objects such as pedestrians.

3 Main Idea

Our key idea is to use a linear array of cameras, collect several image frames while the vehicle moves, and combine certain regions across image frames such that the disparity between the regions is minimal. By reducing the amount of disparity, many of the common difficulties of motion segmentation from a moving vehicle disappear. In this section, we demonstrate how careful geometric analysis leads to these conclusions, building from an example of a single camera pair.

3.1 *Disparity of a Single Camera Pair*

Disparity is the measurable effect of parallax that occurs between two images. When static objects are observed from different viewpoints, they typically appear at different locations on the image plane. The exact observed locations depend both on the orientation and position of the cameras and on the object depth w.r.t. the cameras. To be able to discriminate between stationary and non-stationary objects, we have to determine the disparity of stationary objects as accurately as possible. In this section, we present a detailed analysis of the disparity for cameras on a moving platform.

We use the basic pinhole model for each camera, and for convenience we assume that all cameras have the same focal length f. We compute the disparity for an arbitrary pair of camera positions. We also assume that the viewing directions are parallel to each other, i.e., the cameras are translated but not rotated relative to each other. Note that, with camera calibration (Sec. 4.1), a nonparallel orientation could be detected and the videos could be rectified if necessary.

Let $\mathbf{X} = (X, Y, Z)$ be a point in 3D space. When two snapshots are taken from two different camera positions of the same scene, the projection of $\mathbf{X}$ on the two image planes will generally

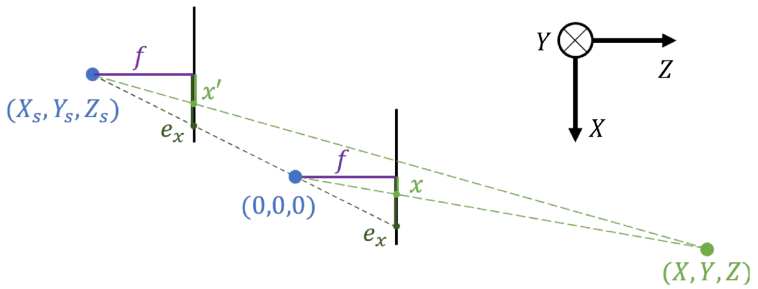


Fig. 2 Top view schematic overview of a single camera pair observing a point $\mathbf{X} = (X, Y, Z)$, where both cameras are aligned. The second camera is displaced by (X_S, Y_S, Z_S) w.r.t. the first camera. The black solid lines represent the image planes. The epipoles are found on the intersection of baseline (i.e., the line joining the camera centers) with the respective image planes.

not be at the same position. Due to parallax, there will be an offset, or disparity, between the two projections of $\mathbf{X}$. Figure 2 shows a schematic top view of the situation.

Let $\mathbf{x}'$ denote the image of $\mathbf{X}$ as seen by the camera with projection center $\mathbf{C}_S = (X_S, Y_S, Z_S)$ and $\mathbf{x}$ denote its image in the reference camera, which is positioned at the origin $\mathbf{C}_R = (0,0,0)$. As shown in Fig. 1, we assume that the reference camera is in front of the other camera; therefore $Z_S < 0$. A straightforward calculation shows that, for a static object at (X, Y, Z), the disparity is

$$(\Delta_{\text{stat},x}, \Delta_{\text{stat},y}) = (x - x', y - y') = -\frac{f}{Z - Z_S}\left(X\frac{Z_S}{Z} - X_S, Y\frac{Z_S}{Z} - Y_S\right). \tag{1}$$

The disparity is zero when $XZ_S = X_S Z$ and $YZ_S = Y_S Z$. Both equalities hold when $\mathbf{X}$ lies on the baseline of the two cameras. A closer analysis of Eq. (1) reveals the following:

i. The disparity is zero for any point on the common line of sight, that is, any point of the form $(X, Y, Z) = (kX_s, kY_S, kZ_s)$, provided $k \neq 0,1$.
ii. The x-component of the disparity is independent of any of the y-coordinates, and vice versa.
iii. The components of the disparity vector are linear functions of X and Y when Z is fixed.
iv. The disparity of the more distant static objects is smaller than the disparity of the closer objects, when considering a given line of sight.

Equation (1) gives the disparity between the two image points for a static object at location (X, Y, Z). We can also express the disparity as a function of the location of the projection of (X, Y, Z) onto one of the image planes and the depth Z of the object. Let

$$\mathbf{x} = (x, y) = \frac{f}{Z}(X, Y), \tag{2}$$

denote the projection of (X, Y, Z) onto the image plane of the reference camera. Replacing X by xZ/f and Y by yZ/f in Eq. (1), we obtain

$$(\Delta_{\text{stat},x}, \Delta_{\text{stat},y}) = \frac{-Z_S}{Z - Z_S}\left(x - f\frac{X_S}{Z_S}, y - f\frac{Y_S}{Z_S}\right), \tag{3}$$

or equivalently

$$(\Delta_{\text{stat},x}, \Delta_{\text{stat},y}) = \frac{-Z_S}{Z - Z_S}(x - e_x, y - e_y), \tag{4}$$

where

$$(e_x, e_y) = \left(f\frac{X_S}{Z_S}, f\frac{Y_S}{Z_S}\right), \tag{5}$$

represents the epipole in the image plane of the reference camera defined by the common line of sight of $\mathbf{C}_R$ and $\mathbf{C}_S$.

We now assume that the displacement of the second camera during a time interval Δt arises from the ego-motion of the vehicle. We also assume that the principal axes of both cameras are parallel to the driving directions. Hence, for a vehicle driving in a straight line at a constant speed v, cameras with frame rate r, and difference in frame index Δk,

$$Z_S = -v\Delta t = -v\frac{\Delta k}{r}. \tag{6}$$

An inspection of Eqs. (4)–(6) leads to key insights about the observed disparity. Specifically, the disparity at image point (x, y) is

i. proportional to the vehicle speed,
ii. inversely proportional to the depth of the object with respect to the non-reference camera, i.e., $Z - Z_S$,
iii. proportional to the difference vector between the projected point (x, y) and the epipole (e_x, e_y).

We note that the individual components in the disparity vector $(\Delta_{\text{stat},x}, \Delta_{\text{stat},y})$ are independent, given the other parameters. Therefore, the disparity can be treated separately for x and y. Thus, for notational compactness and clarity and without loss of generality, we focus on the disparity along the x dimension in the remainder of this section. The behavior of the y-coordinate is analogous.

Except for the depth, all of the variables in Eqs. (4)–(6) are obtained in a straightforward manner during setup (X_S), after camera calibration (f), or directly at runtime (v, x, y, and Δt). Hence, to determine the disparity in the camera pair for a stationary point, only the object depth Z is not directly available. This is discussed in the next paragraph. We also demonstrate that, with an array of cameras, selecting appropriate camera pairs actually partially avoids the need for accurate depth information to estimate the disparity.

3.2 *Determination of Object Depth*

In this section, we identify potential sources for obtaining object depth. In the following sections, we then demonstrate that even low-accuracy depth information can already impose tight bounds on the expected disparity for static objects.

Object depth cannot be observed directly from RGB cameras. There are three main approaches to extracting this information with additional processing. The first approach is to indirectly infer the depth from image context, e.g., with a neural network architecture.[27] This approach has the drawback that, to obtain accurate depth estimates, the scene context needs to be semantically rich enough in all scenarios, which cannot always be guaranteed.

The second approach is to calculate the depth from stereo within a multicamera setup. In the "classical" approach, images from a camera pair selected from an array of cameras, but taken at the same time (as opposed to the proposed system in this paper), are compared. The disparity in a camera pair is inversely proportional to the depth, and therefore, a depth estimate can be directly inferred from the camera parameters and image feature matching. This has the main drawback that there is a trade-off between depth accuracy and the computational complexity of finding correspondences. In particular, for nearby objects, the search window in the matching step can be very large. However, if the correspondence is accurately found, the depth estimate itself can be very accurate for nearby objects, but much less so for objects that are further away. More global stereo depth estimation methods partially mitigate these issues, e.g., by enforcing additional smoothness constraints, but at the cost of larger processing and/or memory demands.[28,29]

A final approach is to use separate depth sensors, such as ultrasonic sensors, radar, or lidar systems. From these options, lidar is generally the most suitable choice regarding accuracy in both range and azimuth spatial dimensions, although the spatial resolution in commercially available systems is low compared with modern RGB camera resolutions. Therefore, additional

upscaling operations are required when both accurate and dense characteristics (i.e., a depth value for every corresponding camera pixel) are necessary.[30]

Fast, low-complexity techniques are naturally preferred for real-time processing. Therefore, it is beneficial to be able to cope with approximate but faster-to-obtain depth information. A notable example is an implicit depth map that only comprises the distance to the closest object in front of the vehicle, i.e., a free zone in front of the camera. It is sufficient to select the minimal distance from detected objects above the ground plane (e.g., in lidar or radar reflections). This avoids the need for dense upsampling.

Another valuable alternative is to manually set the free zone equal to the braking distance of the vehicle. This coincides with the detection of pedestrians before it is too late. Therefore, such a setting has high potential for avoiding collisions with vulnerable road users. Note that objects (also static ones) that are within the braking distance could potentially cause false positive detections in this manner; however, these objects should have been detected earlier to avoid potential collisions. Therefore, we consider these cases to be less relevant within the specific scope of this paper, albeit still paramount for autonomous driving in general.

A sensitivity analysis for Eq. (4) reveals that, for a camera pair in our array setup,

$$\frac{d\Delta_{\text{stat},x}}{dZ} = \frac{Z_S}{(Z - Z_S)^2}(x - e_x). \tag{7}$$

The sensitivity to errors for the object depth is inversely proportional to the square of its actual value (with respect to the second camera). Therefore, the closest object is the most relevant, and obtaining this information is more impactful to the disparity estimate compared with the depth values for other objects in the scene.

Furthermore, the sensitivity is also proportional to $(x - e_x)$, i.e., the distance from the epipole. Hence, a system that is tailored to operate with image regions close to these epipoles has very little depth sensitivity: a large uncertainty on the depth measurement only results in a low uncertainty about the expected disparity. Thus, for image patches close to the epipole, even crude depth approximations (e.g., one free zone with a constant depth for every time instance) yield accurate estimations of the expected, i.e., very small, disparity.

3.3 *Regions of Limited Disparity for Static Objects*

The disparity always gets smaller when an object is further away from the cameras. As discussed in Sec. 3.2, we assume that there is a free zone with a depth d in front of the vehicle in the reference camera. Hence, for a certain camera pair, the distance along the depth (Z) direction between the second camera and the object is at least d. Therefore, $|\Delta_{\text{stat},x}|$ will always be smaller than or equal to the maximal disparity $\Delta_{\text{max},x}$, defined as

$$\Delta_{\text{max},x} = \frac{-Z_S}{d - Z_s}|x - e_x|. \tag{8}$$

Furthermore, we observe that the sign of the components of the disparity vector only depends on the location w.r.t. the epipole, i.e., points to the right of the epipole will always have a positive x-component for the disparity, and vice versa. Hence, objects that are observed at an image point $\mathbf{x}'$ in one camera will appear within a well-defined vertical strip S_x in the reference camera, defined as

$$S_x = \{x : 0 \leq \text{sgn}(x - e_x)(x - x') \leq \Delta_{\text{max},x}\}. \tag{9}$$

Similarly, we define S_y as the set of vertically bounded possible image point locations, limited by the maximal disparity at a given vertical distance w.r.t. the epipole. Thus, the image of a static object in the first camera at $\mathbf{x}$ will appear in the reference camera within a rectangle $S = S_x \cap S_y$, defined by the intersection of two infinite strips.

The bounds in Eq. (9) allow us to define regions of limited disparity in the image plane of the reference camera that only depend on the position of the image point relative to the epipole.

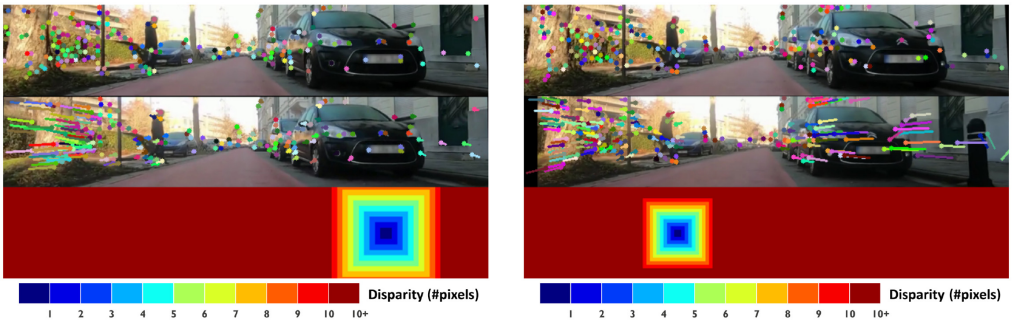


Fig. 3 Examples of regions of limited disparity for two image pairs, with the same reference image. Sparse optical flow[31] was calculated for illustrative purposes. The vehicle was driving at 2.6 m/s. Top row: reference image from central camera; middle row: image captured from one of the side cameras' Δk frames before the reference image; bottom row: maximal disparity (#pixels) for static objects at a minimal depth of 4 m, clipped at a maximum of 10 pixels of disparity. The epipoles are situated in the middle of the concentric squares. Note that the optical flow vectors for static objects are always small for static objects that are further away from the camera or with projections that are close to the epipoles. Note that for larger Δk, the epipoles are located closer to the center. (a) Left camera background, $\Delta k = 16$ frames; (b) right camera background, $\Delta k = 25$ frames.

The limits will only hold for objects that do not lie in the free zone. Examples of regions of limited disparity in a realistic setup are shown in Fig. 3.

So far, we demonstrated that the closer the image points lie to an epipole, the smaller and less sensitive to object depth the observed disparity for static objects is. Hence, in the region surrounding this epipole, the observed disparity is very predictable, and consequently, it is relatively easy to find matches for static objects. Ideally, the epipoles and the aforementioned regions around them are spread out across the image. This can be obtained by considering multiple image pairs, e.g., by choosing one frame from one of the cameras as the reference frame at a given time instance and then creating multiple the image pairs by combining this reference frame with multiple frames from all cameras. Specifically, in a video recording with multiple cameras on a moving platform, comparing frames from different cameras (different baseline X_s) and from different time intervals (different forward moved distance $|Z_s| = v\Delta t$) leads to different locations of the epipoles. This observation naturally leads to the first key idea behind our approach: the combination of different frames and cameras on a moving platform leads to very efficient disparity analysis.

The elements that are essential to applying this idea in (semi-)autonomous setups are as follows:

i. The topology of the camera array can be optimized to minimize the disparity caused by the parallax at prescribed regions in the image.
ii. For each point in the scene, there is always at least one pair of cameras for which the disparity is minimal. The selection of this pair does not depend on the distance between the cameras and the object, but only on the location of this point with respect to the epipoles.
iii. There is a simple method for automatically choosing image regions across time and across cameras that yields minimum disparity. More specifically, it is possible to construct a discrete 2D map such that we can determine the most appropriate camera pair for each pixel by a simple look-up operation.

3.4 *Disparity of Moving Objects*

In the previous paragraphs of Sec. 3, we derived formal expressions for the disparity in a single camera pair. We implicitly assumed that the point (X, Y, Z) does not physically move during the capturing of both images. For moving targets in the scene, the disparity observed in the cameras is actually a superposition of the effects from ego-motion of the cameras and the motion of the target itself. If we can distinguish these effects in the cameras, it is possible to detect moving

objects. Specifically, any observed disparity outside the bounds in Eq. (9) must be caused by a moving object.

Although the analysis can be repeated in a more general setting, to demonstrate the validity of our approach, here we focus on one specific, important scenario in which the moving target is moving parallel to the image plane or, equivalently, perpendicular to the driving direction. For example, this occurs when a pedestrian is crossing the road and crossing the path of an oncoming vehicle in the process (cfr. Car to vulnerable road user scenarios in Euro NCAP tests[32]).

Let $\mathbf{X_0} = (X_0, Y, Z)$ be the original location of the target in 3D space. After moving parallel to the image plane for some time, the target reaches a new location denoted by $\mathbf{X_1} = (X_1, Y, Z)$. From the pinhole camera model, the image point displacement w.r.t. the reference camera, only considering the target's motion, is expressed as

$$(\Delta_{\text{virt},x}, \Delta_{\text{virt},y}) = \frac{f}{Z}(X_1 - X_0, 0) = \frac{f}{Z}\left(v_t \frac{\Delta k}{r}, 0\right), \tag{10}$$

where v_t is the transversal speed of the target and $\Delta_{\text{virt},x} = x - x''$ the resulting disparity component from its motion, i.e., its apparent motion to x from a virtual location in image x''. The disparity component from the ego-motion is again expressed with Eq. (4) by substituting x for x'' and adding $\Delta_{\text{virt},x}$. Hence, the total observed disparity of the moving target along the horizontal direction is

$$\Delta_{\text{dyn},x} = \frac{-Z_S}{Z - Z_S}\left(x - e_x - \frac{f}{Z} v_t \frac{\Delta k}{r}\right) + \frac{f}{Z} v_t \frac{\Delta k}{r}, \tag{11}$$

$$= \frac{-Z_S}{Z - Z_S}\left(x - e_x + f \frac{v_t}{v}\right), \tag{12}$$

where Eq. (6) is used for simplification. We observe that the disparity for a crossing object thus depends on the ratio of its transversal speed v_t and the vehicle speed v.

Similar to the bounds for static objects, established in Eq. (9), we can also establish bounds on the disparity that will occur for targets moving at a certain minimal speed. This leads to another important observation that is key for the application of motion detection: for a given location in the image, if one can establish a certain minimal free zone with depth d in front of the camera, it is possible to distinguish targets such as pedestrians that are moving transversely faster than a certain minimal speed. Moving objects can therefore be distinguished from static objects by analyzing image correspondences. If the observed disparity is outside the bounds set in Eq. (9), it must be due to a moving object.

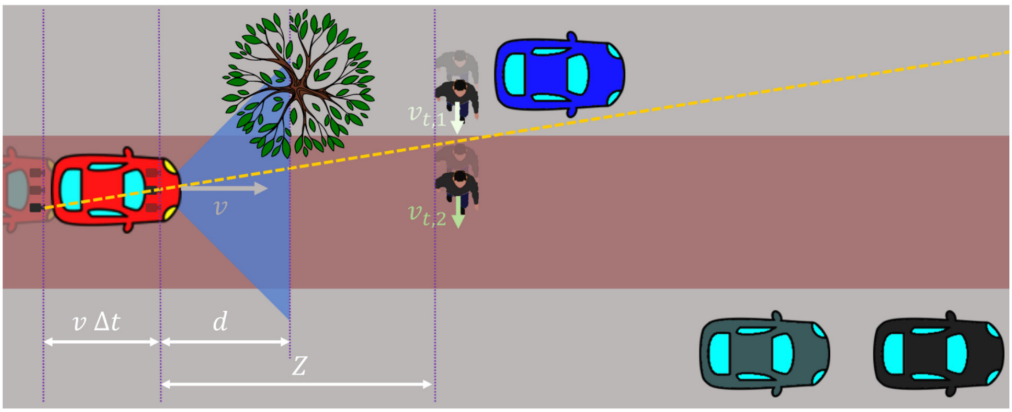
Assume that the moving object (e.g., pedestrian) is crossing the path of the vehicle from left to right, i.e. with a positive transversal speed $v_t > 0$, as shown in Fig. 4. Specifically, this crossing target can be detected in two distinct cases, depending on whether or not the object is moving away from or toward the epipole.

3.4.1 *Object moves toward epipole*

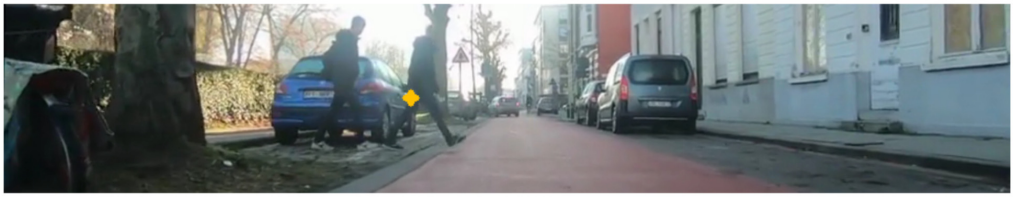
When an object appears to be moving toward the epipole in the image, instead of away from it, it cannot be static. According to Eq. (4), for a static object at the left side the epipole, i.e., $x < e_x$, the disparity $\Delta_{\text{stat},x}$ should always be negative. However, if the pedestrian's speed is sufficiently large, $\Delta_{\text{dyn},x}$ will be positive if the pedestrian moves from left to right. This is demonstrated for the left person in Fig. 4. Specifically, from Eq. (11), the disparity $\Delta_{\text{dyn},x}$ is larger than 0 when the transversal speed exceeds a certain value, i.e.

$$v_t > -v \frac{x - e_x}{f}, \tag{13}$$

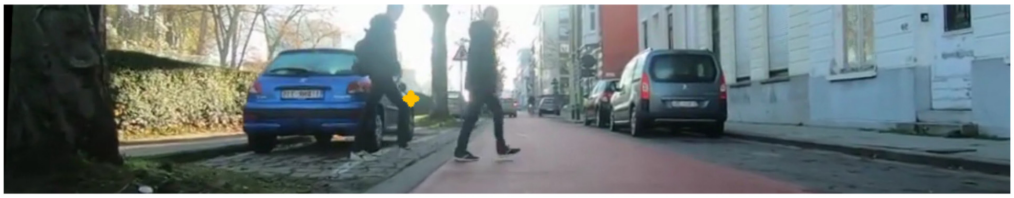
for $x \leq e_x$. Thus, the minimal crossing speed that is necessary for detection is proportional to the vehicle speed, scaled by the normalized image distance to the epipole. The minus sign indicates motion toward the epipole because $(x - e_x) \leq 0$. We note that no defined free zone in front of



(a)



(b)



(c)

Fig. 4 Example of two pedestrians crossing the street from left to right, with transversal speeds $v_{t,1}$ and $v_{t,2}$. The image pair under consideration comprises the current input frame from the reference (central) camera and an older frame in the buffer ($\Delta k = 50$) from the right camera. This image pair generates a specific epipole (orange). The first pedestrian moves toward the epipole in the video sequences, while the second pedestrian moves away from the epipole. (a) Top view schematic of the described situation; (b) older buffer input from right camera; (c) current input from the reference camera.

the camera is required to detect these moving objects because d does not appear in the expression. Hence, if no depth information is available, one can theoretically still detect moving objects by carefully selecting image pairs, such that the epipoles are located in the direction of the object motion, i.e., $e_x > x$, for objects moving from left to right, and vice versa.

3.4.2 *Object moves away from epipole faster than any potential static object*

The maximal disparity, given a free zone with depth d is expressed with Eq. (8). An object that appears to be moving faster in the image than the maximal disparity of a stationary object, that is when $\Delta_{\text{dyn},x} > \Delta_{\text{max},x}$, must be travelling at a minimal transversal speed away from the epipolar line. This is demonstrated in Fig. 4 for the right pedestrian. Specifically, the minimal transversal speed that is necessary to be detected is

$$v_t > v\frac{x - e_x}{f}\frac{Z - d}{d - Z_s}, \tag{14}$$

for $x > e_x$. The minimal crossing speed is thus again proportional to the vehicle speed and the normalized distance to the epipole. However, it is now also scaled w.r.t. a ratio determined by the

object depth and the free zone depth. Objects can be detected at lower speeds either when they are closer to the border of the free zone or when the free zone depth is larger. The first case is related to the fact that it is the apparent velocity that is important, which is higher for close pedestrians. The second case is related to the property that a larger free zone allows us to put a tighter constraint on the maximal disparity.

3.5 *Combining Different Camera Pairs*

We assume that there is a buffer of snapshots that were each captured at a given frame rate by one of the cameras. The proposed setup is capable of detecting crossing objects in a region in front of the vehicle. For any given point on the ground plane, the lowest speed at which a crossing object can theoretically be detected is the minimum value of v_t in Eqs. (13) and (14) for all camera pairs that are considered. Let a snapshot in the buffer be indexed by i. If we always consider the same reference camera for convenience as in Fig. 1, the epipole from the reference camera with snapshot i is denoted $\mathbf{e}^i = (e_x^i, e_y^i)$. The lowest speed at which the pedestrian can be detected is thus

$$v_{\min,t} = \min_i \begin{cases} -v\frac{x-e_x^i}{f} & \text{for } x \leq e_x \\ v\frac{x-e_x^i}{f}\frac{Z-d}{d-Z_s^i} & \text{otherwise.} \end{cases} \tag{15}$$

Because d is typically much larger than $|Z_s^i| = v\Delta t^i$ and the other parameters in Eq. (15) are constant for all camera pairs, the selection of i mainly depends on $x - e_x^i$. Therefore, in our proposed method, the appropriate camera pair is determined by a Voronoi tessellation w.r.t. the epipoles, that is, an image point in the reference camera $\mathbf{x}$ is always compared with the image with index i in the buffer, for which the corresponding epipole $\mathbf{e}^i$ is closest to $\mathbf{x}$. This is elaborated further in Sec. 4.2.

Figure 5 demonstrates an example of the minimal transversal speeds for a three-camera setup with a free zone of 5 m in front of the vehicle. Objects that are crossing more slowly than these minimal speeds can be confused with static objects, depending on their location in the scene. Note that the coverage obtained from the central camera alone is smaller than for the cameras on the sides because all epipoles lie at $x = 0$. Furthermore, one can observe that the coverage for detecting a crossing object is not fully symmetric for left and right, even if the baselines between the left and central cameras are the same as between the central and right cameras. However, if the object were crossing from right to left instead ($v_t < 0$), the coverage would be inverted about the depth axis. Finally, as can be observed from Eqs. (13) and (14), on the epipoles $x = e_x$, moving objects with any non-zero transversal speed can be detected, which agrees with our earlier discussion of the disparity from static objects. We note that, because only the transversal component of the pedestrian's speed (perpendicular to the vehicle trajectory) is taken into account in our analysis, the proposed method is potentially less sensitive for pedestrians that do not cross the trajectory perpendicular to the driving direction.

In the next section, we discuss how these bounds can be practically implemented to obtain a very efficient detector for crossing objects.

4 Efficient Crossing Object Detection

Finding image correspondences typically comes at a high processing cost. Instead, when using the disparity constraints discussed in Secs. 3.3 and 3.4, and selecting appropriate image pairs for every image region, detecting moving objects essentially boils down to checking pixel correspondences in a very small image patch, located around the previous image location. This approach vastly limits the search area in our proposed approach, thus sharply reducing the required computational cost in the process. Furthermore, this approach avoids the computation of explicit image features. This makes our approach suitable for applications with a strong need for real-time processing, critical for, e.g., obstacle avoidance in traffic scenarios.

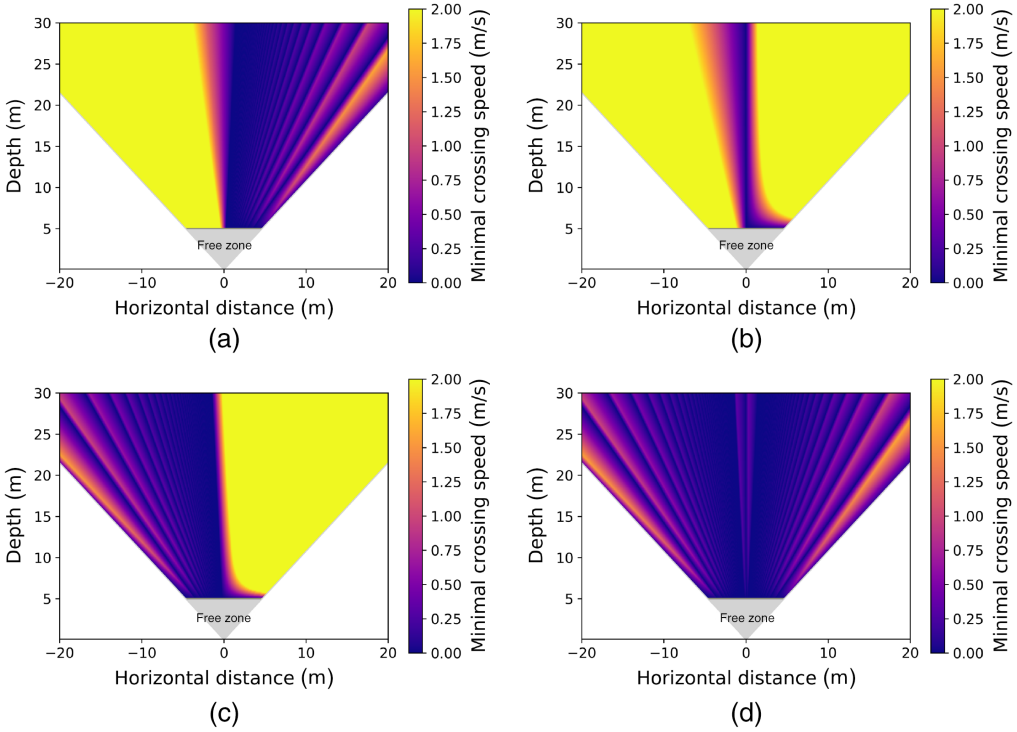


Fig. 5 Top view visualization: pedestrians can be detected in a region front of a vehicle, depending on their minimal crossing speed. A pedestrian is assumed to cross the path of the vehicle from left to right. All potential camera pairs, from different time instances, from the given camera with respect to the current time of the reference (central) camera are considered. In this example, three cameras were placed on a line, with 20 cm between the camera centers. The vehicle is driving at a constant speed of 30 km/h. The frame rate in this example is 119.88 frames per second, and the free zone is set at 5 m. The white region is outside the field of view of the central camera. The pedestrian speed in the graph was clipped at 2 m/s, meaning that in practice they would need to be travelling unrealistically fast to be detected in the yellow regions. (a) Left camera/central camera pairs; (b) central camera/central camera pairs; (c) right camera/central camera pairs; (d) all cameras/central camera pairs.

In this section, we describe our proposed method to detect moving objects. First, we explain which initial calibration and preprocessing steps are required. Then we translate the theory described previously to a practical approach for constructing a so-called background image by combining multiple frames from multiple cameras. Next, we demonstrate how the comparison of this background image with the current input from a reference camera allows us to distinguish moving and static objects in a very fast manner, with a background subtraction-based approach. Finally, we demonstrate the further potential of our proposed approach further by exploring straightforward fusion with camera-based object detectors.

4.1 *Calibration and Preprocessing*

Assuming that all cameras are pointing in the same direction as the speed vector of the vehicle, the following parameters fully determine the disparity bounds for static objects at runtime:

i. The horizontal distance between the reference camera and the second camera on the vehicle X_s
ii. The vehicle velocity v
iii. The frame rate r
iv. The focal length f
v. The horizontal image position with respect to the epipole $(x - e_x)$
vi. The free zone depth d

The focal length can be inferred from intrinsic camera calibration.[33] It can be shown from sensitivity analysis that, in our setup, an error of 3.8% or lower on the focal length amounts to an error below 1 pixel length on the estimation of the disparity. In our experiments, the errors on the focal length were below this boundary.

The baseline can be physically measured with a tape measure during the platform setup. Alternatively, it can be inferred from solving perspective-n-point for each camera pair w.r.t. known calibration objects.[34] Note that this technique can also be used to detect non-parallel orientations of the cameras.

The minimal object depth can in principle be inferred from any of the sensors mentioned in Sec. 3.2. However, as mentioned, we argue that accurate depth information is not necessary in many cases. Therefore, we make use of the free zone, which was discussed in the same section, and set it to the maximal value of the estimated braking distance and the closest object in the scene. Finally, if the vehicle speed, baseline, frame rate, and focal length are known, the image position w.r.t. the epipole can be directly measured from the image.

Note that so far we assumed that the vehicle perfectly drives in a straight line and that there is no camera rotation w.r.t. the scene or jitter between frames. In practice, this naturally will occur in most scenarios, even on (mostly) straight road sections, depending on the local traffic situation (e.g., speed bumps, slight road bends, or steering movements). It is entirely possible to extend the disparity analysis to these more complex scenarios. In this paper, we limit the application to scenarios in which these occurrences cause only minor deviations from a straight line, which can be solved by video stabilization techniques.[35]

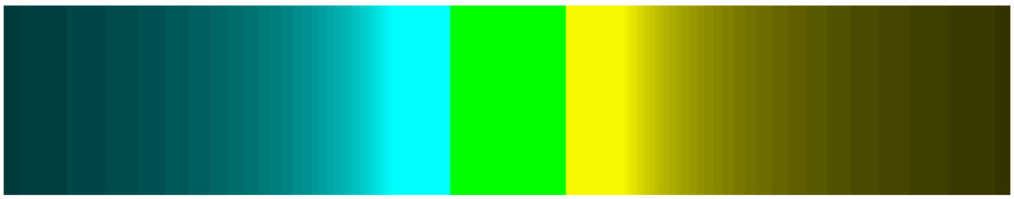
4.2 *Background Image Composition*

The goal is to detect moving objects, which can be achieved by analyzing the disparity w.r.t. the bounds set in Eq. (9). From a computational point of view, this is easiest to perform when the same types of image operations are executed on the entire image, enabling efficient array computation. In this section we explain how keeping a buffer of a certain number of past frames and selecting appropriate image patches from this buffer at runtime leads to a very efficient implementation.

Frames that are captured from different cameras/time instances w.r.t. a certain reference frame cause the epipoles to be located at different locations in the image, as can be observed from Eq. (5). Therefore, we propose keeping a buffer of the last M frames for all cameras into memory. Image patches around the epipoles are used for creating a background image at every time instance. Every frame can thus be reused multiple times for different frame intervals Δk until it is removed from the buffer. Frames that were captured more recently (smaller Δk) or frames from cameras with a larger baseline (larger X_s) w.r.t. the reference camera will result in the epipoles lying further from the image center. Therefore, increasing the buffer size will only increase the epipole density near the center of the reference frame. Furthermore, the epipoles determined by successive frames lie closer together the farther back in time the corresponding frames were captured. Hence, as long as a large enough region in front of the vehicle is covered (see Sec. 3.5), adding more frames to the buffer is no longer worth the linearly increasing memory requirements. An example of the epipole density for a given buffer size is demonstrated in Fig. 6.

The epipoles serve as the input points for a Voronoi tessellation in the image plane. In the resulting partitioning, the image plane is thus divided into subregions, with every subregion comprising all points that are closest to a certain epipole. Hence, each subregion corresponds to a point set for which the disparity for static objects is small when analyzing the image pair corresponding to that epipole. Therefore, in our approach, at each new captured frame, we construct a single background image tiling a plane with the same dimensions as the input. Each tile corresponds to the points in the respective subregion in the reference buffer. The background image lends itself to efficient array processing, which is exploited in our change detection-based detection mechanism, explained in the following subsection. Note that using a single background image essentially amounts to a frame differencing approach of background subtraction. Thus, there are drawbacks regarding ghosting and moving background objects. A more elaborate, e.g., statistical, background model could theoretically be devised. This option is not

(a)



(b)

Fig. 6 Example of the epipole density for a vehicle moving at 2.8 m/s, with a buffer of 40 frames, captured at 119.88 frames per second for three cameras with a baseline of 20 cm. The epipoles are denoted on the current input (central) image with colored crosses. Cyan, green, and yellow epipoles originate from pairwise comparison between the input image and earlier snapshots from the right, central, and left cameras, respectively. Note that all epipoles for the central camera overlap in the center of the image. The innermost epipoles from the left and right cameras originate from the oldest frames in the buffer, i.e., for $\Delta t = 40/119.88$ s. (a) Input image with superimposed epipoles; (b) Voronoi tessellation of the image plane w.r.t. to the epipoles.

explored in this paper, but the effects of these drawbacks and how they can be mitigated in an alternative fashion are discussed further.

4.3 *Change Detection by Background Subtraction*

Full disparity analysis for motion detection requires explicitly finding correspondences between the background and the input images for all pixels, calculating the disparity vectors, and finally checking the obtained vectors w.r.t. to the (typically narrow) bounds from Eq. (9). This could be obtained by analyzing, e.g., matched feature correspondences or optical flow vectors. However, the problem can be simplified even further because the exact disparity vector does not need to be known to decide whether or not a pixel corresponds to a moving object. One only needs to know whether it is within the aforementioned bounds.

Therefore, we reformulated the disparity analysis step as a pure change detection problem: given some model of the scene, changes are detected when there is no local correspondence between the current input and the model. Formally, let us denote the greyscale input image as I and the background model, in this case, a single image that was obtained as described in Sec. 4.2, as R. A pixel at image point $\mathbf{x}$ is classified as follows:

$$C(\mathbf{x}) = \begin{cases} 0 & \text{if } \exists\ \mathbf{x}' \in S : |I(\mathbf{x}) - R(\mathbf{x}')| < T \\ 1 & \text{otherwise,} \end{cases}, \tag{16}$$

where S is the region around $\mathbf{x}$, defined by the static object disparity bounds as described in Sec. 3.3, and T is a constant change detection threshold. Note that we assumed that odometry and stabilization are robust and accurate enough to fully compensate for non-linear vehicle motion. However, when this assumption does not hold, the disparities in the image could fall outside the calculated bounds, potentially impairing detection performance. This effect could be mitigated with a transformation of the input images w.r.t. the rotation of the vehicle, e.g., by warping the input images according to additional data from, e.g., visual odometry or inertial measurement unit measurements. The algorithmic performance will however be sensitive to the accuracy of this data and will require (potentially nontrivial) additional processing steps. We consider handling significant non-linear vehicle motion to be outside the scope of this paper.

Therefore, to cope with small nonlinear vehicle motions, we extend S along both image axes to both sides of the region. S is specifically extended more along the vertical direction (three pixels) than along the horizontal direction (one pixel) because we observed that vertical oscillations were more pronounced in our recordings, even after stabilization.

From a computational perspective, the only operations needed to perform this detection step are absolute differencing, thresholding, and index shifts, which can all be executed very fast in modern processing devices due to the high potential for parallelism. All pixels in the input image can essentially be processed in the same manner because the bounds are small and similar for the entire image. One can very efficiently perform these operations using matrix/array processing, e.g., on multithreaded CPU or GPU cores.

On the flipside, the very low complexity of the previously described solution, sets two notable limitations to the expected performance as a detector.

i. Selecting an appropriate threshold T is highly dependent on the appearance of the environment and the moving targets. When the targets have a locally similar appearance to the background (e.g., dark colored clothes compared with a black car), a low threshold is desirable. Conversely, the effects of illumination variance, image noise, reflections, etc. could be partially mitigated by a high threshold.
ii. So far, we implicitly used the terms "changes" and "moving objects" interchangeably. However, changes actually occur in a bidirectional fashion: a moving target covers a new region of the image, but the region where the target was previously is now uncovered, and the background also changes by becoming visible. This phenomenon is called ghosting.

These potential issues are undesirable for applications in (semi-)autonomous driving. A more optimal solution should combine the strengths of our proposed detector with those of other techniques. Therefore, in the next subsection, we investigate the potential to overcome the final limitations by fusing our proposed camera array-based detector with a neural network-based pedestrian detector, and in Sec. 5 the performances of these different approaches (camera array only, neural network only, and fusion of both) are compared.

4.4 *Extension: Cooperative Fusion with a Pedestrian Detector*

As mentioned in the introduction, current single-frame object (pedestrian) detectors have a few notable downsides, such as poor generalization to unseen scenarios. However, being nontime- and nongeometry-based solutions, they do not suffer from ghosting or potentially insufficient image stabilization, unlike the stand-alone proposed method. Furthermore, if the training set was versatile enough, they should also (mostly) be capable of handling variations in observed intensity differences between the background and the person. Therefore, there is potential in overcoming the aforementioned limitations by combining both approaches.

As an example, we consider the Yolo V4 object detector.[22] This is a dense prediction architecture, meaning that it attempts to find a label for every pixel in a given input image. The architecture uses CSPDarknetNet-53[36] as the backbone network for feature extraction. Furthermore, SPP[37] was used to increase the receptive field and PANet[38] for backbone parameter aggregation. The so-called "head network" of Yolo V4 is described extensively by Redmon and Farhadi.[6] The resulting dense predictions are matched with candidate bounding boxes and, as a final step, only the most likely detections (i.e., all candidate detections with a detection score S_d above a user-specified fixed threshold T_d) are kept. Yolo-Tiny[25] was developed with a similar design, only with much fewer layers and weights in the network.

The choice of the detection threshold poses challenges similar to the proposed method; however, the nature of the errors is different. Mostly, partial occlusion and similarities between object classes are the main causes for misclassification, which amount to missed or falsely detected pedestrians in the envisioned (semi-)autonomous driving applications.

We propose a straightforward cooperative fusion mechanism to combine the strengths of our camera array-based crossing pedestrian detector and the Yolo V4 and Yolo-Tiny V4 object detector. Essentially, Yolo's binary threshold is converted to a ternary threshold, and the camera array is used for making a decision when the detection score is neither high nor low. Specifically, we keep the network structure of Yolo V4/Yolo-Tiny V4, but only consider the detected objects in

the "person" class and replace the single detection threshold with a low threshold $T_{d,l}$ and a high threshold $T_{d,h}$. Let us denote a Yolo candidate detection bounding box Y_d with detection score R_d. We consider three distinct scenarios.

i. $R_d \leq T_{d,l}$ This is a candidate box with a very low detection score. We thus assume that there is a very low probability that there is a person present and omit the candidate from the final set of detections.
ii. $R_d \geq T_{d,h}$ This is a candidate box with a very high detection score and thus very likely to coincide with an actual person that is present in the scene. Therefore, this candidate is kept in the final set of detections. We expect only a low number of false positives to be introduced in this manner.
iii. $T_{d,l} < R_d < T_{d,h}$ These bounding boxes have a medium detection score, signaling no clear preference to either keep or discard it as a detection. Therefore, this range benefits the most from additional information: if there is detection in our camera array-based detection mask C that is (partially) overlapping with this candidate, then it is more likely that there is a crossing person. Hence, for this region, we keep the candidate if there is at least 1 pixel at image point $\mathbf{x} \in Y_d$ that is classified as motion, i.e., $C(\mathbf{x}) = 1$. Otherwise, Y_d is discarded.

To conclude this section, we note that the proposed fusion mechanism does not fully take the probabilistic nature of C into account. Because Yolo's detection score cannot be directly related to absolute differences in intensity, a more intricate, Bayesian approach requires further analysis. Exploring this any further is beyond the scope of this paper. Our main goal concerning the proposed fusion method in this paper, is to demonstrate the potential for combining the camera array with a state-of-the-art neural network-based object detector, even within a straightforward fusion framework.

5 Experimental Results

In this section, we evaluate the performance of our crossing object detection algorithm both as a stand-alone detector and fused with Yolo V4/Yolo-Tiny V4, as described in the previous sections.

5.1 *Dataset*

We captured a dataset in a city center, with three cameras facing toward the driving direction, mounted on an electric cargo bicycle. The videos were recorded at 119.88 frames per second with 3 GoPro Hero 7 cameras, at a baseline of 0.200 m between left/middle and middle/right. Additionally, the recording setup comprised a radar, used for determining the ego-velocity with the CFAR algorithm,[39] and a lidar, which is used for depth estimation. The dataset consists of 21 sequences that were each recorded on a straight road section. In every sequence, one or two people are crossing the road, either individually or together.

All sequences were stabilized with the OpenCV stabilization library, which is based on the work of Grundmann et al.[35]

5.2 *Evaluation*

In total, 33,126 frames were semiautomatically annotated for evaluation. SSD[26] and DSST[40] were used to respectively generate and track candidate bounding boxes. These bounding boxes were reinitialized at regular intervals, i.e., every 30 frames, to mitigate tracking drift. Finally, manual corrections for false, missing, or offset detections were performed on these candidate bounding boxes. The lidar was used to discard image regions closer than the braking distance (estimated at 4 m) from evaluation.

All non-crossing pedestrians and other moving road users, such as cyclists and cars, are considered to be "do not care" regions and thus do not contribute to the overall detection scores. Therefore, the results should be interpreted only as an indication of the performance for specifically detecting crossing objects, not as a general road user detector. Visual examples are shown in Fig. 7.

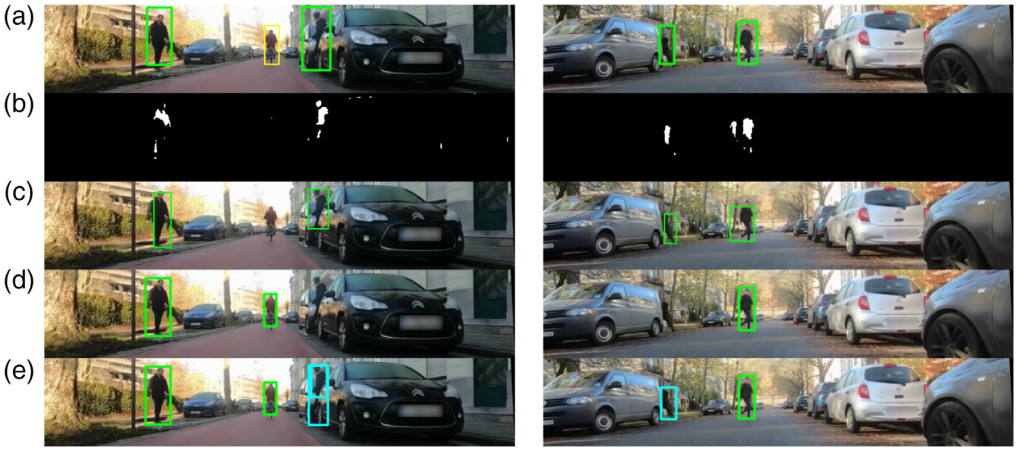


Fig. 7 Examples of processed results from the dataset for different methods. (a) Ground truth, green boxes contain positive samples, yellow boxes are do not care regions; (b) camera array, binary change detection mask C; (c) camera array, bounding boxes from change detection mask; (d) Yolo V4[22] boxes with $T_d = 0.25$; (e) Camera array + Yolo V4 fusion result, green boxes are detections above $T_{d,h} = 0.7$, cyan boxes are detections between $T_{d,l} = 0.1$ and $T_{d,h} = 0.7$.

For the proposed, stand-alone camera array method, the binary output was postprocessed to be able to compare it qualitatively and quantitatively with the bounding boxes from the neural networks: nearby foreground blobs in C are joined and evaluated as a single detected object by the bounding box around them. Consecutively, all resulting bounding boxes that were too small (i.e., smaller than 16 pixels high or 6 pixels wide) were discarded.

A test sample is considered to be a true positive when there is overlap between the candidate bounding box from the detector and an annotated bounding box. Analysis of the total number of true positives, combined with the number of false positives and negatives for different parameter settings, yields the precision-recall curves in Fig. 8(a). We also repeated the experiment for a minimal intersection-over-union (IOU) for the overlap between the ground truth and the detections. This experiment also takes the quality of the bounding boxes into account. These resulting precision recall-curves are shown in Fig. 8(b). For both experiments, we calculated the highest obtained F_1 score and mean average precision (MAP). These metrics can be found in Table 1.

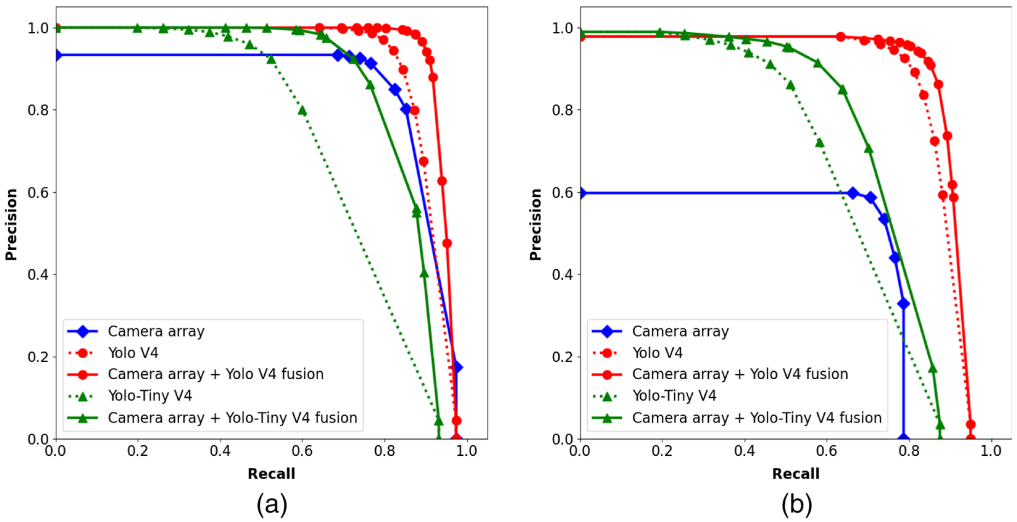


Fig. 8 Precision-recall on the self-recorded dataset. The performance of the proposed method (camera array) is compared with Yolo-Tiny and V4 Yolo V4,[22] stand-alone and in cooperative fusion, for multiple parameter configurations. (a) Overlap test; (b) minimal IOU = 0.25.

Table 1 Detection results on the self-recorded dataset. MAP and the highest obtained F_1 score ($F_{1,max}$) are given for all methods. The experiment was conducted as an overlap test (overlap with ground truth counts as true positive detection) and as a minimal IOU test, in which the minimal IOU was 0.25. Note that the IOU test places stricter constraints on the quality of the bounding boxes, hence the overall lower results.

	Overlap		IOU = 0.25	
Method	MAP	$F_{1,max}$	MAP	$F_{1,max}$
Camera array	0.8479	0.8366	0.4614	0.6406
Yolo-Tiny V4[22]	0.7232	0.6857	0.6600	0.6443
Camera array + Yolo-Tiny V4 fusion	0.8525	0.8116	0.7366	0.7290
Yolo V4[22]	0.9084	0.8786	0.8650	0.8518
Camera array + Yolo V4 fusion	0.9430	0.9268	0.8866	0.8802

The fused camera array + Yolo V4 detector obtains the highest detection scores on our dataset for both experiments. Specifically, this method obtained a maximal $F1$ score of 92.68% and an MAP of 94.30%, which is notably higher than the maximal scores for stand-alone Yolo V4. The fused camera array + Yolo-Tiny V4 method similarly has a significantly better performance than stand-alone Yolo-Tiny V4, in this case with even larger differences. The absolute scores are still lower than for the fusion with the full Yolo V4 network. The proposed camera array method, when used as a stand-alone detector, does not obtain the scores from stand-alone Yolo V4, but still outperforms Yolo-tiny V4 by a significant margin in the overlap test. However, for an IOU of 0.25, the camera array as a stand-alone method has worse performance than the other methods under investigation.

5.3 *Discussion*

Notably, the overlap test demonstrates that the proposed camera array method is able to obtain a high recall even, compared with Yolo V4. This can be attributed to a fundamental benefit of the proposed method compared with object detectors: motion can already be detected from a very small patch of moving pixels, whereas a significant portion of the object needs to be clearly visible for an accurate detection through a neural network. In other words, the proposed method is less sensitive to partial occlusion. This is valuable for the envisioned application as it makes early detection of crossing pedestrians more likely.

Conversely, Yolo-Tiny V4 and Yolo V4 obtain a higher maximal precision, causing fewer false detections. Therefore, applications that require a very low false alarm rate benefit more from using these detectors compared with the stand-alone camera array-based framework. The limitations for stand-alone usage are more clearly visible in the experiment with a minimal IOU of 0.25. Because the bounding boxes are obtained in a crude manner, the delineation is very sensitive to camouflage (e.g., parts of the pedestrian with similar colors as the background) and ghosting. Therefore, the quality of the bounding boxes themselves is lower.

However, it is clear from the results that the camera array has great potential in boosting the detections by acting as a tie-breaker for the more uncertain cases (medium confidence levels) in other object detectors. Combining neural network object detectors such as Yolo and the camera array as described in Sec. 4.4 yields a "best of both worlds" on this dataset. In essence, using the camera array detections as a secondary decision mechanism, one obtains both high precision of using the neural network with a high detection threshold and high recall of using it with a low detection threshold. The strengths of combining multiple approaches are shown in the examples in Fig. 7.

Based on these findings, we conclude that the proposed method is useful in practice, both when the amount of available processing power is limited as a stand-alone detector, to further boost the performance of an already high-performing object detector, and in combination with a

lower-power object detector. We conclude the discussion by acknowledging a few limitations of our current dataset and validation, suggesting follow-up research and development related to our proposed method could be valuable.

- The main speed of the bicycle in our dataset was limited to 25 km/h or below. Because the minimal speed (that is necessary for a pedestrian to be detected) is proportional to the vehicle ego-speed [see Eqs. (13) and (14)], the proposed method is less suitable for higher vehicle speed scenarios. Future research could comprise finding the threshold (car) speed at which the proposed method is no longer able to provide relevant feedback or further improving the performance in these scenarios. However, we argue that our proposed method is mainly useful in urban scenarios with a high number of pedestrians, in which the speed limits are low (30 km/h in many cities, e.g.). On highways, for example, the car speeds are much higher, but there are also in principle significantly fewer pedestrians that could appear.
- The dataset only comprised a maximum of two people crossing at the same time. Our proposed method should still be applicable to more people being present in the scene. The standalone proposed method essentially works on a pixel level, not on an object level. However, when combined with an object detector and when there are more people present in the scene, there is generally a higher chance of partial (dynamic) occlusion, and the overall detection scores might be lower in those cases.
- We focused on detecting crossing pedestrians specifically in our dataset. In principle, any type of (crossing) moving object can be detected with our method. However, the fused method depends on an additional object detector. Hence, if the object detector is not properly trained for detecting, e.g., small children or other road users (such as nonhumans), the fused architecture will also likely miss it.
- There was no noticeable motion in the background, except for other road users who were labeled as "do not care." If a background object is moving, the proposed standalone method will typically generate a false positive detection. This could be handled by a more complex background maintenance mechanism, e.g., from one of the methods that were mentioned in Sec. 2. However, this effect can also mostly be mitigated in the cooperative fusion method, by only selecting appropriate object classes for detection.

6 Conclusions and Future Work

We described a method for detecting crossing pedestrians and other road users with a linear array of cameras on a moving vehicle. We have shown how this setup can be exploited to extract multiple virtual camera pairs that are optimal for distinguishing true motion (perpendicular to the driving direction) from disparity caused by the ego-motion of the vehicle. We first demonstrated how this setup enforces geometric constraints on the disparity of objects in the scene. When combining multiple virtual camera pairs, a detector using these constraints can cover a significant portion of the driving area in front of the vehicle. We have also demonstrated a low processing power, practical implementation of these ideas for detecting crossing targets, based on straightforward, very fast operations such as frame differencing, thresholding, and image shifts. The current (reference) image only needs to be compared with a single background image, composed of well-chosen snapshots from different cameras and different time instances, up to a certain, albeit low, maximal disparity. The performance of the proposed method on our self-captured dataset is superior to the low power Yolo-Tiny V4 object detector on a ground truth overlap test, while still obtaining 59 frames per second without GPU acceleration. Furthermore, this paper demonstrates that, combining the proposed method with the Yolo V4 object detector, the proposed method boosts its F_1 score and MAP on this dataset significantly, from 87.86% to 92.68% and from 90.85% to 94.30%, respectively. When accurate bounding boxes are required, the camera array as a stand-alone network shows more limitations in our experiments, but it is still clearly valuable for boosting the detection scores of Yolo V4 and Yolo-Tiny V4.

A few limitations of the current incarnation can be resolved in future work. In addition to further general improvements on the desired detection rate, a few other notable remaining challenges are ghost removal; dealing with camouflage, illumination and specular reflection

invariance; and the extension to non-flat or curved road segments. Ghost removal and camouflage could theoretically be handled internally in the motion segmentation part by building a background model from multiple samples per pixel instead of just one. Illumination invariance could be obtained, e.g., by switching from intensity-based to edge-based processing.[41] Finally, with additional, accurate 3D odometry, it is possible to transform the background (or equivalently, the reference input) at runtime. However, all of these mentioned approaches require non-trivial steps to be added to the processing pipeline. Further research will determine the next steps in the further development and expansion of these ideas.

Acknowledgments

The authors would like to thank Tomas Vojir for making his software repository for KCF publicly available at https://github.com/vojirt/kcf. This research received funding from the Flemish Government under the "Onderzoeksprogramma Artificiële Intelligentie (AI) Vlaanderen" program.

References

1. Y. Wang et al., "CDnet 2014: an expanded change detection benchmark dataset," in *IEEE Conf. Comput. Vision and Pattern Recognit. (CVPR) Workshops* (2014).
2. A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Conf. Comput. Vision and Pattern Recognit. (CVPR)* (2012).
3. H. Caesar et al., "nuScenes: a multimodal dataset for autonomous driving," in *Comput. Vision and Pattern Recognit. (CVPR)* (2020).
4. I. Hasan et al., "Generalizable pedestrian detection: the elephant in the room," in *Proc. IEEE/CVF Conf. Comput. Vision and Pattern Recognit. (CVPR)*, pp. 11328–11337 (2021).
5. P. Veelaert, D. Van Hamme, and G. Allebosch, "Motion segmentation in video from non-stationary cameras," European Patent EP3803790 B1, https://register.epo.org/application?number=EP19726028 (2022).
6. J. Redmon and A. Farhadi, "Yolov3: an incremental improvement," CoRR abs/1804.02767 (2018).
7. T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: an overview," *Comput. Sci. Rev.* **11–12**, 31–66 (2014).
8. C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," in *IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recognit.*, Vol. 2, p. 252 (1999).
9. T. Bouwmans, F. El Baf, and B. Vachon, "Background modeling using mixture of Gaussians for foreground detection - a survey," *Recent Patents Comput. Sci.* **1**, 219–237 (2008).
10. P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, "A self-adjusting approach to change detection based on background word consensus," in *IEEE Winter Conf. Appl. of Comput. Vision (WACV)*, Waikoloa Beach, Hawaii, pp. 990–997 (2015).
11. G. Allebosch et al., "C-EFIC: color and edge based foreground background segmentation with interior classification," in *Computer Vision, Imaging and Computer Graphics Theory and Applications, Communications in Computer and Information Science*, J. Braz et al., Eds., Vol. **598**, pp. 433–454, Springer-Verlag, Berlin (2016).
12. G. Allebosch et al., "Robust pan/tilt compensation for foreground-background segmentation," *Sensors* **19**(12), 2668 (2019).
13. D. Kim and J. Kwon, "Moving object detection on a vehicle mounted back-up camera," *Sensors* **16**(1), 23 (2016).
14. K. M. Yi et al., "Detection of moving objects with non-stationary cameras in 5.8 ms: bringing motion detection to your mobile device," in *IEEE Conf. Comput. Vision and Pattern Recognit., CVPR Workshops 2013*, 23–28 June 2013, Portland, Oregon, pp. 27–34 (2013).
15. S. W. Kim et al., "Detection of moving objects with a moving camera using non-panoramic background model," *Mach. Vision Appl.* **24**(5), 1015–1028 (2013).
16. R. Sabzevari and D. Scaramuzza, "Multi-body motion estimation from monocular vehicle-mounted cameras," *IEEE Trans. Rob.* **32**(3), 638–651 (2016).

17. A. Kundu, K. M. Krishna, and J. Sivaswamy, "Moving object detection by multi-view geometric techniques from a single camera mounted robot," in *IEEE/RSJ Int. Conf. Intell. Robots and Syst.*, 11–15 October 2009, St. Louis, Missouri, pp. 4306–4312 (2009).
18. A. Kundu, K. M. Krishna, and C. V. Jawahar, "Realtime motion segmentation based multibody visual SLAM," in *Seventh Indian Conf. Comput. Vision, Graphics and Image Process., ICVGIP '10*, 12–15 December 2010, Chennai, India, pp. 251–258 (2010).
19. A. Kundu, C. V. Jawahar, and K. M. Krishna, "Realtime moving object detection from a freely moving monocular camera," in *IEEE Int. Conf. on Rob. and Biomimetics, ROBIO 2010*, 14–18 December 2010, Tianjin, China, pp. 1635–1640 (2010).
20. V. Frémont, S. A. R. Florez, and B. Wang, "Mono-vision based moving object detection in complex traffic scenes," in *IEEE Intell. Veh. Symp., IV*, 11–14 June 2017, Los Angeles, California, pp. 1078–1084 (2017).
21. S. Ren et al., "Faster R-CNN: towards real-time object detection with region proposal networks," in *NIPS*, C. Cortes et al., Eds., pp. 91–99 (2015).
22. A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: optimal speed and accuracy of object detection," CoRR abs/2004.10934 (2020).
23. C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "Scaled-YOLOv4: scaling cross stage partial network," in *Proc. IEEE/CVF Conf. Comput. Vision and Pattern Recognit. (CVPR)*, pp. 13029–13038 (2021).
24. M. Tan, R. Pang, and Q. V. Le, "EfficientDet: scalable and efficient object detection," CoRR abs/1911.09070 (2019).
25. P. Adarsh, P. Rathi, and M. Kumar, "Yolo v3-tiny: object detection and recognition using one stage improved model," in *Proc. 2020 6th Int. Conf. Adv. Comput. and Commun. Syst. (ICACCS)*, 6–7 March 2020, Coimbatore, India, pp. 687–694 (2020).
26. W. Liu et al., "SSD: single shot multibox detector," *Lecture Notes in Computer Science*, B. Leibe et al., Eds., Springer, Cham (2015).
27. I. Alhashim and P. Wonka, "High quality monocular depth estimation via transfer learning," arXiv e-prints abs/1812.11941 (2018).
28. P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient belief propagation for early vision," *Int. J. Comput. Vision* **70**(1), 41–54 (2006).
29. H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. Pattern Anal. Mach. Intell.* **30**(2), 328–341 (2008).
30. M. Dimitrievski et al., "High resolution depth reconstruction from monocular images and sparse point clouds using deep convolutional neural network," *Proc. SPIE* **10410**, 104100H (2017).
31. J.-Y. Bouguet, "Pyramidal implementation of the lucas kanade feature tracker description of the algorithm," Tech. Rep., Intel Corporation Microprocessor Research Labs (2000).
32. R. Schram et al., "Euro NCAP's first step to assess autonomous emergency braking (AEB) for vulnerable road users," in *24th Int. Tech. Conf. Enhanced Saf. of Veh. (ESV)*, Gothenburg (2015).
33. Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.* **22**, 1330–1334 (2000).
34. G. Terzakis and M. Lourakis, "A consistently fast and globally optimal solution to the perspective-n-point problem," *Lect. Notes Comput. Sci.* **12346**, 478–494 (2020).
35. M. Grundmann, V. Kwatra, and I. Essa, "Auto-directed video stabilization with robust l1 optimal camera paths," in *IEEE Conf. Comput. Vision and Pattern Recognit. (CVPR 2011)* (2011).
36. C.-Y. Wang et al., "CSPNet: a new backbone that can enhance learning capability of CNN," in *IEEE/CVF Conf. Comput. Vision and Pattern Recognit. Workshops (CVPRW)*, pp. 1571–1580 (2020).
37. K. He et al., "Spatial pyramid pooling in deep convolutional networks for visual recognition," *Lecture Notes in Computer Science*, D. Fleet et al., Eds., Springer, Cham (2014).
38. S. Liu et al., "Path aggregation network for instance segmentation," in *IEEE/CVF Conf. Comput. Vision and Pattern Recognit.*, pp. 8759–8768 (2018).

39. H. Rohling, "Some radar topics: waveform design, range CFAR and target recognition," in *Advances in Sensing with Security Applications*, J. Byrnes and G. Ostheimer, Eds., pp. 293–322, Springer Netherlands, Dordrecht (2006).
40. M. Danelljan et al., "Discriminative scale space tracking," *IEEE Trans. Pattern Anal. Mach. Intell.* **39**(8), 1561–1575 (2017).
41. G. Allebosch et al., "Edge based foreground background estimation with interior/exterior classification," in *Proc. 10th Int. Conf. Comput. Vision Theory and Appl.*, SCITEPRESS, Vol. 3, pp. 369–375 (2015).

Gianni Allebosch is researcher at imec and Ghent University at the Department of Telecommunications and Information Processing, in the "Image Processing and Interpretation" research group. He received his master of science degree in electronics and ICT engineering from Ghent University College, Belgium, in 2012. His current research interests include motion segmentation, object tracking, camera auto-calibration, and sensor fusion.

David Van Hamme received his degree of master of science in electronics and ICT engineering from Ghent University, Belgium, in 2007. In 2016, he obtained a PhD in computer science in the field of intelligent vehicles at the same university, and currently holds a position there as post-doc researcher at the Department of Telecommunications and Information Processing. His main research topics are related to perception systems for intelligent vehicles and traffic infrastructure.

Peter Veelaert received his degree in electronic engineering, after which he worked as an engineer at the company DISC (now ESKO graphics) where he developed computer graphics software. In 1986, he joined the Laboratory for Electronics at the University of Ghent from which he received his PhD in 1992. He currently teaches and does research at Ghent University. His current research interests include computer vision, geometric uncertainty modelling, and sensor fusion.

Wilfried Philips is senior full professor at Ghent University's Department of Telecommunications and Information Processing and heads the research group, "Image Processing and Interpretation." He leads sensor fusion research within imec. His main research interests are in the domains of image and video quality improvement and estimation, real-time computer vision, and sensor data processing. He is the cofounder of Senso2Me, which provides internet of things solutions for elderly care.