Hindawi Publishing Corporation
EURASIP Journal on Embedded Systems
Volume 2007, Article ID 82174, 12 pages
doi:10.1155/2007/82174

Research Article

Embedded Vehicle Speed Estimation System Using an Asynchronous Temporal Contrast Vision Sensor

D. Bauer, A. N. Belbachir, N. Donath, G. Gritsch, B. Kohn, M. Litzenberger, C. Posch, P. Schön, and S. Schraml

Austrian Research Centers GmbH - ARC, 1220 Vienna, Austria

Received 28 April 2006; Revised 12 September 2006; Accepted 30 October 2006

Recommended by Udo Kebschull

This article presents an embedded multilane traffic data acquisition system based on an asynchronous temporal contrast vision sensor, and algorithms for vehicle speed estimation developed to make efficient use of the asynchronous high-precision timing information delivered by this sensor. The vision sensor features high temporal resolution with a latency of less than 100 μs, wide dynamic range of 120 dB of illumination, and zero-redundancy, asynchronous data output. For data collection, processing and interfacing, a low-cost digital signal processor is used. The speed of the detected vehicles is calculated from the vision sensor's asynchronous temporal contrast event data. We present three different algorithms for velocity estimation and evaluate their accuracy by means of calibrated reference measurements. The error of the speed estimation of all algorithms is near zero mean and has a standard deviation better than 3% for both traffic flow directions. The results and the accuracy limitations as well as the combined use of the algorithms in the system are discussed.

1. INTRODUCTION

The need for traffic data acquisition is growing as increasing traffic density calls for extensive use of traffic management systems that rely on high-quality real-time traffic flow information. Such systems, for example, support the operation of variable speed limit or lane closing signs which adapt to the actual traffic situation, possibly preventing critical traffic situations, such as congestions, before they occur. Lane occupation and average velocity are important input parameters for traffic management systems but also detailed knowledge about the variation of the velocity distribution of the vehicles helps to predict immanent traffic congestion [1–3]. Basis for such prediction methods is precise speed estimation for single vehicles and not only average traffic velocity.

As traffic speed monitoring and vehicle counting using buried induction loops involves relatively high installation and maintenance costs, highway authorities switch to noninvasive technologies such as microwave radar, infrared, ultrasound, and video detection. The advantages of nonoptical techniques are their relative independence from weather and light conditions, however, most of these devices need to be installed centered above the single lane they service. Advanced video detection is capable of monitoring several lanes simultaneously from a side mount position and delivers an image of the traffic situation, but performance suffers from poor lighting or bad weather conditions. Furthermore, real-time video detection is computationally expensive and relies on high-performance signal processing hardware, large amounts of memory, and high-bandwidth data links. As costs per installation for advanced video detection systems remain high, the density of installation along a highway route is limited and may not reach the density necessary to acquire enough data for reliable traffic flow prediction.

Traffic speed monitoring and vehicle counting systems using video detection and image processing techniques have been reported, for example, by Coifman et al. [4], Cathey and Dailey [5], and Grammatikopoulos et al. [6]. None of these however have been implemented as embedded systems but run their algorithms on workstations. In [4] a network of DSPs in combination with a Pentium processor was used to estimate vehicle speeds and compare to 5 min. average traffic speed as reference. In [5, 7] a theoretical estimate of the speed measurement accuracy using uncalibrated cameras is given. In [6] video tracking and speed estimation of vehicles is performed. However, [5, 6] do not provide comparison to

reference speed measurements, while [7] compares estimation results to average traffic speed. In [8] real-time detection of vehicles in a tunnel using a dual-DSP smart camera is presented, but with no speed estimation. Reference [9] gives a general overview on video processing techniques for traffic applications but most of the presented examples are computationally too expensive for embedded processing.

The embedded vision system presented in this paper overcomes some of the limitations of traditional video processing mentioned above. A specialized optical sensor [10, 11] delivers zero-redundancy, asynchronous data containing precise timing information of moving objects at a comparatively small data rate. The analog preprocessing of the visual motion information on the sensor focal plane allows for using a low-cost low-power DSP for data post processing, limiting system size, cost, and price. The proposed system features vehicle detection and counting and individual vehicle speed estimation and has been implemented as a compact low-power embedded system. Three different algorithms for speed estimation which exploit the unique characteristics of the asynchronous data are described and compared. The accuracy of the speed estimations is evaluated by means of ground truth data from a calibrated reference system.

The article is organized as follows: Section 2 describes the vision sensor and the embedded system, Section 3 explains the algorithm for vehicle detection as a prerequisite for the vehicle speed estimation, the algorithms for which are described in Section 4. Section 5 contains an estimation of the accuracy and error analysis. In Section 6, the speed estimation results are compared to ground truth data and the accuracy is evaluated for the different algorithms. As all algorithms run concurrently on the processor, a measure for the level of confidence is established, and a procedure for establishing the final system output is presented.

2. DESCRIPTION OF THE EMBEDDED SYSTEM

In this section, the temporal contrast vision sensor and the embedded traffic monitoring system that has been specifically designed to process the asynchronous data stream produced by the imager are described.

In contrast to traditional CCD or CMOS imagers that encode image irradiance and produce constant data volume at a fixed frame-rate irrespective of scene activity, the sensor contains an array of autonomous self-signaling pixels which individually respond to relative changes in light intensity by placing their address on an asynchronous arbitrated bus with a latency of less than 100 μs. Pixels that are not stimulated by a change in illumination, or temporal contrast, are not triggered hence static scenes produce no output. Because there is no pixel readout clock, no time quantization takes place at this point. The sensor operates largely independent of scene illumination, directly encodes object reflectance, and greatly reduces redundancy while preserving precise timing information. Because output bandwidth is automatically dedicated to dynamic parts of the scene, a robust detection of fast moving vehicles is achieved. The high dynamic range of the photosensitive element (> 120 dB or 6 decades) makes the sensor ideal for applications under uncontrolled light conditions.

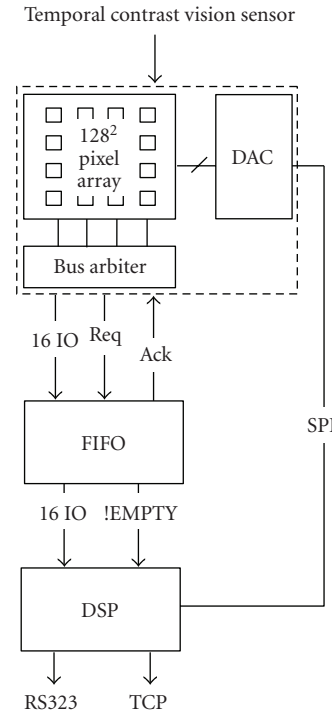
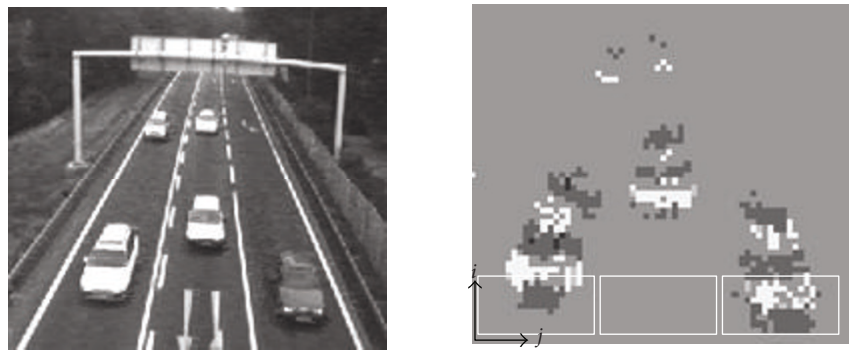


Figure 1: Schematics of the embedded vision system architecture.

The pixel location in the imager array is encoded in the event data that are reflected as coordinates in the resulting image space by address-event-representation (AER) [12]. The scene information is transmitted event-by-event to an Analog Devices BF533 "blackfin" DSP via an asynchronous data bus. The imager is capable of transmitting 100 kilo-events per second (kevents/s) and more; however, for a typical traffic surveillance scenario the peak data rate from the 128×128 pixel imager is not higher than 50 kevents/s on average.

Figure 1 depicts the general architecture of the embedded sensory system, which comprises the vision sensor, a first-in first-out (FIFO) buffer memory and the DSP. Following a data available request (Req) from the sensor, the location (i.e., address) of the event generating pixel within the array is transmitted to a FIFO on a 16-bit parallel bus, implementing a simple 4-phase handshake protocol. The FIFO with a depth of 512 events responds with an acknowledge signal (Ack) making the bus available for the next address event (AE) data transmission [12]. The FIFO is placed between the sensor and processor to cope with peaks of AE activity and is capable of handling up to 40 MHz memory access frequency. A process running on the DSP buffers the data for further processing as long as the FIFOs "not empty" (!EMPTY) signal is active. Every AE received by the DSP is labeled by attaching the processor clock ticks with 4 millisecond precision as a time stamp. These data are the basis for the vehicle speed estimation.



(a) (b)

FIGURE 2: Still video image from a conventional camera (a) and image representation of a typical address-event data stream (b) from the image sensor pointed at a highway scene. The frame integration time in (b) is 100 ms. Three ROIs covering the three lanes are marked in (b) by white rectangles.

The DSP also controls 24-bit resolution bias generator DACs on the sensor chip that generate all internal bias currents, thus allowing on-the-fly adjustments of functional parameters like contrast detection thresholds [13]. Data for the DAC setting are clocked and latched into shift registers via a 3-line serial interface to the DSP. The embedded system provides an Ethernet connection to interface to host computer or any other IP-client.

The traffic data acquisition system consists of a sensor board and a DSP board, both of $7 \times 7\,\text{cm}^2$ dimensions. The system is mounted above the road at a mounting height of 7 to 12 m, either overhead or at the roadside. It is capable of monitoring traffic on up to four lanes simultaneously, depending on the mounting position. Due to its low power consumption, the system is suitable for autonomous solar or battery supply operation. Detailed technical specifications of the embedded traffic data system can be found in [14].

3. VEHICLE DETECTION

The images in Figure 2 show a comparison of a still video picture and 64×64 pixel frame of AER data of a highway traffic scene. In order to visualize the AER data, events have been collected for a 100-millisecond interval and rendered like a video frame. The different gray shadings encode pixel activity per unit time. Note that the white and black vehicles both have very similar representation in the AER data stream illustrating the sensitivity of the imager even to small contrast changes.

In order to take full advantage of the efficient coding of the visual information in the AER data, a reasonable algorithm directly processes the spatiotemporal information contained in the data stream. Figure 3 depicts two example illustrations of 4 second of AER data streams produced by vehicles moving along a road on two different lanes moving towards the imager ($v = -100.2$ km/h) and away ($v = +41.5$ km/h) from the imager, respectively. The pixel event activity is plotted for the imager i and j axis versus time.

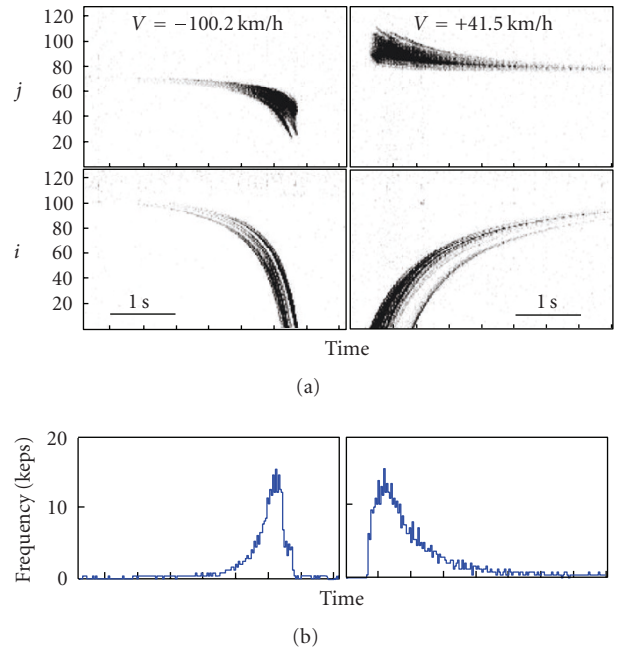


FIGURE 3: (a) Spatiotemporal representation of the AE data for two vehicles. (b) Instantaneous AER data rates for the two examples.

The event rate is encoded in grey levels from 0 (white) to 15 kevents/s (dark). Each vehicle produces compact point clouds of AEs in (t, i) and (t, j) space representing the vehicles track when moving towards the imager. The temporal development of the instantaneous address event activity in kevents/s is plotted below the vehicle traces.

Due to the image sensors specific sensitivity to temporal contrast, each vehicle produces a distinct activity peak of almost equal magnitude in sceneries of highly variable or wide dynamic range illumination. The system thus reliably detects vehicles without the need to adjust parameters such as optical aperture or sensor gain.

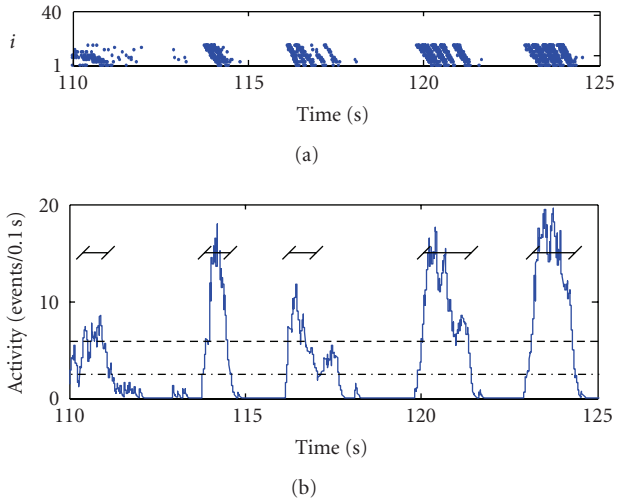
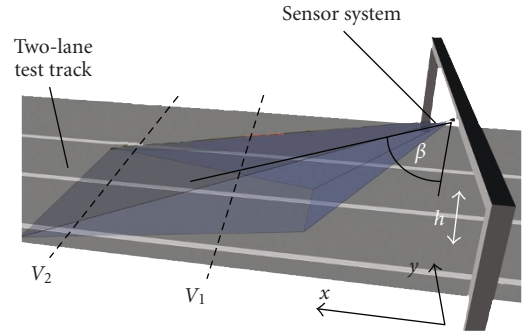


Figure 4: (a) Example of the vehicle detection shown for the AER data from an ROI produced by five vehicles. (b) Development of the smoothed event activity within this ROI. The black horizontal lines indicate five detected vehicles in time interval 110 to 125 s. Dashed and dashed-dotted lines indicate the lower and upper detection thresholds.



Mounting height	h	7.3 m
Mounting tilt angle	β	71.9°
Lens aperture angle	α	42.6°

Figure 5: Setup of the reference measurement test site.

The vehicle detection algorithm is based on the observation of activity peaks in predefined regions-of-interest (ROIs) corresponding to highway lanes [15]. The AER activity is accumulated in 10-millisecond frames in every ROI separately and the resulting activity is stored in one element of the ring buffer. For every ROI, a ring buffer with a length of 10 elements is installed. The sum activity, that is, the sum of all activity values in the ring buffer (in total 100 milliseconds), is compared with the "high" threshold. If the sum activity is larger than the "high" threshold, then the beginning of a vehicle is detected and the corresponding AE stream is buffered as long as the sum activity is above the "low" threshold. If the sum activity falls below the "low" threshold, then the end of the vehicle is detected and the buffering of the AE stream is stopped. The threshold hysteresis is used to avoid spurious vehicle detected triggered by imager noise on one hand and to allow a robust detection of the vehicle end on the other hand. Figure 4 shows an example for the detection of 5 vehicles on the center lane of a highway together with the temporal development of the ROI ring buffer value. The high and low thresholds are indicated by dashed and dash-dot lines. Horizontal black lines show the detection times for the vehicles. In a further processing step, the stored AE buffer corresponding to one vehicle is used to calculate the velocity, which is explained in Section 6.

4. TEST DATA ACQUISITION AND EVALUATION

Figure 5 depicts the test site used for evaluating the vehicle speed estimation where the system is mounted on a bar above a two lane test track. The dashed lines indicate the positions of two calibrated light-barrier speed measurement units with a resolution of 0.1 km/h and a precision of better than 0.1 km/h. Reference speeds V_1 and V_2 are measured at distances 7 m and 16 m from the sensor base point. These speed data are used as ground-truth for the verification of the speed-estimation algorithms. Cases where V_1 and V_2 differ by more than 1 km/h are rejected to assure near constant vehicle speed. A total number of 273 test cases have been evaluated. For the selected test cases, the average of V_1 and V_2 was used as the reference speed V_{ref}.

The difference of the estimated speed to the mean reference speed is the error v_{err}. Assuming that all systematic errors have been removed by calibration, the quality of the speed estimation for a set of measurements is given by the width of the standard deviation $\sigma(v_{\text{err}})$. The measurements have been conducted under bright daylight conditions.

5. SPEED ESTIMATION ALGORITHMS

In order to estimate the velocity of the detected vehicles, the pixel indices i have to be transformed into the real world coordinate x. Note that the x-axis is parallel and y-axis is perpendicular to the moving direction of the vehicles.

From the geometric setup involving the sensor mounting height h, the aperture angle α and the tilt angle β, the real-world coordinate x (Figure 5) can be calculated from the pixel indices i according to the following equation:

$$x = h \cdot \tan\left(\beta + \arctan\left(\tan\frac{\alpha}{2} \cdot \left(\frac{2 \cdot i}{M-1} - 1\right)\right)\right). \quad (1)$$

M is the total number of pixels of a sensor column. Only a downward tilt angle was assumed and the transformation holds only for objects on the road surface. The above equation is evaluated for $i = 1, \ldots, 128$ and the corresponding x values are stored in a look-up table. Thus, a fast transformation from the pixel index i to the coordinate x is performed by applying this table.

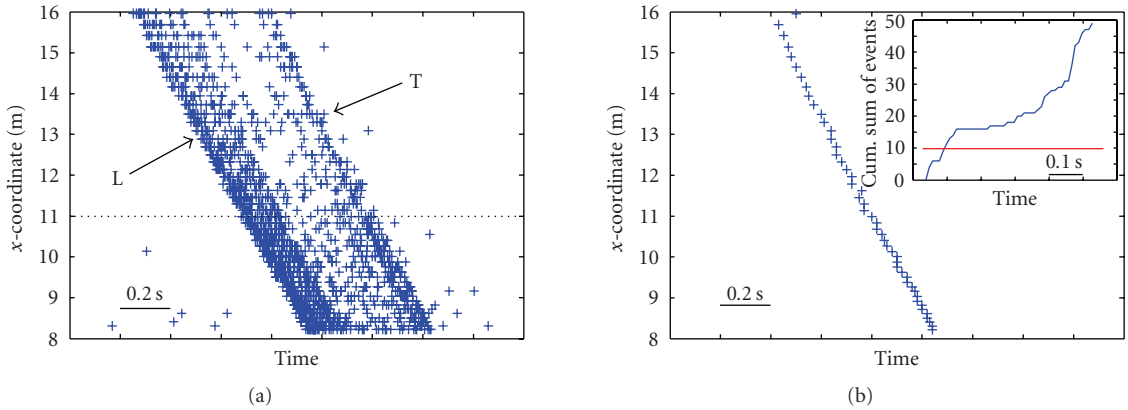


Figure 6: (a) The (x, t)-point cloud representing a vehicle, the leading and trailing edges of the vehicle are indicated by the letters L and T. (b) The (x, t)-point cloud representing the leading edge extracted from (a). The inset shows the cumulative event sum over time at the $x = 11$ m position.

5.1. Edge-based approaches

5.1.1. Edge extraction

The vehicle detection algorithm provides an (x, t)-point cloud of the complete vehicle. Figure 6 shows an example for the resulting point cloud for a vehicle extracted from real-live data. The negative slope of the point cloud corresponding to a vehicle moving towards the sensor system in the $-x$ direction (corresponding to $v < 0$) is clearly visible in the figure. The higher density for points with smaller distance (i.e., small x) is explained by the denser geometric projection of the imager rows onto the ground near the sensor system.

For the edge-based velocity estimation algorithms, a certain edge has to be extracted out of the vehicle's point cloud. In general, it is possible to detect different edges in the vehicle. For velocity estimation purposes, the most important edges are the leading and the trailing edges typically caused by the vehicle's shadow on the ground.

Based on the (x, t)-point cloud representing the vehicle (Figure 6(a)), a time histogram with a temporal resolution of 10 milliseconds for a certain x value is generated. Note that this temporal resolution of the AER processing corresponds to a 100 frames per second video processing if compared with traditional techniques. The calculation of the cumulative activity is based on the time histogram. For extracting the leading edge, the summation is performed from the smallest time instance to the largest time instance of the histogram (inset, Figure 6(b)). For the trailing edge the direction of summation is done vice versa.

The time at which the activity exceeds a certain threshold is stored for the considered x value. For the shown example, one point of the leading edge at $x = 11$ m is extracted. 20% of the maximum of the cumulative activity has shown to be an appropriate threshold for this kind of edge detection.

This is done for a number of N x-values of a predefined region of interest (ROI) corresponding to a certain lane, thus obtaining a new point cloud in (x, t). For the example considered here, $N = 48$.

Due to the edge extraction routine, the amount of data is further reduced to N points (x, t). This point cloud represents the track of the leading edge (Figure 6(b)) of the vehicle's point cloud shown in Figure 6(a).

5.1.2. Velocity-estimation algorithms

(a) Histogram method

This section describes an efficient method for vehicle speed estimation from AER data streams. The method is based on the estimation of the slope of the AER point cloud representing a certain edge of a vehicle.

In a first step, vehicles are detected in the unstructured AER data stream by a vehicle detection algorithm detailed in Section 3. The next processing step is to isolate the trace of either the leading or the trailing edge of a vehicle, which is already described in the section above. The edge extraction provides an (x, t)-point cloud consisting of N points.

For the point cloud of the leading edge (for the case shown in Figure 6) consisting of N points, the velocities from every combination among these points are computed. More precisely, we calculate $N \cdot (N - 1)/2$ velocities according to

$$v_{kl} = \frac{x_k - x_l}{t_k - t_l} \tag{2}$$

with $l, k = 1, \ldots, N$. Note that, as $v_{kl} = v_{lk}$ holds, this value is considered only once. These velocities are entered in a histogram which ranges from 20 to 300 km/h.

The width of the histogram bins is 2 km/h. Empirical tests have shown that a too fine resolution, for example, 1 km/h, yields a cliffy pulse in the histogram. Finding the correct maximum in a cliffy pulse is not straightforward and therefore broader histogram bins are used. The drawback of the resulting rough velocity resolution is eliminated by calculating the center of gravity (CoG) of the pulse. Details regarding the CoG are explained further ahead. In Figure 7, the velocity histogram for the leading edge shown in Figure 6 is depicted. The empty histogram bins for velocities > 120 km/h

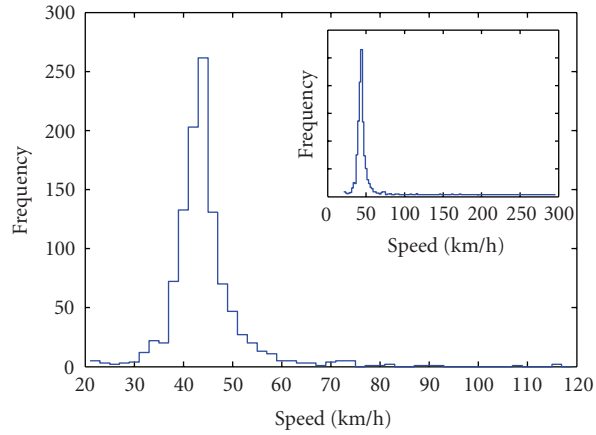


Figure 7: The velocity histogram corresponding to the leading edge depicted in Figure 6. The inset shows the full range of the velocity histogram (0–300 km/h).

are not shown for a better visibility of the histogram peak. Furthermore, it can be seen that due to the rough velocity resolution the resulting pulse is very smooth.

The remaining part of the algorithm is to find the position of the maximum in the histogram and the calculation of the CoG. For this purpose, the weighted average of the maximum bin and its two neighboring bins are calculated:

$$v = \frac{f_{\max-1} \cdot v_{\max-1} + f_{\max} \cdot v_{\max} + f_{\max+1} \cdot v_{\max+1}}{f_{\max-1} + f_{\max} + f_{\max+1}}, \quad (3)$$

where f_k denotes the frequency of a certain velocity and thus is the height of a histogram bin and v_k denotes the corresponding velocity. The subscript k denotes the number of the histogram bin. Due to the calculation of the CoG, the effective resolution is better than the histogram bin width. Intensive investigations based on a huge amount of test data have shown that it is optimal to use only 3 histogram bins for the CoG calculation. For the histogram shown in Figure 7, the velocity estimate is 42.8 km/h.

Using relatively wide histogram bins and the CoG method leads to an efficient determination of the pulse maximum. Nevertheless, the velocity resolution is sufficiently high. Furthermore, the present velocity estimation algorithm allows for calculating a confidence measure for assessing the quality of the calculated speed value:

$$c = \frac{f_{\max-1} + f_{\max} + f_{\max+1}}{N \cdot (N-1)/2} \cdot 100\%. \quad (4)$$

The denominator normalizes the confidence measure, because the entire number of calculated velocity values is $N \cdot (N-1)/2$ and therefore, the maximum confidence is 100%. A confidence value of almost 100% is only achieved for very smooth edges. For this reason, the above stated confidence measure is very conservative. For the histogram in Figure 7, the confidence value is 52.8%.

The requirements for a good confidence measure are that it yields small values, if the detected edge is very noisy or if the edge extraction algorithm provides an edge which is actually a mixture of two edges. Large confidence values are expected if the considered edge is well pronounced. All these requirements are satisfied by the above stated confidence measure.

A noisy edge results in a broad pulse in the histogram. The broad pulse results in a small numerator in (4), because only a small part (3 bins) of the complete broad pulse is considered. A small numerator and a constant denominator result in a small confidence value. On the other hand, if a well pronounced edge is present, the histogram consists of a sharp pulse consisting of only a few bins. Therefore, the numerator and thus the confidence measure are large. If the edge is a mixture of two edges (suboptimality of the edge extraction algorithm), then two or more peaks can be found in the histogram and thus the numerator is small, because there are a lot of histogram bin with high frequency, which are not considered in the numerator. The defined confidence measure for the quality of the velocity estimate allows rejecting results from broad distributions or distributions without a pronounced maximum originating from measurement artifacts.

(b) Line-fit method

This method is also based on the extracted edge of a detected vehicle. As the name of the algorithm already reveals, the main principle of the considered method is to fit a line into the extracted edge. This is done by the least squares (LS) approach. The general line equation adapted to the velocity estimation problem can be written as

$$x = k \cdot t + d, \quad (5)$$

where k is the velocity and d is only an x-shift (useless for the actual velocity calculation). The (x, t)-point cloud representing the extracted edge consisting of N points is used to find the line characterized by k and d which minimizes the mean square error to the points of the point cloud. With the following relationship between the parameters k and d of the line and the coordinates x and t of the point cloud:

$$\underbrace{\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{pmatrix}}_{\mathbf{x}} = k \cdot \begin{pmatrix} t_1 \\ t_2 \\ \vdots \\ t_N \end{pmatrix} + d = \underbrace{\begin{pmatrix} t_1 & 1 \\ t_2 & 1 \\ \vdots & \vdots \\ t_N & 1 \end{pmatrix}}_{\mathbf{T}} \cdot \underbrace{\begin{pmatrix} k \\ d \end{pmatrix}}_{\mathbf{p}}, \quad (6)$$

the best estimates (in the LS sense) for the parameters k and d can be calculated using

$$\mathbf{p} = \left(\mathbf{T}^T \cdot \mathbf{T}\right)^{-1} \cdot \mathbf{T}^T \cdot \mathbf{x}. \quad (7)$$

Note that $(\mathbf{T}^T \cdot \mathbf{T})^{-1}$ is a 2×2 matrix and therefore a very efficient implementation of the matrix inversion is possible.

In order to suppress the influence of outlier on the estimated velocity, the line fit is repeated 3 times. Between the line-fit iterations, outliers are removed by a distance criterion. If the distance of a certain point of the point cloud to

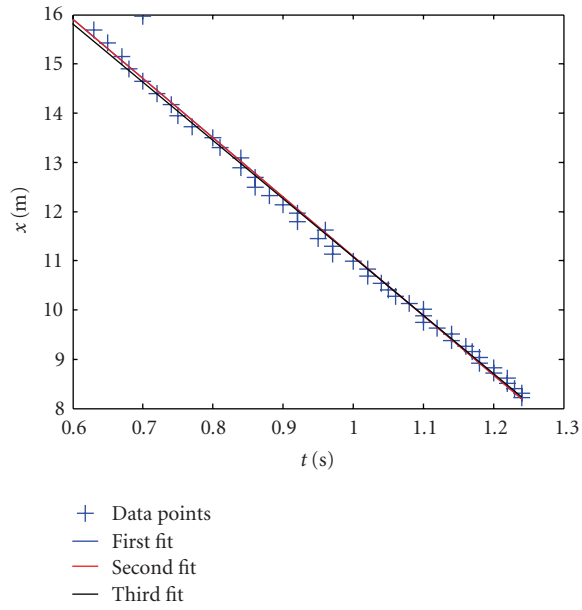


FIGURE 8: The point cloud of the extracted edge and the fitted lines.

the estimated line is larger than a certain threshold

$$|x_i - (k \cdot t_i + d)| > d_{\text{THRESHOLD}}, \tag{8}$$

then this point is assumed to be an outlier and is not used for the following iteration step.

The work flow of the whole algorithm is as follows:

First line fit based on the whole point cloud $\Rightarrow k_1, d_1$

First outlier rejection: $|x_i(k_1 \cdot t_i + d_1)| > d_{\text{THRESHOLD 1}}$

Second line fit based on a reduced point cloud $\Rightarrow k_2, d_2$

Second outlier rejection: $|x_i(k_2 \cdot t_i + d_2)| > d_{\text{THRESHOLD 2}}$

Third line fit based on a further reduced point cloud $\Rightarrow k_3, d_3$

Note that the second threshold is stricter than the first threshold in order to further thin out the point cloud. The resulting parameter k_3 is the final velocity estimate.

Furthermore, a confidence measure for the velocity estimate can be calculated:

$$c = \frac{N - N_{\text{OUTLIER}}}{N} \cdot 100\%, \tag{9}$$

where N_{OUTLIER} denotes the number of outliers. Thus, if no outliers are present,that is, the extracted edge is very smooth, then the confidence value is 100%. If the edge is noisy or the edge is actually a mixture of two or more edges, then the number of outliers increases and thus the confidence value decreases.

For the edge shown in Figure 6, the line fit algorithm provides the lines within the point cloud shown in Figure 8.

For this special case, the first and the second fit are identical because the distance threshold for the first outlier rejection is 2 m and thus the outlier on the top of Figure 8 is not rejected. Thus the basis for the line fitting in both cases is identical and therefore obviously the results are identical. At the stage of the second outlier rejection the above mentioned outlier is rejected because the second threshold is set to 1 m. The resulting slope of the line allows calculating a velocity measure which is in this case 42.6 km/h with a confidence level of 97.9%.

The confidence measure for this velocity estimation algorithm provides rather larger values. Really small confidence values (< 30%) are very unlikely even for very noisy edges.

5.2. AER projection

Other than the algorithms described before, this method does not rely on a leading or trailing edge extraction. With this method the AEs belonging to a vehicle's point cloud in (t, x)-space are time shifted for a speed hypothesis v, yielding a corrected time t',

$$t' = t - \frac{x}{v}. \tag{10}$$

The data are then projected onto t'-axis using the corrected time by computing the distribution $N(v, t')$ with a resolution of 10 milliseconds. The maximum value of the histogram for this projection is stored. Figure 9 shows the (t', x) data for a vehicle with reference speed −105.5 km/h and histograms for two speed hypotheses. The left pane shows the general case where v does not match the reference speed (−70 km/h) and the right pane shows the case of the best estimate (−106 km/h).

The algorithm relies on the fact that a correct hypothesis for v will yield the largest maximum in the corresponding $N(v, t')$ histogram. This is explained by the compensation of the slope of the point cloud resulting in a projection of the largest number of the t' values into a single histogram bin. Figure 10 shows the maximum histogram values versus the speed hypothesis v with a resolution of 1 km/h and indicates the speed estimate of −106 km/h. A second peak at −122 km/h is produced by the trailing edge of the vehicle. As the trailing edge is always at roof height of an approaching vehicle, h is smaller than assumed and the speed estimate is too high. As seen from the shape of the distribution with its central maximum and a pronounced roll-off to both sides, the production of extreme outliers is very improbable with this method.

As the computational cost is high implementation on the embedded processor is only possible by reducing the number of events in a first step by simple random picking of 100 events and testing a number of speed hypothesis with a coarse speed resolution. In a second step, the procedure is repeated with 250 data points and a finer speed resolution around a neighborhood of the v found in the first iteration.

5.3. Implemented speed selection

The implemented firmware routines first evaluate the speed with the line-fit and histogram methods and determine the best result by the confidence measure. If none of the two confidence values exceeds the confidence threshold the

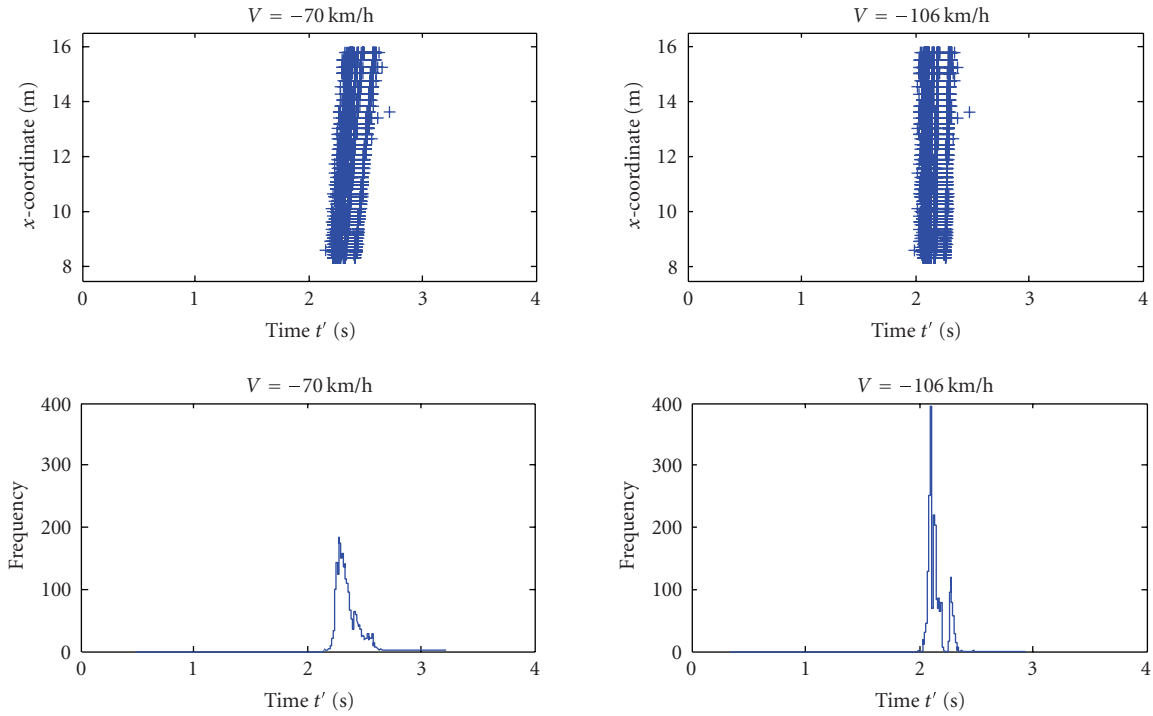


Figure 9: AE data with corrected time t' for two speed hypotheses and their projection histograms.

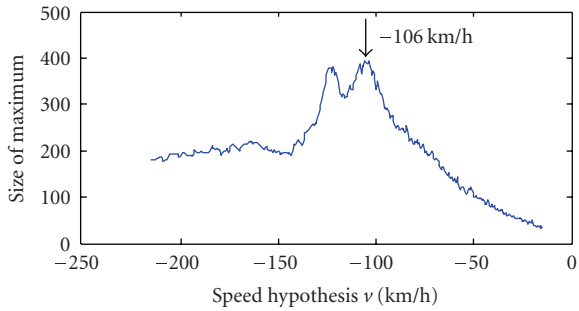


Figure 10: Dependence of the maximum value of the projection histograms on the speed hypothesis. The best estimate is indicated by an arrow.

projection method is performed to achieve a robust speed result at the risk of a lower estimation quality.

The three algorithms have been implemented in C/C++ with minor code optimization, such as using fixed point notation for some variables. On the embedded system processor, all three algorithms run simultaneously for each detected car and use roughly 30% of the CPU time.

6. ACCURACY ESTIMATION

The following section gives an estimation of the accuracy of the vehicle speed estimation method. The system accuracy is limited by the finite temporal resolution of the AER time stamping and the precision of the world coordinate transformation that depends on a correct calibration of the optical system.

Speed is calculated by the quotient of a distance d and the time delay t that it takes a vehicle to pass this distance,

$$v = \frac{d}{t}. \tag{11}$$

In the presented system, d is the length of the ROI used for the speed estimation of typically 8 m and t is the time that the vehicle takes to cross the ROI. Although the presented algorithms frequently use much shorter inter-pixel distances and time delays, the speed estimation basically depends on the precision of d and t. Their uncertainty is governed by the uncertainty of the length of the ROI and the temporal resolution, respectively.

As d is a difference of pixel positions x_i and the pixel event response to a moving edge will occur uniformly in the pixels center, the pixel resolution of the imager itself is not influencing the accuracy of d. The uncertainty of d is mainly influenced by the calibration error caused by an imprecise measurement of the mounting parameters during the installation procedure. We further assume an ideally flat road surface and the object size being substantially larger than the geometrical projection of the pixel on the road surface.

Errors in mounting parameters result in an erroneous transformation from imager pixel- to real-world coordinates and consequently in an uncertainty in d. From (1), we can derive d provided that the ROI is situated between imager lines 1 and 48. With parameters α, β, and h from Figure 5

and (1), we get

$$d = x_{i=48} - x_{i=1} = x(h,\beta)_{i=48} - x(h,\beta)_{i=1} = 7.73\,\text{m}. \tag{12}$$

It is assumed that the error of the aperture angle α is negligible, because it is a constant value that can be calibrated very precisely under lab conditions and is not influenced by installation procedures. Alternatively, the aperture angle can be derived precisely from parameters found in the lens datasheet and the imager dimensions.

For a simple manual calibration with $\Delta h = 0.1\,\text{m}$ and $\Delta\beta = 0.5^\circ$ the new distance results in

$$\begin{aligned} d_{\Delta h,\Delta\beta} &= x_{\Delta h,\Delta\beta;i=48} - x_{\Delta h,\Delta\beta;i=1} \\ &= x(h+\Delta h,\beta+\Delta\beta)_{i=48} \\ &\quad - x(h+\Delta h,\beta+\Delta\beta)_{i=1} = 8.13\,\text{m}. \end{aligned} \tag{13}$$

The resulting distance error is $\Delta d = 0.4\,\text{m}$. These uncertainties results in a *systematic error* in the speed measurement leading to a nonzero mean error. The mean velocity error due to a suboptimal calibration is

$$\Delta v(d,t) = v \cdot \frac{\Delta d}{d} = v \cdot 5.2\%. \tag{14}$$

With a larger calibration effort, for example, by using specialized equipment during installation, the values for h and β could be quantified with much higher accuracy. For such a calibration the error could be neglected ($\Delta d = 0$) and this will result in a purely statistical error for the speed measurement.

The temporal precision is found in the error of the time delay measurement, given by the sum of the variance of the pixel latencies (indexed L) and the error introduced by the activity histogram time stamping for edge extraction with 10-millisecond resolution (indexed TS). For a set of events both values are *statistic in nature* and not a fixed value, thus the width of their distribution is used to derive an error,

$$\sigma t = \sqrt{(\sigma t_L)^2 + (\sigma t_{\text{TS}})^2}. \tag{15}$$

The typical pixel response time is $< 100\,\mu$s for daylight conditions with a variance of $< 50\,\mu$s and is therefore negligible compared to the temporal resolution of the time stamp. As events occur randomly in time, the time stamping error is distributed equally between 0 and 10 milliseconds, the variance of this distribution will be assumed as the time measurement error,

$$\sigma t \sim \sigma t_{\text{TS}} = \frac{10\,\text{ms}}{2\sqrt{3}} \sim 2.9\,\text{ms}. \tag{16}$$

The error in the time measurement leads to a nonzero standard deviation in the speed error distribution and is independent of the calibration. Using σt, we calculate the standard deviation of the error. Assuming a vehicle passing the ROI with speed 100 km/h, we get

$$\sigma v(d,t) = \left(\frac{\partial v}{\partial t}\right) \cdot \sigma t = 0.29\,\text{km/h}. \tag{17}$$

The precision of the speed measurement is therefore mainly influenced by the quality of the optical calibration, regarding the system mounting height and camera tilt angle.

Other possible sources for errors are imager noise and lens imperfections, such as lens distortion. Imager noise influences the quality of the edge extraction and adds to the statistical error. Lens distortion has a systematic effect on real-world coordinate transformation and adds to the systematic error. These error contributions have been neglected.

7. RESULTS AND DISCUSSION

Figures 11 and 12 show the results of 273 single vehicle speed estimations for both driving directions on both lanes, using the histogram method of Section 5.1.2(a) and the line-fit method of Section 5.1.2(b), for measured speeds between -120 km/h and 80 km/h. The dashed line in the figures is the $y = x$ line. The standard deviation of the error for discrete speed intervals is shown as a bar chart. The results for the leading and trailing edges are shown separately. Note that the leading edge of the approaching vehicles ($v < 0$), formed by the ground shadow, shows a smaller error than the leading edge of departing vehicles ($v > 0$) which is formed by the car body and is smeared out. A similar effect can be seen vice versa for the trailing edges. An additional systematic error is caused by the trailing edge of approaching cars, which is always the roof edge and appears faster than the ground shadow. For both methods the standard deviation of the error is below 3 % for most of the speed intervals[1] when the approaching vehicles leading edges and departing vehicles trailing edges are considered. The errors for the rest of the test cases are in the 5 to 7 km/h regime for the above mentioned reasons.

Figure 13 shows the results of the projection method of Section 5.2. This method does not extract the leading or trailing edge but rather picks the most dominant edge of the vehicle, that is, the one with the strongest contrast. The figure shows that the standard deviation of the error is much smaller for the departing vehicles. As the back view of cars is rather featureless, the dominant edge is the ground shadow for the departing vehicles, thus delivering good speed results with a standard deviation of the error below 3 km/h. The front view of the approaching cars seen from an elevated position is a mixture of high contrast edges from bonnet, windshield, and roof. Therefore, the velocity is calculated from a set of "smeared" edges, producing a worse result with standard deviation of roughly 6%.

All results presented have been produced under daylight conditions. Note that the speed estimation also works in the night, by extracting the AE point cloud produced by the high contrast head- and taillights. Due to the nature of the temporal contrast imager, the sensor produces no blooming or overexposure effects, thus allowing the robust speed estimation also during night operation. The speed result,

[1] The calculation of the standard deviation is based on the data of 5–20 vehicles for each speed interval.

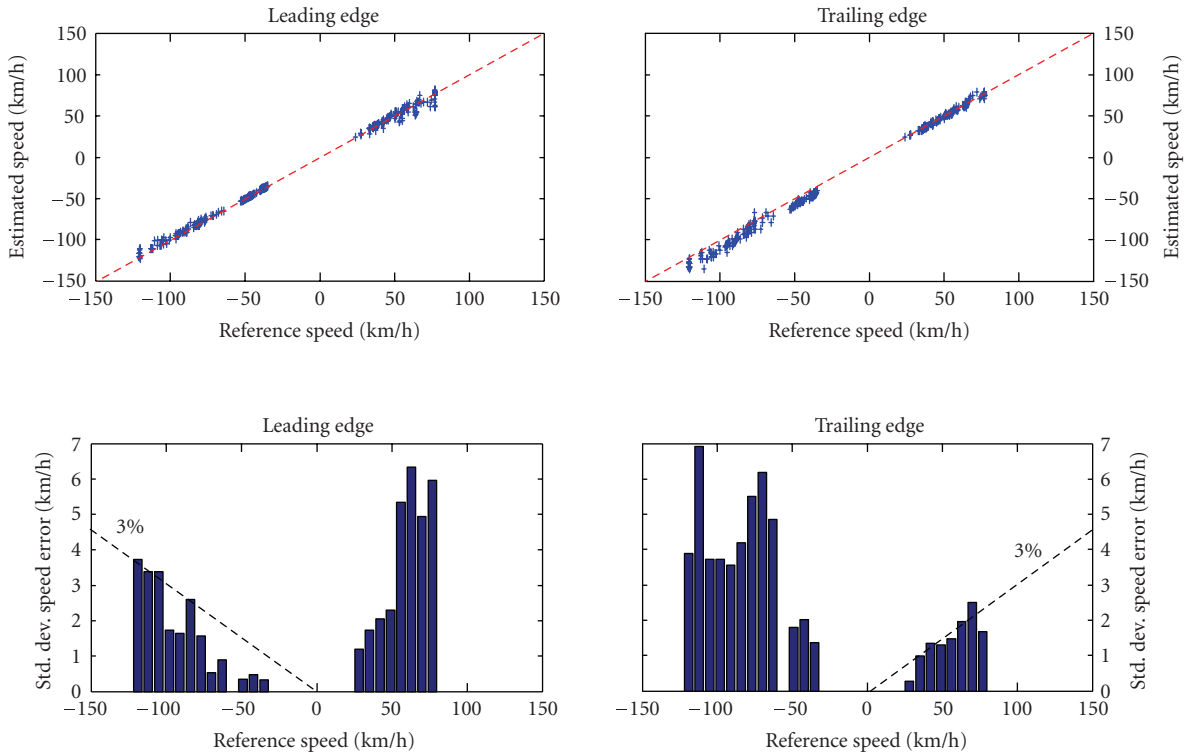


Figure 11: Results of the histogram method for the extracted leading and trailing edges.

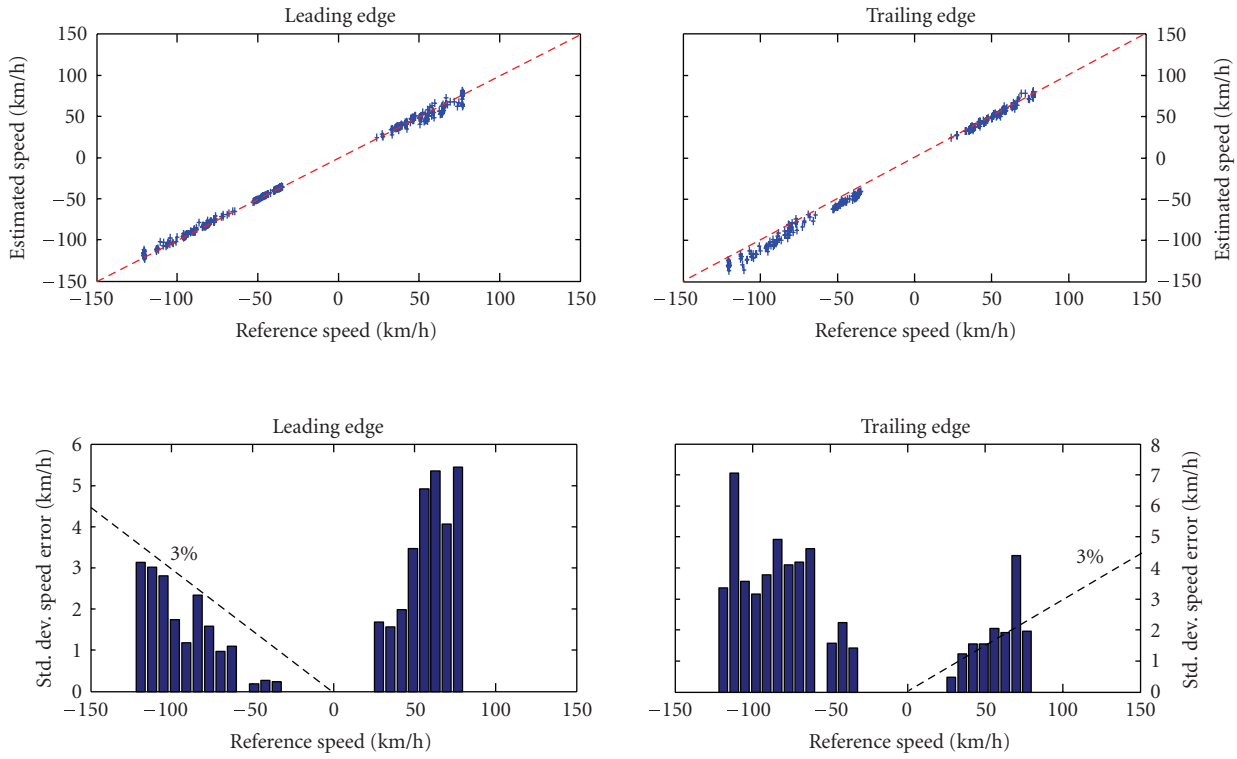


Figure 12: Results for the LS line fitting method for the extracted leading and trailing edges.

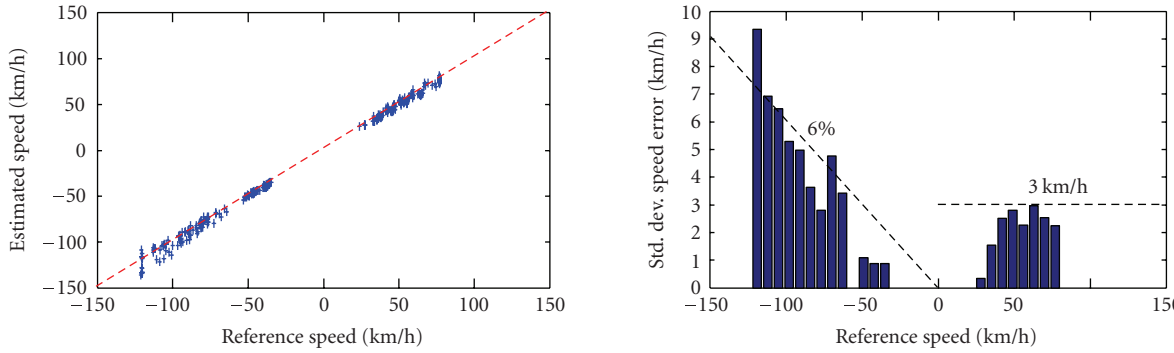


Figure 13: Results of the AE projection method. Note that this method does not rely on edge extraction, thus only one result per test drive is given.

Table 1: Comparison of the three speed estimation algorithms.

Method	Driving direction	Edge	Mean (v_{err}) km/h	σ (v_{err}) km/h	Success rate (%)	Applied confidence threshold[2]
Histogram	v > 0	leading	0.65	4.4	100	10%
	—	trailing	0.61	1.8	100	
	v < 0	leading	−0.83	2.3	99.3	
	—	trailing	8.50	4.3	98.6	
Line-fit	v > 0	leading	1.90	4.3	100	60%
		trailing	0.63	2.1	98.5	
	v < 0	leading	−0.68	2	99.3	
	—	trailing	8.50	4.2	95	
Projection	v > 0	—	−0.08	2	100	N/A
	v < 0	—	0.72	5	100	

[2] For definitions of the confidence measures for both methods, compare Sections 5.1.2(a) and 5.1.2(b).

however, has to be corrected to the mean height of vehicle headlights above ground.

The advantages of the histogram method are that it is simple, robust, and easy to implement on the DSP. The method is able to produce good velocity estimates even in case of noisy or spurious AE data by robustly finding the slope of the most dominant velocity in a point cloud from the largest peak in the velocity histogram.

The line fit method is able to produce similar quality estimates for good noise-free AE data but is rather sensitive even to the presence of single noise events degrading the line fit result.

The projection method is very robust against noisy AEs appearing in the data stream and against badly pronounced leading or trailing edges, which may cause problems with the previously described algorithms. Due to the integrating nature of the method, it will always pick the most pronounced edge. This might not be the ground shadow but, for example, the vehicle's roof for which the geometric projection into world coordinates does not hold. Consequently, the quality of the velocity estimate is not as good as for the two other methods. Furthermore, no confidence measure can be derived by this algorithm, however, due to the robustness of the method, extreme outliers do not occur.

Table 1 shows an overview of all results with the mean errors and the success rates. The rate of successfully estimated car speeds is determined by the confidence level produced by each method. The table also gives the confidence threshold that was used to accept a result. As obvious from the presented results, only the leading edges of the approaching and the trailing edges of departing vehicles are evaluated by the embedded system firmware when using the histogram and line-fit methods. Using the three methods combined allows to derive a good speed estimate with a better than 3% standard deviation of the error for almost any of the presented test cases.

The results are comparable to results postulated in [6], where an error of ±3 km/h is estimated from a single reference measurement using GPS data. In [7] the distribution of single vehicle speed estimates were compared to speeds averaged over 20 second intervals measured with induction loops.

Our work is the first to report the comparison of a substantial number (273) of single vehicle speed estimates directly to single vehicle speed reference measurements.

Possible limitations to the reliability of the system stem from the approach to detect vehicles within regions-of-interests. If a passing vehicle occupies more area of the lane 1 ROI than of the lane 2 ROI, then the car is detected on lane 1, and the corresponding velocity is calculated correctly and vice versa. However, lane changes are critical if the vehicle is passing the ROI-zone more or less exactly between two neighboring lanes. In this case, the system's behavior cannot be predicted and is possibly erroneous.

The systems firmware comprises a shadow suppression algorithm, which is capable of distinguishing between two vehicles moving in parallel or a vehicle and its shadow. However, as the problem is not trivial, the algorithm might fail in cases where shadows are very long or are cast diagonally on the street.

8. CONCLUSION

A compact embedded vision system for traffic data acquisition on up to 4 lanes was presented comprising an Analog Devices "blackfin" DSP and a contrast sensitive asynchronous vision sensor. The sensor features a 120 dB wide dynamic range, focal-plane contrast change detection, and an efficient AER data coding. Three different algorithms for vehicle speed estimation based on processing the AER data stream with 10-millisecond temporal resolution and their implementation in the embedded system firmware have been described. All three algorithms run concurrently on the embedded processor consuming 30% of the CPU time without special code optimization. The selection of the best speed estimate is made by computing a confidence measure for two of the speed estimations. For the first time, the speed estimation error has been evaluated by comparing with single vehicle reference speed measurements performed on a two-lane test track. Measuring the leading and trailing edges of the approaching and departing vehicles, respectively, a nearly unbiased measurement with a mean error below 1 km/h has been achieved. The statistic error was less than 3% for the majority of speed intervals. A patent application covering the system concept and speed estimation algorithms has been filed [16].

ACKNOWLEDGMENTS

The authors would like to thank Tobi Delbruck and Patrick Lichtsteiner (Institute of Neuroinformatics, ETH-Uni, Zürich) for their work and support on the temporal contrast vision sensor, Vittorio Dante and Paolo Del Giudice (Instituto Superiore di Sanita, Rome, Italy) for the original design of the PCI-AER hardware, and Adrian Whatley, Gerd Dietrich, and all other members of the Institute of Neuroinformatics ETH-Uni, Zürich, involved in the development of the PCI-AER board, its drivers, and software library components.

REFERENCES

[1] M. Bandoi, K. Hasebe, K. Nakanishi, A. Nakayama, A. Shibata, and Y. Sugiyama, "Phenomenological study of dynamical model of traffic flow," *Journal de Physique I*, vol. 5, no. 11, pp. 1389–1399, 1995.

[2] Y. Pei and Y. Lang, "Research on dynamic traffic assignment applied in traffic congestion analyze and management," in *Proceedings of IEEE Intelligent Transportation Systems*, vol. 2, pp. 1032–1035, Shanghai, China, October 2003.

[3] J. K. Lam, J. Kerr, P. Korpal, and C. Rayman, "Development of compass - an advanced traffic management system," in *Proceedings of the IEEE-IEE Vehicle Navigation and Information Systems Conference (VNIS '93)*, pp. 200–203, Ottawa, Canada, October 1993.

[4] B. Coifman, D. Beymer, P. McLauchlan, and J. Malik, "A real-time computer vision system for vehicle tracking and traffic surveillance," *Transportation Research Part C*, vol. 6, no. 4, pp. 271–288, 1998.

[5] F. W. Cathey and D. J. Dailey, "A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras," in *Proceedings of IEEE Intelligent Vehicles Symposium*, pp. 777–782, Las Vegas, Nev, USA, June 2005.

[6] L. Grammatikopoulos, G. Karras, and E. Petsa, "Automatic estimation of vehicle speed from uncalibrated video sequences," in *Proceedings of International Symposium on Modern Technologies, Education and Profeesional Practice in Geodesy and Related Fields*, pp. 332–338, Sofia, Bulgaria, November 2005.

[7] T. N. Schoepflin and D. J. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 2, pp. 90–98, 2003.

[8] M. Bramberger, J. Brunner, B. Rinner, and H. Schwabach, "Real-time video analysis on an embedded smart camera for traffic surveillance," in *Proceedings of the 10th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS '04)*, vol. 10, pp. 174–181, Toronto, Canada, May 2004.

[9] V. Kastrinaki, M. Zervakis, and K. Kalaitzakis, "A survey of video processing techniques for traffic applications," *Image and Vision Computing*, vol. 21, no. 4, pp. 359–381, 2003.

[10] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128x128 120dB 30mW asynchronous vision sensor that responds to relative intensity change," in *Proceedings of IEEE International Solid-State Circuits Conference (ISSCC '06)*, San Francisco, Calif, USA, February 2006.

[11] P. Lichtsteiner and T. Delbruck, "64x64 event-driven logarithmic temporal derivative silicon retina," in *Proceedings of IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, pp. 157–160, Nagano, Japan, June 2005.

[12] K. A. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, vol. 47, no. 5, pp. 416–434, 2000.

[13] T. Delbruck and P. Lichtsteiner, "Fully programmable bias current generator with 24 bit resolution per bias," in *Proceedings of IEEE International Symposium on Circuits and Systems (ISCAS '06)*, p. 4, Kos, Greece, May 2004.

[14] Smart-eye traffic data sensor, http://www.smart-systems.at/products/products_video_tds_en.html.

[15] D. Bauer, P. Bühler, N. Donath, et al., "Embedded vehicle counting system with 'silicon retina' optical sensor," in *Workshop on Information Optics (WIO '06)*, Toledo, Spain, June 2006.

[16] Austrian patent application no. A 1011/2005, June 2005.