EMBEDDED VISION SYSTEM FOR REAL-TIME OBJECT TRACKING USING AN ASYNCHRONOUS TRANSIENT VISION SENSOR

M. Litzenberger, C. Posch, D. Bauer, A.N. Belbachir[1)], Member, IEEE
P. Schön, B. Kohn, and H. Garn, Senior Member, IEEE

ARC Seibersdorf research GmbH, Tech Gate Vienna, Donau-City-Str.1, A-1220, Austria
[1)] Corresponding author: ahmed.belbachir@arcs.ac.at

ABSTRACT

This paper presents an embedded vision system for object tracking applications based on a 128×128 pixel CMOS temporal contrast vision sensor. This imager asynchronously responds to relative illumination intensity changes in the visual scene, exhibiting a usable dynamic range of 120dB and a latency of under 100µs. The information is encoded in the form of **A**ddress-**E**vent **R**epresentation (AER) data. An algorithm for object tracking with 1 millisecond timestamp resolution of the AER data stream is presented. As a real-world application example, vehicle tracking for a traffic-monitoring is demonstrated in real time. The potential of the proposed algorithm for people tracking is also shown. Due to the efficient data pre-processing in the imager chip focal plane, the embedded vision system can be implemented using a low-cost, low-power digital signal processor.

Index Terms— embedded vision system, real-time object tracking, address-event representation, neuromorphic vision sensor.

1. INTRODUCTION

In the past decades, object tracking has significantly evolved from an academically pattern recognition problem [1] into emerging applications for different purposes [9]. Existing object tracking applications include, but not limited to, pedestrian and vehicle tracking and surveillance [11], bubbles tracking [2] and soccer players tracking [6].

Vehicle tracking is one of the most spread applications due to the increasing number of cars and the increasing demand in traffic safety. Most of the existing vehicle-tracking systems are based on the video cameras. Many video tracking systems identify vehicles by virtue of their motion. In cases where objects are moving quickly past the sensing camera, the motion segmentation techniques are fast and robust. Unfortunately, in cases where the sensing camera observes a largely stationary traffic light queue, motion estimation based systems mostly fail to follow-up the objects.

Previous research in automotive tracking systems has not been completely successful. In [9], Kalman-Snakes technique is used to provide automobile contours after initial motion segmentation step. These contours are used for tracking purpose. The authors in [3] make use of block matching to find optical flow, which is combined with a priori knowledge of the road geometry to handle stationary vehicles. In [15], the authors apply a background estimation technique to isolate foreground objects "blobs". Afterwards, the principal component analysis is used to classify the blobs and estimate their orientation.

In addition to the limitation of the developed tracking methods in terms of performance, video systems usually produce a huge amount of data that likely saturate any computational unit responsible for data processing. Thus, real-time object tracking based on video data processing requires large computational effort and is consequently done on high-performance computer platforms. As a consequence, the design of video-tracking systems with embedded real-time applications, where the algorithms are implemented in Digital Signal Processor (DSP) is a challenging task.

A further weakness of video detection is the limitation of conventional camera systems to operate under wide dynamic range lighting, which is typical for outdoor applications. Therefore, real-time video-based tracking applications are mostly constrained with limited resources at the price of the optimal performance.

There also exist other systems for vehicle tracking using the support of satellites for the vehicle follow-up [1]. Although those systems might be efficient, they are very costly as they are requiring a sensor, a communication link and a workstation mounted on each vehicle.

The aim of this work is to develop a low-cost tracking system with real-time capability for embedded applications. The tracking system has been developed using the asynchronous transient vision sensor [11] and the algorithm has been implemented on the Blackfin DSP from Analog Device. The system has already been demonstrated its performance for vehicle speed estimation and vehicle counting in previous papers [13][3]. This work presents an extension application of this system to object tracking.

The paper is structured as follows. In section 2, the embedded sensory system including the DSP unit is described. The tracking algorithm is described in section 3. The experimental results after the application of the presented algorithm on the real data are discussed in section 4. Section 5 concludes the paper with a short summary.

2. THE ASYNCHRONOUS TRANSIENT VISION SENSOR

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 asynchronous vision sensor contains an array of autonomous, self-signaling pixels which individually respond in real-time to relative changes in light intensity by placing their address on an asynchronous arbitrated bus. Pixels that are not stimulated by a change in illumination 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 at variable lighting conditions is achieved. The scene information is transmitted event-by-event to a DSP via an asynchronous bus. The pixel location in the imager array are encoded in the event data that are reflected as i,j coordinates in the resulting image space in the form of **A**ddress-**E**vents (AE) [12]. An effective way of processing AE data takes advantage of the efficient coding of the visual information by directly processing the spatial and temporal information contained in the data stream.

The high dynamic range of the photosensitive element (>120dB or 6 decades) makes the imager ideal for applications with uncontrolled light conditions.

Fig. 1 depicts the general architecture of the concerned embedded sensory system, which comprises an imager, a **F**irst-**I**n, **F**irst-**O**ut (FIFO) buffer memory and the Blackfin DSP BF537 from Analog device. It has a maximum frequency of 600 MHz, 128 KB internal memory and 32 MB external SDRAM memory. This limited memory resource might be far-too low for the data processing of any high-resolution video system as it does not fit the traditional video processing needs. The imager and the DSP consume in total roughly 2.5 W of electrical power. The location (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 is placed between the imager sensors and the DSP to cope with peaks of AE activity and is capable of handling up to 40 MHz memory access frequency. In the processing stage, every AE received by the DSP is labeled by attaching the processor clock ticks with 1ms precision as a timestamp. These data are the basis for the vehicle tracking.

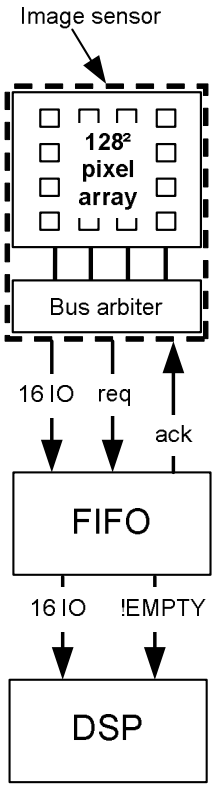

Fig. 1. Schematics of the embedded system architecture.

Other AE processing algorithms for the vehicle speed estimation have also been implemented on this DSP [13]. The full processing comprises the AE acquisition and time stamping, clustering and tracking including rough speed estimation. The following sections of this paper focus on the tracking algorithm.

The images in Fig. 2 show a comparison of a still video picture and 64×64 pixel transient imager [11] AE data of a highway traffic scene. In order to visualize the AE data, events have been collected for a 20 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 AE data stream illustrating the sensitivity of the imager even to small contrast changes.

Detailed technical specifications of the embedded traffic data system can be found in [16].

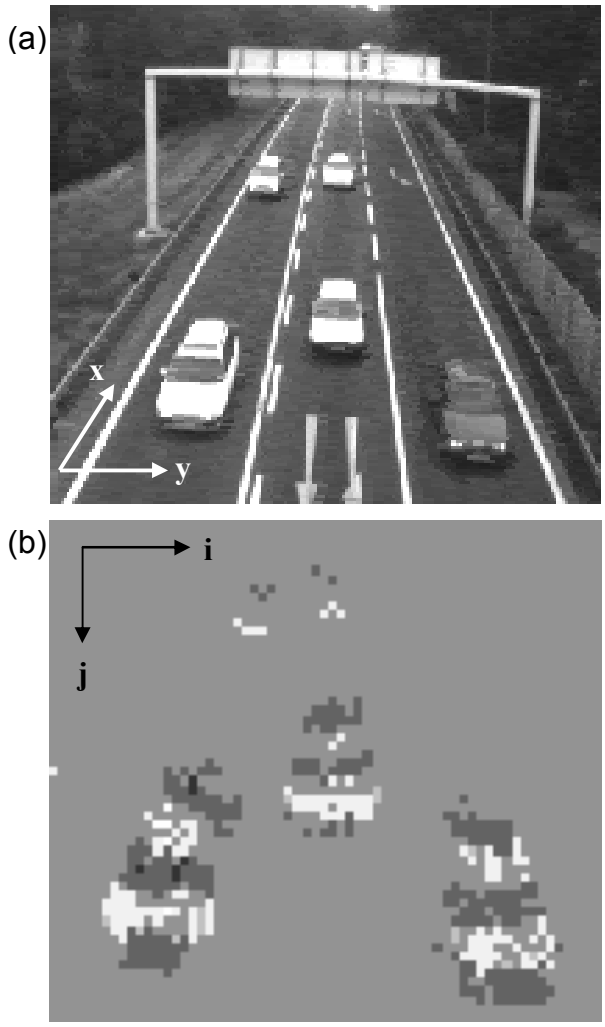

Fig. 2. Still image from (a) a conventional video camera (b) and representation of the AE data stream from a 64×64 pixel transient imager. The axis of the imager coordinate system i,j and world coordinate system x,y are displayed in the images.

3. THE PROPOSED TRACKING METHOD

The proposed algorithm is inspired by the mean-shift approach [6][7] and implements a continuous clustering of AE's and tracking of clusters. This algorithm processes each AE as it is received without data buffering (no memory consumption). This is of special importance when using low-cost and low memory resource systems.
Each new event can be assigned to a cluster based on a distance criterion and is used to update this clusters weight and (x,y) - center position for tracking.

The algorithm can be briefly described as follows:

1. Find one cluster in the cluster list that the new AE with address x_E=(i,j) lies within a seek-distance R_K of its center (see Fig. 3):

$$R = | x - x_E | < R_K, \text{ for all clusters.} \quad (1)$$

2. If a cluster is found where $R < R_K$, update all the clusters features accordingly (see below).
3. If no cluster is found, create a new one with center x_E and initialize weights and size, creation time and velocity. A new label (unique identification number) is assigned to the cluster.

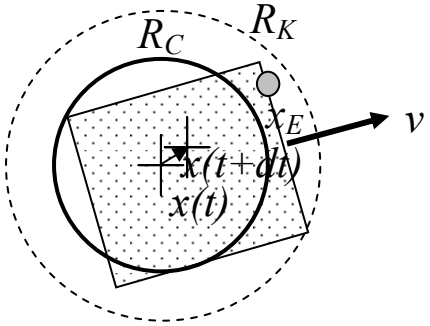

Fig. 3. Continuous clustering of address-events.

The cluster update process is sketched in Fig. 3. Let x_E be the coordinate of an AE produced by the edge of a moving (velocity v) object. Let $x(t)$ be the original cluster center, then the new center-coordinate $x(t+dt)$ is calculated as:

$$x(t+dt) = x(t)\cdot\alpha + x_E\cdot(1-\alpha), \quad (2)$$

where $0 \leq \alpha < 1$ is a parameter of the algorithm and dt is the timestamp difference between the current and the last AE's that were assigned to the cluster. This shifts the clusters center by a certain amount controlled by α, which is usually chosen near 1 to obtain a smooth tracking.
Furthermore the cluster size R_C is updated:

$$R_C(t+dt) = \max(R_{min},\ R_C(t)\cdot\alpha + R\cdot(1-\alpha)) \quad (3)$$

R_{min} is a parameter of the algorithm and the max-condition assures that the clusters size is kept within certain limits. A seek-distance larger than cluster size allows the clusters to grow in size and thus adapt dynamically to the size of the tracked object. Otherwise, clusters would be just allowed to shrink. We define the seek-distance R_K for each cluster as a multiple of the cluster size R_C:

$$R_K = \min(R_{max},\ R_C \cdot R_{multiple}); \quad (4)$$

where $R_{multiple}$ (usually $1 < R_{multiple} < 3$) and R_{max} are parameters of the algorithm and the min-condition assures that the clusters size is kept within certain limits. The $R_{multiple}$ parameter is most important when two clusters approach each other during tracking. A reasonable value will prevent one cluster from instantaneously "overtaking" the other cluster because its seek radius is limited. Also the maximum change of a clusters size R_C is limited by the parameter α.
For the weight of the cluster W the mean frequency of events in this cluster is used, so that

$$W(t+dt)=W(t)\cdot\alpha + 1/dt\cdot(1-\alpha). \quad (5)$$

Very inactive clusters have low AE-frequency and consequently low weights.

For a practical application of the algorithm it has turned out that a separate definition of α for each single parameter position α_X, size α_R and weigh α_W is advantageous.

The list of existing clusters is scanned periodically (10 to 20 times/s) for too old and inactive clusters which are then deleted from the list. The velocity vector of the cluster is also updated at this occasion. To ensure smooth changes of the velocity vector it is also recalculated using α

$$v(t+dt)=v(t)\cdot\alpha + v\cdot(1-\alpha). \quad (6)$$

If a new event for an object edge is created outside the seek radius of the cluster the object belongs to, the object will split up into two overlapping clusters. This situation is sketched in Fig. 4. Successive events might then be attributed to the new cluster C2 instead of the long-standing C1, and the object might be permanently split into two (or more) clusters. The new cluster might also "overtake" the long-standing one, by catching all new events from to the object for itself. This leads to a discontinuity in the objects track because the object will consequently change its label.

To prevent this situation the cluster list is kept sorted by creation time. Consequently, when looping through the list to attribute AE's to clusters, older clusters will be preferred. Therefore the new, overlapping cluster will "starve" and be deleted very quickly because fewer or no events will be attributed to it. By this measure a very smooth and continuous tracking of address-event clusters can be achieved with the presented algorithm.

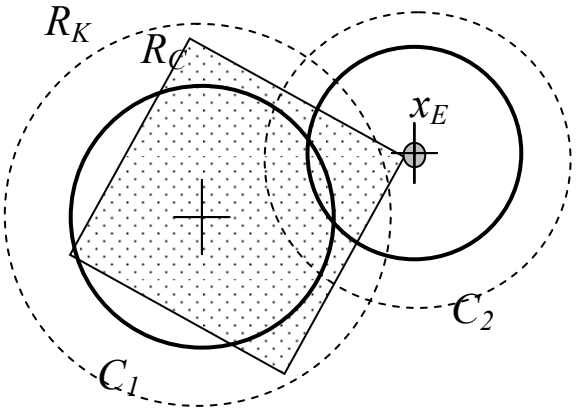

Fig. 4. The problem of overlapping clusters in tracking.

The algorithm consumes only a small amount of memory as only the cluster list has to be kept in memory. Trials showed that a cluster list of ~20 clusters is sufficient to solve most of the test scenarios. As only a few variables are used to describe the features of a cluster, 2 kB of memory is sufficient for the cluster list. The computational complexity is moderate. However, event-address to cluster-center distances have to be computed several times for each new event. Using quadratic clusters instead of circular ones can further decrease the computational effort. The features of each cluster have to be updated once for every event.

4. EXPERIMENTAL RESULTS

The presented tracking algorithm runs in real time on vehicle AE data with a timestamp resolution of 1 millisecond. The system is used for lane change monitoring in a traffic surveillance application. The prototype vision system was mounted above a test track and AER data originating from driving-by vehicles were processed. Fig. 5 shows three still images from a 3 second tracking sequence of two vehicles travelling at a speed of approximately 30 km/h. To visualize the AER data, the pixel activity occurring during a fixed time interval is mapped onto an image for an image-like representation of the AER data. The tracking result is presented as squares in the images encircling the tracked objects. A trace is shown to visualize the objects track within the past second. For both lanes, lane changes can be observed with the tracking algorithm over a distance of typically 50 meters.

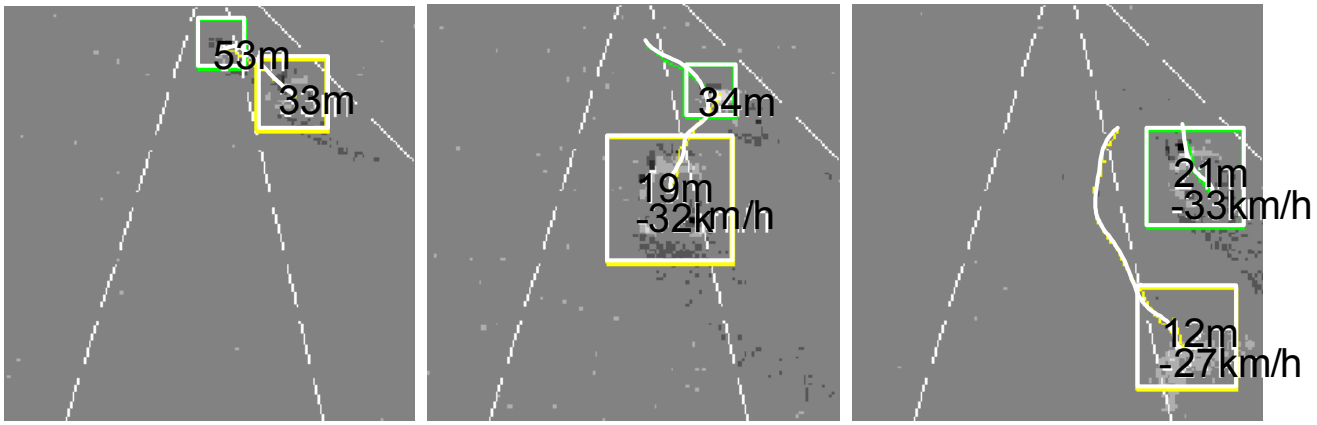

Fig. 5. Three still images from a 3 second tracking sequence showing AER-data with two vehicles driving towards the vision system. Given inside the cluster markers (rectangles) are the distance from the sensor and –if available- the velocity extracted from the track. The dashed lines indicate lane boundaries not visible to the sensor.

Fig. 6 depicts examples of six vehicle tracks observed on a two-lane road. Imager coordinates have been transformed to world coordinates using a simple geometric projection based on the imager mounting height and optical parameters. The x-coordinate shows the road length in meter (containing the vehicle direction) while the y-coordinate gives of the road width in meter. The distance between two adjacent circles on the vehicle track is 0.2 seconds.

Therefore, the traces with closer circles distance reflect the low-speed vehicle compared to the longer distances between circles.

Fig. 7 shows a demonstration of the algorithm on simulated AE data for a person tracking application. The AE data have been simulated from a video sequence with a resolution of 140×180 pixels that is close to our developed 128×128 imager. On the left part, two images from a 2 second video sequence have been extracted for illustration. On the right side, the simulated AER data of the scene including the tracking results is provided.

The two right subfigures show the different persons locations indicated by circles, a unique ID number identifying the object and an arrow indicating the direction and speed of movement. As an example, ID 198 has been correctly tracked while ID 227 was a shadow effect that disappeared in the next sequence

The number listed above each subfigure gives a rough estimation of the vehicle speed extracted from the track (negative values for approaching vehicles and positive values for departing vehicles).

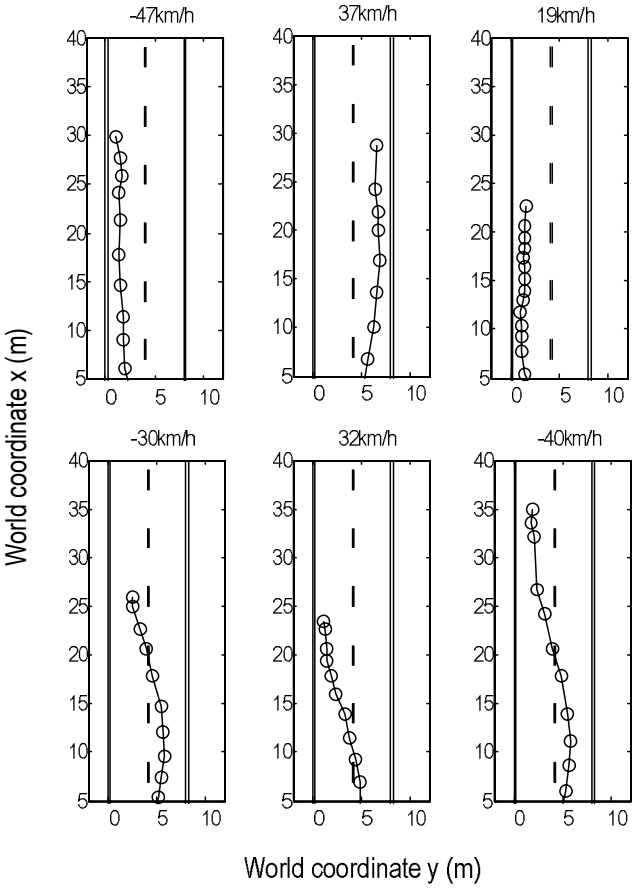

Fig. 6. Example of 6 vehicle tracks. The lane boundaries are shown by straight and dashed lines.

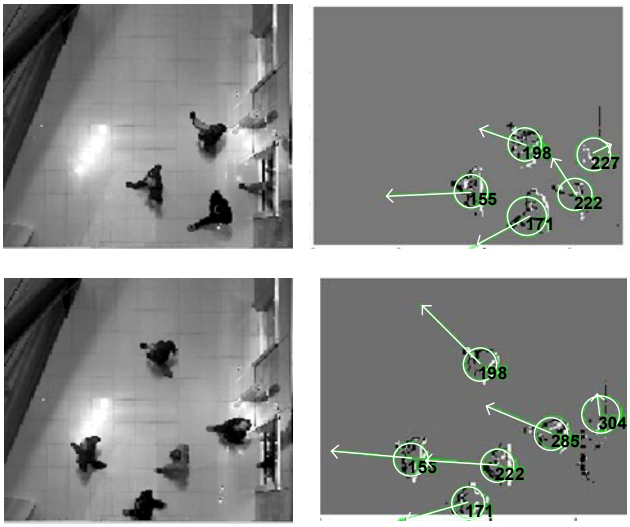

Fig. 7. Example results for person tracking.

5. CONCLUSIONS

This paper presented an embedded vision system and tracking algorithm using data from an asynchronous transient vision sensor for real-time applications. The tracking algorithm benefits from the capability of this imager to detect relative intensity changes and on its efficient asynchronous communication, which significantly reduce the computational burden as compared to traditional video-based traffic surveillance systems, enabling a low-cost, embedded DSP implementation.

This concept has been demonstrated for real-time tracking of vehicles on a two-lane test track. Further application for people tracking has been simulated and the preliminary results are promising.

6. ACKNOWLEDGMENT

The authors thank Tobi Delbruck and Patrick Lichtsteiner from the Institute of Neuroinformatics at ETH Zürich for their work and support on the vision sensor.

7. REFERENCES

[1] L. Alexander and M. Donath, "Differential GPS based control of a heavy vehicle," *International Conference on Intelligent Transportation Systems*, 1999.

[2] Y. Anzai, "Pattern Recognition & Machine Learning," *Morgan Kaufmann*, 1992.

[3] F. Bartolini, V. Capellini, and C. Giani, "Motion Estimation and Tracking for Urban Traffic Monitoring," *International Conference on Image Processing*, volume 3, 1996

[4] D. Bauer, P. Bühler, N. Donath, H. Garn, B. Kohn, M. Korber, M. Litzenberger, J. Meser, C. Posch and P. Schön "Embedded Vehicle Counting System with 'Silicon Retina' Optical Sensor" Workshop on information Optics 2006, Toledo, Spain, June 5-7, 2006.

[5] D.C. Cheng and H. Burkhardt, "Bubble Tracking in Image Sequences," *International Journal of Thermal Sciences*, Volume 42, Number 7, pp. 647-655(9), July 2003.

[6] D. Comaniciu and V. Ramesh, "Mean Shift and Optimal Prediction for Efficient Object Tracking," *International Conference on Image processing*, vol.3, p.70-73, 2000.

[7] D. Comaniciu and P. Meer, "Mean-shift: A Robust Approach Towards Feature Space Analysis," *IEEE PAMI*, vol.24, no.5, 2002

[8] P. Figueroa, N, Leite, R.M.L. Barros, I. Cohen and G. Medioni, "Tracking soccer players using the graph representation," *Proceedings of the 17th International Conference on Pattern Recognition*, UK, 2004.

[9] D. Koller, J. Weber and J. Malik, "Towards Realtime Visual based Tracking in Cluttered Traffic Scenes," *In: Proc. of the Intelligent Vehicles Symposium,* Paris, France, October 1994.

[10] D. Li, K.D. Wong, Y.H. Hu and A.M. Sayeed, "Detection, Classification and Tracking of Targets in Distributed Sensor Networks," *IEEE Signal Processing Magazine*, 19(2):17--30, March 2002.

[11] P. Lichtsteiner and T. Delbruck, "64×64 Event-Driven Logarithmic Temporal Derivative Silicon Retina," *in IEEE Workshop on Charge-Coupled Devices and Advanced Image Sensors*, Nagano, Japan, 2005.

[12] P. Lichtsteiner, C. Posch and T. Delbruck, "A 128x128 120dB 30mW Asynchronous Vision Sensor that Responds to Relative Intensity Change," *in IEEE International Solid-State Circuits Conference (ISSCC2006*), San Francisco, USA, Februay 2006.

[13] M. Litzenberger, A.N. Belbachir, N. Donath, G. Gritsch, H. Garn, B. Kohn, C. Posch and S. Schraml "Estimation of Vehicle Speed Based on Asynchronous Data from a Silicon Retina Optical Sensor," *IEEE Conference on Intelligent Transportation Systems*, Canada, September 2006.

[14] N.T. Siebel and S.J. Maybank , " Real-Time Tracking of Pedestrians and Vehicles," *Proceedings 2nd IEEE Int. Workshop on PETS*, Kauai, Hawaii, USA, December 9 2001

[15] H. Veeraraghaven and O. Masoud. N. Papanikolopoulos, "Managing Suburban Intersections Through Sensing," Technical Report at *Intelligent Transportation Systems* Institute University of Minnesota December 2002

[16] www.smart-systems.at/products/products_video_tds_en.html