Real-Time Vision Using a Smart Sensor System

Ahmed Nabil Belbachir, *Member, IEEE,* Martin Litzenberger, Christoph Posch and Peter Schön

Austrian Research Centers GmbH - ARC, Smart Systems Division, Donau-City Str. 1, 1220 Vienna, Austria

{ahmed.belbachir, martin.litzenberger, christoph.posch, peter.schoen}@arcs.ac.at

http://www.smart-systems.at

***Abstract*— This paper presents a real-time data processing concept for asynchronous "Address Event" image sensors in high-speed machine vision applications. In the implemented system, a dual-line temporal contrast vision sensor asynchronously responds to relative illumination intensity changes in the visual scene, and encodes the information in the form of "Timed Address Event" representation (TAE) data. The TAE stream consists of the event generation time concatenated to the Address Event (AE), which encodes the coordinates of the pixel in the 2×256 matrix. The event data processing includes object detection, denoising and scaling, and feature extraction. The feature extraction step comprises circle fit and orientation estimation techniques that enable robust recognition of a wide range of objects. The data processing algorithms take advantage of the efficient information encoding in the TAE data protocol, yielding a flexible, compact and low-cost real-time machine vision system for high-speed shape detection, object orientation extraction and monitoring.**

I. INTRODUCTION

Traditional CMOS Active Pixel Sensors (APS) or CCDs encode image irradiance and produce constant data volume at a fixed frame rate irrespective of the scene activity. The data processing unit, attached to the imager, is continuously handling the same amount of data, independent of the image content, which represents an unnecessary load on the system resources.

The vision sensor used in this system delivers data only upon presence of activity in the scene as they only react on illumination variation. These variations are usually caused by changes in object reflectance and thus signify object movements. Pixels that are not stimulated by an illumination change remain idle hence static scenes do not produce output.

This image data redundancy suppression is beneficial for high-speed machine vision applications, like for the object inspection in the quality control area or for autonomous processes in the automated fabrication area, as the reduced data volume allows an extremely efficient real-time processing. The algorithms presented in this paper are optimized to take full advantage of this encoding of visual information.

The paper is structured as follows. In section II, a general description of the proposed system is given. Example data from three objects are shown in Section III. In Section IV, the data processing concept and its modules are presented in detail. Section V gives an evaluation of the data processing performance and discusses potential machine vision applications. Section VI concludes the paper with a brief summary and some perspectives.

II. THE SMART SENSOR SYSTEM

The proposed machine vision system (see Fig.1) is inspired by the human visual system, being a combination of retinal sensors and massively parallel computing.

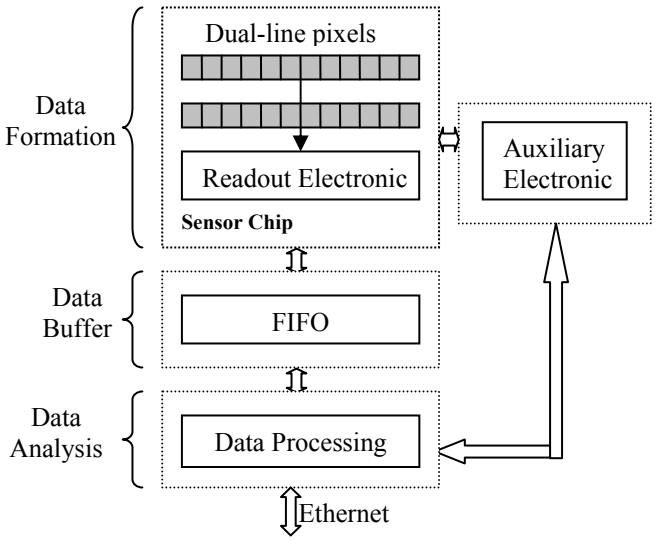

Fig. 1 A system diagram of the smart vision system

The goal is to build a simplified model of an eye-brain machine that can automatically recognize, localize, count, classify, and track objects in real time. The eye is analogical to a compact optoelectronic subsystem that integrates different sensors with geometric, radiometric and spectral parameters to adapt to the actual scene. The brain is equivalent to a high performance control and data handling subsystem with computing resources for different vision applications.

The proposed vision system comprises the following modules:

1. A "silicon retina" sensor chip [6] that contains a dual-line arrangement of autonomous pixels which are sensitive to local temporal contrast [5][6].
2. Auxiliary electronics for on-the-fly configuration of the sensor chip. This allows adapting the sensor to different

scene conditions like varying illumination, object reflectance and speed.

3. A FIFO to handle peak data rates.
4. A data processing unit that is responsible for treating and interpreting the data. The processing concept is detailed in Section IV.

TAEs are classified into two types; ON-events that represent a fractional increase in intensity and OFF-events that reflect a fractional decrease.

III. Data Examples from the Presented Vision System

Fig. 2 depicts data examples from a quadrangle, a hexagon and a ring passing across the sensor's field-of-view with a velocity of ca. 5 m/s at a distance of about 15 cm. Only the edges of the moving objects, generating temporal contrast, trigger events. The right side figures show the TAEs delivered by one of the pixel lines of the sensor in response to the visual stimulus. The x-axis is event generation time in units of the timestamp period, the y-axis is the pixel address (0-255).

The time-stamps can be converted to isogonal spatial information on the basis of the known object speed measured by correlating the data from the two pixel lines.

The total data amount required to represent those three objects, using one line of pixels, ranges between 400 – 650 events per object. This sparse representation of the object shape is exploited by the data processing algorithms as described in the following sections.

IV. Real-Time Data Processing

The data processing unit consists of a Digital Signal Processor (DSP) board with a Blackfin® DSP BF537 (Analog Devices®). It runs at a maximum frequency of 600 MHz, has 128 KB internal memory and 32 MB external SDRAM memory.

The data processing concept is illustrated in Fig. 3. It can be subdivided in five parts:

A. Data Interface

This module is responsible for the data acquisition from the FIFO and their storage in the processor memory. This module can receive data at a rate of up to 2M events / s. One TAE contains 32 bit of data (or less if several pixel addresses share one timestamp, i.e. more than one pixel signals during one timestamp period).

B. Pre-processing

The processing part aims to prepare the data for the analysis step. It includes four modules: object detection, outliers rejection, edges thinning and data normalization.

Object Detection: This module extracts the individual objects from the data stream by monitoring the data rate in time slots under the assumption that the objects are sequentially crossing the sensor field of view. An object is detected when the event rate exceeds a first threshold (threshold 1 is set typically to 5 events). When the event rate falls below second threshold (threshold 2 < threshold 1), the detection algorithm assumes the end of the object. The event rate is calculated for time slots of 1 ms. The identified cluster of events is collected and associated to one object.

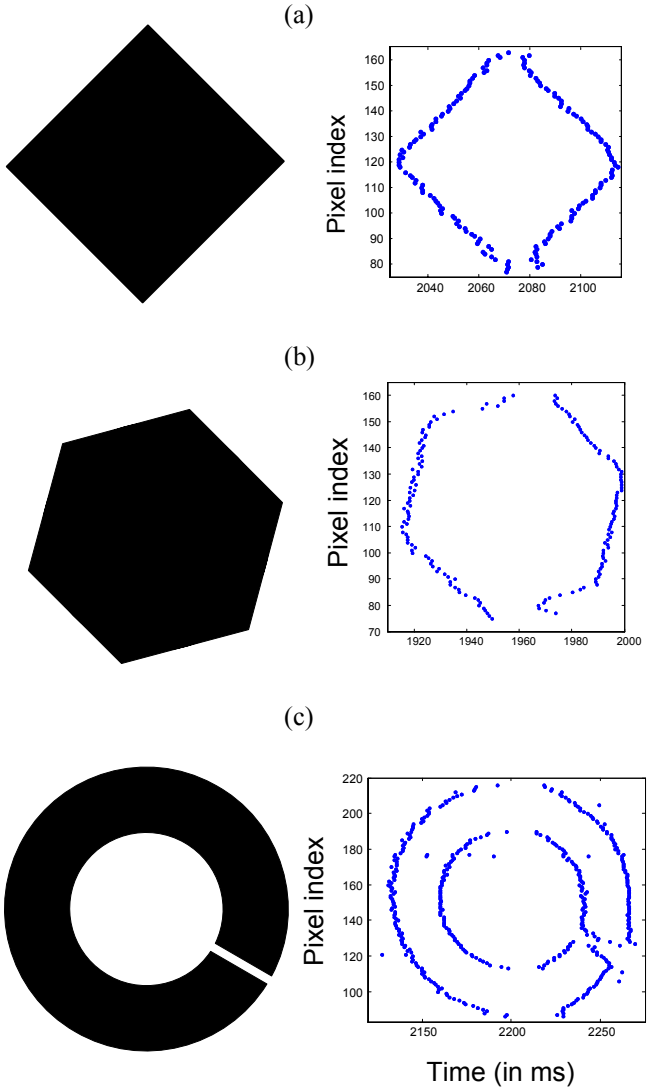

Fig. 2 Original shapes of a quadrangle (a), hexagon (b) and a ring (c) depicted on the left figures. The data representation of those images with the presented vision system are, respectively, given on the right figures

This method is fast and easy to implement. It has however the disadvantage that several objects can be considered as one if they are too close together when moving across the sensor field of view. A minimum time interval of 50 ms between two successive objects guarantees robust detection.

Outlier Rejection: This step makes use of a sigma clipping method for removing outliers. It starts by calculating the mean and the standard deviation (σ) for each cluster of event. The events with large deviations relative to 3σ are

excluded. This method is very effective in removing isolated events that rise from fluctuations in the scene. This method may also remove events from the object structure, however the significant object shape is preserved as 3σ has proven to be an adequate threshold parameter for our data case.

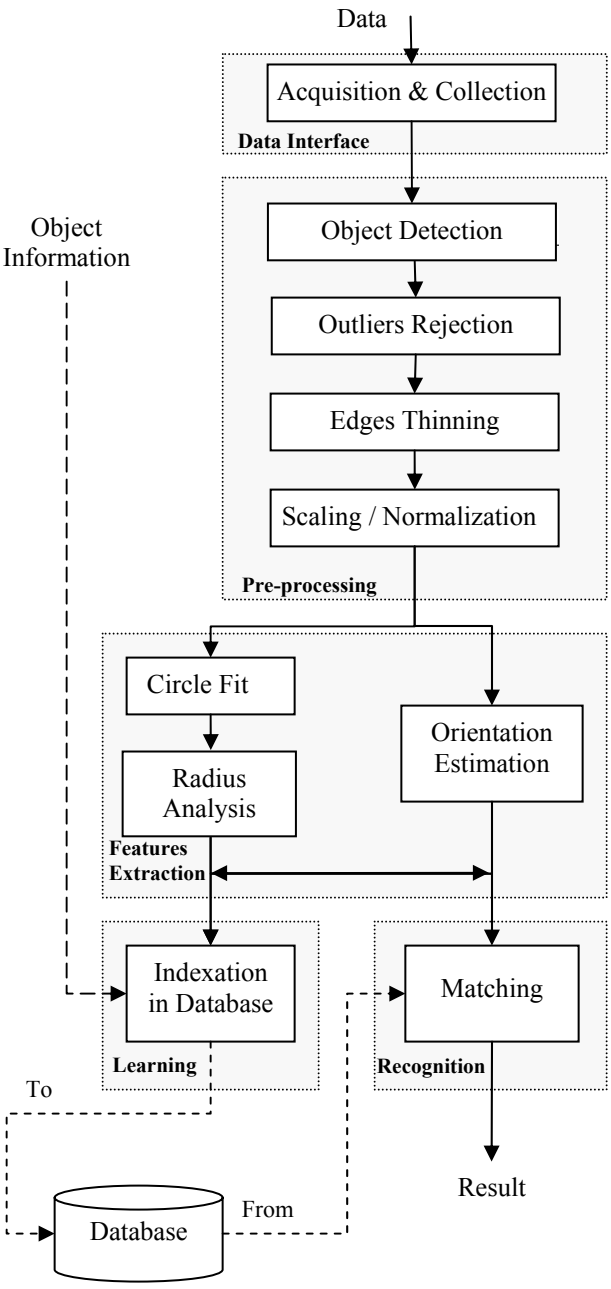

Fig. 3 Overview of the data processing concept

Edge Thinning: This module is intended for sharpening the object edges by removing redundant events. Each pixel can deliver one to several events per time slot and local temporal contrast. This algorithm reduces the data to one event per pixel and slot duration, thus only the first pixel event is kept per slot. The time slot is adjusted according to the prior knowledge of the object velocity.

Scaling / Normalization: The x and y coordinates of the object shape representation do not have the same scale. The time (x-coordinate) has a range of [0,∞] while the pixel index range is [0,255]. Therefore, a scaling of the x-coordinate range is required for shape-based post-processing. The time is divided by the inverse object velocity, measured by the sensor, in order to transform time into isogonal spatial coordinates.

Implemented on the Blackfin-processor, the pre-processing algorithms are able to process up to 500k events / s.

C. Features Extraction

For recognition of a given object, a set of features are extracted to characterize the object. Two techniques are used for extracting specific object characteristics like the dimension and the orientation using the "circle fit" and "orientation estimation", respectively. These feature extraction techniques are detailed in the following paragraphs.

Circle Fit: This module aims to fit the data points to a circle for a minimum relative error between the original data points and the fit data points. Equation (1) shows the circle fit formulation where (x, y) are the data point coordinates, (x_c, y_c) is the circle center coordinate and r is the circle radius.

$$(x - x_c)^2 + (y - y_c)^2 = r^2. \quad (1)$$

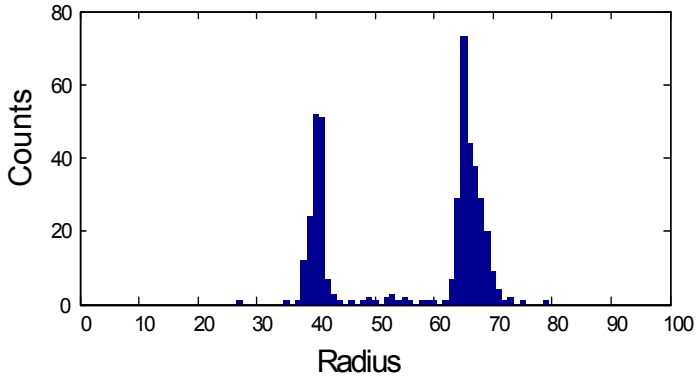

Fig. 4 Radius histogram of the shape depicted in Figure 2(c)

The barycenter of the data points is considered to be the circle center. Then, the distance to this center is calculated for every data point and a histogram of this distance is built. Fig. 4 illustrates this radius (distance) histogram for the ring shape depicted in Fig. 2 (c). The x-coordinate represents the calculated distances for every data point while the y-axis shows the frequency of occurrence of this distance. It can be noticed that the distances are grouped in two clusters, which is analogical with the ring appearance as it contains two circles with different radii. The local maxima in this histogram are the fit circle radii. These radii are used as a recognition feature as described in the 'recognition' subsection.

The circle fit algorithm is able to process 1M events / s.

Radius Analysis: The information resulting from the circle fit is used to build an additional list of features for the given object. An analysis of the average radius distribution per angle resolution (1 – 360) is made. Afterwards, an analysis of this distribution is performed to extract these parameters:

1. The dispersion range of the average radius (Max – Min).
2. The number of local maxima.
3. The distance between the local maxima in degree.

Those parameters are grouped and sent to the "Learning" and "Recognition" sections.

Fig.5 depicts the average radius distribution for the hexagon depicted in Fig. 2(b) versus the angle. It can be noticed that the average radius dispersion range is ~7. There exist six local maxima with a quasi-equivalent distance of ~60 degree. These six maxima correspond to the hexagon corners. There are two discontinuities in the curve close to the second and fifth maxima, which correspond to the missing events on the top and bottom edges of the hexagon, respectively. This radius distribution can be further smoothed using an average filter to improve the dispersion regularity.
The radius analysis algorithm processes up to 80k events/s.

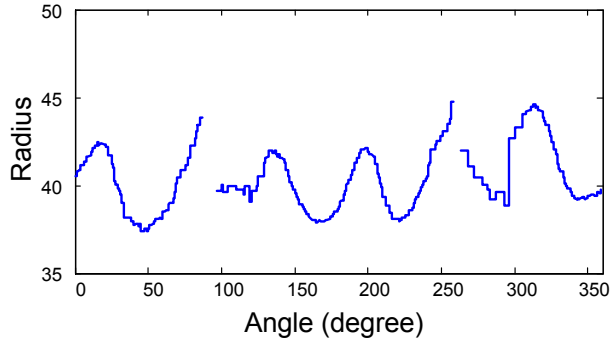

Fig. 5 Average radius distribution of the shape depicted in Figure 2(b) versus the angle (15° segment)

Orientation Estimation: This module aims to estimate the object edges orientation as an additional feature for the recognition. The dual-line sensor provides output sorted by pixel address which greatly reduces the computational complexity of the orientation estimation. The orientation estimation method performs the following steps:

1. It searches the next N-neighbors (N = 1 : 1000) within the sorted TAE list for every event.
2. A linear fit is performed over those selected events and the resulting slope is used to compute the local edge orientation.
3. The event polarity is used to extract the gradient direction that is 90° clockwise for ON Events and 90° counter-clockwise for OFF events. Thus, the gradient vectors point from dark to bright regions in the shape.

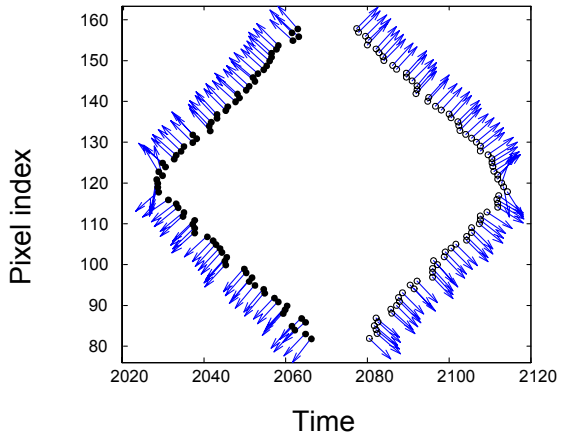

Fig. 6 Local orientation estimation for the quadrangle depicted in Fig.2(a)

Fig. 6 shows the local orientation estimation for the quadrangle edges. In this case, the event polarity (OFF or ON) is used for the estimation of the event direction. The filled dots represent the OFF events while the circles show the ON events. The computed gradient orientations are displayed as vectors originating from the data points. It can be noticed that every edge is providing a quasi-uniform orientation vector. Using these orientation vectors, a histogram showing the frequency of the gradient orientations vs. the orientation angle is built. Subsequently, a statistic over the orientation histogram is performed by extracting the local maxima in order to estimate the global orientation for every edge in the object.
Fig. 7 depicts the histogram of the local orientation estimation. The four local maxima represent the orientation of the four edges of the quadrangle.
The information (number of local maxima) from this histogram is stored in the database as a feature for the recognition purpose.
The orientation estimation algorithm is able to process 13k events / s on the target processor.

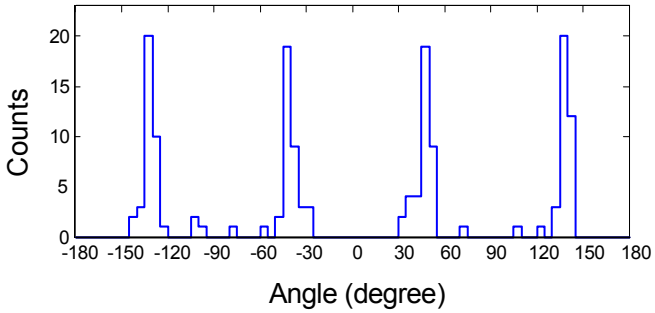

Fig. 7 Histogram of events orientation (counts vs. orientation angle)

D. The Learning

The aim of this step is to memorize the object and its features in the database. This task receives the object information (name) from the user interface and allocates an ID for it. Then, it assigns to the object ID the set of features that were collected in the previous steps – Fit circle' radius and

radius statistics as well as the orientation statistics – and includes them to the database. This step is called the indexation of the object. The information stored in the database is used as a prior knowledge for the recognition.

E. The Recognition

This is the targeted vision task that aims to identify the captured object. This module receives the features from the previous step and matches them to one of the objects stored in the database. The matching consists of comparative steps between the extracted and stored features. The following parameters are tested:

1. First, the normalized object length is checked. The length (in µs) represents the time required for an object to traverse a line of pixels. It is scaled by a ΔT factor (ΔT is the time needed for the object to pass from the 1st line to the 2nd line of pixels) to obtain a normalized length, independently from the object velocity.
2. Afterwards, the radius error resulting from the circle fit of the actual object is checked to be below 8%. If this parameter matches, then the object has a near-to-circle shape form.
3. It compares the fit circle radius of the actual object to the stored radius within a given deviation 'Δ' (Δ is set to 20%). If the parameters match, then step 4 will be performed otherwise the object is set as 'non-identified'.
4. In this step, the number of local maxima is compared for a deviation under 10%. If the number matches one of the stored objects, then, this one will be assigned to the current observed pattern. Otherwise, the distance between the local maxima will be used as a parameter to extract a matching object. In both cases, the recognition procedure stops after a success in finding an object matching the current pattern.
5. The recognition procedure can be pursued if step 4 does not identify the current pattern. In this step, the dispersion range of the radius and the orientation histogram are compared for the actual pattern and the stored object. If both parameters match for a deviation of 20%, then the candidate object is assigned to the actual pattern otherwise, the object is set to 'non-identified'.

Thus, the recognition procedure will be identifying the actual object whenever the targeted object is present in the database and the parameters are robustly extracted. Otherwise, the learning process has to be run in order to include the actual pattern in the object database. The algorithmic complexity of the recognition step includes the feature extraction complexity and the database access duration. In Section V, the algorithmic complexity for the recognition of test objects is given.

F. The Database

This is a memory region containing a listing of objects and their features that is built up by the learning process. This database is updated by the learning step for every new target. It is used by the recognition process for the object identification.

A typical exploitation of this vision system for machine applications consists of starting the system by learning its environment and indexing the dedicated objects in the database. Afterwards, the system can be run for the recognition of the prior learnt objects.

V. Evaluation of the Proposed Concept

The proposed vision system has the advantage of providing a sparse object representation, which consists of timed address-events. The events are typically located on the object contours facilitating the shape recognition. The processing concept includes a pre-processing step for the data preparation, feature extraction techniques for parameters generation and a decision-making process for object memorization or identification. This vision concept is efficient and easy to implement and the object recognition is computationally fast. This concept has been successfully evaluated using objects with regular shapes.

Fig.8 depicts four shapes (ball, screw nut, cube and cuboid) for testing this recognition concept. A resulting 3D features space is shown in Fig. 9 from using the test shapes 1000 times at different velocity. In this case, the used features are the fit circle radius, the radius error and the normalized object length.

Fig. 8 Original shapes of test-objects (from left to right: ball, screw nut, cube and cuboid)

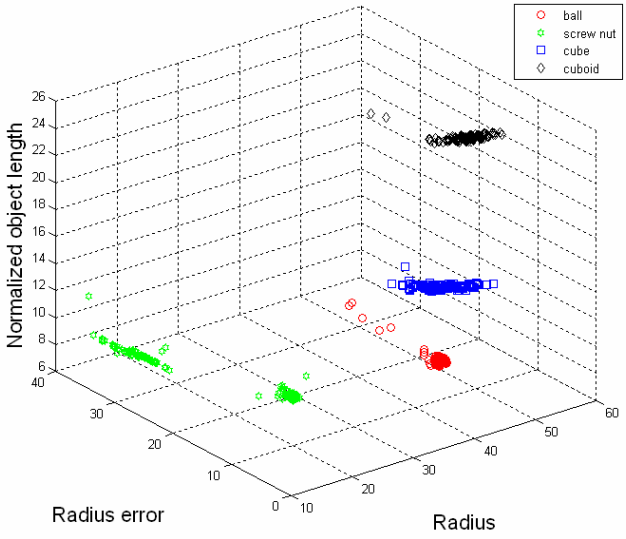

Fig. 9 The resulting features distribution in 3D space from the test-objects using the radius, the radius error and the normalized object length

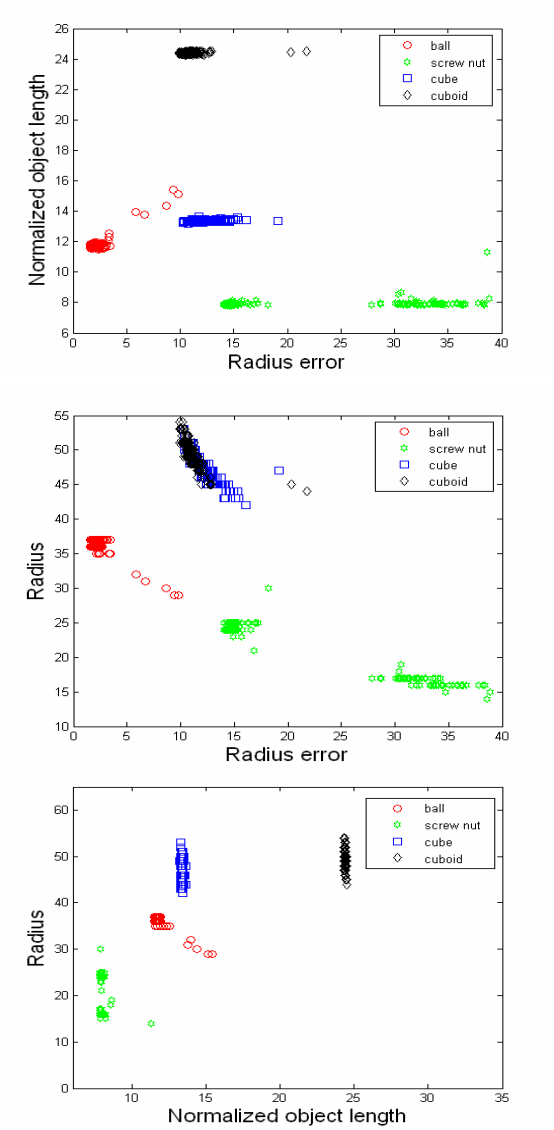

Fig. 10 Illustration of the resulting features distribution in 2D space using the normalized object length vs. the radius error (Top), the radius vs. the radius error (middle) and the radius vs. the normalized object length (bottom)

For a better visibility of the features distribution, Fig.9 is split into 3 subfigures (Fig.10), each depicting the dependency between two parameters. In Fig.10 (top), it can be noticed that the cuboid can be fully distinguished using the normalized object length as it builds an independent cluster. The screw nut builds two clusters of features representing the inner and outer contours. Besides one outlier, the screw nut can be discerned using the object length and the outlier can be isolated using the radius error. A major part of the radius error distribution from the ball lies under 8%. The radius parameter (Fig.10 bottom) can be used to fully separate between the ball and the cube.

In this example, the circle fit and the object length calculation seem to be sufficient to distinguish between the four test objects. Both techniques are computationally efficient as they are using a closed form and thus, the complexity is proportional to the number of AE. For these test examples, the features extraction and recognition steps require less than 1ms/object. One major contribution in the processing time is the object detection, which may take many ms, depending on the object velocity, as the object has to fully traverse both lines of pixels before the processing starts. In other words, a simple, low-cost DSP can process more than 35 objects per second for shapes that produce 500 events in average with less than 30ms travel time across the sensor field of view. The recognition step is robust for object shapes that are close to a circle e.g. ball, cube, pentagon, hexagon… etc. The method can be extended to other shapes by adapting the fitting method to these shapes.

Many applications can exploit the advantage of this system in monitoring high-speed objects for surface vision (defect recognition, remote diagnosis…), quality vision (presence inspection, contour inspection, position inspection…) and for automated fabrication processes (sorting, classification…).

VI. Conclusions

In this paper, a real-time data processing concept for asynchronous "Address Event" image sensors in high-speed machine vision applications has been presented. The processing steps include data denoising and normalization, features extraction and indexation or recognition of objects. Using a circle fit and orientation estimation, the features extraction has proven to allow the recognition of objects with regular shapes (like Geon e.g. ball, cube, hexagon…etc). Furthermore, those methods are computationally efficient as the complexity is linearly proportional to the number of events.

This processing concept can be extended to other shapes by considering other models fitting for the features extraction.

References

[1] D.H. Ballard and C.M. Brown, "Computer Vision," *Prentice-Hall Inc. Englewood Cliffs, New Jersey*, 1982

[2] W.-C. Fang, "A System on-Chip Design of a Low-Power Smart Vision System," *IEEE Workshop on SIPS 98, Cambridge*, pp.63-72, USA, 1998

[3] M. Hoffstaetter, A.N. Belbachir, E. Bodenstorfer and P. Schoen, "Multiple Input Digital Arbiter with Timestamp Assignment for Asynchronous Sensor Arrays," *IEEE ICECS06*, Nice, France, 2006

[4] T. Komuro, I. Ishii, M. Ishikawa and A. Yoshida, "A Digital Vision Chip Specialized for High-speed Target Tracking," *in IEEE Transactions on Electron Devices*, vol.50, pp.191-199, 2003

[5] Lichtsteiner, P.; Posch, C.; Delbruck, T., "A 100dB dynamic range high-speed dual-line optical transient sensor with asynchronous readout," *in the Proceedings of ISCAS 2006*, pp 1659- 1662 , May 21-24, 2006

[6] Lichtsteiner, P.; Posch, C.; Delbruck, T., "A 128×128 120db 30mW asynchronous vision sensor that responds to relative intensity change," *Solid-State Circuits, 2006 IEEE International Conference ISSCC, Digest of Technical Papers* , pp. 2060- 2069, Feb. 6-9, 2006.

[7] C. Posch, M. Hofstätter, D. Matolin, G. Vanstraelen, P. Schoen, N. Donath and M. Litzenberger, "A Dual-Line Optical Transient Sensor with On-chip Precision Timestamp Generation," *IEEE International Conference ISSCC, Digest of Technical Papers* , pp., Feb. 11-15, 2007

[8] O. Schrey, J. Huppertz, G. Filimonovic, A. Bussmann, W. Brockherde and B.J. Hosticka, "A 1 K×1 K High Dynamic Range CMOS Image Sensor with On-chip Programmable Region-of-Interest Readout," *in IEEE Journal of Solid-State Circuits*, vol.37, pp.911-915, 2002

[9] G.P. Stein, E. Rushinek, G. Hayun and A. Shashua, "A Computer Vision System on a Chip: a Case Study from the Automotive Domain," *IEEE Conference for CVPR*, vol.3, pp.130-134, 2005.