

# Efficient Feedforward Categorization of Objects and Human Postures with Address-Event Image Sensors

Shoushun Chen, *Member, IEEE*, Polina Akselrod, Bo Zhao, *Student Member, IEEE*,  
 Jose Antonio Perez Carrasco, *Member, IEEE*,  
 Bernabe Linares-Barranco, *Fellow, IEEE*, and Eugenio Culurciello, *Senior Member, IEEE*

**Abstract**—This paper proposes an algorithm for feedforward categorization of objects and, in particular, human postures in real-time video sequences from address-event temporal-difference image sensors. The system employs an innovative combination of event-based hardware and bio-inspired software architecture. An event-based temporal difference image sensor is used to provide input video sequences, while a software module extracts size and position invariant line features inspired by models of the primate visual cortex. The detected line features are organized into vectorial segments. After feature extraction, a modified line segment Hausdorff distance classifier combined with on-the-fly cluster-based size and position invariant categorization. The system can achieve about 90 percent average success rate in the categorization of human postures, while using only a small number of training samples. Compared to state-of-the-art bio-inspired categorization methods, the proposed algorithm requires less hardware resource, reduces the computation complexity by at least five times, and is an ideal candidate for hardware implementation with event-based circuits.

**Index Terms**—Human posture categorization, bio-inspired categorization, event-based circuits, address-event image sensor.

## 1 INTRODUCTION

PRIMATES' vision is extremely accurate and efficient in the categorization of objects. The current theory of the cortical mechanism responsible for object categorization has been pointing to a hierarchical and mainly feedforward organization [1], [2], [3], [4], [5], [6], [7], [8], where short-range feedback is believed to play a secondary role. This organization can provide hierarchical features of increasing complexity and invariance to size and position, making object categorization a multilayered and tractable problem.

In this paper, we present an energy-efficient system which combines 1) a custom designed smart image sensor, and 2) a biologically inspired efficient categorization algorithm. The image sensor is equipped with temporal difference processing hardware and outputs data in the format of binary event stream, in which "1" stands for a pixel on a motion object and "0" represents a still background pixel. The algorithm filters the individual motion events to extract a very limited number of line features. A modified line segment Hausdorff distance classifier is then employed to measure the similarity of the features with

those extracted from a small set of library objects, as explained in Section 4. The goal of our research is to allow embedded platforms to perform sophisticated object categorization tasks for indoor environments such as assisted living. The proposed approach is innovative due to its high data encoding efficiency, large saving in computation complexity, as well as an efficient way to achieve robustness to translations and scale while categorizing objects. This is also the first address-event categorization algorithm that provides size and position invariance [9], [10]. Particular care was taken in the design of the algorithm to allow for a straightforward and efficient hardware implementation.

We herein show the application of our system and algorithm toward the categorization of human posture. This application is gaining increasing attention, especially in the area of assisted living applications and sensors networks [11], [12], [13], [14], [15], [16], [17], [18]. Posture categorization can be used to monitor human behavior, in particular for home care of the elderly [19], [20]. But the results presented in this paper have very broad applicability: personal health care, environmental awareness, intelligent visual human machine interface, video game systems, and human-robot interaction, just to name a few.

Based on commercially available image sensors and powerful personal computers, an impressive series of research work has been reported for human posture categorization [21], [22], [23], [17], [18]. In general, those approaches first detect moving objects by the analysis of video stream, then extract human silhouettes using background subtraction technique [24], [25]. Blob metrics are represented into multiple appearance models [26] and, finally, posture profiling is conducted based on frame-by-frame posture classification algorithms. Due to the complexity, these algorithms need to be implemented on

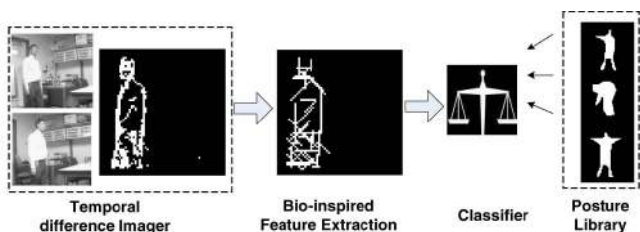


Fig. 1. Efficient feedforward system used to categorize objects and human postures. The system collected visual information with a  $64 \times 64$  pixels address-event temporal difference image sensor. A bio-inspired contour-based feature extraction algorithm, a classifier, and a reference posture library.

powerful computers (1 GHz processors or better), even when categorizing only a small subset of human body postures [27]. These requirements limit the use of these algorithms in real-life applications with low-cost and lightweight wireless platforms, such as embedded computers, sensor networks, or smart cellular phones.

In addition to the complexity of the algorithms, the conventional frame-based (fb) image sensors employed in these systems also contribute to lower energy efficiency. In fact, the output of conventional image sensors, as a matrix of pixel color values, contains a very high level of redundancy. Large amounts of unimportant data have to be read and processed before obtaining the features of interest [12]. As a matter of fact, the first step of many computer vision algorithms is to remove the background and extract object line segments or motion contours [21], [28]. Smart image sensors combine focal-plane signal processing and implement novel approaches to improve the computation efficiency when compared to conventional discrete sensor-processor systems. Among these are various image sensors for motion detection, resolution reduction, and even object tracking [29], [30], [31], [32], [33], [34], [35], [36]. The system presented in this paper is based on one type of these sensors. Combined with ultraefficient bio-inspired object categorization algorithms, the system allows implementation and execution on a small FPGA and a cellular phone platform [37]. Since no raw video data are involved, patients' privacy is protected when they are monitored.

This approach and algorithm is very lightweight when compared to more sophisticated systems [38], [39] that can operate in more general conditions. The paper is organized as follows: Section 2 introduces the system. Section 3 describes the proposed line segment feature extraction algorithm, and Section 4 describes the size and position invariant categorization algorithm. Section 5 discusses the computation complexity. Section 6 reports the experimental results as well as comparison to other algorithms. Section 7 discusses similarities and differences with previous relevant work, and Section 8 concludes the paper.

## 2 SYSTEM OVERVIEW

The architecture of the proposed system is illustrated in Fig. 1. We use a temporal difference image sensor named MotoTrigger [19], [36], combined with a software implementation of a bio-inspired feature extraction unit and a classifier. A known set of posture library (or object library) is used for evaluating the categorization performance.

The temporal difference image sensor compares two consecutive image frames and only outputs the addresses of those pixels whose illumination changes by an amount larger than a predefined threshold. If the scene illumination and object reflectance are constant, the changes in scene reflectance only result from object movements or camera translation. The background information is thus filtered by the MotoTrigger camera, sparing the processor from this computation after image acquisition with a standard intensity camera [21], [28]. The merits of employing such an image sensor result not only from the kind of data collected, but also from the lower amount of data that need to be communicated. The image sensor encodes the addresses of the motion-sensing pixels into a stream of events and communicates through a protocol called Address Event Representation (AER) [40], [35], [41]. In AER terminology, events are communication packets that are sent from a sender to one or more receivers. The MotoTrigger sensor compares the pixel integration voltage to that of the previous frame. When this difference reaches a threshold voltage, the pixel will generate an event and request communication with an outside receiver. An "address-event" refers to the image coordinates of a certain pixel. MotoTrigger has a nominal pixel count of  $64 \times 64$ . We have used this image size in our work and experiments.

The feature extraction algorithm of Fig. 1 performs directly on individual pixel events, rather than frames. Each address-event is sent in parallel to a battery of orientation filters based on the Gabor functions, and convolution operation is performed on the fly. The responses of the filters are analogous to feature map neurons in biological networks, where individual synapses deliver charge pulses to targeted neurons. These filters extract zero-crossing or line information from the image, as explained in Section 3.1.

After that, a MAX-like operation is applied in order to find the maximal response among the feature maps or "neurons." Only those who reach the maximal response can survive during the competition and each "neuron" represents a vectorial contour segment in the image (explained in Section 3.2). The extracted line segments are fed to the classifier to measure the similarity of the input line segments with those of a set of library objects. The classifier is based on a modified line segment Hausdorff-distance scheme. Size and position invariance are achieved by using event-cluster-based methods that can be easily computed from individual pixel events.

## 3 BIO-INSPIRED FILTERS AND FEATURE EXTRACTION

The feature extraction unit is inspired by a recent model of object categorization in the primate visual cortex [4]. The key idea of the model can be summarized as: 1) a hierarchical visual processing, to build invariance to position and scale first and then to viewpoint and other transformations, 2) along the hierarchy, the feature maps size increases, 3) the processing of information is feedforward.

As shown in Fig. 2, an image is first processed by a network of simple filters "S1" (after nomenclature in [4]). Each filter models a neuron cell with certain size of feature maps and responds best to basic feature at certain orientation. In the second stage, layer "C1" combines all the outputs from "S1" cells that have the same orientation and finds the

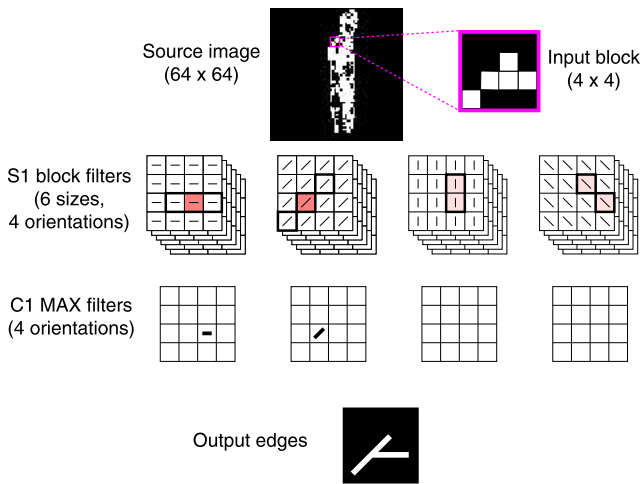


Fig. 2. Hierarchical organization of the feature extraction unit. The highlighted square contains a zoomed-in part of the original image on the left. For the sake of clarity, the feature extraction is exemplified on this 4 × 4 subimage. It is first processed by a network of simple filters “S1.” Each filter models a neuron cell with a specific feature map size that responds best to the basic feature at certain orientation. Each pixel has 24 neurons associated with it (four orientations and six sizes). The neurons of the same feature map size and orientation are organized into 4 × 4 squares. The latter are shown as four piles (by orientation), each pile containing six different sizes. The neurons with maximal response among their neighbors are highlighted. In the second stage, layer “C1” combines the outputs from the same orientation “S1” cells whose response is maximal (highlighted) and sufficiently high. For example, the 3-pixel horizontal line gives one high peak, while the 2-pixel vertical line gives two low peaks. In the “C1” layer above, only the surviving neurons are shown. Thus, the image is represented by two line segments of size 3: one horizontal and one of 45 degree angle. The line segments are visualized as thick (multiple pixel of width) white lines on the output image at the bottom.

maximal response (MAX) among them. A neuron cell which reaches the peak response stands for a feature (line or edge) at the same size and orientation as that neuron cell.

Our approach is summarized as Algorithm 1 and the following sections will explain the implementation of the algorithm in detail.

**Algorithm 1.** Procedure for line segment extraction

- (1) **S1:** each input image is filtered by 24 filters: 4 orientations ( $\theta = 0^\circ, 45^\circ, 90^\circ$  and  $135^\circ$ ), and 6 kernel scales ( $s = 3, 5, 7, 9, 11, 13$ ). This generates 24 feature maps.
- (2) **C1-1st Max operation across neighborhood and orientation:** each neuron output, representing different orientation maps, will be compared to all other neurons within the same size feature map. After this step, only the neurons located at the center of the feature and with the right orientation feature map will survive (will be non-zero).
- (3) **C1-2nd Max operation across scales:** The neurons from the previous step contribute potential line segments of the corresponding scale. Within each orientation, overlapping “edge-candidates” are merged to create a new line segment with a neighboring maximum operation.

### 3.1 Simple Cells and Local Filter Response

Simple cells are used to build object selectivity. The temporal difference image is convoluted with a multidimensional

array of simplified Gabor filters. Gabor filters are able to achieve selectivity in space, spatial frequency, and orientation [4]. Their function is described in

$$F(x, y) = \exp\left(-\frac{x_0^2 + \gamma^2 y_0^2}{2\sigma^2}\right) \cos\left(\frac{2\pi}{\lambda} x_0\right) \quad (1)$$

$$x_0 = x \cos \theta + y \sin \theta, y_0 = -x \sin \theta + y \cos \theta.$$

Selection of the filter parameters, i.e., the aspect ratio,  $\gamma = 0.3$ , the orientation  $\theta$ , the effective width  $\sigma$ , and the wavelength  $\lambda$ , were extensively addressed by Serre et al. [5], Chen et al. [42], and a similar set of parameters is adopted in our work. Moreover, for implementation simplicity, the orientation filters are normalized to integer values by scaling the minimum value to 1 and by taking the nearest integer. Notice that the filter size depends on the image size and the size of the features to be categorized. In this work, we arrange the filters to have six different sizes, ranging from 3 to 13 (filter kernels of  $3 \times 3$  to  $13 \times 13$ ), and four orientations, i.e., 0, 45, 90, and 135 degrees. Therefore, the network of filters is able to detect features (transitions from black to white or vice versa) as short as 3 and as long as 13, at 4 orientations. The convolution result of each filter will be one matrix of neuron cells. Since the filters are scaled to integer values, the output of the filters will have large integer values also.

Notice in 1 that here we use an even Gabor filter (cosine function), as opposed to the odd Gabor (sine). The even Gabor is better suited for highlighting line segments, while the odd Gabor is better suited for detecting edges between a dark region and a light region. Processing a luminance image with an odd Gabor detects the edges and gives you “segments” for those detected edges. In this case, though, we use a temporal difference camera that gives us directly line segments, therefore an even Gabor filter set is better suited.

Fig. 3 illustrates the feature maps of neurons for a test image. One can note that, if the size of the feature is larger than the filter size, i.e., the neuron feature map size, a trapezoid-shaped response is obtained along the direction of the feature. In this case there is no single maximum of the function. When the size of the feature matches the neuron feature map size, a triangle-shaped response is obtained, also resulting in a high-peak response. If the size of the feature is smaller than the neuron feature map size, either a low peak is observed or there is no local maximum at all (multiple pixel will have the same maximum values, as in Fig. 3b). Finding a single peak is thus indicative of what size filter best describes the feature detected.

### 3.2 Complex Cells and Neighborhood Competition

Now we proceed to find the orientation, the location, and the size of the features. This is done in two steps of MAX operations: first to find the right position and orientation filter—or direction of the line, then to determine the length of the line by examining the responses of the same orientation filter but at different sizes.

First, we find the maximum response across neighborhood and all orientations. A maximum operation (MAX) is performed [4] by comparing each neuron response to the one of the other neurons (feature maps with different orientation) that fall within its feature map. The feature maps in our implementation are square areas of the size of the filter

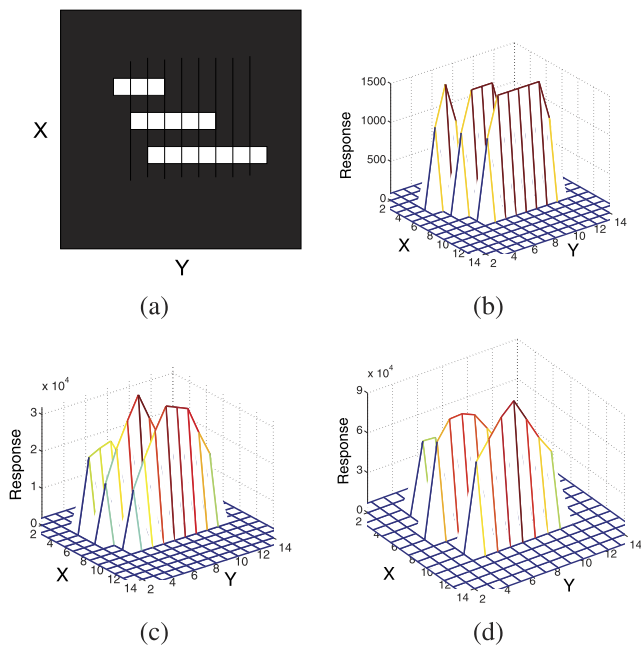


Fig. 3. Feature map of S1 neuron cells for a test image. This represents the output of the line segment Gabor filters. (a) Source image which consists of three horizontal lines, with lengths of 3, 5, and 7, respectively. (b)-(d) Neuron cells responses, implemented as convolution of the image with horizontally oriented Gabor filters of sizes 3, 5, and 7, respectively. The size of the image in this example is reduced for simplicity. Pixel (0, 0) is on the top left corner in these images.

centered at the corresponding pixel (e.g.,  $3 \times 3$ ,  $5 \times 5$ , etc.). For instance, a neuron with feature map size of  $3 \times 3$  will be compared to the three other  $3 \times 3$  maps with different orientation.

In our implementation, each neuron is built with an attached digital flag bit to indicate whether or not this neuron can survive during competition with other neurons. A neuron will deselect itself from local competition by turning the flag bit to "0" if at least one of its neighbors has a higher response. The principle behind this choice is the following: A neuron has a higher response than its neighbor of the same orientation and feature map size due to a better position.

Second, a MAX operation will be performed to find the size of the feature, and thus the line length. In our system, the size of the filters ranges from 3 to 13. Each peak gives rise to a potential line segment of the corresponding size. Still, line segments of the same orientation but of different sizes may overlap and hence make the representation redundant. We find the line length by comparing all the neurons of the same orientation (the one that won the first MAX operation), but with different sizes. Only the one reporting the maximum response will survive as the best descriptor of the size of the feature. Fig. 4 shows the feature map of surviving neurons of Fig. 3c. Compared to the original map, one can note that only the sufficiently high peak neurons are left.

In some images there are features (lines) larger than the largest feature map size, resulting in the detection of multiple overlapping line segments. To avoid this, we postprocess these line segments and merge them. By doing this, the size of the maximum extractable line is not restricted to the maximum filter size. Fig. 5 shows an example of filter

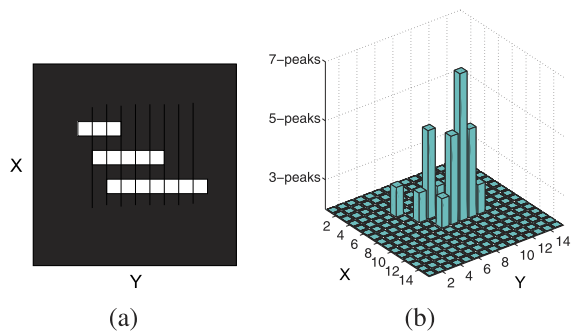


Fig. 4. Selection of the most appropriate filter length based on neighborhood maximum operation. (a) The processed image, the same as in Fig. 3a. (b) Feature map of surviving neurons of Figs. 3b, 3c, and 3d after neighborhood MAX operation. The surviving neurons corresponding to the sizes 3, 5, and 7 are shown as low (3-peaks), medium (5-peaks), and high (7-peaks) bars, respectively. The size of the image in this example is reduced for simplicity.

response to an image with a line longer than the largest filter size ( $13 \times 13$ ). By keeping the number of line segments as low as possible we maximize the algorithm efficiency because each line segment needs further processing by the classifier. Fig. 6 shows the extraction result of two temporal difference images. In the source image, the outline of the human is composed by scattered pixels, while in the reconstructed image, the outline is replaced by a straight line that best estimates the feature.

### 3.3 Discussion

Compared to the previous work [5], our approach differs in the way the MAX operations are performed. There, C1 cells are obtained by performing max-like operation over simple S1 units with the same preferred orientation, but slightly different positions, in order to gain position tolerance. Each neuron compares its response to its surroundings and will copy the maximum response within its neighborhood as its own response. Therefore, the final resolved feature becomes wider, and it is harder to reduce this to a line with single pixel width, as desired. This effect is illustrated in Fig. 7. One can note that, compared to the feature maps of Fig. 3, the feature is highlighted by a much larger number of neurons (about three times). Larger neuron populations and wider maximal filter responses reduce the precision of the

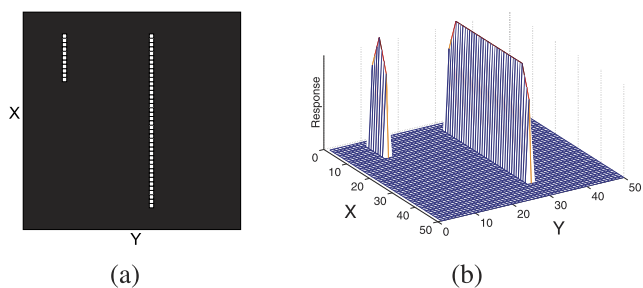


Fig. 5. Example of line extraction of two lines that do not exactly match the size of filters in S1. (a) The source image contains two lines of sizes 11 and 40 pixels. (b) Response of the horizontally oriented Gabor filter of size 11 to (a). The shorter line exactly matches the size of the filter. Although the longer line does not match any of the filter sizes, it is successfully detected by the algorithm. The size of the image in this example is reduced for simplicity.

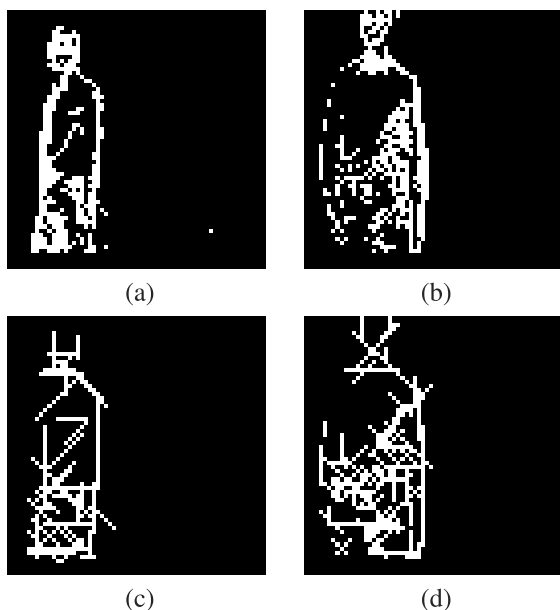


Fig. 6. Feature detection examples with real data. (a) and (b) Two source temporal difference images. (c) and (d) The corresponding extracted line segments.

line extraction algorithm. As a matter of fact, with the results in Fig. 7 the exact position and size information of the feature is lost.

## 4 SIZE AND POSITION INVARIANT CLASSIFIER

In the previous section, we described our methodology for extracting line features from input images or frames. This line extraction technique is applied to all input images or frames. A subset of inputs is used to generate a tagged library of line features. The library is used to compare subsequent input images or frames to the ones in the library by means of a classifier. In this section, we describe the classifier used in our algorithm, and how our object classification algorithmic implementation achieves invariance to size and position.

### 4.1 Modified Line Segment Hausdorff Distance

In computer vision, the Hausdorff distance has also been applied to categorization with conventional frame-based image sensors and with good results [43], [44], [45], [46],

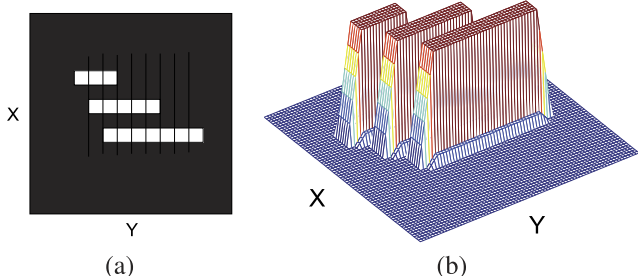


Fig. 7. The filter response of [5] involves a larger neuron population than the one extracted with our methodology (compare to Fig. 3). (a) The processed image, the same as in Fig. 3a. (b) A corresponding neuron cells response. This reduces the precision in the localization of the feature. The size of the image in this example is reduced for simplicity.

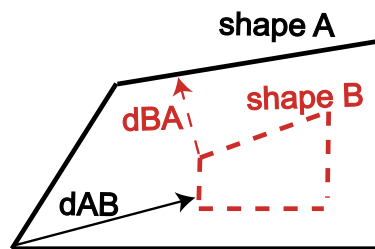


Fig. 8. The classical Hausdorff distance between two geometrical shapes, which measures how close the shapes are to each other [48]. The solid arrow is the distance between the solid figure and the dashed figure. The dashed arrow is the distance between the dashed figure and the solid figure. The classical Hausdorff distance is the maximum between the two. In our approach we use a similar idea to compare two sets of line segments by means of the line segment Hausdorff distance.

[47]. The idea to measure distance between shapes goes back to Hausdorff [48], see Fig. 8. This approach naturally fits in our case as well. The classifier computes the modified line segment Hausdorff distance between the line segments of the test image and each one of the predefined library images. Our definition is a modified version of the one given in [47]. The test image is identified with the library image yielding the minimal distance.

### 4.2 Size and Position Invariant Categorization

Once line segments information is extracted, the input image is tested for similarity with each library image. The two images first need to be aligned before comparing their distance. For example, face recognition algorithms operating in modern digital cameras align a face template on the location of the eyes found in the input image [47]. To achieve this for human postures or even generic objects, we propose to align two objects using their center position. In addition, the two objects also need to be stretched to the same size to make the comparison invariant to the object's distance to the camera.

In order to perform the alignment and stretching, we need to first find the size and position of the object. The challenge here is mainly about how to effectively find the object when noise pixels and multi-objects (human or pets) inevitably exist in the scene. Here, we propose an event-based clustering algorithm which is inspired by the object tracking techniques reported in [49], [50]. The key processing element is so called "cluster," which is a block of pixels belonging to the same object. A cluster is described by its four boundaries (rectangular shape), center point, and number of events. By trading off the immunity against noise objects and the implementation complexity, we employ three clusters and consider the largest one as the object-of-interest. The algorithm is implemented in an on-the-fly fashion. Each time when a pixel event is received, the three clusters will be updated as the following procedure:

- Examine the distance of the pixel event to the existing clusters. If the distance is beyond a certain threshold ( $\hbar$ ), a new cluster will be built which is centered at the address of the new pixel. The distance of a new pixel to a cluster is examined by the equation

$$dx < \hbar, \text{ and } dy < \hbar, \quad (2)$$

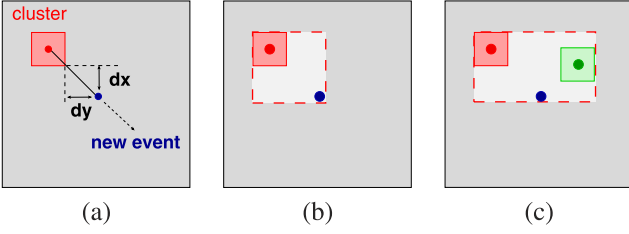


Fig. 9. Cluster update procedure. (a) Distance measurement of a pixel to an existing cluster. (b) When the distance falls within the threshold, the cluster grows to enclose the new pixel. (c) Two clusters merge when the pixel belongs to both of them.

where  $dx$  and  $dy$  are the horizontal and vertical distance from the new pixel to the boundary of the cluster, respectively. The threshold  $\hbar$  is an empirical parameter, obtained by simulation.

- If the pixel falls into the boundary of an existing cluster, the cluster will simply increase its number of events by 1. If the pixel falls out of the boundary of a cluster but within the threshold ( $\hbar$ ), the pixel is still considered to belong to the cluster and the latter will grow its boundary to enclose the new pixel and without forgetting to increase its number of pixels.
- In the case when the pixel belongs to more than one cluster at the same time, the clusters will be merged into a larger one.
- If the pixel belongs to none of the existing clusters, the cluster containing the least number of pixels is considered a noise object and dropped. A new cluster will be built at the address of the new pixel.

The procedure mentioned is clearly illustrated by Fig. 9.

With the center and boundary information, both position and size invariant categorization can be achieved. The test object can be aligned and resized with respect to the library object. The alignment of the centers is followed by a resizing operation to make them have the same size. The line segment Hausdorff distance is updated to

$$\vec{D}(I_t, I_l) = \frac{\sum_{e_t \in I_t} \left( \min_{e_l \in I_l} d \left( \frac{S(I_t)}{S(I_l)} (e_t - C(I_t)), e_l \right) \right) |e_t|}{\sum_{e_l \in I_l} |e_l|}, \quad (3)$$

where  $C(I_t) = (C_x^{I_t}, C_y^{I_t})$  is the center of the test object, while  $e_t \in I_t$  and  $e_l \in I_l$  denote the line segments.  $S(I_l)$  and  $S(I_t)$  is the size of the test and library object, respectively.

Both alignment and rescaling preserve the angles between the line segments and hence are consistent with the representation.

Fig. 10 shows the intermediate clusters when doing size and position calculation on a testing image and the effect of resizing and alignment. We note that our approach demonstrates a great implementation efficiency of the image scaling. For instance, to resize the centered image by the ratio of  $\alpha$ , we simply multiply the coordinates of the line segments by  $\alpha$ . This is a built-in advantage of the vectorial feature representation, while, in conventional approaches, scaling an image involves complex operations such as nearest-neighbor interpolations, supersampling, and resolution synthesis. The drawback of this approach is that if multiple objects are present in the view, then it is not possible to scale the image with this technique as the

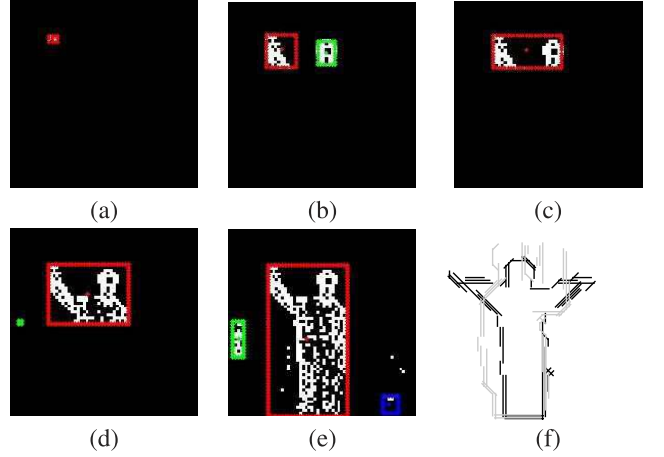


Fig. 10. (a)-(e) Intermediate clusters when doing size and position calculation on a testing image. (f) The graphical representation of the aligned and resized test human outline (black) with a library object (gray).

scaling dimensions will become the average between all points of the objects.

## 5 IMPLEMENTATION COMPLEXITY

In this section, we examine the algorithmic complexity of the proposed object categorization technique. First of all, the event streams produced by the address-event image sensor are sent in parallel to a battery of S1 orientation filters. Within each feature map, the neurons are updated on the fly. In order to evaluate the algorithmic complexity of this step, consider the following example on a  $3 \times 3$  kernel that will also apply to larger kernels. Let  $S$  be the source image and  $F$  a filter designed to detect vertical bars, for instance,  $F$  is the kernel in the equation

$$F = \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}. \quad (4)$$

The response  $R$  is defined as the convolution of  $S$  with  $F$ , as reported in the equation

$$R(i, j) = \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} S_{k,l} F_{2+i-k, 2+j-l}. \quad (5)$$

Typically, the image  $S$  has to first be buffered in a memory as a frame, then each  $3 \times 3$  pixels undergo convolution by (5), and finally the result is written into the memory allocated for response  $R$ . The number of operations is thus  $n \times n \times 3 \times 3$ , where  $n \times n$  is the size of the frame.

However, in the event-based approach, this operation can be optimized as follows: Suppose the pixel  $S(i, j)$  generates an event. In this case, (5) involves only  $3 \times 3$  pixels. The change  $\Delta$  in  $R$  is given by the equation

$$\Delta \begin{pmatrix} R_{i-1, j-1} & R_{i-1, j} & R_{i-1, j+1} \\ R_{i, j-1} & R_{i, j} & R_{i, j+1} \\ R_{i+1, j-1} & R_{i+1, j} & R_{i+1, j+1} \end{pmatrix} = \pm F. \quad (6)$$

In  $\Delta$ , the positive sign is taken if  $S_{i,j}$  changed to 1 and the negative sign is taken if  $S_{i,j}$  changed to 0. In other words, all

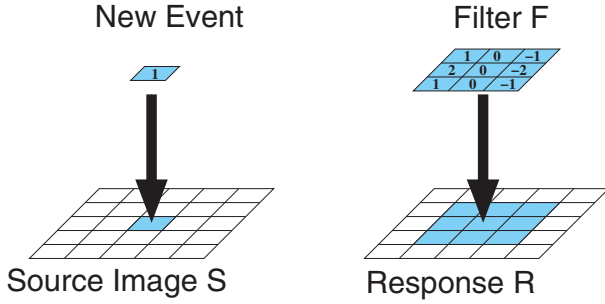


Fig. 11. The principle of our event-based implementation of convolution. The squares on the left and on the right depict the pixels of the source image  $S$  and the filter response  $R$ , respectively. Now an event occurs (the square on the left is the incoming value of the center pixel). The filter response  $R$  is updated by adding the filter kernel  $F$  (a  $3 \times 3$  matrix on the right, above  $R$ ) to the previous values of  $R$  at the corresponding submatrix, according to (6).

we need to do is to add  $\pm F$  to a  $3 \times 3$  submatrix of  $R$  centered at  $(i, j)$  (the number of operations is 9 per event). See Fig. 11. For implementation simplicity, instead of using floating-point weights and multiplication, the orientation filters are normalized to integer values and therefore the projective fields are implemented only as integer addition and subtraction operations.

We now report on the cost of extraction of the features described in Section 3. For a given image with dimensions  $n \times n$  we perform convolution of the image matrix with each of the filters matrices. The convolutions are computed only when an event occurs (one pixel can change per event). Within the MAX operation only the surviving neurons are taken into account as we extract line segments. The total number of operations is

$$1816 \times 36\% \times n^2 + e, \quad (7)$$

where  $e$  is the number of the extracted line segments. In our system,  $n = 64$  and  $e \approx 60$ , which results in  $\approx 2.7 \times 10^6$  operations per image.

We now report on the cost of image classification described in Section 4. Suppose our library consists of  $k$  images (referred to as library images). To classify the given image  $I_{test}$ , its distance to each library image  $I_{lib}$  has to be calculated according to (3). The total number of operations is

$$30ke^2 + 10ke, \quad (8)$$

on average. The value of  $k$  for our posture library is about 30, yielding  $\approx 3.5 \times 10^6$  operations.

A naive frame-based approach to perform the same feature extraction we presented requires a higher number of computation. The reason is: 1) Convolutions are performed on all pixels in the frame, while in our case they are performed on the “active” portion of the frame that reports events (only 25 percent of the pixels). In addition, 2) the second MAX operation is applied only to the surviving neurons (11 percent of the neurons) (refer to Section 3.2).

From Table 1 one can see that our approach yields improvement by roughly five times versus the frame-based approach in terms of number of operations. Moreover,  $e_{fb}$  counts redundant overlapping line segments and is larger than  $e$ . Even though the rate  $e_{fb}/e$  does not have a significant

TABLE 1  
Cost Estimation:  
Our Approach versus Frame-Based Approach

| Algorithm Stage | Our Approach                    | Frame-based Approach                  |
|-----------------|---------------------------------|---------------------------------------|
| Convolution     | $1816 \cdot n^2 \cdot 25\%$     | $1816 \cdot n^2$                      |
| MAX operation   | $1816 \cdot n^2 \cdot 11\%$     | $1816 \cdot n^2$                      |
| Line segments   | $e$                             | $e_{fb}$                              |
| Total           | $1816 \cdot n^2 \cdot 36\% + e$ | $1816 \cdot n^2 \cdot 200\% + e_{fb}$ |

effect on the line segment extraction computational cost (see (7)), its role for the classifying operation cost is crucial: For instance, if  $e_{fb}/e$  is merely 2, the classifier based on  $e$  line segments rather than  $e_{fb}$  line segments will run 4 times faster (see (8)).

The proposed algorithm achieves great computational savings, resulting from several novel techniques. First, the object of interest is directly obtained from the output of temporal difference image sensor without any image preprocessing. Only the active pixels are permitted to send address events. Second, the contour of the object is decomposed into a limited number of line segments. Compared to the previous work [5], our approach requires extremely least amount of memory to store library features. The average number of line segments is only around 60 per image, while [5] is believed to be memory hungry due to the fact that a large pool of patches of various sizes and at random positions is extracted from a target set of images at the level of the C1 layer for all orientations. Third, size and position invariance is an integral part of our approach and no additional scaling and shift preprocessing is needed.

We note that another fundamental advantage of event-based convolution computation is that convolution output is built continuously, event after event, and the output is available at any moment. We do not have to wait for an entire frame time to “see” or “use” the output since this is continuously updated after each event. So, in practice, input and output AER data flow are virtually simultaneous, except for collecting the few input events that would generate a corresponding output event, but this delay is very small compared to the frame time. A system like the one presented in [51] contains convolutional processing circuits that can accelerate in hardware the algorithms presented here.

## 6 EXPERIMENTAL RESULTS

The algorithm was implemented in C++. We have implemented a working demo system that can operate in real time ( $>30$  frames/s) on a laptop and at 5 frames/s on cellular phone platforms (Nokia Symbian S60 implemented in Java, and iPhone in C++). The codes and demo videos can be accessed from our lab website [52], [53]. In order to evaluate the system performance, we first built libraries by choosing a number of representative images for each human posture. Standard libraries with the address-event temporal-difference format were not found in the literature or online, so we had to specifically make one ourselves for testing our algorithms. We extracted the line segments of every such image and stored them as library of features. Next, we compared each image in the test database to each image in the library. The number of successful matches

TABLE 2  
Experimental Results for Images Taken by a Web Camera







|              | Bend  | Hand1   | Hand2   | Stand   | Squat  | Swing Hand  | Total |
|--------------|---|---|---|---|--|---|-------|
|              |  |  |  |  |  |  |       |
| No. Images   | 317   | 262   | 236   | 283   | 327  | 288   | 1713  |
| Categorized  | 289   | 178   | 205   | 268   | 288  | 210   | 1438  |
| Success Rate | 91%   | 68%   | 87%   | 95%   | 88%  | 73%   | 84%   |

TABLE 3  
Experimental Results for Images Taken by Our Temporal Difference Image Sensor  
(Use 30 Training Images for Each Group)











|              | Bend  | Hand1   | Hand2   | Stand   | Squat  | Swing Hand  | Total |
|--------------|---|---|---|---|--|---|-------|
|              |  |  |  |  |  |  |       |
| No. Images   | 100   | 100   | 100   | 100   | 100  | 100   | 600   |
| Categorized  | 98  | 83  | 93  | 99  | 95   | 74  | 542   |
| Success Rate | 98%   | 83%   | 93%   | 99%   | 95%  | 74%   | 90%   |

TABLE 4  
Experimental Results for Images Taken by Asynchronous Motion Detection Image Sensor  
(Use 30 Training Images for Each Group)

|              | Hand2  | Stand  | Squat  | Swing Hand  | Total |
|--------------|--|--|--|---|-------|
|              |  |  |  |  |       |
| No. Images   | 50   | 50   | 50   | 50  | 200   |
| Categorized  | 48   | 35   | 42   | 37  | 162   |
| Success Rate | 96%  | 70%  | 84%  | 74%   | 81%   |

(successfully categorized postures) that the algorithm yielded are recorded.

Three sets of live images have been captured. During the data acquisition, the person stands in front of the sensor with a distance ranging from 2 to 5 meters. As long as the person's main body is enclosed in the field of view (FOV), our algorithm can effectively localize it and perform categorization. As for the viewpoint, the person shows his lateral profile for the posture "bend," and shows his frontal or rear profile for postures "hand1," "hand2," "squat," and "swing." These postures can have a tilt angle of up to  $\pm 30^\circ$ ; while for "stand" the view angle can be anywhere. The first test set consists of six groups of samples using a web camera (with a scaled resolution of  $64 \times 64$ ), approximately 1,700 images. We used a training set of only 30 images taken from the larger test set. As reported in Table 2, the average success rate is 84 percent. The second set consists of six groups of postures, each group has 200 images (with a resolution of  $64 \times 64$ ), among which 100 are used for test and 30 of the others are taken for training. The success rate is 90 percent (see Table 3). The third set was obtained from another type of image sensor, which is not based on differencing full frames, but on focal plane pixel light intensity temporal derivative computation and normalization with respect to ambient light [41]. When the change of light in a pixel passes




a threshold, an event is triggered. The corresponding pixel address is transmitted and, at the receiver side, the silhouette of a moving object can be reconstructed [54]. Based on a set of recorded data, we derived four groups of postures and each group contains 100 binary images (at a resolution of  $128 \times 128$ ), among which 50 are used for test and 30 of the others are taken for training. The average success rate is 81 percent (see Table 4).

For purposes of comparison, a fourth data set has been extracted from the Yann LeCun and Fu Jie Huang's library small NORB object data set, V1.0 [55], available online [56]. The original images are in gray scale, and were simply thresholded to obtain a binary image compatible with our temporal difference inputs. Sixty test images were used in this case, with a training set of only 15 images taken from the larger test set. The success rate is 87 percent (see Table 5).

The proposed algorithm has been compared to the original HMAX scheme [57] and the model by Serre et al. [5]. Matlab implementations of the two approaches can be found on the Internet. Both of the two approaches use Support Vector Machine (SVM) as classifier. To perform multiclass categorization on the groups of postures, a one-versus-one (OVO) SVM scheme is employed. For  $c$  classes,  $c \times (c - 1)/2$  times OVO SVMs are needed. For [5], each



TABLE 5  
Experimental Results for Images  
from Yann LeCun and Fu Jie Huang’s Library  
Small NORB Object Data Set, V1.0 [55]

|              | Human   | Plane   | Truck   | Total |
|--------------|---|---|---|-------|
|              |  |  |  |       |
| No. Images   | 20  | 20  | 20  | 60    |
| Categorized  | 20  | 15  | 17  | 52    |
| Success Rate | 100%  | 75%   | 85%   | 87%   |

The original images were converted into the binary format first.

image is described with a 150-dimension C2 feature vector, and 15 OVO SVMs are used for this 6-group categorization problem.

Fig. 12 show the simulation results of the three methods, namely, the original HMAX+SVM, Serre’s model+SVM, and our algorithm, using the first data set obtained from our own image sensor. The simulations were performed on a laptop computer equipped with Intel Core I5-540M CPU and 4 GB RAM. Categorization success rate and CPU time are measured with respect to different number of training images. Our algorithm gives the highest success rates and consumes mediate CPU time. This simulation also showed the tradeoff between the size of training image set and system performance (success rate and runtime). Larger size of training set leads to higher success rate but at the expense of scalded execution time. One can note that our algorithm does not require a large training set. Using 10 to 30 training images per group can have a pretty good result and, at the same time, achieve more than 50 percent save in CPU time than Serre’s model.

## 7 HARDWARE IMPLEMENTATION AND DISCUSSION

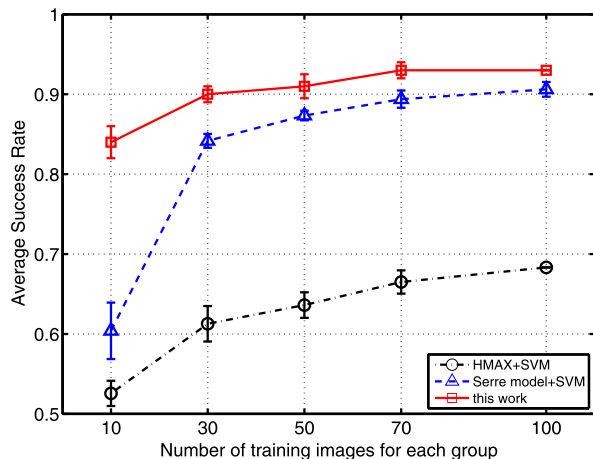
### 7.1 Hardware Implementation Considerations

In this paper, we reported on a C++ coded application that performs posture categorization. The algorithm was

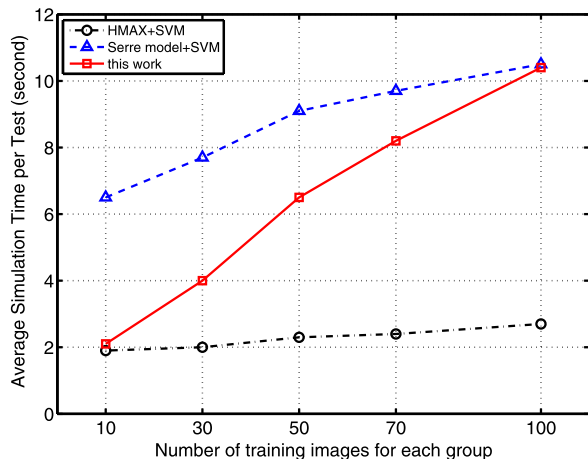
designed with the intention of being implemented into event-based or address-event hardware. Our algorithm performs as fast as convolutional neural networks [55], but has the advantage of not needing large data sets (10,000+ images) for training. Our work is also related to the use of artificial neural networks for human posture detection [58], but our emphasis is on biologically inspired preprocessing with address-event cameras and filter banks. We note that it is possible to train spiking convolutional neural networks using “Spike Time Dependent Plasticity” [59].

At present, there is no general-purpose hardware that can directly operate on address-events as a microprocessor operates on digital data. Several groups have proposed some version of general hardware, and the most notable is IFAT [60], from which most of our algorithmic work is inspired and targeted. Another clever and sophisticated example of general-purpose address-event processing hardware is the CAVIAR project [9], [10], [61], [41], [51], [62]. CAVIAR uses programmable convolution filters [9], [10], [51], [62] and 2D WTA and is meant for direct implementation of trained convolutional neural networks [55]. Kernel weights cannot be learned in hardware in CAVIAR, although a high-level features trajectory/classification scheme [51] was included. Although the demonstration system in CAVIAR [51] illustrated a setup for performing a single convolutional filter for direct template matching, the infrastructure and principles provided can be directly extended to implement convolutional neural networks capable of invariant categorization [62].

Both the CAVIAR project and the ideas in this paper intend to provide a framework for the implementation of artificial vision systems and other bio-inspired processing systems. In general, when implementing hierarchical convolutional neural networks, one adjusts the kernel weights to extract visual features which are very simple in the lower layers (oriented segments at different scales), and aggregate at higher layers to identify more sophisticated forms, shapes, or full objects. If the objects are very simple and of fixed size, then a single convolution filter could perform direct object categorization, as in the illustrative examples



(a)



(b)

Fig. 12. Comparison between our algorithm, original HMAX [57], and Serre’s Model [5] using the first data set obtained from our own image sensor. Better categorization results are obtained when more images are used as training images, but at the expense of scaled execution time. (a) Average success rate versus number of training images per group. (b) CPU time versus number of training images per group.

TABLE 6  
VLSI Implementation Results of the Feature  
Extraction Part of the Proposed System

|                       |                             |
|-----------------------|-----------------------------|
| Technology            | UMC 0.18 $\mu\text{m}$ CMOS |
| gate count            | 170k                        |
| MAX freq.             | 14.8 MHz                    |
| processing capability | 51 fps                      |
| power consumption     | 90 mW @30fps                |

in [9], [10] (e.g., a rotating propeller, a ball, etc.). On the other hand, we work with small fixed set of kernels (24 filters) for the purpose of providing generic categorization of any object. Moreover, we use the convolution output to extract line segments from the compared images, opposite to [9], [10]. The differences arise from the distinction between the goal to find a concrete object in the image and the goal to recognize a general object by testing similarities with the predefined images. Our approach appeals for its simplicity and generality.

The work presented here can be thus be implemented on both IFAT and CAVIAR hardware, with the appropriate extensions and modifications. Our long term goal is to make these platform converge with address-event algorithms. As a preliminary attempt, we have implemented the feature extraction part into VLSI using UMC 0.18 $\mu\text{m}$  CMOS technology [63]. Table 6 summarizes the implementation results. The design can operate at a maximum clock frequency of 14.8 MHz. Each extraction procedure involves on-the-fly convolution, 2-step MAX operation, and in total, needs about  $70 \times 4,096 = 286,720$  clock cycles. Therefore, a maximum frame rate of 51 (64  $\times$  64 resolution) can be attained, which means the design can support most real-time applications.

## 7.2 Discussion

As seen in Section 6, the combination of our custom hardware and feedforward characterization algorithm performs well with both objects and human postures. This system is not completely free of problems: One typical problem is when multiple objects are moving back and forth in the scene, or the background is moving. In this case, categorization fails because the algorithm loses the person-of-interest. When monitoring a single person in a room, like in assisted living applications, this is not an issue [15]. However, for real-world application, an object tracking stage should be added to the system. At present, the event-based clustering algorithm can locate the size and position of one human in the scene and reject a small disturbing moving object in the background, such as a cat [64]. Further challenge emerges when multiple objects run into one another and then separate. A more advanced object tracking algorithm or facility is to be employed. One possible way is to have the target person carry a detectable tag or marker [65], [66]. Another concern is the system robustness against viewpoint variance and field of view full coverage. In our present experimental setup, the person should show his lateral profile for the posture "bend," and show his frontal or rear profile for posture "hand1," "hand2," "squat," and "swing." These postures can have a tilt angle of up to  $\pm 30^\circ$ . For practical usage, multiple camera nodes should be used and, at this point, the proposed system is superior. Due to its high-computation efficiency, it allows making a compact,

small footprint embedded system that can be easily installed. Since no raw video data are involved, patients' privacy is protected when they are monitored.

A further problem is adaptation to lighting conditions. Even indoors, light intensities can vary by 10 times or more, making it difficult for the temporal difference camera to always extract complete contours of objects and human postures. Our camera can globally compensate for lighting intensity by performing adjustments on the internal clock [19]. Another imaging sensor design, not based on differencing full frames but on focal plane pixel light intensity temporal derivative computation and normalization with respect to ambient light, results in highly efficient micro-second resolution ambient light independent event generation [41]. As mentioned in the experimental results, our approach is fully compatible with such a sensor. We have adapted our algorithm to the recorded data from one of this type of cameras.

Notice also that the filter sizes used to detect line segments depend on the image size and the size of the features to be categorized. In this work, we assumed the size of the object was always between 25 and 50 percent of the size of the image. Longer lines need to be divided into segments and processed serially in order to perform categorization. Larger size filters will be investigated in a future version of the system.

## 8 CONCLUSION

This paper reports a size and position invariant human posture categorization algorithm. The image is first acquired using an address event temporal difference image sensor and followed by a bio-inspired hierarchical line segment extraction unit. A simplified line segment Hausdorff distance scheme is employed for similarity measurement, while size and position invariance are achieved by deriving size and position information from event clusters. The proposed algorithm achieves about 90 percent average categorization rate while featuring five times computational saving as compared to a conventional approach.

## ACKNOWLEDGMENTS

This project was funded in part by US National Science Foundation (NSF) award 0622133 and a Nanyang Assistant Professorship (M58040012). The authors also thank Berin Martini for his help on software development.

## REFERENCES

- [1] R. Van Rullen and S. Thorpe, "Rate Coding versus Temporal Order Coding: What the Retinal Ganglion Cells Tell the Visual Cortex," *Neural Computation*, vol. 13, pp. 1255-1283, 2001.
- [2] S. Thorpe, A. Delorme, R. Van Rullen, and W. Paquier, "Reverse Engineering of the Visual System Using Networks of Spiking Neurons," *Proc. IEEE Int'l Symp. Circuits and Systems*, vol. 4, pp. 405-408, 2000.
- [3] C. Koch, *The Quest for Consciousness: A Neurobiological Approach*. Roberts and Company Publishers, 2004.
- [4] T. Serre, "Learning a Dictionary of Shape-Components in Visual Cortex: Comparison with Neurons, Humans and Machines," PhD dissertation, MIT, Apr. 2006.
- [5] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust Object Recognition with Cortex-Like Mechanisms," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411-426, Mar. 2007.

- [6] J.J. Peissig and M.J. Tarr, "Visual Object Recognition: Do We Know More Now than We Did 20 Years Ago?" *Ann. Rev. Psychology*, vol. 58, pp. 75-96, 2007.
- [7] J.K. Tsotsos, A.J. Rodriguez-Sanchez, A.L. Rothenstein, and E. Simine, "Different Binding Strategies for the Different Stages of Visual Recognition," *Proc. Second Int'l Conf. Advances in Brain, Vision and Artificial Intelligence*, pp. 150-160, 2007.
- [8] S. Chikkerur, T. Serre, C. Tan, and T. Poggio, "What and Where: A Bayesian Inference Theory of Attention," *Vision Research*, vol. 50, no. 22, pp. 2233-2247, 2010.
- [9] R. Serrano-Gotarredona, T. Serrano-Gotarredona, A. Acosta-Jimenez, and B. Linares-Barranco, "A Neuromorphic Cortical-Layer Microchip for Spike-Based Event Processing Vision Systems," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 53, no. 12, pp. 2548-2566, Dec. 2006.
- [10] L. Camunas-Mesa, A. Acosta-Jimenez, C. Zamarreno-Ramos, T. Serrano-Gotarredona, and B. Linares-Barranco, "A  $32 \times 32$  Pixel Convolution Processor Chip for Address Event Vision Sensors with 155ns Event Latency and 20Meps Throughput," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 58, no. 4, pp. 777-790, Apr. 2011.
- [11] T. Teixeira, A. Andreou, and E. Culurciello, "An Address-Event Image Sensor Network," *Proc. IEEE Int'l Symp. Circuits and Systems*, pp. 4467-4470, May 2006.
- [12] E. Culurciello and A. Savvides, "Address-Event Image Sensor Network," *Proc. IEEE Int'l Symp. Circuits and Systems*, pp. 955-958, May 2006.
- [13] E. Culurciello and A. Andreou, "CMOS Image Sensors for Sensor Networks," *Analog Integrated Circuits and Signal Processing*, vol. 49, no. 1, pp. 39-51, 2006.
- [14] T. Teixeira, E. Culurciello, E. Park, D. Lymberopoulos, A. Barton-Sweeney, and A. Savvides, "Address-Event Imagers for Sensor Networks: Evaluation and Modeling," *Proc. Information Processing in Sensor Networks*, pp. 458-466, Apr. 2006.
- [15] Z.M. Fu and E. Culurciello, "A 1.2mw CMOS Temporal-Difference Image Sensor for Sensor Networks," *Proc. IEEE Int'l Symp. Circuits and Systems*, May 2008.
- [16] S. Pellegrini and L. Iocchi, "Human Posture Tracking and Classification through Stereo Vision and 3D Model Matching," *J. Image and Video Processing*, vol. 2008, pp. 7:1-7:12, 2008.
- [17] C. Wu, H. Aghajan, and R. Kleihorst, "Real-Time Human Posture Reconstruction in Wireless Smart Camera Networks," *Proc. Int'l Conf. Information Processing in Sensor Networks*, pp. 321-331, 2008.
- [18] R. Jafari, W. Li, R. Bajcsy, S. Glaser, and S. Sastry, "Physical Activity Monitoring for Assisted Living at Home," *Proc. Int'l Workshop Wearable and Implantable Body Sensor Networks*, Mar. 2007.
- [19] Z. Fu and E. Culurciello, "Fall Detection Using an Address-Event Temporal Contrast Vision Sensor," *Proc. IEEE Int'l Symp. Circuits and Systems*, May 2008.
- [20] G. Virone, M. Alwan, S. Dalal, S.W. Kell, B. Turner, J.A. Stankovic, and R. Felde, "Behavioral Patterns of Older Adults in Assisted Living," *IEEE Trans. Information Technology in Biomedicine*, vol. 12, no. 3, pp. 387-398, May 2008.
- [21] H. Su and F.-G. Huang, "Human Gait Recognition Based on Motion Analysis," *Proc. Int'l Conf. Machine Learning and Cybernetics*, vol. 7, pp. 464-468, Aug. 2005.
- [22] B. Boulay, F. Bremond, and M. Thonnat, "Posture Recognition with a 3D Human Model," *Proc. IEEE Int'l Symp. Imaging for Crime Detection and Prevention*, pp. 135-138, June 2005.
- [23] J. Isaacs and S. Foo, "Hand Pose Estimation for American Sign Language Recognition," *Proc. 36th Southeastern Symp. System Theory*, pp. 132-136, 2004.
- [24] K. Takahashi, T. Sakaguchi, and J. Ohya, "Remarks on Real-Time Human Posture Estimation from Silhouette Image Using Neural Network," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics*, pp. 370-375, Oct. 2004.
- [25] E. H-Jaraha, C. Urunuela, and J. Senar, "Detected Motion Classification with a Doublebackground and a Neighborhood-Based Difference," *Pattern Recognition Letters*, vol. 24, pp. 2079-2092, 2003.
- [26] L.H.W. Aloysius, G. Dong, H. Zhiyong, and T. Tan, "Human Posture Recognition in Video Sequence Using Pseudo 2-D Hidden Markov Models," *Proc. Eighth Control, Automation, Robotics, and Vision Conf.*, pp. 712-716, Dec. 2004.
- [27] P. Spagnolo, M. Leo, A. Leone, G. Attolico, and A. Distanto, "Posture Estimation in Visual Surveillance of Archaeological Sites," *Proc. IEEE Conf. Advanced Video and Signal Based Surveillance*, pp. 277-283, July 2003.
- [28] J. Triesch and C. von der Malsburg, "A System for Person-Independent Hand Posture Recognition against Complex Backgrounds," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 12, pp. 1449-1453, Dec. 2001.
- [29] M. Gottardi, N. Massari, and S. Jawed, "A  $100\mu\text{m} \times 64$  Pixels Contrast-Based Asynchronous Binary Vision Sensor for Sensor Networks Applications," *IEEE J. Solid-State Circuits*, vol. 44, no. 5, pp. 1582-1592, May 2009.
- [30] E. Artyomov and O. Yadid-Pecht, "Adaptive Multiple-Resolution CMOS Active Pixel Sensor," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 53, no. 10, pp. 2178-2186, 2006.
- [31] A. Teman, S. Fisher, L. Sudakov, A. Fish, and O. Yadid-Pecht, "Autonomous CMOS Image Sensor for Real Time Target Detection and Tracking," *Proc. IEEE Int'l Symp. Circuits and Systems*, pp. 2138-2141, May 2008.
- [32] A. Fish, L. Sudakov-Boresha, and O. Yadid-Pecht, "Low-Power Tracking Image Sensor Based on Biological Models of Attention," *Int'l J. Information Theory and Applications*, vol. 14, no. 2, pp. 103-114, 2006.
- [33] S. Mizuno, K. Fujita, H. Yamamoto, N. Mukozaka, and H. Toyoda, "A  $256 \times 256$  Compact Cmos Image Sensor with On-Chip Motion Detection Function," *IEEE J. Solid-State Circuits*, vol. 38, no. 6, pp. 1072-1075, June 2003.
- [34] J. Choi, S.-W. Han, S.-J. Kim, S.-I. Chang, and E. Yoon, "A Spatial-Temporal Multiresolution CMOS Image Sensor with Adaptive Frame Rates for Tracking the Moving Objects in Region-of-Interest and Suppressing Motion Blur," *IEEE J. Solid-State Circuits*, vol. 42, no. 12, pp. 2978-2989, Dec. 2007.
- [35] P. Lichtsteiner, C. Posch, and T. Delbruck, "A  $128 \times 1280$  120dB  $15\mu\text{s}$  Latency Asynchronous Temporal Contrast Vision Sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566-576, Feb. 2008.
- [36] D. Kim, Z. Fu, and E. Culurciello, "A 1-mw CMOS Temporal-Difference AER Sensor for Wireless Sensor Networks," *IEEE Trans. Electron Devices*, vol. 56, no. 11, pp. 2586-2593, Nov. 2009.
- [37] E-Lab, E. Culurciello's Laboratory at Yale Univ., <http://www.eng.yale.edu/elab/>, 2011.
- [38] C. Farabet, B. Martini, P. Akselrod, S. Talay, Y. LeCun, and E. Culurciello, "Hardware Accelerated Convolutional Neural Networks for Synthetic Vision Systems," *Proc. IEEE Int'l Symp. Circuits and Systems*, pp. 257-260, 2010.
- [39] C. Farabet, B. Martini, P. Akselrod, B. Corda, S. Talay, Y. LeCun, and E. Culurciello, "Bio-Inspired Vision Processor for Ultra-Fast Object Categorization," *Proc. High Performance Embedded Computing Workshop*, 2010.
- [40] E. Culurciello, R. Etienne-Cummings, and K. Boahen, "Arbitrated Address Event Representation Digital Image Sensor," *Proc. IEEE Int'l Solid-State Circuits Conf. Digest of Technical Papers*, pp. 92-93, May 2001.
- [41] J.A. Lenero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A  $3.6\mu\text{s}$  Asynchronous Frame-Free Event-Driven Dynamic-Vision-Sensor," *IEEE J. Solid-State Circuits*, vol. 46, no. 6, pp. 1443-1455, June 2011.
- [42] L. Chen, G. Lu, and D. Zhang, "Effects of Different Gabor Filter Parameters on Image Retrieval by Texture," *Proc. 10th Int'l Multimedia Modelling Conf.*, vol. 1, pp. 273-278, July 2004.
- [43] D. Huttenlocher, G. Klanderman, and W. Rucklidge, "Comparing Images Using the Hausdorff Distance," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 850-863, Sept. 1993.
- [44] D.-G. Sim, O.-K. Kwon, and R.-H. Park, "Object Matching Algorithm Using Robust Hausdorff Distance Measures," *IEEE Trans. Image Processing*, vol. 8, no. 3, pp. 425-429, Mar. 1999.
- [45] J. Chen, M.K. Leung, and Y. Gao, "Noisy Logo Recognition Using Line Segment Hausdorff Distance," *Pattern Recognition*, vol. 36, no. 4, pp. 943-955, 2003.
- [46] C.-H. T. Yang and S.-H. Lai, "Hybrid Image Matching Combining Hausdorff Distance with Normalized Gradient Matching," *Pattern Recognition*, vol. 40, no. 4, pp. 1173-1181, 2007.
- [47] Y. Gao and M. Leung, "Face Recognition Using Line Edge Map," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 764-779, June 2002.
- [48] M. Barnsley, *Fractals Everywhere*. Academic Press, 1988.

- [49] M. Litzengerger, C. Posch, D. Bauer, A. Belbachir, B.K.P. Schon, and H. Garn, "Embedded Vision System for Real-Time Object Tracking Using an Asynchronous Transient Vision Sensor," *Proc. 12th Signal Processing Education Workshop*, pp. 173-178, Sept. 2006.
- [50] T. Delbruck and P. Lichtsteiner, "Fast Sensory Motor Control Based on Event-Based Hybrid Neuromorphic-Procedural System," *Proc. IEEE Int'l Symp. Circuits and Systems*, pp. 845-848, May 2007.
- [51] R. Serrano-Gotarredona, M. Oster, P. Lichtsteiner, A. Linares-Barranco, R. Paz-Vicente, F. Gomez-Rodriguez, L. Camunas-Mesa, R. Berner, M. Rivas-Perez, T. Delbruck, S.-C. Liu, R. Douglas, P. Hafliger, G. Jimenez-Moreno, A. Ballcells, T. Serrano-Gotarredona, A. Acosta-Jimenez, and B. Linares-Barranco, "CAVIAR: A 45k Neuron, 5M Synapse, 12G Connects/s AER Hardware Sensory Processing Learning Actuating System for High-Speed Visual Object Recognition and Tracking," *IEEE Trans. Neural Networks*, vol. 20, no. 9, pp. 1417-1438, Sept. 2009.
- [52] <http://www.eng.yale.edu/elab/research/svision/svision.html>, 2011.
- [53] <http://www.eng.yale.edu/elab/research/postures/postures.html>, 2011.
- [54] Java AER Open Source Project, <http://jaer.wiki.sourceforge.net/>, 2011.
- [55] Y. LeCun, F. Huang, and L. Bottou, "Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting," *Proc. IEEE CS Conf. Computer Vision and Pattern Recognition*, 2004.
- [56] F.J. Huang and Y. LeCun, "NORB Object Database," <http://www.cs.nyu.edu/~yiclab/data/norb-v1.0-small/>, 2009.
- [57] M. Riesenhuber and T. Poggio, "Hierarchical Models of Object Recognition in Cortex," *Nature Neuroscience*, vol. 2, pp. 1019-1025, 1999.
- [58] K. Takahashi and T. Uemura, "Real-Time Human Body Posture Estimation Using Neural Networks," *JSME Int'l J. Series C Mechanical Systems, Machine Elements, and Manufacturing*, vol. 44, no. 3, pp. 618-625, 2001.
- [59] T. Masquelier and S.J. Thorpe, "Unsupervised Learning of Visual Features through Spike Timing Dependent Plasticity," *PLoS Computational Biology*, vol. 3, no. 2, p. e31, Feb. 2007.
- [60] R. Vogelstein, U. Mallik, E. Culurciello, R. Etienne-Cummings, and G. Cauwenberghs, "Saliency-Driven Image Acuity Modulation on a Reconfigurable Silicon Array of Spiking Neurons," *Proc. Advances in Neural Information Processing Systems*, Dec. 2004.
- [61] M. Oster, Y. Wang, R. Douglas, and S.-C. Liu, "Quantification of a Spike-Based Winner-Take-All VLSI Network," *IEEE Trans. Circuits and Systems I: Regular Papers*, vol. 55, no. 10, pp. 3160-3169, Nov. 2008.
- [62] A. Linares-Barranco, M. Oster, D. Cascado, G. Jimenez, A. Civit, and B. Linares-Barranco, "Inter-Spike-Intervals Analysis of AER Poisson Like Generator Hardware," *Neurocomputing*, vol. 70, no. 70, pp. 2692-2700, May 2007.
- [63] B. Zhao and S. Chen, "Realtime Feature Extraction Using MAX-Like Convolutional Network for Human Posture Recognition," *Proc. IEEE Int'l Symp. Circuits and Systems*, May 2011.
- [64] Z. Fu, T. Delbruck, P. Lichtsteiner, and E. Culurciello, "An Address-Event Fall Detector for Assisted Living Applications," *IEEE Trans. Biomedical Circuits and Systems*, vol. 2, no. 2, pp. 88-96, June 2008.
- [65] T. Teixeira, D. Jung, G. Dublon, and A. Savvides, "PEM-ID: Identifying People by Gait-Matching Using Cameras and Wearable Accelerometers," *Proc. ACM/IEEE Int'l Conf. Distributed Smart Cameras*, pp. 1-8, 2009.
- [66] T. Teixeira and A. Savvides, "Lightweight People Counting and Localizing in Indoor Spaces Using Camera Sensor Nodes," *Proc. ACM/IEEE Int'l Conf. Distributed Smart Cameras*, pp. 36-43, 2007.



**Shoushun Chen** received the BS degree from Peking University, the ME degree from the Chinese Academy of Sciences, and the PhD degree from the Hong Kong University of Science and Technology in 2000, 2003, and 2007, respectively. He held a postdoctoral research fellowship in the Department of Electronic & Computer Engineering, Hong Kong University of Science and Technology for one year after graduation. From February 2008 to

May 2009, he was a postdoctoral research associate within the Department of Electrical Engineering at Yale University. In July 2009, he joined Nanyang Technological University as an assistant professor. He serves as a technical committee member of Sensory Systems, IEEE Circuits and Systems Society (CASS); an associate editor of *Sensors Journal*; program director (Smart Sensors) of VIRTUS, IC Design Centre of Excellence. His research interests include mixed signal integrated circuits design for sensors, feature extracting biomimetic sensors for sensor networks, energy-efficient algorithms for object recognition, smart vision sensors, asynchronous VLSI circuits and systems. He is a member of the IEEE.



**Polina Akselrod** received the BSc degree in computer engineering from the Hebrew University of Jerusalem, Israel, in 2008. Since 2008, she has been working in the e-Lab in the Department of Electrical Engineering at Yale University as a software and hardware engineer/researcher. Her fields of interest include high-speed computations, digital design, and applications to image processing and computer vision.



**Bo Zhao** received the BS and MS degrees in electronic engineering from Beijing Jiaotong University, Beijing, China, in 2007 and 2009, respectively. He is currently working toward the PhD degree in the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. His research interests are in design and VLSI implementations of pattern recognition algorithms. He is a student member of the IEEE.



**Jose Antonio Perez Carrasco** received the degree in telecommunication engineering from the University of Seville, Spain in 2004. After working toward the PhD degree in event-based vision processing at the Instituto de Microelectrónica de Sevilla, Spain, he received the PhD degree in vision processing in March 2011 from the University of Seville. His research interests include image processing and its medical applications, visual perception, and real-time processing. He is the author of several papers both in journals and international conferences. He is a member of the IEEE.



**Bernabe Linares-Barranco** received the BS degree in electronic physics in June 1986 and the MS degree in microelectronics in September 1987, both from the University of Seville, Spain. He received the first PhD degree in high-frequency OTA-C oscillator design in June 1990 from the University of Seville, Spain, and the second PhD degree in analog neural network design in December 1991 from Texas A&M University, College Station. Since September

1991, he has been a tenured scientist at the Sevilla Microelectronics Institute (IMSE), which is one of the institutes of the National Microelectronics Center (CNM) of the Spanish Research Council (CSIC) of Spain. On January 2003, he was promoted to tenured researcher and, in January 2004, to full Professor of Research. Since March 2004, he has also been a part-time professor at the University of Seville. From September 1996 to August 1997, he was on sabbatical at the Department of Electrical and Computer Engineering of The Johns Hopkins University, Baltimore, Maryland, as a postdoctoral fellow. During Spring 2002, he was a visiting associate professor at the Electrical Engineering Department of Texas A&M University. He has been involved with circuit design for telecommunication circuits, VLSI emulators of biological neurons, VLSI neural-based pattern recognition systems, hearing aids, precision circuit design for instrumentation equipment, bio-inspired VLSI vision processing systems, transistor parameters mismatch characterization, Address Event Representation (AER) VLSI, RF circuit design, memristive type neuromorphic systems, AER vision chips, real time AER vision processing chips, and extending AER to nanoscale. He was a corecipient of the 1997 *IEEE Transactions on VLSI Systems* Best Paper Award for the paper "A Real-Time Clustering Microchip Neural Engine," and of the 2000 IEEE CAS Darlington Award for the paper "A General Translinear Principle for Subthreshold MOS Transistors." He organized the 1994 Nips Post-conference Workshop on Neural Hardware Engineering. From July 1997 to July 1999, he was an associate editor of the *IEEE Transactions on Circuits and Systems II*, and from January 1998 to December 2009 he was also an associate editor for *IEEE Transactions on Neural Networks*. He has been an associate editor of *Frontiers in Neuromorphic Engineering* since May 2010. He was a chief guest editor of the 2003 *IEEE Transactions on Neural Networks* special issue on neural hardware implementations. From June 2009 to May 2011, he was a chair of the Sensory Systems Technical Committee of the IEEE CAS Society. He is a coauthor of the book *Adaptive Resonance Theory Microchips* (Kluwer, 1998). He was the coordinator of the EU-funded CAVIAR project. He has been an IEEE fellow since January 2010.



**Eugenio Culurciello** received the Laurea (MS) degree in electronics engineering from the University of Trieste, Italy, in July 1997. His MS thesis work was developed at The Johns Hopkins University with Professor Ernst Niebur. He joined Professor Andreas G. Andreou's laboratory in January 1998 as a graduate student. He received the second MS degree in electrical and computer engineering from The Johns Hopkins University, Baltimore. In September 2004, he received the

PhD degree in electrical engineering from The Johns Hopkins University. In July of 2004 he joined the Department of Electrical and Computer Engineering at Yale University, where he currently is an associate professor and directs Yale's E-lab, a VLSI laboratory. His research aims at extending the performance of CMOS circuits by means of advanced VLSI technologies. He focuses on topologies and circuits that take advantage of the native properties of devices to augment their computational and communication capabilities. His research interests originate from the identification of the physical limitations of current integrated circuits technologies. These limitations suggest efficient algorithms to encode information in ways that are compatible with the physical medium where computation and communication are performed. His research interests include: analog and mixed-mode integrated circuits with applications to biomedical instrumentation, biological sensors and interfaces, implantable sensors, telemetry sensors, biomimetic sensors, bio-inspired vision sensory systems and application in sensor networks, efficient communication systems, event-based communication and processing, silicon on insulator and silicon on sapphire circuit design, models of devices, analog-to-digital conversion, radio circuits, radiation tolerant design, and isolation amplifiers. He is the recipient of the Presidential Early Career Award for Scientists and Engineers (PECASE) and Young Investigator Program from the US Office of Naval Research, is a Distinguished Lecturer of the IEEE (CASS), and is the author of the book *Silicon-on-Sapphire Circuits and Systems, Sensor and Biosensor interfaces* (McGraw Hill, 2009). He is a senior member of the IEEE