A Hardware Accelerated Object Recognition System Using Event-based Image Sensor

DING Ruoxi

School of Electrical & Electronic Engineering

A thesis submitted to the Nanyang Technological University

in partial fulfillment of the requirement for the degree of

Master of Engineering

2015

M.Eng Thesis

version 0.2.1

August 27, 2015

Nanyang Technological University

Smart Sensor Group

Electrical and Electronics Engineering

VIRTUS, S3.2-B2, 50 Nanyang Avenue

639798 Singapore

Abstract

The goal of this research is to explore the design and implementation of a hardware accelerated highly-efficient engine for the categorization of objects, using asynchronous event-based image sensor. The image sensor, namely Dynamic Vision Sensor (DVS), is equipped with temporal difference processing hardware. It 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 proposed system fully utilizes the precise timing information in the output of DVS. The asynchronous nature of this system frees computation and communication from the adamant clock timing in typical systems. Neuromorphic processing is used to extract cortex-like spike-based features through an event-driven MAX-like convolution network.

Real-time light-weighted object recognition system is found more and more important, in particular in a number of emerging applications, such as in Unmanned Aerial Vehicle (UAV), where a recognition system that can detect and avoid obstacle; e-health applications such as human activity categorization; automobile application such as active car collision avoidance system, to name a few. A lot of research work has been focused on this before, mainly using traditional picture-based images sensor running at a high frame rate. Approaches like background subtraction can provide foreground object skeleton, followed by various techniques to represent the features in the image and at last, utilizing statics regression and classification algorithms, raw recognition results can be obtained. However, due to algorithm complexity and the large quantity of frame based image data, these algorithms have to be carried out on super powerful computers. This limits the application to wider usage, not to mention its high power consumption, mass and volume. In addition, conventional frame image sensor contains tons of data re-

dundancy for image processing. Usually the output data form can be represented as 3 layer matrix. The color information and the redundant background lag the later processing speed in a great deal. Actually, the very first of many image processing algorithm is trying to remove those background as much as possible. For example, the Phantom Miro 3 high speed camera from Vision Research [1] with a resolution of 1920×1080 and a frame rate of 2570 will produce 5.3 GB data per second and consume around 12W power. While the second generation DVS with 2048×2048 resolution will only output 10 MB data per second. Different from the conventional frame based cameras, event based silicon retina, or so called DVS, is able to perform temporal difference in hardware within the sensor chip to reduce output data.

In this thesis we proposed a seamless combination of a bio-inspired, event driven object recognition system. The system is tailored for frame-free asynchronous Address Event Representation (AER) vision sensor to receive and process the absolute binary event stream. For each event, in the form of $(address, time)$, is sent to a batch of Difference of Gaussian (DoG) filters and convoluted, in parallel. Modern neuron model, leaky and integration models are used to model dynamic membrane potential and fire status. Each neuron competes with others within its receptive field. The extracted spike patterns are then classified by an online sequential extreme learning machine with lookup table. Using a lookup table, the system can be made virtually fully connected by physically activating only a very small subset of the classification network. Experiment results show that the proposed system has a very fast training speed while still maintaining a competitive level of accuracy.

The main contributions of this thesis can be summarized as three aspects: (1) Fully event driven, biology inspired, hardware friendly classification system; (2) The seamless integration of online sequential learning to an AER classification system; (3) The use of a lookup table to achieve a virtually fully connected system by physically activating a very small subset of the classification network.

Acknowledgments

It is a great opportunity to express my in depth gratitude, to my supervisor, Prof Chen Shoushun, for his always insightful guidance, constant backing and unwavering support in my whole M.Eng program. I would like to extend my appreciation to Dr Yu Hang and all members in our research group, thank you all for your continuous support. Special thanks to Dr Zhao Bo for guiding me into neuromorphic engineering. My research could not complete without any one of you.

Thank my parents and family for your endless and selfless love, it is you who encourage and support me throughout.

Contents

List of Figures

List of Tables

List of Abbreviations

ADLs	Activity of Daily Living
AER	Address Event Representation
AIT	Anterior Inferotemporal (cortex)
ANN	Artificial Neural Network
ASIC	Application Specific Integrated Circuits
BPNN	Back-propagation Neural Network
CBIR	Content Based Image Retrieval
DFT	Discrete Fourier Transform
DBN	Deep Belief Network
DoG	Difference of Gaussian
DVS	Dynamic Vision Sensor
ELM	Extreme Learning Machine
ERP	Event-related Potentials
FNN	Feedforward Neural Network
FOV	Field of View
FPGA	Field Programmable Gate Array
FPN	Fixed Pattern Noise

GEI	Gait Energy Image
GLCM	Gray-level Co-occurrence Matrix
HMAX	Hierarchical Model and X
IT	Inferotemporal
k-NN	k Nearest Neighbor
LHD	Line-segment Hausdorff Distance
LSE	Least Square Estimation
LUT	Lookup Table
ML-ELM	Multi-layer Extreme Learning Machine
MEMS	Microelectromechanical System
MEI	Motion Energy Image
MHI	Motion History Image
MLP	Multi-layer perceptron
MT	Middle Temporal (area)
OCR	Optical Character Recognition
OSELM	Online Sequential Extreme Learning Machine
OVO	One-versus-one
OVA	One-versus-all
PCA	Principle Component Analysis
PIT	Posterior Inferotemporal (cortex)
RBF	Radial Basis Function
SVM	Support Vector Machine
SLFN	Single Hidden Layer Feedforward Neural Network
VTU	View-tuned Units
WSN	Wireless Sensor Network
WTA	Winner-take-all

Chapter 1

Introduction

1.1 Research Motivation

Object recognition has many applications. A wide range of utilizations are related to machine based artificial intelligence, among them there are some new emerging applications like UAV navigation, car collision avoidance, security surveillance etc., as shown in Fig. 1.1.

Unmanned Aerial Vehicle (UAV), or drone is gaining more and more attention at present. Up to now, the most associated situations are military applications like remote sensing and precise strike. Also, civil aviation have developed UAV for aerial survey of crops, emergency rescue, video shooting etc. Widely used as it is, most of present UAVs are still remotely piloted aircraft instead of autonomous. In addition, since the pilot is absent, camera provided limited narrow angle vision, Field of View (FOV), might not guarantee a fast and precise response to sudden unforeseen situations. Hence, how to provide aided vision to aircraft pilot or even implementing autonomous drone relies on the development of both machine vision and control system.

For our daily life, motion object recognition is also important. Situations like car collision avoidance can be handled in some part by radar or sonar, but radar can only detect the location of obstacle and cannot recognize what the obstacle is or even whether it is an obstacle. For the recent hot topic, pilotless

automobile, motion object recognition now seems to be more important. Object recognition can provide helpful information for more accurate judgment, for both human and autonomous vehicle.

Also, for security or nursing surveillance applications, object or human posture detection can be helpful. Dynamic Vision Sensor (DVS), the main topic of this thesis, is then an ideal way to implement such a system. On one hand, the vision of DVS can reduce a great deal of human resources to monitor patient's or elderly's activities day and night; on the other hand, DVS can remove the detailed facial information but only keep the posture and movement information, hence can protect the privacy of them.

Figure 1.1: Applications of Object Recognition. From left to right, this figure shows applications of object recognition in UAV, collision avoidance and nursing surveillance, separately.

In previous research, there are so much work devoted to image processing and object recognition. But most of the them are focusing on traditional frame based cameras. Since the frame based camera outputs huge amount of data, the later processing and algorithm will require much computation resources. Letting alone the huge investment on high speed data transmission and storage equipment.

Dynamic Vision Sensors, on the contrary, can be a very promising solution to the situation. Pioneering work are also found in the literature. Such as an object tracking image sensor architecture seen from [2]. This type of sensor can be switched between two modes: object acquisition and tracking. In its Field of View (FOV), it locates some salient objects and their centroid. A predefined window can then extract features centered as the pre-found centroid. In addition, spatial temporal CMOS sensor with

multi resolution is reported in [3]. This sensor possesses the capability to output both high resolution background and low resolution foreground object at Region of Interest (ROI) at the same time.

For the back end processing algorithm, existing research is focused on implementing algorithms to softwares, to name a few, active contours, principle component analysis (PCA), hidden Markov models (HMM) [4]. Recently, biology evolution, especially in visual cortex mechanism shed light on bio-inspired feature extraction and recognition. Since primates' object recognition is found to be fast and accurate, researchers are suggesting a feed-forward hierarchy of cortex neurons [5].

1.2 Main Contributions

The goal of this research is to find a solution to efficiently process and recognise objects in high speed motion. Through discussion above, our task is clear: to combine the inherent advantage of event based DVS sensor with bio-inspired feature extraction and recognition system. This work mainly contributes in the following three aspects:

- **Event driven, biology inspired, hardware friendly classification system.** To better utilize the power of event based DVS sensor, we propose a fully event based architecture. The whole processing flow is on the fly and real time considered. The inherent nature of DVS sensor can provide event stream. This can largely reduce data but requires the later processing to be complete event driven. Luckily for this system, the algorithm is designed to be able to operate on individual pixel. In addition, the whole system is designed according to most recent visual cortex research that can perform with high efficiency. Last but not least, the system is hardware friendly. Mechanism like LUT utilizes the advantage of hardware for better performance and less resources.

- **The seamless integration of online sequential learning method to an AER classification system.** In this system we design the feature extraction and classification to be seamless and straightforward. The extracted features, along with their addresses, are fetching their corresponding

weights in Initial Weight LUT, and then trained/tested by later classification network. The procedure is direct and without any lag. This performs better when considering the online sequential training method of classifier.

- **The use of a LUT to achieve a virtually fully connected system by physically activating a very small subset of the classification network.** In principle, all the features after extraction needs classifying. Suppose an image with $m \times n$ resolution, for a four maps feature extraction network and N classes task, it will require $4 \times m \times n \times N$ neurons. However, with this virtually fully connected network, every feature can fetch its weight and the size of network can be greatly reduced.

1.3 Thesis Organization

The rest of this thesis is organised as follows:

Chapter 2 is about literature review. Following the procedure of image recognition, this part is separated into three parts: data acquisition, feature extraction and classification. Data acquisition is mostly on various image sensors, this part will compare their difference in mechanism and propose potential usage; feature extraction talks about different feature extraction and representation algorithm; at last some kinds of widely used classification methods are discussed;

Chapter 3 explains the proposed classification system in details, first about DVS sensor from our group, which serves as the data acquisition method in this system; then the second part is about bio-inspired cortex-like feature extraction algorithm; discussion on the online sequential classification performance and comparison with other research are provided at the end of this part.

Chapter 4 is about the prospective hardware implementation architecture. The overall architecture, FPGA hardware software co-processing method will be discussed in detail.

Chapter 5 concludes the whole thesis, illustrates some limitations at present and sets some goals for our future work.

Chapter 2

Review of Literature

In the literature there are so much work devoted to different aspects of motion object recognition. For a better and systematic view, this chapter first deliberates data acquisition, on a great variance of vision based sensors; then feature extraction theories; finally about diverse classification methods.

2.1 Data Acquisition

Object recognition, within computer vision field, is to find and identify objects in image or video sequence. Ironically, human eyes can recognize multiple objects with little effort, regardless of their size or orientation (size, orientation invariance). We human beings can easily recognise different kinds of objects even when objects are partially behind obstacle (occlusion problem). However, this task remains a great challenge to computer visions.

Previously, image/video cameras are widely used in security surveillance and daily activity analysis. Usually with one or a series of cameras observing objects, it still requires human to stand by monitors to recognize. There are already various kinds of computer vision and pattern recognition algorithms, static images or video sequence. In general, there are gray level cameras, colour cameras, thermal infrared sensors, depth cameras and the most recent Dynamic Vision Sensors.

2.1.1 Gray Scale and Color Camera

In the literature, most studies adopted normal camera as data acquisition device [6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17]. In fact, image processing, pattern recognition and computer vision are essentially developed based on binary or grey level images, color images and video sequences. Reviews on video-based human activity recognition are given in [4, 18]. Usually, background subtraction techniques are first used to obtain the silhouette or contour of objects, then various feature extraction and representation approaches are adopted to describe objects in a single frame or object's motion in spatial-temporal volume. Different classifiers are thereafter employed to do the classification.

Yuan and Yang[7] reported a human action recognition system based on a single camera. The authors used chromaticity and gradient based background subtraction to extract human silhouettes and further human contours from video sequences. Human posture was represented by star skeleton and classified by Support Vector Machine (SVM). String matching technique was further used to perform higher-level action recognition. In [12], human postures were presented by a chain-code signature of human contours. While in [8], location, speed information together with Hu Moment Invariants -based shape representations were combined in a hierarchical action decision tree. Moments descriptors were also used in [13] to approximate an ellipse for the human silhouette. Parameters of the ellipse and coefficients of Discrete Fourier Transform (DFT) were used to represent human postures.

Efros et al. [15] performed human action recognition at a distance based on broadcast videos of football games. They adopted an optical flow based motion descriptor and a nearest neighbor classifier. A general method for human activity recognition in video was introduced in [6], which also used optical flow based local motion descriptor but further considered position and velocity information as additional features. High-level activity recognition was achieved by using Hidden Markov Models (HMM). Another optical-flow based representation was proposed in [11], each video clip was described by a bag of models of kinematic features derived from optical flow.

2.1.2 Infrared Sensor

Normal cameras can only capture images or videos in visible spectrum. To obtain good recognition results, the lighting illumination and visibility of the air must be well controlled. For example, when in a low-illumination environment or in a space full of smoke, the image captured by normal cameras will be either very dark or quite unclear as shown in Figure. 2.1. To realize security surveillance or human activity monitoring in such extreme conditions, we have to utilize some special image sensors, such as thermal infrared sensors. An un-cooled infrared sensor was used in [19] to bring robustness to hard visibility conditions. People in an environment filled by smoke cannot be detected by normal camera, however, they can be clearly seen and easily segmented from background using thermal infrared sensors. Han et al. [20] performed human activity recognition in thermal infrared imagery. An efficient spatio-temporal representation, Gait Energy Image (GEI) was proposed. Generally speaking, the introduction of thermal infrared imagery into computer vision has helped to extract human silhouettes from background and allowed various existing algorithms to work well regardless of lighting conditions and visibility issues.

2.1.3 Depth Sensor

Another kind of new emerging camera is depth sensor (also time-of-flight camera). Depth sensor consists of an infrared laser projector combined with a monochrome CMOS sensor, which provides a depth map of the scene through time of flight principle as shown in Figure 2.2. Each pixel in the depth map represents the distance from that point to the camera. As active vision sensor, depth sensor can work under any ambient light conditions. It has higher accuracy compared to stereo vision. Available depth sensors in the market include MESA SwissRanger SR4000 [22] (relatively expensive) and Microsoft Kinect for Xbox 360 (150 USD) [23]. Diraco et al. [24] used a wall-mounted time-of-flight camera (MESA SwissRanger SR3000) for fall detection and geodesic distance map-based posture recognition for elderly homecare applications. The same sensor was adopted by Zhu et al. in [25] to perform human

Figure 2.1: Comparison of smoke scene captured by normal sensor and thermal infrared sensor: smoke scene captured by normal camera on the left; the thermal image on the right is the same scene captured from infrared camera[21]

pose estimation by means of a model-based, Cartesian control theoretic approach. Ganapathi et al. [26] reported a hybrid, GPU-accelerated filtering approach and made a large improvement toward real-time human motion capture. Amoretti et al. [27] introduced time-of-flight cameras into wireless sensor networks and studied data fusion for user activity monitoring. In [28], Shotton et al. proposed a real-time human pose recognition system based on single depth images capture by Kinect depth sensor. they mapped the difficult pose estimation problem into a simpler per-pixel classification problem. By using randomized decision forests trained with a large and highly varied training dataset, 31 body parts can be recognized and joint positions can be thereafter obtained. A very high recognition rate was achieved and the hardware system could run in real time.

2.1.4 Dynamic Vision Sensor

Normal cameras usually have little or no computing capability, which makes the captured image containing huge redundancy. These data put heavy pressure on the transmission bandwidth and the following

Figure 2.2: Depth sensor using in Microsoft XBOX Kinect (left) and its captured images (right) [29]

processing unit. Smart image sensors, that have certain computation capabilities, provide a possible way to alleviate these problems. This kind of sensor [30, 31, 32, 33, 34] usually integrates some image processing on the focal plane and only outputs useful information (such as motion objects or spatial contrast) in the scene, largely reducing the bandwidth requirement.

Lichtsteiner et al. [31] designed a 128×128 asynchronous temporal contrast vision sensor. The logarithmic transformation in each pixel allows for very high dynamic range operation ($120dB$), which makes it appropriate for uncontrolled light conditions. The whole value of pixels change more than the user-defined thresholds to generate the ON/OFF events. The events are rendering in a so-called address event representation (AER). Based on this asynchronous transient vision sensor, real-time mean-shift-like object tracking [35], high way vehicle speed estimation [36] and a fast sensory motor control goal keeper have been reported. In addition, the CAVIAR project [37], an AER hardware sensory-processing-learning-actuating system, has successfully demonstrated the computing power of spike-based neuromorphic processing framework.

Chen et al. [34] proposed a 64×64 temporal difference image sensor with UWB wireless transceiver. The sensor features sequential scan readout strategy, high speed (160FPS) and low power consumption (sensor 0.9mW, UWB 15mW), making it an appealing candidate for wireless sensor node.

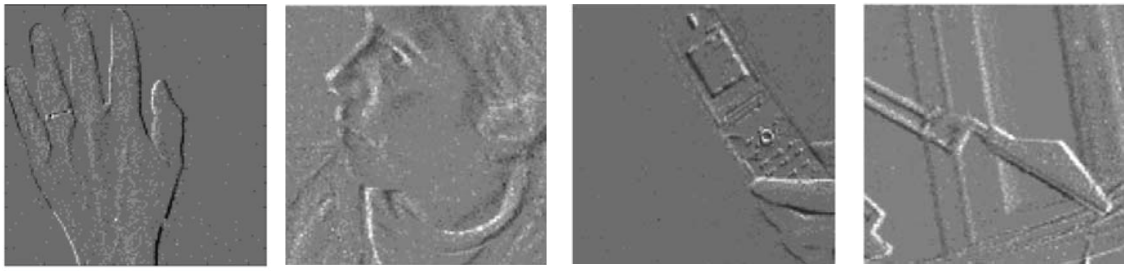

Figure 2.3: Sample images created by histogramming events captured by Dynamic Vision Sensor: (a) hand. (b) head. (c) cellular phone. (d) camera watching lamp in front of window where outside object can be seen. [38]

2.1.5 Summary on Data Acquisition

We have made a brief review on data acquisition devices for human activity recognition. Most studies in the literature are based on images or videos captured by normal gray level or color cameras. Normal cameras usually have little or no computing capability and contains huge redundancy. Circumstance can be even worse in the case of wireless sensor network (WSN). Smart vision sensors, which have certain computation capability, provide only the useful information, greatly saving the bandwidth and making themselves promising in WSN. In addition, since smart vision sensors capture only the motion but not raw images, the privacy of the target person (e.g. in the case of elderly care) can be protected. Moreover, smart vision sensors usually have high frame rate. This property makes it suitable for high-speed object tracking and fast action detection such as people falling down. Smart image sensors still have some limitations. Like normal cameras, they also require a good lighting condition and cannot work in extreme circumstances such as dark or smoky environments. In these cases, thermal infrared or depth sensors can be employed.

2.2 Feature Extraction

Feature extraction is one of most essential steps in object recognition. It extracts distinguishable and salient features from the original raw images. Hence it is both a process to reduce redundant information and a guarantee to the accuracy of later classification. Different features and representation methods have been proposed. This part will try to cover some of the most inspiring thoughts and ideas.

2.2.1 Object Property Descriptors: Color, Texture and Shape

Human vision can recognize and discriminate objects by their color, texture and shape. All these features have also been exploited in computer vision. Compared to color and texture, shape features are mostly adopted in object recognition area. Biederman [39] claims this is due to the reason that the object's class identity is more intrinsically related with its volumetric description (shape) rather than surface characteristics such as color and texture. However, this does not mean color and texture are useless. There are many examples in nature and also artificial environments where color (or texture) correlates with class identity.

Swain and Ballard [40] illustrated the use of color histogram and histogram intersection for efficient indexing to large databases. Given a color space (e.g. RGB), a color histogram makes statistics of the frequency of each color (or each range of colors) occurs. Figure 2.4 illustrates the color histogram (generated by Java program Color Inspector 3D [41]). Color histogram is an efficient representation of color images; it is widely used in content based image retrieval (CBIR). Wang [42] proposed a robust CBIR approach based on local color histogram. The image was first divided into several blocks and color histogram was calculated in each block. Similar technique was adopted in [43], Wang et al computed color histogram on local feature regions which were obtained by using multi-scale Harris-Laplace detector and feature scale theory. While Han et al. [44] presented a new histogram, so-called fuzzy color histogram, and demonstrated its superior retrieval performance than conventional color histogram. Huang et al. [45] introduced spatial correlation of colors into color histogram and defined a new image feature called color correlogram. Color correlogram-based object tracking was illustrated in [46].

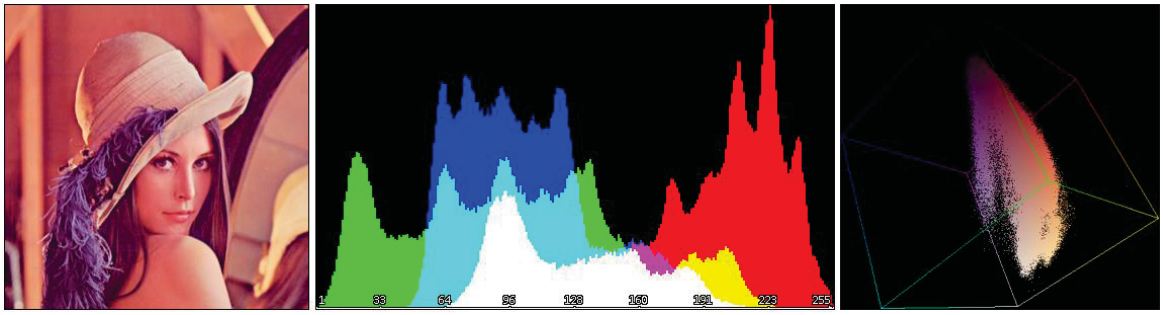

Figure 2.4: An example of color histogram. In the middle is the color histogram of *Lena* standard image, and the third image is color distribution in $3D$ RGB space. [53]

Texture features are also widely used in pattern recognition, especially in the analysis of remote sensed imagery, medical imagery and so on. Typical texture features include gray-level co-occurrence matrix (GLCM), Law's texture energy and Gabor filter banks analysis. GLCM technique was first proposed by Haralick in 1973 [47] and by far it has become the prevalent technique for texture analysis. This technique first constructs a co-occurrence matrix which involves the spatial relationship of pairwise pixels (usually neighbors). Laws' texture energy [48] is another effective texture representation. The image is first convolved with a variety of kernels and then texture energy is calculated on each map. A feature vector can be constructed using these texture energy measures. In addition, Gabor filters are also adopted in texture analysis [49, 50]. Manjunath and Ma [49] applied a batch of Gabor filters (see Figure 2.5 for example) to an image, resulting R $\times$ S filtered images. Then feature is extracted as the mean and standard deviation of these filtered images. Gabor filter bank-based texture features were used in [51, 52] for optical character recognition.

Color and texture features are appealing in the sense of low complexity and high efficiency in circumstances where images contains apparent unique colors or textures. However, in general object recognition, there are many cases that colors or textures are not available or not obviously discriminative for different objects. Therefore, shape-based features and representations are prevalently used in recent

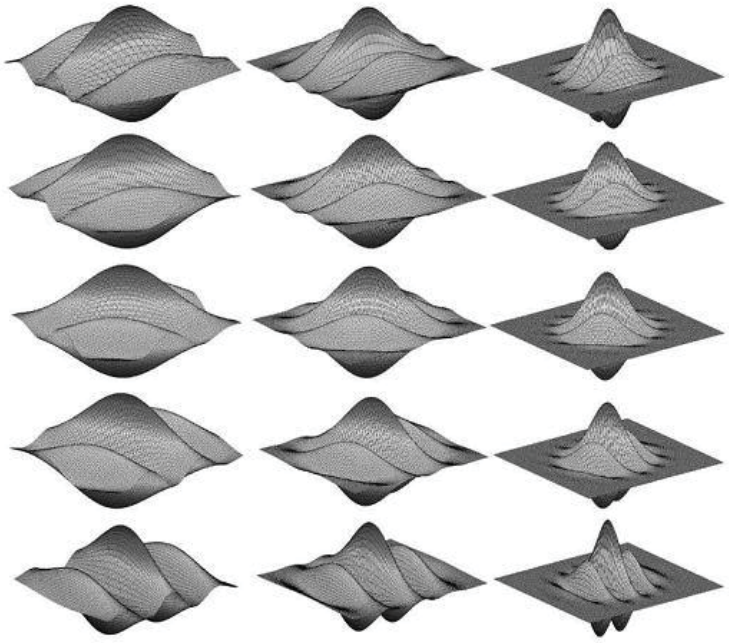

Figure 2.5: Gabor filters for texture analysis. A bank of Gabor filters with three scales (in column) and five orientations (in row) used for texture analysis [54].

research. Shapes such as edges or lines, corners, contours or silhouettes have been widely adopted in object tracking, pattern recognition, human activity analysis and so on. Various shape representations will be discussed in the following subsection.

2.2.2 Object Shape Representation

A number of shape representations have been proposed in the literature for the description of shape features [55]. Generally speaking, these methods can be classified into global shape representation, where shape is described as a whole, and local shape representation, where shape is described in parts or in local regions. In addition, from another point of view, shape representations can also be divided into contour-based methods and region-based methods. Furthermore, if we look at the domain on which the algorithms are based, dividing the approaches into space domain and transform domain can be viewed

as a third classification method of shape representations.

Unlike the conventional classifications, we divide shape description methods according to their processing approaches. Approaches that are covered here include Moments (geometric moment invariants and Zernike Moments), shape transform domain methods (e.g. Fourier Descriptor, Angular Radial Transform, $\Re$ transform), spatial correlation representations (chain code, star skeleton and shape context) and statistical methods (projection histogram, zoning, PCA).

2.2.2.1 Fourier Descriptors

Fourier descriptor is an old technique which lasts nearly 40 years. Although old, people regard it as an effective descriptor of shape. Generally speaking. It is obtained by Fourier transform of the shape contour points and taking the normalized Fourier transform coefficients as the representation of the shape [56, 57]. The points on the shape boundary (contour) are first organized into a set $s(t)$ in clockwise (or counter-clockwise) manner. Each point in the set is described as a complex form.

$$s(t) = [x(t) - x_c] + i[y(t) - y_c] \; t = 0, 1, \cdots, L \tag{2.1}$$

where (x_c, y_c) is the centroid of the shape, and L is the total number of points in the contour. The discrete Fourier transform (DFT) of $s(t)$ is:

$$u_n = \frac{1}{N} \sum_{t=0}^{N-1} s(t) \exp(\frac{-j2\pi nt}{N}), \; n = 0, 1, \cdots, N-1 \tag{2.2}$$

The coefficients u_n are called Fourier descriptors (FD). The first one (DC component) is only related to shape position. A comparative study on shape retrieval using Fourier descriptors was given in [58]. Zhang et al. analyzed the describing ability of Fourier descriptors on different shape signatures. Hu et al. [59] used FDs on body contour for human posture recognition.

2.2.2.2 Angular Radial Transform

Angular radial transform (ART) is a region based shape descriptor adopted in MPEG-7. Similar to Zernike moments, ART is also defined on a unit disk in polar coordinates. It can be described as follows [60]:

$$F_{nm} = \int_0^{2\pi} \int_0^1 V_{nm}(\rho,\theta) f(\rho,\theta) d\rho d\theta \tag{2.3}$$

where $f(\rho,\theta)$ represents an image in polar coordinates and $V_{nm}(\rho,\theta)$ is ART basis function. The ART descriptors are defined as normalized magnitudes of a set of ART coefficients. Shapes described by ART descriptors can be easily compared using L1 norm [60]. Richard et al. [60, 61] generalized the use of ART, made it robust to rotations and deformations, and applied it to color images and 3D shapes.

2.2.2.3 Spatial Correlation Representations

Unlike the above-mentioned transform-based methods, representations covered in this part are all based on spatial correlation of the shape. More specifically, shape contour can be described by using chain code, star skeleton, shape context, etc.

Chain code is a contour-based shape descriptor which can be found in classic image processing textbooks. It describes the movement along the shape contour pixels. The direction of each movement is encoded as an integer number in the range of [0,3] (for 4-connectivity) or [0,7] (for 8-connectivity) [55], see Figure 2.6. Basic chain code description is translation invariant but it is very sensitive to noise. Singh et al. [12] derived chain code-based directional vectors from human contours and adopted vector space analysis for human activities clustering and recognition.

Star skeleton is another contour-based shape representation. It was first proposed by Fujiyoshi et al. [62] in 1998. Several extreme points were first found on the human contour and then connected to the centroid, giving a "star" skeleton of the human shape. Usually, the local maxima of the human contour reside in head, two arms and two legs. Yuan et al. [7] used star skeleton as feature vector to

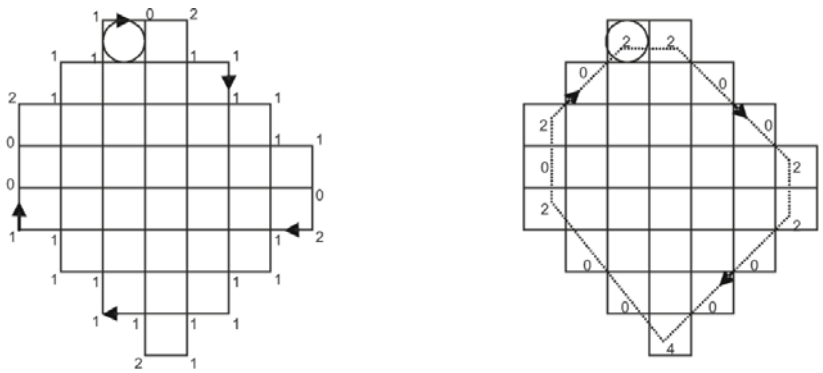

Figure 2.6: Basic chain code directions. The left figure shows chain code with 4 orientations and block border as boundary; the right figure shows chain code with 8 directions and block centroid as boundary.

represent human postures, then performed posture recognition using SVM and further action recognition by means of string matching. Chuang et al. [63] adopted constrained Delaunay triangulation to divide human contour into triangular meshes, and then a graph searching scheme was employed to extract the skeleton. Yu et al. [64] proposed variable star skeleton representation (VSS). Multi stars on the medial axis of human contour were studied and they showed superior performance than single star.

Shape context is also a powerful shape contour descriptor by Belongie et al. in [65]. As local shape descriptor, each point on the contour is associated with a shape context descriptor, which describes the coarse distribution of other contour points with respect to this point. For a point on the shape contour, its shape context is a log-polar histogram of other points' coordinates measured by using this point as origin. Shape context is invariant to translation and insensitive to small perturbations of parts of the shape. Guo et al. [66] generalized shape context and applied it to human silhouettes; they proposed region-based shape context technique and used it for categorizations of six static postures.

2.2.2.4 Statistical Methods

Another type of shape descriptor is based on statistical methods, such as projection histogram, zoning, PCA, etc.

Projection histogram was first introduced in 1956 by Galuberman [67], who used it for a hardware optical character recognition (OCR) system. This technique is now widely used in character segmentation and text lines detection. There are also some studies in which projection histogram is employed for recognizing human postures [68, 16, 13].The horizontal and vertical projection histogram of a shape silhouette is obtained by counting the number of pixels in each row and each column, respectively. Since projection histograms are location sensitive, a normalization step is usually required for the silhouette (e.g. re-scaling the silhouette to a fixed length [16], or using DFT [13]). Guo et al. [68] and Nasution et al. [16] directly used projection histograms of foreground silhouette as feature vector for human posture recognition. Projection histograms do not provide rich description, instead, important information about the shape seems to be lost. Therefore, using projection histograms alone may not be a good representation. Foroughi et al. [13] combined the approximated ellipse of human body with projection histograms for posture recognition, and achieved promising results.

Zoning is another feature extraction approach widely used in optical character recognition [69, 70, 71, 72, 73]. The segmented character image is first divided into several equal zones, and statistical information is obtained for each zone. Zoning description can preserve more detail information than projection histograms. Vamvakas et al. [70] divided the binary character image into 25 equal zones (grids), and for each zone they calculated the density of the character pixels. Several character profiles were also obtained by using projection histogram and distance to boundary. Blumenstein et al. [71] marked direction information onto each character pixel through searching in the thinned character image, and then made statistics of directional lines for each of the nine equal-sized zones. Bo et al. [73] partitioned the character image into 16 partially overlapped zones of equal size. For each zone, they evaluated the density of active pixels and the extent to which the sub-matrix shape matches six main directions. An analog VLSI chip was designed to do the feature extraction.

Principle component analysis (PCA) is an orthogonal linear transformation. It transforms the high-dimensional correlated image to low-dimensional uncorrelated image. The first coordinate (first principle component) contains the greatest variance, and so on. PCA is commonly used to reduce dimension in feature extraction [17, 74]. Turk and Pentland [75, 76] applied PCA on the training set of face images and used the first M principle components (called Eigenfaces) to define the face space. An image was projected onto this face space and a set of weights were obtained and used as feature vector for face matching. Figure 2.7 shows an example of the Eigenfaces.

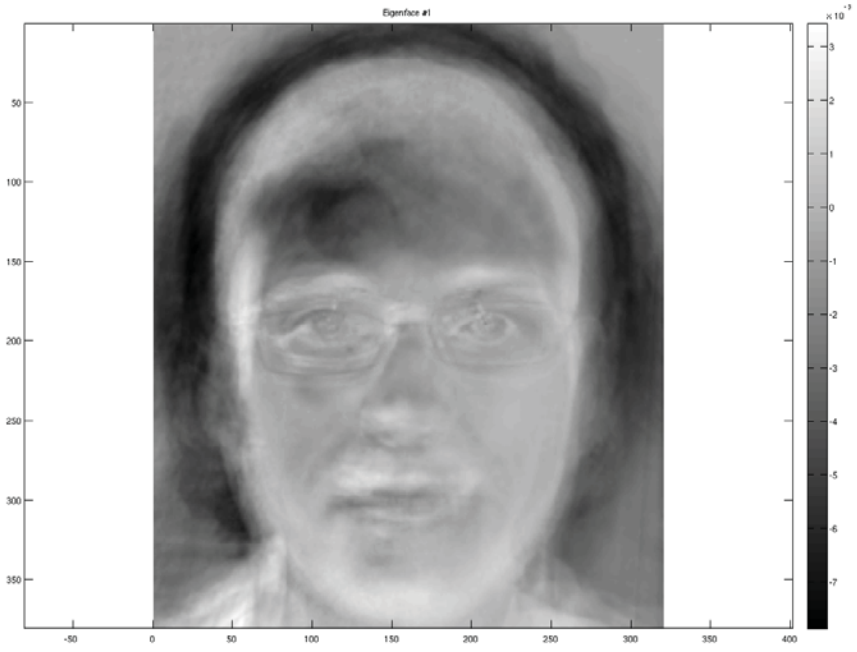

Figure 2.7: Examples of Eigenfaces [75]. Here shows one Eigenface built from 400 face images.

2.2.3 Motion Based Representations

Several shape representations have been discussed in the previous section. In image-based object recognition, shape features and representations are mostly used. While for video-based human action/activity

analysis, besides shape feature, we can also employ another kind of feature: motion feature. This section focuses on the motion-based representations, such as motion history image (MHI), motion energy image (MEI), optical flow-based motion descriptors, etc.

Motion history image (MHI) and motion energy image (MEI) were proposed by Bobick and Davis [77] in 2001. In a given video sequence, the foreground human silhouette in each frame is accumulated and displayed in one binary motion image, called motion-energy image (MEI). MEI describes *where* there is motion in the spatial pattern. To further describe *how* the motion is moving, motion-history image (MHI) is proposed. Newer frames tend to be brighter and older frames tend to be darker. MEI and MHI have sufficient discriminating ability for several simple actions such as "sitting", "bending" and "crouching" [77]. However, it seems that MEI and MHI would lose discriminative power for complex activities due to the overwriting of the motion history.

Optical flow constructs the basis of another type of motion descriptors. It shows the distribution of apparent velocities of movement in an image [78, 79] as shown in Figure 2.8. Efros et al. [15] proposed a local motion descriptor based on half-wave rectified optical flow, and used it for far-field action recognition. Yang et al. [80] adopted the same motion descriptor for patch based human action recognition from a single clip per action. Robertson and Reid [6] extended Efros' motion descriptor by introducing temporal context; they concatenated the motion-descriptor data from five consecutive frames which resulted in better matching results. Ali et al. [11] extracted kinematic features based on optical flow and further built up a bag of kinematic modes for each video sequence. The coordinates of the video in the kinematic-mode-based feature space were used for classification using the nearest neighbor algorithm.

2.2.4 Visual Cortex Recognition Models

This part focuses on the biologically plausible feature models proposed by computational neuroscientists to account for visual processing in the ventral stream of visual cortex. It starts from the illustration of architecture of visual cortex, then move on to the simple (S) and complex (C) cells proposed by

Figure 2.8: Optical flow descriptors using Horn and Schunck method [78]. Given image sequence from the first to the second, optical flow is then calculated showing in the third image

Hubel and Wiesel. After that, it will deliver a brief review of the hierarchical feedforward models of increasingly sophisticated representations (with the emphasis on the HMAX model of Riesenhuber and Poggio[81], and a modified version proposed by Serre et al. [82]).

2.2.4.1 Architecture

Anatomically, primate cerebral cortex can be divided into four regions, namely occipital lobe, parietal lobe, frontal lobe and temporal lobe [83]. The part that related to visual information processing, so-called visual cortex, is mostly located in the occipital lobe, in the back of the brain [84]. Visual cortex consists of striate cortex ($V1$) and extra striate cortical (such as $V2, V3, V4 and V5$). It is well known that two visual processing streams exist in the cortex: dorsal stream and ventral stream (see Figure 2.9). The dorsal stream (also known as the "where" pathway), which starts from $V1$, goes through $V2$, $V3$ and Middle temporal area (MT, also called $V5$) to inferior parietal cortex (PG), is considered as related to motion[85]. While the ventral visual stream, which begins from $V1$, goes through extra striate cortical areas $V2$ and $V4$ to the inferotemporal cortex (IT, further divided into TEO and TE), is treated as associated with object recognition, and thus it is also called "what" pathway [85]. Along the ventral visual pathway, neuron cells show tuning properties to increasingly complex stimuli. For instance, a

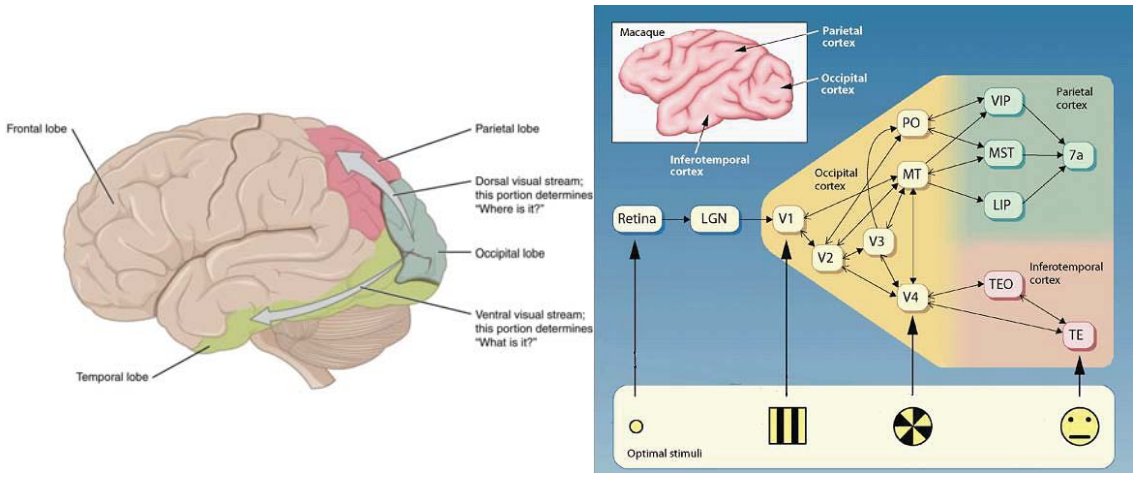

Figure 2.9: Ventral stream and dorsal stream of information processing in primate cortex. Ventral stream, also called "what" pathway, mediates the sophisticated task of object recognition; while dorsal stream, all called "where" pathway, is related to motion, object localization and control of eye and arms. Along the ventral visual pathway, neuron cells show tuning properties to increasingly complex stimuli. e.g. A V1 cell responses best to an edge-bar of particular orientation [86, 87], V4 cell responses to patterns composed by different orientations, while TE cells of the inferotemporal cortex show strong responses to complex visual stimuli, such as faces [88, 89].

$V1$ cell responses best to an edge-bar of particular orientation [86, 87], $V4$ cell responses to patterns composed by different orientations, while TE cells of the inferotemporal cortex show strong responses to complex visual stimuli, such as faces [88, 89] (see Figure 2.9).

2.2.4.2 Simple and Complex Cell

The studies on hierarchical processing in visual cortex began with the ground breaking work of Hubel and Wiesel. Through the investigation on primary visual cortex of cat [86] and macaque [87], they proposed a hierarchical conceptual model of simple to complex cells. Simple cells have small receptive

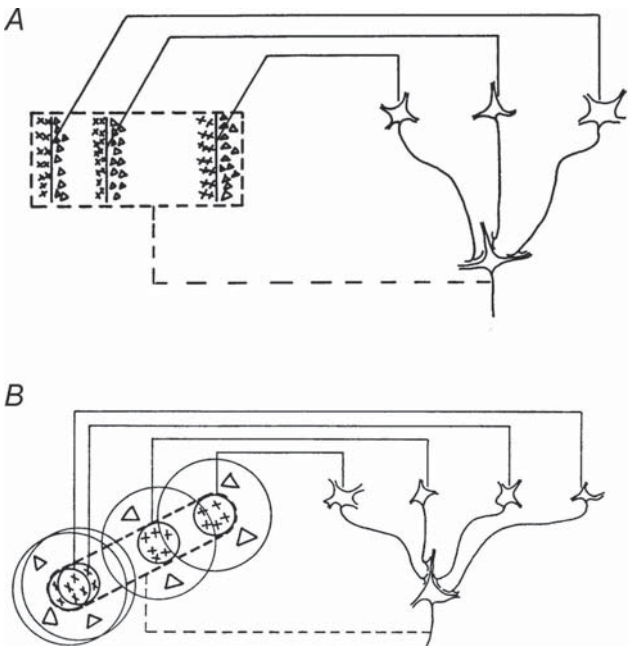

Figure 2.10: The Hubel and Wiesel [86] hierarchical model of building complex cells from simple cells . Fig (A) simple cells with small receptive field and tuning property to bar-like stimuli with orientation and position selective; Fig (B) complex cells can be built by integrating convergent input from simple cells.

field and show tuning property, i.e. respond best to bar-like stimuli at a particular orientation and position; while complex cells have larger receptive field and respond best also to bars at a particular orientation. However, they show strong response to the bar anywhere within its receptive field (i.e. position invariance). Hubel and Wiesel [86] suggested a hierarchical model of the visual cortex, in which complex cells can be built by integrating convergent inputs from simple cells. As illustrated in Figure 2.10, position invariance property in complex cells can be obtained by pooling over simple cells at different positions but with same orientation.

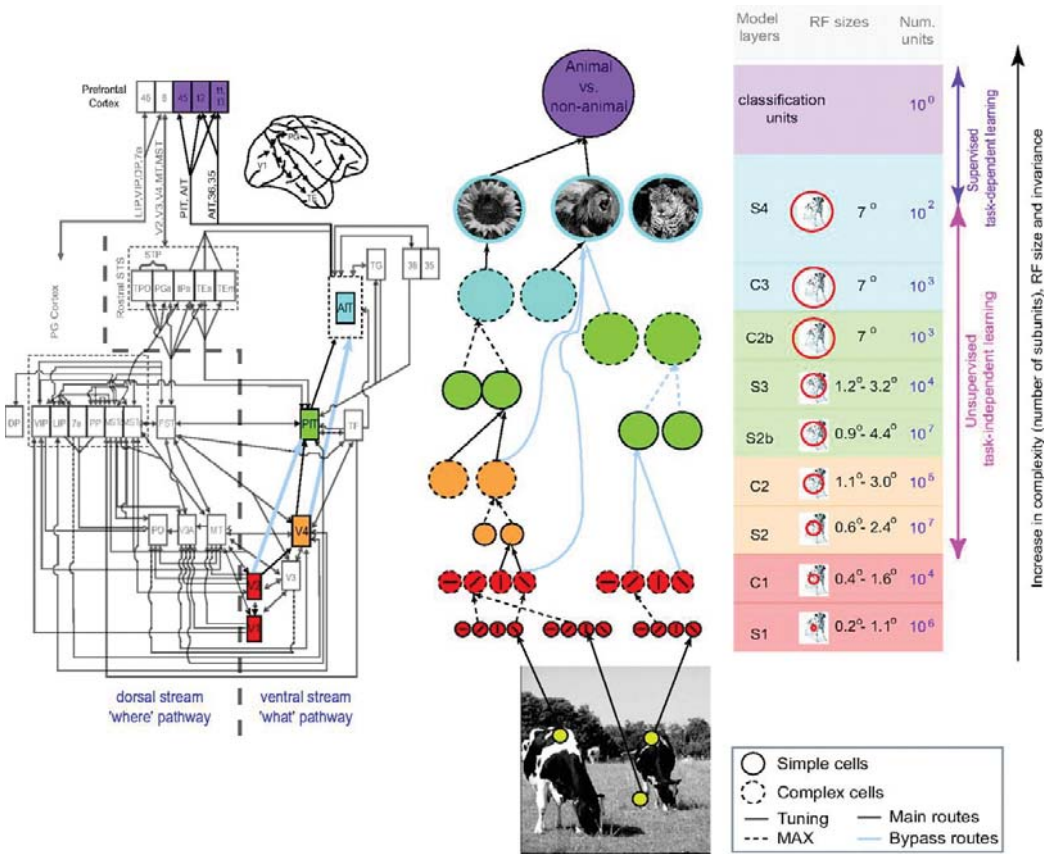

Figure 2.11: The extended HMAX model proposed by Serre et al. [90] The feed forward architecture is $S1-C1-S2-C2$. $S1$ and $C1$ resembles simple and complex cells in visual cortex ($V1$), while $S2$ and $C2$ roughly related to $V2$ and $V4$, respectively. This figure shows extended HMAX model with more layers and can handle more difficult recognition task.

2.2.4.3 Hierarchical Feedforward Model

Hubel and Wiesel [87] conceptually modeled the primary visual cortex ($V1$) as a feedforward hierarchy of simple cells and complex cells. This idea is commonly accepted, however for the visual processing in the whole ventral stream ($V1-V2-V4-IT$), there have been controversy between object-centered models and view-based models, and also between feedforward processing and feedback connections [91]. Object-centered models, such as "Recognition-by-Component" (RBC) theory of Biederman [39],

directly extracts a view-invariant structural description of the object for matching with stored object descriptions in the library. On the contrary, view-based models represent objects as collections of specific view features (view-tuned units) [91]. The view-base models can be further divided into two major groups, one that employing purely feedforward processing and are that utilizing feedback connections for the feature processing (excluding learning) in the recognition process.

Neurophysiological data from monkeys point to view-based theory. Cells in inferotemporal cortex (IT) of macaque show tight tuning property to views of complex objects [88, 92]. Logothetis et al. [92] found that the cell responded selectively to one view of a paperclip and showed strong invariance to scaling and translation but little invariance to rotation in depth. This supports the view-based models.

In addition, the studies on processing speed in the visual cortex have constrained the "immediate" object recognition to be mainly feedforward. Thorpe et al. [93] performed rapid categorization experiments on human subjects. The subjects were shown with previously unseen images flashed on for just 20ms and requested to decide whether an animal existed or not in the image. Event-related potentials (ERP) analysis revealed that humans could perform rapid categorization tasks in natural image approximately 150ms after stimulus onset. Thus the visual processing from V1 to IT (in human cortex) must be within 150ms. The corresponding time for monkeys is about 100ms, which is shorter probably due to their smaller brain size. Figure 2.10 shows typical neuron latencies at different cortical stages of monkeys involved in rapid categorization. The short processing time along ventral visual stream (V1-V2-V4-PIT-AIT. It should be noted that PIT and AIT are subareas of IT, roughly corresponding to another type of division: TEO and TE [89]) strongly suggest that the flow of information is mostly feedforward [5].

Based on simple and complex cells model, Fukushima [94] developed a computer vision system, called "Neocognitron", to perform position invariant recognition in unsupervised manner. Simple and complex layers were alternatively constructed. C-cells in one layer are excited by pooling over local neighborhood of afferent S-cells; while S-cells respond to conjunctions of C-cells in previous layer. As times went on, in late 1980s, by introducing back-propagation multi-layer perceptron into a series

of simple-complex layers, another feedforward system called convolutional network was proposed by LeCun et al. [95]. Such a system showed good performance in handwritten digit recognition [95, 96] and more recently in the domain of generic object recognition and face detection [97]. However, systems like Neocognitron and Convolutional network, although biologically inspired to some extent, lack a direct correspondence with cortical stages (not neurophysiologically plausible) and thus cannot be used to make predictions for physiologists [5].

As for neurophysiologically plausible models, HMAX proposed by Riesenhuber and Poggio [98] was probably one of the most successful feedforward theories. HMAX extends the Hubel and Wiesel classical models of complex cells built from simple cells. It summarizes the basic facts about the ventral visual stream ($V1-V2-V4-IT$). The neurophysiological plausibility of HMAX lies in that it accounts for the experimental data of Logothetis et al. [92] on the tuning properties of macauqe's IT neurons. As mentioned earlier, Logothetis et al. [92] found cells in inferotemporal cortex (IT) showing significant invariance to scaling and translation but little invariance to rotation in depth. These cells, which responded selectively to one view of complex objects (e.g. a paperclip), were modeled as view-tuned units (VTU) at the top of HMAX. HMAX consists of a hierarchy of "S" layers and "C" layers ("S" and "C" follow the notation of Fukushima [94]). The "S" layer cells increase feature complexity by linear weighted summation of inputs; while "C" layer cells increase invariance through the nonlinear MAX operation, which selected the maximum input as the response of the cell. The new pooling mechanism, nonlinear MAX operation, plays a key role in obtaining the VTU's invariance properties [98].

Based on HMAX, Riesenhuber and Poggio also suggested a conceptual feedforward view-based model [91] for the whole process of object recognition. The view-tuned units (VTU) on top of HMAX only have tolerance to scaling and translation but little invariance to rotation in depth. These view-invariant units, together with view-tuned units, can then servers as input to task-related units which performs recognition tasks such as identification and categorization [91].

HMAX model was further extended by Serre et al. [99, 90]. The whole feedforward architecture was remained the same (S1-C1-S2-C2). S1 and C1 layers correspond to simple and complex cells in primary

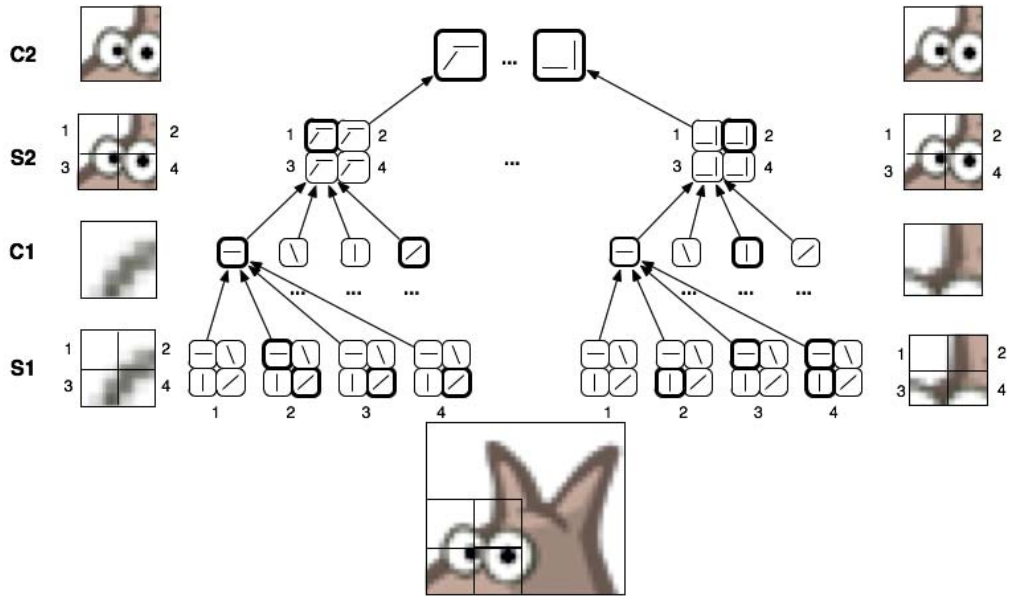

Figure 2.12: Simplified HMAX model proposed by Riesenhuber and Poggio [98]. This figure shows the idea of *S* and *C* layers without too much complex information. The local image pathes are matching with more features in *S* layers and *C* layers select the most salient (highest response) from all previous *S* layers. *S* layers increase the data complexity while *C* layers reduce its complexity. This whole hierarchical architecture can extract features in local area.

visual cortex (V1), while S2 and C2 are roughly related to V2 and V4, respectively. The first two layers of Serre's model are mostly consistent with the old model HMAX (differences exist in the adoption of Gabor filters rather than difference of Gaussians). First, a bank of Gabor filters with different sizes and orientations are used to increase the selectivity, and then MAX operations are taken to achieve slightly scale and position invariance. The last two layers (S2 and C2) are where Serre et al. [99] has made significant modifications; learning is introduced at stage S2 [99]. They first randomly extract a number of patches from C1 maps of training images and use these patches as RBF centers, then for each image, Gaussian radial basis function is applied to the distance between C1-layer map and patches, followed by a MAX operation to generate the shift- and scale- invariant C2 features. Figure 2.11 illustrates the

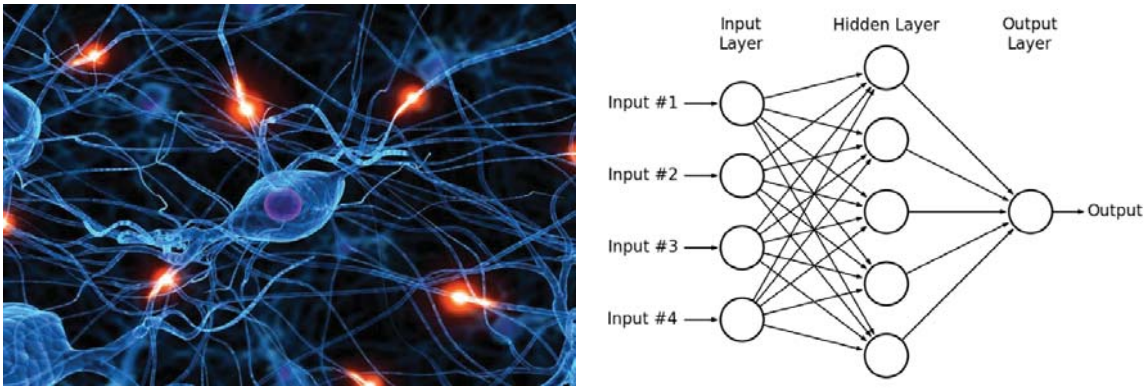

Figure 2.13: Artificial neural network (left) and its math model (right), single hidden layer feedforward neural network (SLFN). A common SLFN contains one input layer, in which each neuron is a feature vector; it also contains one or more hidden layers, in which each neuron is the weighted sum of all previous neurons; and it has an output layer, in which each neuron is a different class. More hidden layer commonly leads to more complex network model and better classification ability but at the same time requires more *training* resources and time.

approach of building C2 features in Serre model. For the detailed description about the algorithms and parameters, one could refer to [99] and [90]. Serre demonstrated the neurophysiological plausibility of this model in [5]. In addition, promising results comparable to state-of-the-art computer vision systems have been obtained in object recognition tasks on natural images [90].

2.3 Classification

After a review of various kinds of feature extraction approaches (including biological-like and non-biological methods), we move on to the classification part. In this section, several popular classifiers, such as k-nearest neighbor (k-NN), artificial neural network (ANN), support vector machine (SVM), and extreme learning machine (ELM), will be discussed.

2.3.1 k-Nearest Neighbor (k-NN)

The k-nearest neighbor algorithm (k-NN) is a simple classification scheme that classifies an unknown example based on the consensus of its neighboring training samples [100]. The feature vectors of all the training samples are stored and labeled during *Training*; while in *Classification* phase, a testing example is classified by assigning the label which most frequently appears in the k-s nearest training samples (simply summarized as "majority voting"). k-NN is an example-based classification approach which requires search among the training samples. The user-defined constant "k" is usually chosen as an odd number for binary (two classes) classification to avoid tie votes. When k=1, it becomes the nearest neighbor algorithm, in which the unknown example is classified by directly assigning to it the label of its nearest neighbor.

2.3.2 Artificial Neural Network (ANN)

An artificial neural network (ANN) is a computational model of nervous system. It consists of an interconnected network of neurons, mimicking the network inside the human brain. Based on the topology of the network, ANN can be divided into feed-forward neural network (FNN) and recurrent neural network (RNN) which has feed-back connections [101]. The feed-forward neural network is commonly used in data classification and object recognition. Typical FNNs include Multilayer perceptron (MLP) and Radial Basis Function (RBF) neural network. MLP network consists of several layers of artificial neurons with fully feed-forward connections. It usually has an input layer, where the number of nodes equal to the dimension of feature vectors, one or more hidden layers and an output layer, in which each output node corresponds to a class. The output is determined by applying an activation function (such as sigmoid) on the weighted summation of previous-layer neurons. The weights of the MLP network are determined by the training process, in which back-propagation algorithm is usually employed (such a network is also called BPNN, back-propagation neural network). Another kind of feed-forward neural network, the RBF network, is composed of an input layer (which receives the feature vector), a hidden

layer (in which a kernel such as Gaussian is applied on the distance between the input and each RBF center) and an linear output layer, in which each node is weighted summation of hidden layer neurons and corresponds to one class. The number of hidden neurons in RBF network is the same as the number of centers, which are usually determined by clustering techniques such as k-means. The training of RBF network only involves the determination of the output weights which can be done by linear least square estimation (LSE) [102]. However, RBF typically requires more training data and the selection of centers is also a major issue.

2.3.3 Support Vector Machine (SVM)

SVM is another widely used technique in data classification. It first maps the data into a different (usually higher dimensional) space and then finds a linear separating hyper plane that best distinguishes two classes [103]. The mapping function is called "kernel". Common kernels include linear, polynomial, RBF (radial basis function) and sigmoid functions. SVM is a binary classifier which only separates two classes; to deal with multi-class classification problem, multiple SVMs have to be employed. There are two common approaches to break the multiclass problem into multiple binary classification problems: One-versus-all (OVA) and one-versus-one (OVO). For a N categories classification problem, One-versus-all requires N SVMs, each one distinguishes the data in a single class from the data in all remaining classes. The classification of new instances by winner-take-all, in which the highest output decides the decision. While one-versus-one (OVO) involves $N(N-1)/2$ SVMs, each one corresponds to a two-class pair, and the classification is max-wins voting, class with most votes is selected as the label. Usually, OVO SVMs have better classification performance than OVA SVMs.

2.3.4 Extreme Learning Machine (ELM)

The Extreme Learning Machine (ELM) is an emerging, efficient learning machine. A Single Hidden Layer Feed-forward Neural Network (SLFN) with nonlinear activation functions and N hidden layer

neurons can classify N distinct features. Moreover, the Initial Weights (IW) and hidden layer biases do not require tuning and can be generated randomly [104]. ELM has been proven to have a fast training speed and competitive classification performance. In [105], a Multi-layer Extreme Learning Machine (ML-ELM) (network structure: 784-700-700-15000-10) achieved $99.03(\pm 0.04)\%$ testing accuracy for a standard MNIST dataset with a training time of $444.655s$, whereas a Deep Belief Network (DBN) took $20,580s$ but achieved lower testing accuracy. For recognition application, a frame-based incremental learning is reported in [106], which proposes *snippets* as image feature representations and uses ELM for incremental learning.

Besides the these kinds of classifiers mentioned above, there are still many other classification tools, such as Naive Bayes, decision tree, etc. The posterior probability of each class can be described by the multiplication of prior probability and class conditional probability densities divided by evidence. Then the classification decision is made according to the maximum posterior probability. While decision tree is a hierarchical classifier which classifies an instance in a top-down fashion from the branches to the leaf nodes according to some category-dependent variables [107]. In the training phase, histograms are built for each leaf node; and in the testing phase, classification is done by first tracing to a leaf node and then checking its histogram, in which the class with largest happening probability is assigned as the label of the unknown instance. The hierarchical essence ensures decision tree an efficient testing performance, but the training of the large number of parameters is computationally expensive.

2.4 Summary on Literature Review

This chapter made an in-depth review on various human posture categorization systems. It divides the system into four parts: data acquisition, preprocessing, feature extraction and classification.

For the data acquisition, various kinds of cameras were covered. Most studies in the literature are based on images or videos captured by normal gray level or color cameras. However, for the normal camera, massive amount of redundant information has to be transmitted, which requires a large

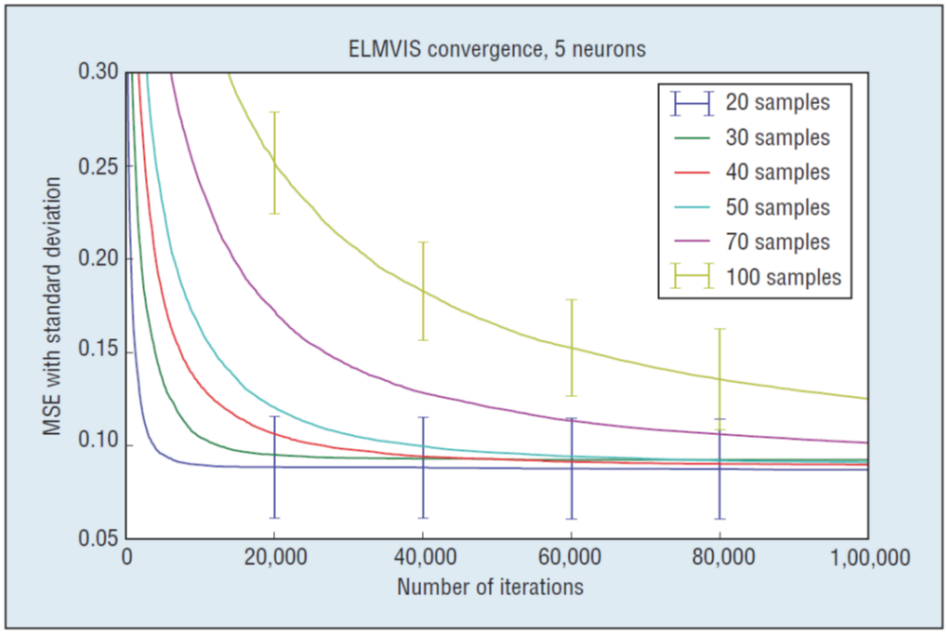

Figure 2.14: Convergence of ELM visualization algorithm on spiral dataset. With 100,000 training steps and 100 restarts, this figures shows the number of iterations requirement for ELM, varying from 20 samples to 100 and proves the fast converging speed of ELM [105]

bandwidth, and things will be even worse in the case of wireless sensor network (WSN). Smart vision sensors, which have certain computation capability, provide only the useful information, greatly saving the bandwidth and making themselves promising in WSN. In addition, since smart vision sensors capture only the motion but not raw images, the privacy of the target person (e.g. in the case of elderly care) can be protected. In some extreme conditions, such as dark or smoky environments, infrared thermal or depth sensors can be employed.

The preprocessing step mainly covered background subtraction techniques in video-based recognition. The purpose of this step is to obtain the human silhouette for further processing.

In feature extraction part, various features and representations (handcrafted and biologically plausible) were illustrated. We reported on color, texture, motion and shape features and emphasized several kinds of shape-based representations, such as moments, transform domain descriptors, spatial corre-

lation approaches and statistical methods. Although these handcrafted features may be discriminative for object recognition, they lack biological plausibility. It is well known that human visual system performs the object recognition tasks extremely well, much better than state-of-the-art computer vision systems. Therefore, developing a biologically inspired system to mimic the visual processing in cortex is a promising way to go for computer vision. We started from the illustration of the architecture of visual cortex, moved on to Hubel and Wiesel model of simple and complex cells, then gave a brief review on several hierarchical feed-forward models of visual processing in the cortex ventral stream, with the focus on HMAX (of Riesenhuber and Poggio) and the extended version proposed by Serre et al. [82].

Last but not least, several commonly used classifiers (e.g. k-NN, SVM, ANN and so on) were discussed in the last section. k-NN is a simple example-based classifier which involves searching in the training samples. It works efficiently for small training dataset. However, it would be very slow once the training library becomes large. In this condition, learning-based classifiers (such as SVM, BPNN and RBF neural networks, which optimize the model parameters in the training phase) would be much more efficient in the classification.

Chapter 3

Proposed Recognition System

This chapter will propose an event-driven classification system with a fast online training speed. The system takes address events data from a DVS. It extracts cortex-like spike-based features [108, 109] through a neuromorphic feature extraction unit and classified the extracted feature spike patterns using a Regularized Online Sequential Extreme Learning Machine (OSELM) [110] with Lookup Table (LUT). A Motion Symbol Detector in the system detects a burst of input AER events and then trigger the successive processing parts. The priorities are a fast online training speed, and a hardware friendly architecture. The major contribution of this work lies in three areas:

- **Event driven, biology inspired, hardware friendly classification system.** To better utilize the power of event based DVS sensor, a fully event based architecture is proposed. This architecture runs on the fly and takes real time processing into consideration. The inherent nature of DVS sensor can provide event stream. This can largely reduce data but requires the later processing to be completely event driven. Luckily for this system, algorithm is designed to be able to operate on individual pixel. In addition, the whole system is designed according to the most recent visual cortex research that can be performed with high efficiency. Last but not least, the system is hardware friendly, using mechanism like LUT to utilize the advantage of hardware for better performance and less resources.

- **The seamless integration of online sequential learning method to an AER classification system.** In this system, the feature extraction and classification are designed to be seamless and straightforward. The extracted features, along with their addresses, are fetching their corresponding weights in Initial Weight LUT, and then trained/tested by later classification network. The procedure is direct and without any lag. It performs better when considering the online sequential training method of classifier.

- **The use of an LUT to achieve a virtually fully connected system by physically activating a very small subset of the classification network.** In principle, all the features after extraction needs classifying. Suppose an image with $m \times n$ resolution, for a four maps feature extraction network and N classes task, it will require $4 \times m \times n \times N$ neurons. However, with this virtually fully connected network, every feature can fetch its weight and the size of network can be greatly reduced.

3.1 Address Event Processing

Though dynamic vision sensor, or silicon retina, has the great advantage in reducing output data amount, its signal processing cannot easily follow previous frame based algorithms. Limited feature extraction can be done at pixel level [111]. The output address events cannot be directly fed to classifiers. Additional processing methods, like segmentation, resizing and repositioning are also different. Therefore, a complete set of innovative event based signal processing algorithms has to be explored.

In previous research, there is attempt to adopt AER processing back to frame. jAERViewer [112] is to divide events into fixed time slides and reconstruct those events to pseudo frames. One example is shown in Figure 3.1. The event stream is separated into slices in time domain, for example, each slice has a time interval of $20ms$. Then, slice 1 will contain events happen between $0-20ms$, slice 2 will contain events happen between $20-40ms$, and so on. A frame can be accumulated from each slice of events, and the fast playback of all frames makes a continuous video. The resolution of each

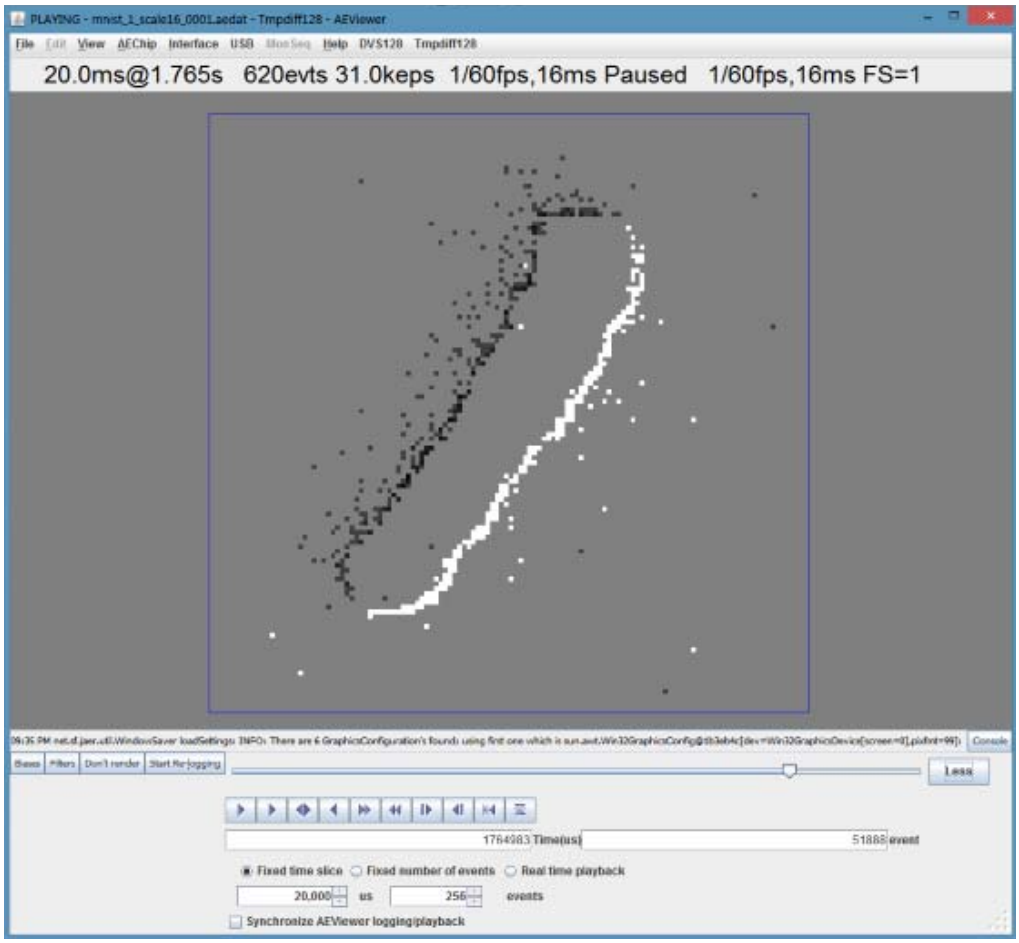

Figure 3.1: jAERViewer open source program for AER events frame reconstruction.

frame will be determined by AER sensor. In this figure, events are extracted from a 128×128 AER sensor. For each frame, the reconstruction takes place as follows: the whole frame value will be reset to 0 at the beginning; for each event within this slice, the corresponding pixel with this event's address will be increased by 1 (for *ON* event) or decreased by 1. After this process, all the pixel values of this frame are determined. The frame can be shown after normalization or filtering. The background with value 0 is gray, *ON* events are shown in bright color, and OFF events in dark color. Figure 3.2 shows a spatiotemporal scatter plot of address events captured by an AER vision sensor [113]. An Arabic number $''1''$ is moving left and right in this recording, lasting for around 6 seconds.

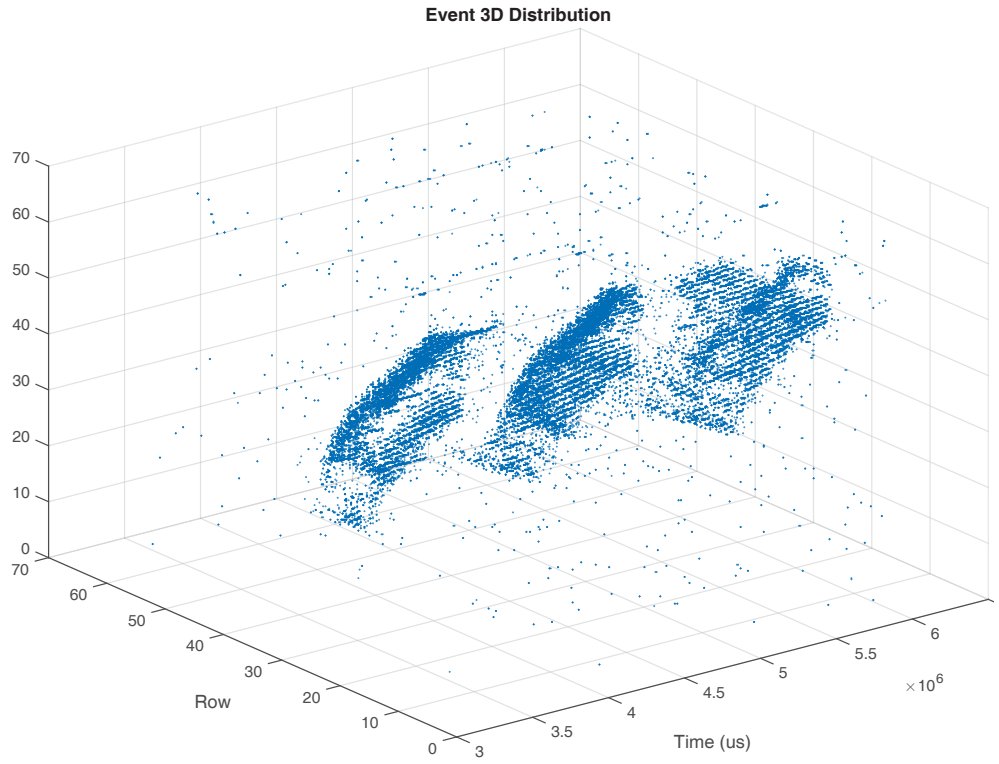

Figure 3.2: AER event address and time plotting of a Arabic number ″1″ in MNIST DVS dataset [113]

In order to fully utilize the advantage of AER sensors, event-based processing should be used, considering the individual events instead of frames. A fully event-based MAX-like convolution network is proposed to extract features, without any idea of frames. Since it is event-based processing, the features will be dynamically changing for each input event. Classification in such high frequency is not only impossible but also inaccurate. Therefore, some good timings to perform classification should be specified and be adaptive to events. This is exactly the reason of introducing Motion Symbol Detector in the later illustration.

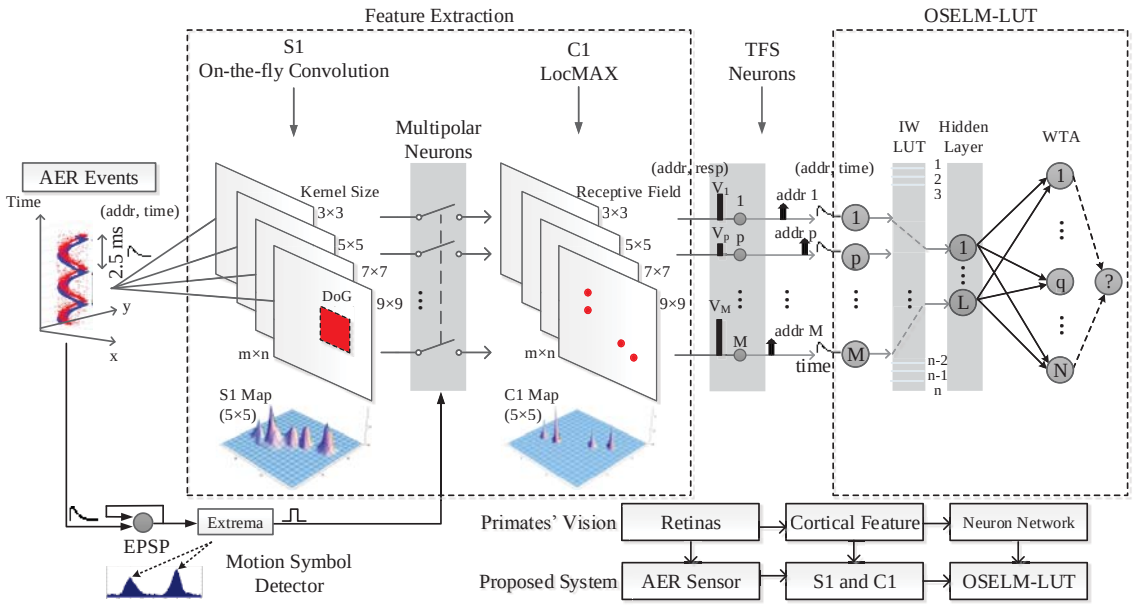

Figure 3.3: Architecture of the propose AER classification system. The lower right part of the figure illustrates the system's neuromorphic strategy. The system consists of several building blocks, namely Convolution and Competition, Feature Spike Conversion, Motion Symbol Detector and an OSELM-LUT classifier. Each input event is projected onto a group of dynamic $S1$ feature maps through on-the-fly convolution with a forgetting mechanism. $S1$ neurons compete within their local neighbors through a LocMAX operation (local area max pooling) to survive in the $C1$ layer. Survived $C1$ neurons represents a number of salient blob features. The Motion Symbol Detector can detect a burst of events and then trigger the successive processing stages. $C1$ responses are sent to a small set of TFS neurons and converted into feature spikes. The spike patterns are then fed to OSELM-LUT for classification. Each feature spike associated with an address fetches its corresponding weight in Initial Weight LUT. The final decision is made by the winner-take-all (WTA) result of the OSELM encoder outputs.

3.2 System Architecture Overview

Fig. 3.1 shows the architecture of the proposed system. It comprises several building blocks: Convolution and Competition, Feature Spike Conversion, Motion Symbol Detector and an OSELM-LUT classifier. The feature extraction algorithm acts on individual events. Each input event is projected onto a group of dynamic $S1$ feature maps through on-the-fly convolution with a forgetting mechanism. $S1$ neurons compete within their local neighbors through LocMAX operation to survive in the $C1$ layer. Survived $C1$ neurons represent a number of salient blob features. The Motion Symbol Detector detects a burst of events and then triggers the successive processing stages. $C1$ responses are sent to a small set of TFS (Time-to-First-Spike) neurons and are converted into feature spikes. The spike patterns generated are then fed to OSELM-LUT for classification. Each feature spike associated with an address fetches its corresponding weight in Initial Weight LUT. The final decision is made by the winner-take-all (WTA) result of the OSELM encoder outputs. The flow of information processing is as follows:

(i) **Feature Maps Construction and Neuron Competition:** The event stream from the AER sensor is sent in parallel to a batch of $S1$ feature maps [114], where on-the-fly convolution with Difference of Gaussian (DoG) filtering is performed. A forgetting mechanism is introduced in the convolution to forget the impact of very old events. After a trade-off between complexity and performance, it was decided to use 4 scales (3, 5, 7 and 9) of DoG filters. Each $S1$ neuron competes with other neurons within its local neighbors (i.e. neurons within its receptive field) and can survive in $C1$ layer if it is the maximum among those local neighbors (LocMAX) [115, 114, 116].

(ii) **Motion Symbol Detector and Feature Spike Conversion:** The convolution maps ($S1$) are updated for each incoming AER event. The classification decision, however, does not need to be made at such a high speed, so a Motion Symbol Detector comprising a leaky integrate neuron and an Extrema detector is included in the system to detect bursts of input AER events and then trigger the successive processing stages. After the LocMAX operation, survived $C1$ responses are sent to a small set of Time-to-First-Spike (TFS) neurons, where they are converted into spikes.

(iii) **Regularized OSELM with Lookup Table:** The generated feature spike patterns are then fed to the regularized OSELM-LUT for classification. Each feature spike is associated with an address, which can be used to access the LUT and fetch a corresponding weight. In this way, a virtually fully connected system is achieved by physically activating only a very small subset of the classification network. The final classification decision is made by applying winner-take-all (WTA) to the auto encoding outputs of the OSELM.

3.3 Dynamic Vision Sensor

We use the so-called Dynamic Vision Sensor (designed by Chen et al. [34]) to capture human motion. The sensor can automatically remove the still background and output only the contour of moving object (as a stream of binary motion events). It avoids the transmission of the massive raw data, largely saving the channel bandwidth and thus making itself a promising candidate of wireless sensor node. In addition, since this sensor only outputs binary images, the privacy of the target person can be protected. This property is appealing especially in the application of elderly care. On the one hand, the elderly people who live alone do need a vision-based care system that can detect their accidents such as falling down and further inform their family members or medical care personnel. However, they may not want their privacy to be exposed to others. Indeed, being monitored by a normal camera makes people feel uncomfortable. Now, by using this kind of temporal difference image sensor, it can still monitor the daily activities of the elderly, but with a high protection of their privacy.

The working principle of this sensor is as follows. Inside the sensor, each pixel is equipped with an analog memory (capacitor) and the whole array is hence capable of storing the current frame as a reference image. The rows are first sequentially selected for reset. Later at another round of scanning, the rows of pixels are selected for readout sequentially. Each pixel will output both the new integration voltage on its photodiode and the previous voltage stored on its capacitor. The two voltages are fed into a global event generator circuit which is composed of a global amplifier with a temporal-difference

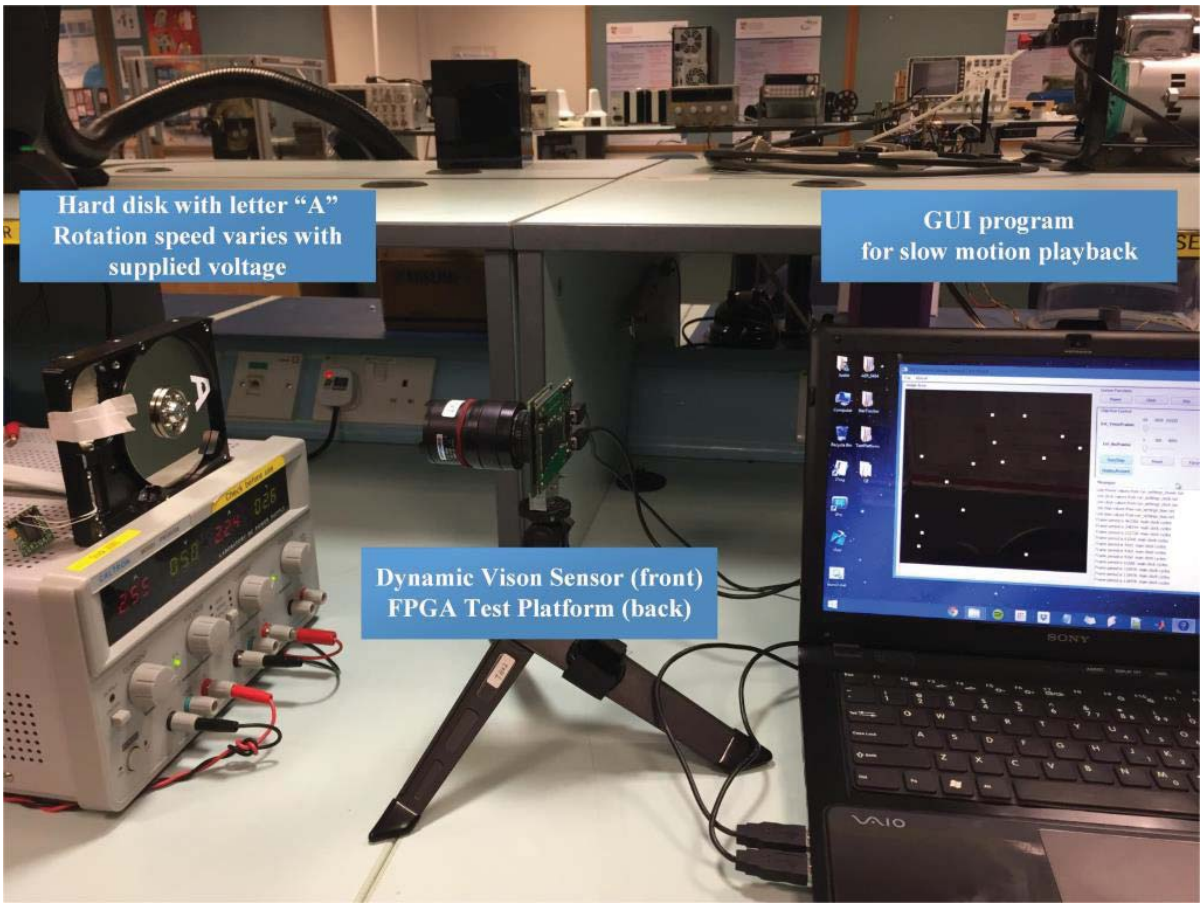

Figure 3.4: Dynamic Vision Sensor along with testing platform. This figure shows DVS in [117]. On the left is a hard disk with capital letter *A* and rotates in high speed; in the middle is the DVS sensor with FPGA universal test platform; On the right side is a GUI program for slow motion playback of the harddisk. This test shows that DVS has the ability to capture high speed moving objects.

computation circuit based on dual comparison. The event generator computes the difference between the two voltages, and compares it to a positive and a negative threshold. An effective event $1'b1$ is generated if this difference exceeds the thresholds; otherwise, $1'b0$ is used as the output of that pixel. In this way, the still background is removed and the human motion is captured in a stream of binary motion

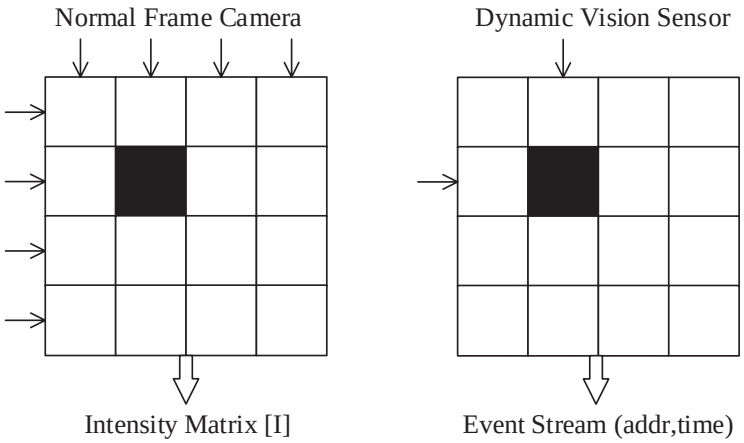

Figure 3.5: Output data comparison between frame camera (left) and DVS (right). For the conventional frame based camera if there is an active pixel showing in dark color, the sensor will output the whole picture as the Intensity Matrix; while for DVS, it only output the address of this active pixel and ignore all the other background, its output data is in the format of $(addr, time)$. This advantage can be more obvious if it is in the case of capturing high speed moving objects.

events.

3.4 Feature Maps Construction and Neuron Competition

Current cortical theory suggest a hierarchical, mainly feedforward structure. The proposed system comprises two layers of feature extraction maps, namely: $S1$ maps (on-the-fly convolution with leaky EPSP) and $C1$ maps (LocMAX over Receptive Field).

3.4.1 $S1$ Maps: On-the-fly Convolution with Leaky EPSP

$S1$ Maps are built by convolving each input event with a bank of DoG filters on the fly [118]. For each filter, it resembles a neuron that has receptive field of a particular size and responds best to a given size

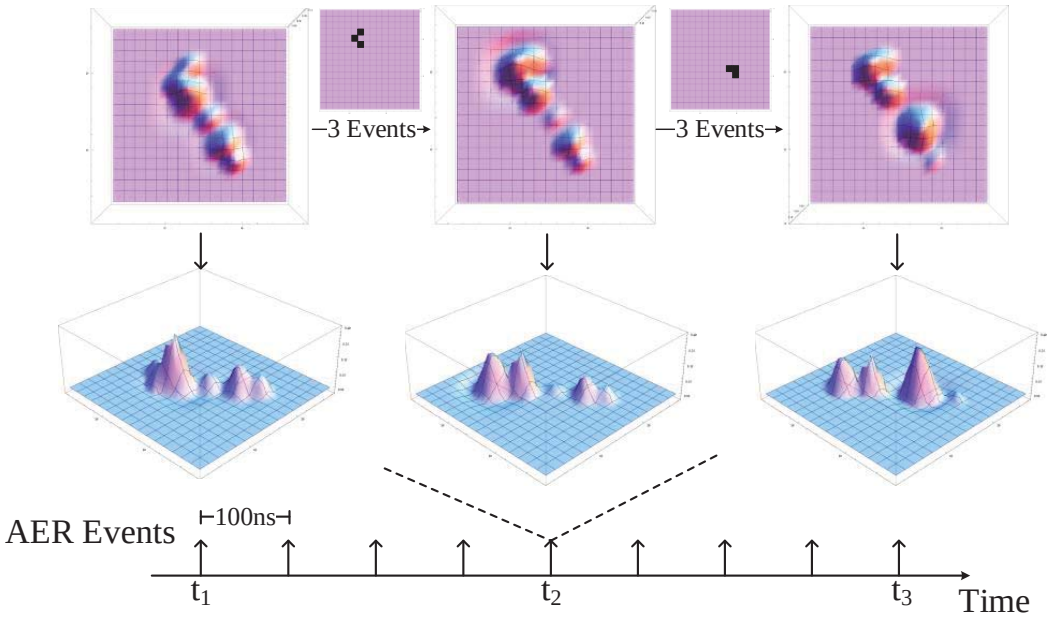

Figure 3.6: Reconstructed images illustrating on-the-fly Convolution with a forgetting mechanism. AER events are projected onto a batch of $S1$ maps with a period of $100ns$ (this figure only shows the $S1$ map corresponding to the 5×5 kernel). Using event address as centroid, it will trigger kernel size local neighbours (superposition). Between t_1 and t_2, there are three events. Each neuron on the $S1$ maps will decrease (or increase) towards the resting potential (usually set as 0) as time goes by. The reconstructed images above show the same $S1$ map at different times.

feature. The DoG filter used in this system is the approximation of the Laplacian of Gaussian (LoG), which is defined as:

$$DoG_{\{s,l_c\}}(l) = G_{\sigma(s)}(l - l_c) - G_{3\cdot\sigma(s)}(l - l_c) \tag{3.1}$$

$$G_{\sigma(s)}(l) = \frac{1}{2\pi \cdot \sigma(s)^2} \cdot \exp\left(-\frac{||l||^2}{2 \cdot \sigma(s)^2}\right) \tag{3.2}$$

where $G_{\sigma(s)}$ is the 2-D Gaussian with variance $\sigma(s)$ depending on the scale s. l_c is the center position of the filter.

The on-the-fly convolution is illustrated in Fig. 3.6. Each input event adds a convolution kernel to the $S1$ map. The center of the kernel is aligned to the map according to the address of the input event. To eliminate the impact of very old events on the current response map, a forgetting mechanism is adopted. Each pixel in the response map will decrease (or increase) towards the resting potential (usually set as 0) as time goes by.

3.4.2 $C1$ Maps: LocMAX over Receptive Field

Convolution with four scales of DoG filters produces, four $S1$ maps. $C1$ maps are then obtained by performing the LocMAX operation on the $S1$ maps. As can be seen in Fig. 3.7, each $S1$ neuron competes with other neurons locating within the local neighbors (i.e. the neurons that locate within the receptive field) and can survive in the $C1$ layer if it is the local maximum. In various $S1$ maps, each neuron has different receptive field, varying from 3×3 to 9×9. After the LocMAX operation, each survival neuron in the $C1$ maps represents a salient feature, i.e. a blob of a certain size.

3.4.3 Feature Extraction using DoG Filters

The DoG (Difference of Gaussian) filters used in $S1$ Maps are believed to imitate the neural process by which the retina of an eye extracts details from external stimuli[119, 120]. Several different scales of DoG filters would construct a variety of $S1$ maps. Also, max pooling over two polarities, different scales, and different local positions produce contrast reversal invariance, scale invariance and positional invariance, respectively [121]. Bio-plausible implementations of the MAX operation were proposed in [81], and biological evidences of neurons displaying MAX-like behavior have been found in a subclass of complex cells in $V1$ [122] and cells in $V4$ [123]. After a trade-off between invariance and computational complexity, the proposed system only adopts MAX over local neighbour (LocMAX) to increase invariance somewhat while maintaining real time processing capability. Table 3.1 shows the number of neurons survived in $C1$ in the *Posture* dataset classification, as specified in Section 3.7. The number

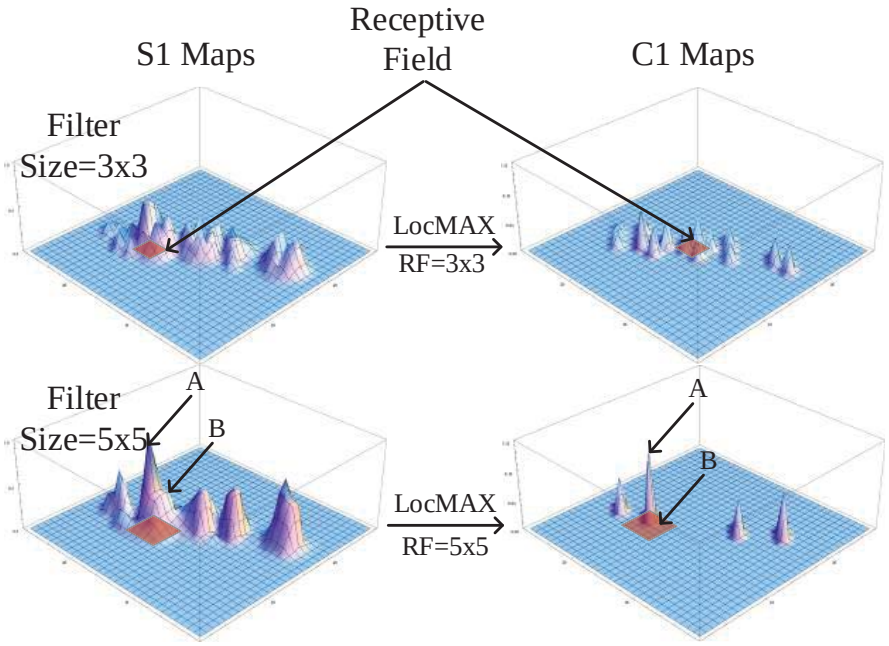

Figure 3.7: The LocMAX operation. Each $S1$ neuron competes with its local neighbors and can only survive in the $C1$ layer if it is the local maximum. This diagram shows the LocMAX operation from $S1$ Maps to $C1$ Maps in 3×3 and 5×5 feature maps, where neuron A triumphs over neuron B and survives in the $C1$ Maps. It should be noted that this $C1$ map figure is rendered with shadow for better visualization, only one neuron survives in each mountain-like response.

of survived neurons drop by more that half when DoG filtering is used instead of the four scales, four orientation Gabor filters described in [124].

Table 3.1: Number of Survived $C1$ Neurons

	BEND	*SITSTAND*	*WALK*
Zhao et. al [124]	40 ± 10	45 ± 7	65 ± 9
This Work	10 ± 8	12 ± 6	16 ± 10

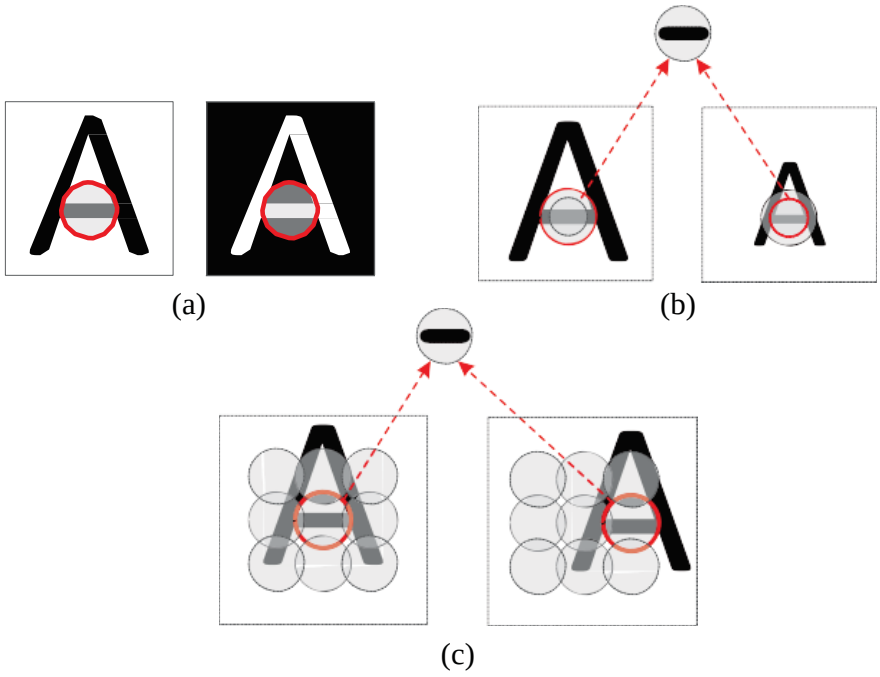

Figure 3.8: Illustration of contrast/scale/position invariance obtained from DoG filter[118]. (a) image patch in red circle has different contrast, but later max pooling over polarities reduces this variance; (b) scale invariance by max pooling over varied scales; (c) local position invariance by max pooling over local positions.

3.5 Motion Symbol Detector and Time-to-First Spike

The proposed system uses a Motion Symbol Detector to detect a burst of input AER events and then trigger the successive processing stages. The Motion Symbol Detector comprises of a leaky integrate neuron and an Extrema detector. Fig. 3.9 shows how each input event contributes an Excitatory Post-Synaptic Potential (EPSP) to the leaky integrate neuron. The total potential of this neuron is obtained by superposition of all the EPSPs. If the neuron's potential exceeds V_{THH} and the curve has reached a peak, an Extrema can be identified and a pulse will then be generated to turn *ON* the Multipolar Neurons (Switches) [125] between $S1$ and $C1$ in Fig. 3.1. After this, the neuron will go through a short refractory period, during which new input events will not be responded. This resembles the way primates' neurons eliminating the reaction potential of the same, constant stimuli. For real time processing, redundant

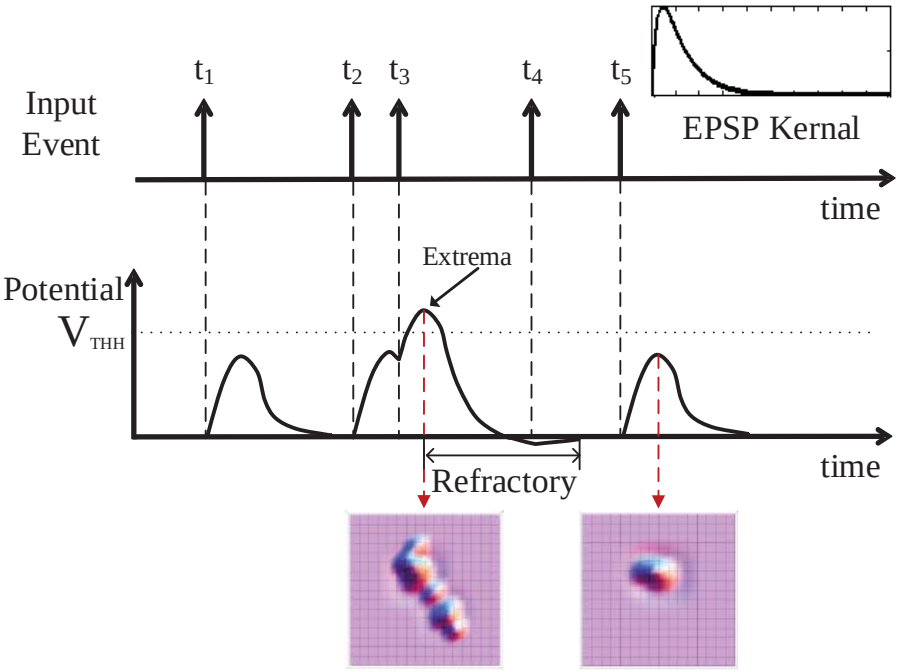

Figure 3.9: Motion Symbol Detector. Each input event initiates an EPSP in the neuron. The total potential of the neuron is obtained by superposition all the EPSPs. This figure shows two reconstructed images of an $S1$ map at two particular timings (Extrema and Refractory).

image transition events of similar features occurring very close together within a short period of time will be eliminated. Fig. 3.9 shows two reconstructed images, at different timings, to demonstrate the effectiveness of our algorithm.

After the LocMAX operation, the survived $C1$ responses are sent to a small set of Time-to-First-Spike (TFS) neurons [124], where they are converted into spikes. Extracted features are again encoded into AER form (spatiotemporal spikes stream), with an address and a time stamp. The time stamp is inversely proportional to the strength of the $C1$ response, and the address indicates the $C1$ neuron's position. All this allows seamless connection to the OSELM-LUT classifier later.

3.6 Regularized OSELM with Lookup Table

In this section, we will explain the idea of transforming an original Extreme Learning Machine (ELM) into an online sequential version capable of handling real-time streaming processing and other strategies to improve classification results and facilitate actual hardware implementation.

3.6.1 Regularized ELM and OSELM

The Extreme Learning Machine (ELM) provides unified solutions to generalized feed-forward neural networks. A Single Hidden Layer Feed-forward Neural Network (SLFN) with nonlinear activation functions and N hidden layer neurons can classify N distinct features. Moreover, the Initial Weights (IW) and hidden layer biases do not require tuning and can be generated randomly [104]. ELM has been proven to have a fast training speed and competitive classification performance. It transforms learning into determining output weights β through calculating the generalized inverse of hidden layer activation function matrix. For N arbitrary different patterns (x_i, t_i), the basic ELM, with hidden neuron number L and activation function $g(x)$ can be modeled by:

$$f_{\tilde{N}}(X_j) = \sum_{i=1}^{\tilde{N}} \beta_i g(a_i, b_i, x) = t_j, j = 1, ..., N. \tag{3.3}$$

where a_i is the weight from input layer to i_{th} hidden neuron, and β_i from the ith hidden neuron to the output layer, respectively. b_i is the bias for the ith hidden neuron. $g(a_i, b_i, x)$ denotes the activation function from the ith hidden neuron corresponding to the input x. The above ELM is proved to be capable of approximating N samples with zero error.

This model produces the smallest training error because it uses the least-square solution to solve the linear functions of $H\beta = \Gamma$ as:

$$||H\hat{\beta} - \Gamma|| = ||HH^{\dagger}H - \Gamma|| = min||H\beta - \Gamma|| \tag{3.4}$$

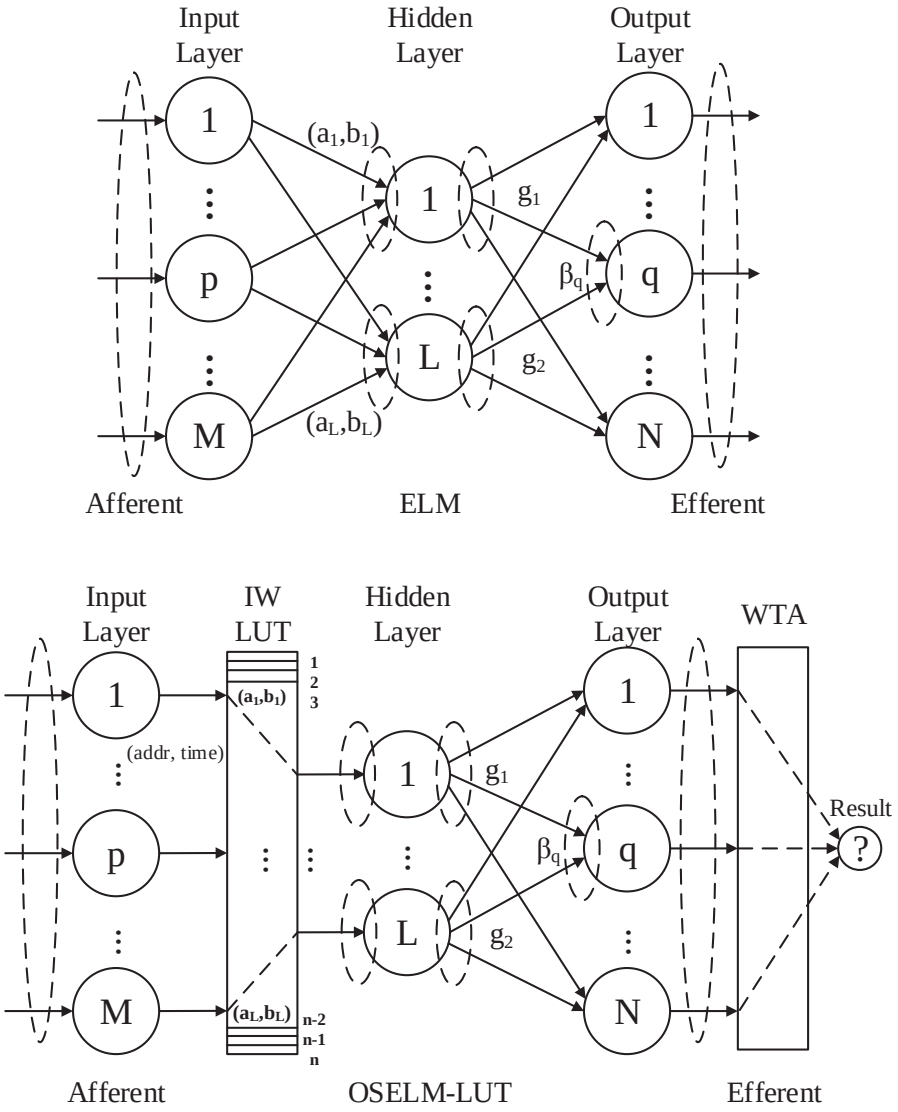

Figure 3.10: Comparison of ELM and OSELM-LUT. The basic ELM is shown in the top illustration, the proposed OSELM-LUT in the button illustration. For every afferent, each event fetches its Initial Weight according to its address in IW LUT and maps to the Hidden Layer. Recursive β update occurs only between the hidden layer and the output layer. Using the encoding potential of ELM, better classification results can be achieved by winner-take-all (WTA).

where H denotes hidden layer output matrix, Γ the targets and $H^{\dagger}$ represent a Moore-Penrose generalized inverse of H. Under the condition of $rank(H) = L$,

$$\beta = H^{\dagger}\Gamma; H^{\dagger} = [H^T H]^{-1} H^T \tag{3.5}$$

The non-singularity of $H^T H$ can be ensured with the method of reducing hidden layer neuron number or, in initialization, increasing the number of training samples N. To improve the stability and generalization capacity of the traditional ELM, Huang et al. [104] proposed an equality-constrained optimization-based ELM. In this method, the solution of ELM can be expressed as:

$$\beta = (\Psi)^{-1} H^T \Gamma; \Psi = H^T H + \frac{I}{C} \tag{3.6}$$

where I is the identity matrix and C is the regularization parameter for balancing the influence of error-term and the model complexity: a constant in a specific training model. This is a general method for stabilizing the least-square regression solution. In statistics it is called ridge regression [126].

However, the solution method above assumes that the number of different distinct training samples to be N (Batch Learning), which is not the case in actual real-time learning. Modifications are therefore required to adapt it to the scenario of online sequential/increamental learning [110].

Let us suppose that we have an initial training set $(x_i, t_i, 1 \leq i \leq N_0)$. The ELM learning method for this set of data is equivalent to solve the error minimizing problem $||H_0\beta - \Gamma_0||$ for $N_0 \geq L$ where

$$H_0 = \begin{bmatrix} g(a_1, b_1, x_1) & \cdots & g(a_L, b_L, x_1) \\ \vdots & \cdots & \vdots \\ g(a_1, b_1, x_{N_0}) & \cdots & g(a_L, b_L, x_{N_0}) \end{bmatrix}_{N_0 \times L} \tag{3.7}$$

and Γ_0 represents the set of targets $t_i (1 \leq i \leq N_0)$.

One solution for minimizing $||H\beta - \Gamma_0||$ is $\hat{\beta}^{(0)} = (\Psi_0^{-1} H_0^T \Gamma)$. Suppose another training data set containing N_1 samples. Now the error minimization problem can be solved with more data:

$$\varepsilon = \left\| \begin{bmatrix} H_0 \\ H_1 \end{bmatrix} \beta - \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix} \right\| \tag{3.8}$$

Solving the error minimization problem for the whole data, the weight $\beta^{(1)}$ becomes:

$$\beta^{(1)} = (\Psi_1)^{-1} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix} \tag{3.9}$$

online sequential learning, $\beta^{(1)}$ need to be represented with $\beta^{(0)}, \Psi_1, H_1$, and Γ_1. Since:

$$\begin{aligned} \begin{bmatrix} H_0 \\ H_1 \end{bmatrix}^T \begin{bmatrix} \Gamma_0 \\ \Gamma_1 \end{bmatrix} &= H_0^T \Gamma_0 + H_1^T \Gamma_1 \\ &= \Psi_1 \beta^{(0)} - H_1^T H_1 \beta^{(0)} + H_1^T \Gamma_1 \end{aligned} \tag{3.10}$$

By combining (3.9) and (3.10), we have:

$$\beta^{(1)} = \beta^{(0)} + (\Psi_1)^{-1} H_1^T \left[\Gamma_1 - H_1 \beta^0 \right] \tag{3.11}$$

The recursive expression for updating the nonlinear model can then be obtained by:

$$\beta^{k+1} = \beta^k + \Psi_{k+1}^{-1} H_{k+1}^T \left(\Gamma_{k+1} - H_{k+1} \beta^k \right) \tag{3.12}$$

$$\Psi_{k+1}^{-1} = \Psi_k^{-1} - \Psi_k^{-1} H_{k+1}^{-1} [I + H_{k+1} \Psi_k^{-1} H_{k+1}^T]^{-1} \tag{3.13}$$

The recursive update algorithm β shown above calculates the error of every training sample, obtains the error weight $\beta = \Psi_{k+1}^{-1} H_{k+1}^T (\Gamma_{k+1} - H_{k+1} \beta^k)$ and updates the previous β^k. OSELM does not require the number of training samples (Block Size) to be identical. The recursive learning approach presented in (3.12) and (3.13) is made up of two phases: (1) Initialization phase and (2) Sequential phase.

The recursive update algorithm β of the online sequential ELM, basically calculates the error of every training pattern $\Gamma_{k+1} - H_{k+1} \beta^k$, and the error weight $\beta = \Psi_{k+1}^{-1} H_{k+1}^T \Gamma_{k+1} - H_{k+1} \beta^k$, and updates

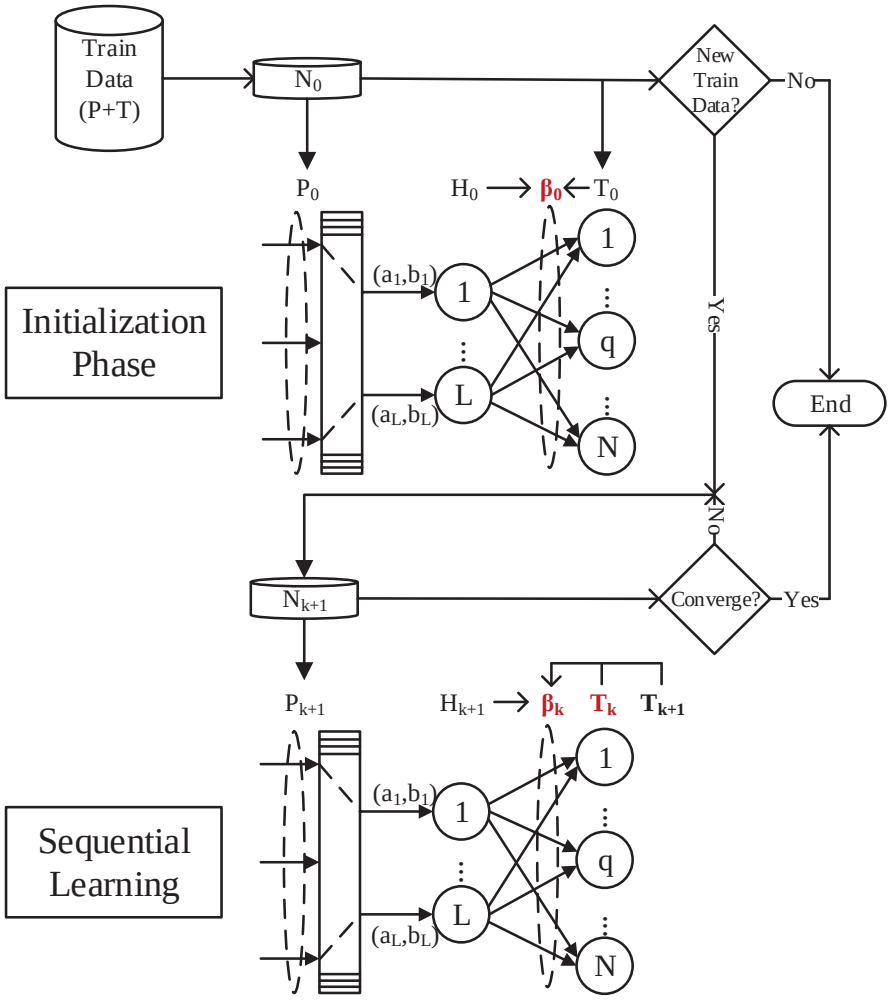

Figure 3.11: The OSELM recursive training process has two phases: Initialization and Sequential Learning. In the initialization phase, N_0 features (with Pattern P and Target T) are used to calculate the initial weight β. Then, in the sequential learning phase, the N_{k+1} feature is used to update the previous weight β_k. The training process terminates when there is no new training data when β converges.

the previous β^k. The OSELM does not require the number of training set patterns (Block Size) to be identical. The recursive learning approach presented in (3.12), (3.13) is made up of 2 phases: (1) Initialization phase and (2) Sequential phase. In Initialization, H_0 is prepared in such a manner that

the number of distinct training samples is equal to or greater than the number of hidden layer neurons $N_0 \geq L$. Then, in sequential phase, the weight β is updated by new training data set in either a one-by-one or a chunk-by-chunk fashion. Fig. 3.11 shows the flow diagram for the OSELM recursive training process.

3.6.2 OSELM with Lookup Table

In principle, all the $C1$ responses are needed for classification. Let $m \times n$ denotes the resolution of the feature maps and N denotes the number of classes. The size of the fully connected ELM network would then be: $4 \times m \times n \times N = M \times N$ in Fig 3.10. The size of the network is large. However, thanks to the LocMAX operation and the AER nature of the feature spikes, it is possible to achieve the same results as with the full network using a very small network with only a few inputs (in our case, around 100 input neurons are connected to hidden layer neurons for every pattern). This reduces the hardware costs. Fig. 3.10 shows how a LUT is used to store the initial weights between the input layer and hidden layer of the ELM. Each input feature spike has an address, which is used to access the initial weight LUT and fetch its corresponding weight. The timestamps of the feature spikes are used as the input. Each class is represented by 10 bits coding. The final classification decision is made by applying winner-take-all (WTA) to the outputs. Fig. 3.11 shows the flow diagram of the recursive training process in the OSELM-LUT.

3.7 Experiment Results

3.7.1 Performance Evaluation on a Posture Dataset

The proposed system was evaluated on the 64×64 AER posture dataset captured by Zhao et. al [127]. This dataset contains three human actions, namely bending to pick something up (*BEND*), sitting down and standing up (*SIT/STAND*) and walking back and forth (*WALK*). Fig. 3.12 shows some reconstructed

images. Each row corresponds to one action; the images are reconstructed from AER motion events, using the fixed time slice approach [127] with a frame interval of 20 *ms*.

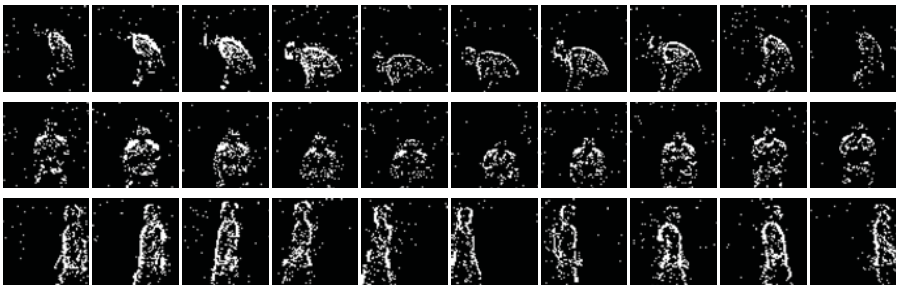

Figure 3.12: Reconstructed images from the AER posture dataset, which contains three kinds of human actions. Each row corresponds to one action.

3.7.1.1 Training Block Size and Hidden Neuron Number

In the OSELM-LUT classifier, two major parameters need to be tuned: the sequential training block size and the number of hidden neurons. As shown in Fig. 3.13, classification accuracy scarcely varies as the block size increases but the training time decreases dramatically. Also, as we can see in Fig. 3.14, when the number of hidden neurons increases, accuracy improves, but the training time gets longer. The number of hidden neurons was set at 800 after a trade-off between accuracy and training speed. Considering the actual real-time processing scenario, in which feature detection is sparse and asynchronous, the block size was set at 2.

3.7.1.2 Weight Convergence and Training Speed

To test the weight convergence and training speed, we recorded the realtime weight training process. Fig. 3.15 shows the first 30 hidden layer neuron weights in three phases. The first, right after the initialization phase; the second, after 100 times recursive training (25*s* duration); and the third, which is the final result, after 142 times recursive training (35*s* duration). Fig. 3.15 also shows the fast training speed and weights' convergence of the OSELM-LUT.

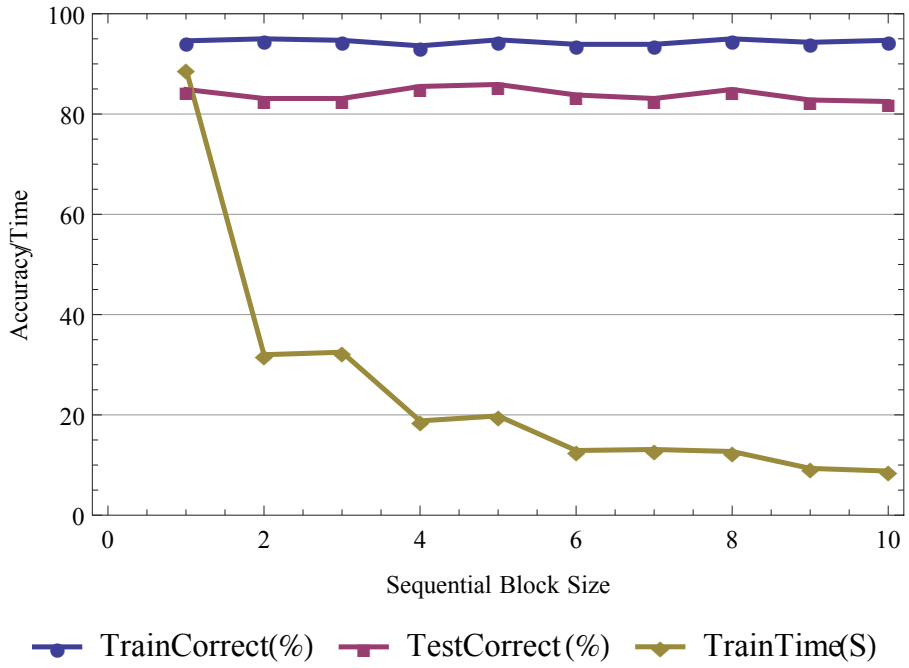

Figure 3.13: Test result for Accuracy/Time with different sequential training block sizes.

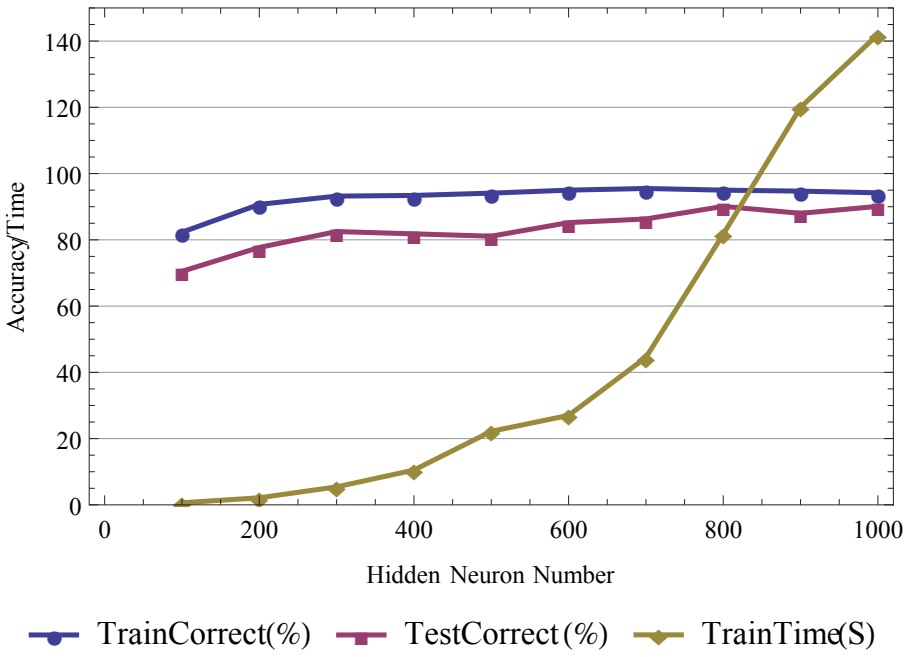

Figure 3.14: Test result for Accuracy/Time with different numbers of hidden neurons.

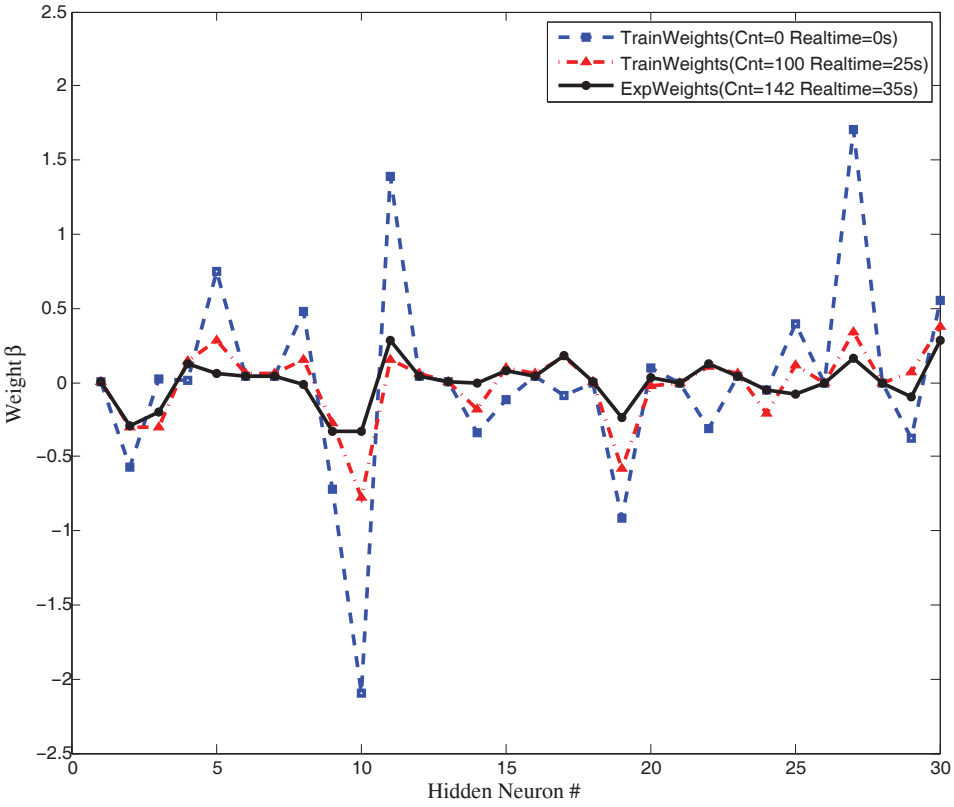

Figure 3.15: Real-time Weight Training on a Posture Dataset. Three different training phases are shown: right after the initialization phase (recursive train cnt=0); after 100 times recursive training (25 *s* duration); and at the end of training. The figure shows that the hidden layer weights can converge within 35 *s*.

Whole system performance was compared between our work (with two different kernels: Singmoid/Hardlim), Zhao et al. [124], and another two biologically inspired algorithms, HMAX [108] and Serre et. al [109], with the same desktop computer. The *Posture* dataset consists of 191 *BEND*, 175 *SIT/STAND* and 118 *WALK*. This test randomly pick out 80% for training and 20% for testing. Evaluation was repeated 20 times to obtain the average performance. More parameters are shown in Table 3.2. The results in Table 3.3 show that the proposed system has a very fast training speed while still maintaining a competitive accuracy. The MATLAB code of the proposed algorithm can be found in the webpage [128].

Table 3.2: Parameters for Posture Dataset

Time Constant τ_m in Motion Symbol Detector	$20ms$
Leakage rate in event-driven DoG filter	$5 \times 10^1 s^{-1}$
Pattern number in Initialization N_0	100
Hidden Neurons Number L	800

Table 3.3: Performance on Posture Dataset

Model	TestCorrect(%)	TrainTime(s)
This Work (Sigmoid)	90.1 ± 3.5	21.1
This Work (Hardlim)	88.0 ± 3.4	26.8
Zhao et. al [124]	98.5 ± 0.8	255
HMAX + SVM [108]	78.1 ± 4.1	240
Serre + SVM [109]	93.7 ± 1.9	702

3.7.2 Performance Evaluation on standard MNIST Datasets

Further evaluation was conducted with a standard dataset MNIST with 10 digits $(0-9)$ and a total of $70,000$ images. Fig. 3.16 shows some images from this dataset. The test randomly choose $60,000$ images for training and $10,000$ images for testing. To process the frame based MNIST dataset, these images were converted to an event stream. A basic thresholding method was used to convert grey level MNIST images into binary images [127]. The black pixels represent the black background and the white ones the foreground. Address events are generated from all the foreground pixels, assuming the pixels fire at the same time and the events are driven out with random priority. Each foreground pixel generates one event; each image generates around 200 events. The average length of a converted event stream is $10\mu s$, and the inter spike interval is about $100ns$. More parameters are listed in Table 3.4.

When the proposed system is compared with Zhao el al. [124] on the MNIST dataset, this algorithm

proves to have a higher potential. Table 3.5 shows how the proposed system (with two different kernels: Sigmoid/Hardlim) also has a better recognition rate and consumes less training time. The results obtained from the original batch learning version of the ELM with Gaussian kernel result are also listed. Noted that the original batch learning ELM can only takes frame-based MNIST as input images.

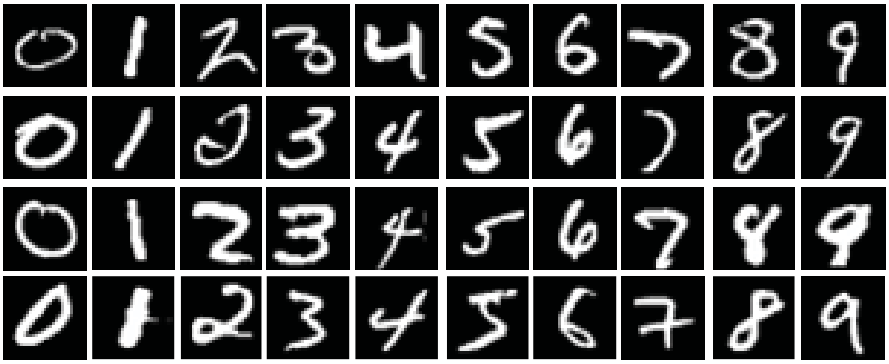

Figure 3.16: Frame images from MNIST dataset with ten kinds of handwritten digits.

Table 3.4: Parameters for MNIST dataset

Time Constant τ_m in Motion Symbol Detector	$20\mu s$
Leakage rate in event-driven DoG filter	$5 \times 10^4 s^{-1}$
Pattern number in Initialization N_0	1000
Hidden Neurons Number L	800

3.7.3 Performance Evaluation on MNIST DVS Datasets

The proposed system was also evaluated on MNIST DVS dataset [113], which contains a total of $30,000$ DVS recordings of 10 different digits $(0-9)$ in 3 different scales, using the DVS sensor reported in [129] . Fig. 3.17 shows some frames from these datasets reconstructed using jAERViwer [112]. A total of $10,000$ original 28×28 pixel MNIST handwritten digit images from MNIST [130] were up-scaled using

Table 3.5: Performances on MNIST Dataset

Model	TestCorrect(%)	TrainTime(s)
This Work (Sigmoid)	91.25 ± 0.14	166.2
This Work (Hardlim)	90.98 ± 0.19	104.0
Zhao et. al [124]	88.0 ± 2.58	5484
ELM(Gaussian) [104]	97.39 ± 0.1	545.9

smoothing/interpolation algorithms. Each up-scaled digit was then displayed on a LCD monitor with slow motion and a 128×128 pixel DVS was used to record the moving digits. The proposed system was evaluated on the scale-4 of the MNIST DVS dataset. For this scale there are a total of $10,000$ recordings, each with a time length of about $2s$. Since a digit in scale-4 roughly fits into a 28×28 patch, event-driven cluster-tracking algorithms [131] were used to track the moving digits from the original 128×128 DVS recordings. The tracked event streams were then set as the input for the proposed system. Each time, we randomly tested 4 groups in this scale (randomly split 90% of the images for training and 10% of the images for testing). The evaluation was repeated 10 times to obtain the average performance. The training accuracy was nearly 100% and the test accuracy was around 80%. The test results are given in detail in Table 3.6. Noted that for this relatively complex classification task, the number of hidden neurons is $6,000$, resulting in a relatively longer training time. The time constant τ_m and leakage rate are identical to *Posture* dataset.

3.8 Discussion on Proposed System

As seen in section 3.2, 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.

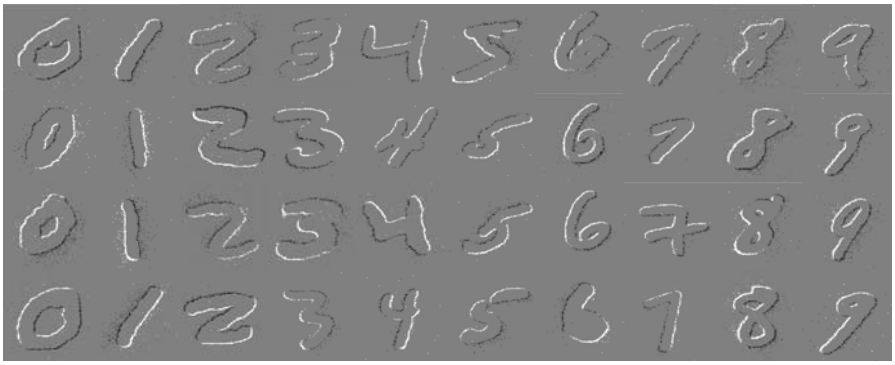

Figure 3.17: Frame images from the MNIST DVS dataset with 10 kinds of handwritten digits.

Table 3.6: Performance on MNIST DVS Dataset

Model	TestCorrect(%)	TrainTime(s)
This Work (Sigmoid)	81.25 ± 2.14	1766.2
This Work (Hardlim)	80.98 ± 1.19	1404.0
Rebuilt Frame (Sigmoid)	41.28 ± 2.27	2564.2
Rebuilt Frame (Hardlim)	40.20 ± 3.25	2178.6

One typical problem is when multiple objects are moving back and forth in the scene, or the background is moving. In this case the 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. 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 small disturbing moving object in the background, such as a cat [132]. Further challenge emerges when multiple objects run into one and then separate. More advanced object tracking algorithm or facility is to be employed.

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 $\pm 30^o$. For practical usage, multiple camera nodes should be deployed, and at this point, proposed system's advantage of reducing data will be more solid. Due to its high computation efficiency, it allows a compact, small footprint embedded system that can be easily installed. Since no raw video data is involved, patients' privacy is protected when they are monitored.

Chapter 4

Hardware Implementation

In this part details on event-based highly efficient hardware architecture will be covered. The whole system will be discussed in separate modules for the cost of their resources and then combined together; FPGA and software co-processing method will also be applied; the implementation of some preliminary parts, will be discussed in detail. More parts will be implemented in the future.

4.1 System Implementation Architecture

This section will discuss about the overall system architecture and cost of resources of each operation modules.

4.1.1 Overall System Architecture

Figure 4.1 shows the hardware architecture of the proposed object recognition system. Each event from the AER vision sensor is time-stamped by Sensor Ctrl module, which handles the hand-shaking with AER sensor and contains a counter for time-stamping of events. The time-stamped event, which is a stream of $(addr, time)$, then goes through feature extraction and classification (which are shown as dashed boxes in Figure 4.1). As mentioned earlier in the algorithm description part, feature extraction

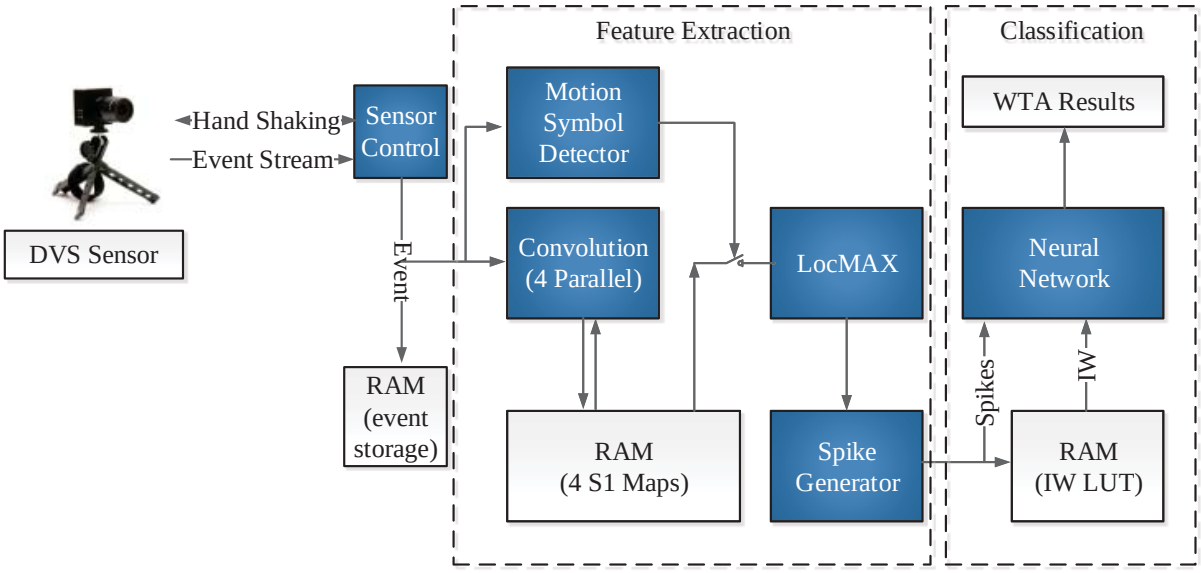

Figure 4.1: Hardware architecture of proposed classification system.

can be divided into several modules, namely Motion Symbol Detector, Leaky Convolution, LocMAX Competition, and Spike Generator; and the classification includes a weights lookup table and single hidden layer feed-forward neural network (SLFN).

4.1.2 Memory and Computation Cost

This section will compute and illustrate the resource cost of the whole system in separate modules. It should be noted that all calculations are based on 32×32 resolution DVS sensor.

4.1.2.1 Convolution with Leakage

As mentioned earlier in the algorithm description, on-the-fly convolution (with leakage) is applied to the address events from the AER vision sensor. The on-the-fly convolution is a simple addition of kernel onto the region specified by the address of incoming event. It also has a linear leakage mechanism to forget the impact of very old events.

In the proposed system, DoG filters F_i of four sizes $|Fi| \in 3,5,7,9$ are used, leading to four convolution modules in parallel. The generated four $S1$ feature maps have the same resolution with the input

($n \times n$, i.e. 32×32). If 48 bits are used to represent the information of each $S1$ (16 bits of $S1$ response, 32 bits of last update time for leakage calculation), then for each $S1$ map, $32 \times 32 \times 48bit = 48Kbit$ storage is needed.

For each $S1$ map, the on-the-fly convolution only affects a neighborhood region. The size of which is specified by the corresponding filter and the center by the input address event. The operations involved are leakage calculation, leakage subtraction and adding the kernel to the old $S1$ response. The new $S1$ ($S1'$ value as well as the event time) will be written back to $S1$ map.

$$S1' = S1 + Kernel - R_{leak} \times (t_e - t_l) \tag{4.1}$$

where R_{leak} stands for the leakage rate, t_e and t_l denote the input event time and the last update time of this $S1$ neuron, respectively. For on-the-fly convolution, the number of pixels involved in each $S1$ map is $|F_i|^2$, where $i = 3,5,\ldots 9$. Let $Int32Sub$ denote a 32-bit integer subtraction, $FP16AddSub$ denote a 16-bit floating point addition/subtraction and $FP16MULT$ denote a 16-bit floating point multiplication, the total number of operations per input event is given by:

$$\begin{aligned} &\sum_{i=1}^{4} |F_i|^2 \times (Int32Sub + FP16MULT + FP16AddSub \times 2) \\ &= 164 \times (Int32Sub + FP16MULT + FP16AddSub \times 2) \end{aligned} \tag{4.2}$$

4.1.2.2 Motion Symbol Detector

As described in the algorithm description section, the Motion Symbol Detector consists of one leaky integrate neuron and an extrema detection unit.

Each input event initiates a PSP to the leaky integrate neuron. The PSP has an exponential decay shape. From the kernel shape, one can see that it nearly reaches zero after a time of $5\tau_m$. This can be interpreted as one input event will have little contribution to the total potential after $5\tau_m$ from the time it is received. Therefore, only the events happened within the time range $[t - 5\tau_m, t]$ need to be cached,

where t denotes the current time. Let R denote the average input event rate, then the number of cached events is $5\tau_m R$. For the posture dataset, $\tau_m = 20ms$, and the event rate of these postures R is smaller than $30K$ events per second (eps).

Thus, for the time range of $\tau_m = 100ms$, there will be about $30Keps \times 0.1s = 3000$ events. This means that at most 3000 time-stamps need to be cached (the addresses of these events are not required for symbol detector). For the AER vision sensor, the time-stamp of each event is generated by a 32-bit counter (with $1\mu s$ time resolution) inside the sensor. Therefore, the memory requirement for event caching is $3000 \times 32 = 96Kbit$.

The PSP kernel is simply implemented as a lookup table, using the time difference between the current time and the event time-stamp as entry index. In order to get the total potential of the leaky integrate neuron, all the PSP kernels generated by the events that happened in the time range of $[t - 5\tau_m, t]$ need to accumulated (up to 3000 events as derived above). The total potential of the leaky integrate neurons does not need to be updated at a very high frequency. It is updated every $1ms$ (dt = $1ms$). The number of entries in the PSP kernel is about $5\tau_m/dt = 100ms/1ms = 100$. The value of each entry is represented as 16-bit floating point number (1 sign bit, 5 bits for exponent and 10 bits for mantissa). Thus the memory requirement for the PSP kernel is $100 \times 16bit = 1.6Kbit$.

The Motion Symbol Detector module has an extrema detection unit to locate the temporal peaks of the total potential of the leaky integrate neurons. The potential at time $(t - t_{SR}/2)$ is compared with other potentials within its temporal search range $[t - t_{SR}, t]$. If this potential shows a sign of decaying, then it is considered as an extrema. This extrema detection unit requires to store all the potentials within $[t - t_{SR}, t]$. For human posture recognition, this search range t_{SR} is usually no more than 1, thus $t_{SR}/dt = 1s/1ms = 1000$ potential values need to be cached in a RAM. Each potential is represented as a 16-bit floating point number. Therefore the memory requirements for the peak detection unit are $16bit \times 1000 = 16Kbit$.

For each time step, the computation of Motion Symbol Detector involves the following: (1) calculating the time difference between the current time and each event time-stamp (32-bit integer Subtraction, denoted as $Int32Sub$); (2) accumulating each PSP value to the total potential (16-bit floating

point Addition, $FP16Add$); (3) checking whether the potential is an extrema or not (16-bit floating point comparison, $FP16Comp$). The total number of computations for Motion symbol detector is 3000 $events \times (Int32Sub + FP16Add) + 1000\ potentials \times (FP16Comp)$.

4.1.2.3 LocMAX and Spike Generation

Once an extrema is detected by the Motion Symbol Detector, LocMAX competition will be conducted over local neighborhood in each $S1$ map to find the features. In order to perform LocMAX competition over local neighborhood, $|F_i| \times |F_i|$ $S1$ responses ($|F_i|$ denotes the kernel size, i.e., $3,5,7,9$) need to be read out. The maximum of these responses needs to be computed and finally the center is judged whether it is a MAX or not. The max is calculated column wise and the computed column MAXs are stored for later reuse. For current neuron, only five new data (of the new column) need to be read out and used to compute a column max, then the MAX of 5×5 local neighborhood is computed from five column max numbers, and finally the current neuron compares itself with the local MAX to decide to survive or die. Note that for the left boundary neurons, all the five columns of $S1$ responses still needs to be read. However, for the following ones (non left boundary), only one new column need to be read in. The number of operations ($FP16Comp$, 16-bit floating point comparison) involved in MAX competition are:

$$\sum_{i=1}^{4}[n|F_i|^2 + n(n-1)|F_i|] = [32 \times (3^2+5^2+7^2+9^2) + 32 \times 31 \times (3+5+7+9)] = 29056 \qquad (4.3)$$

The neurons that win the competition will be converted to spikes in a Time-to-First-Spike manner. The larger the response, the smaller the spike timing. $y = a - bx$, where y and x denote the spike timing and the original response respectively, a and b are two constant parameters. The generated spikes are normalized and rounded into the time range of $[0,256]$. Each spike also inherits the address of the corresponding winner neuron. As mentioned in the algorithm description part, the surviving neurons

are very sparse and 100 TFS neurons are sufficient. Each spike needs 32 bits (16-bit address and 16-bit time). Therefore the generated spikes require $32bit \times 100 = 3.2Kbit$ of storage. The number of operations for feature spike generation is $100 \times (FP16MULT + FP16Sub)$.

4.1.2.4 Classification Network

The generated spikes are first sent to weight LUT to fetch their corresponding weights (using their addresses as entry index), and then the spike timings and the obtained weights are fed to a neural network to get the final classification results.

The number of weights in weight LUT is N times (N denotes the number of categories, e.g., 3 for the posture dataset) of the number of all $S1$ neurons. Thus the number of weights equals $N \times (32 \times 32 \times 4) = 12K$. Each weight is represented as a 16-bit floating point number, so $12K \times 16bit = 192Kbit$ memory is needed to store all the weights.

4.2 FPGA Hardware Software Co-processing

The proposed AER recognition system involves a lot of parallel computations, which lags when running on personal computer (PC). To achieve realtime posture recognition, the system has to be accelerated by implementing the algorithm into hardware, such as FPGA or application specific integrated circuits (ASIC). Compared to ASIC, FPGA is more flexible and convenient for rapid prototyping. FPGA can provide many parallel resources to expedite the computational speed. However, it is more difficult for sophisticated control as compared to software. To sum up, software (i.e. processing by general purpose CPU) is easy for complex control but slow for parallel computation, while hardware (i.e. FPGA or ASIC) is difficult to implement complicated control procedures but it can provide fast parallel computation speed. Therefore, by combining the advantages of both software and hardware, the proposed recognition system is implemented using a FPGA-based software-hardware co-processing architecture, using hardware to handle computation intensive tasks and using software to control the procedure.

4.2.1 Hardware software co-processing architecture

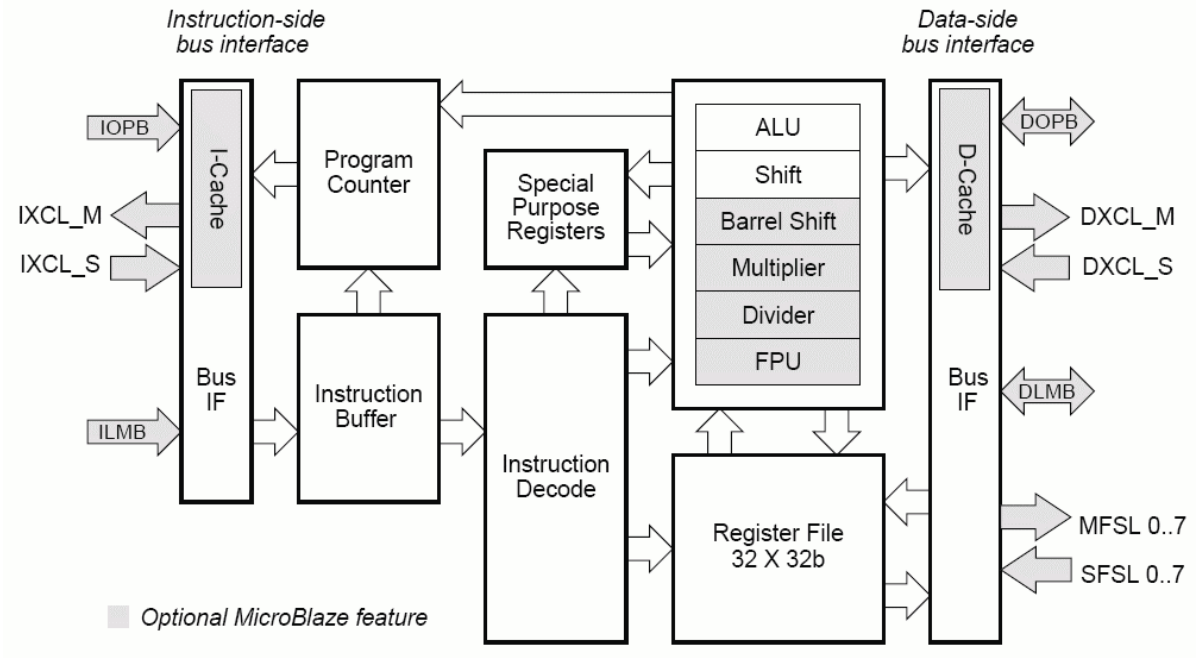

Figure 4.2: Block diagram of Xilinx MicroBlaze soft processor core [133]

Inside Xilinx FPGA, an embedded system can be built using the MicroBlaze soft processor provided by Xilinx. MicroBlaze is a 32-bit reduced instruction set computing (RISC) soft processor IP core. As a soft processor, it can be configured by user to choose or trim optional features and implemented entirely in general purpose memory and logic resource of FPGAs. Figure 4.3 shows the block diagram of the MicroBlaze core. MicroBlaze has separate instruction and data buses (Harvard architecture). It has bus interfaces to two buses, namely Local Memory Bus (LMB) and Processor Local Bus (PLB). The dedicated memory accessing bus LMB reduces loading on PLB. Various types of peripherals can be attached to PLB to build an embedded system. Figure 4.3 shows an example of MicroBlaze-based embedded system. The MicroBlaze core accesses the block RAM-based instruction and data memory through LMB bus, and it can have various peripherals connected through the PLB bus, such as universal asynchronous receiver/transmitter (UART), inter-integrated circuit (I2C), Ethernet, general purpose input/output (GPIO), etc.

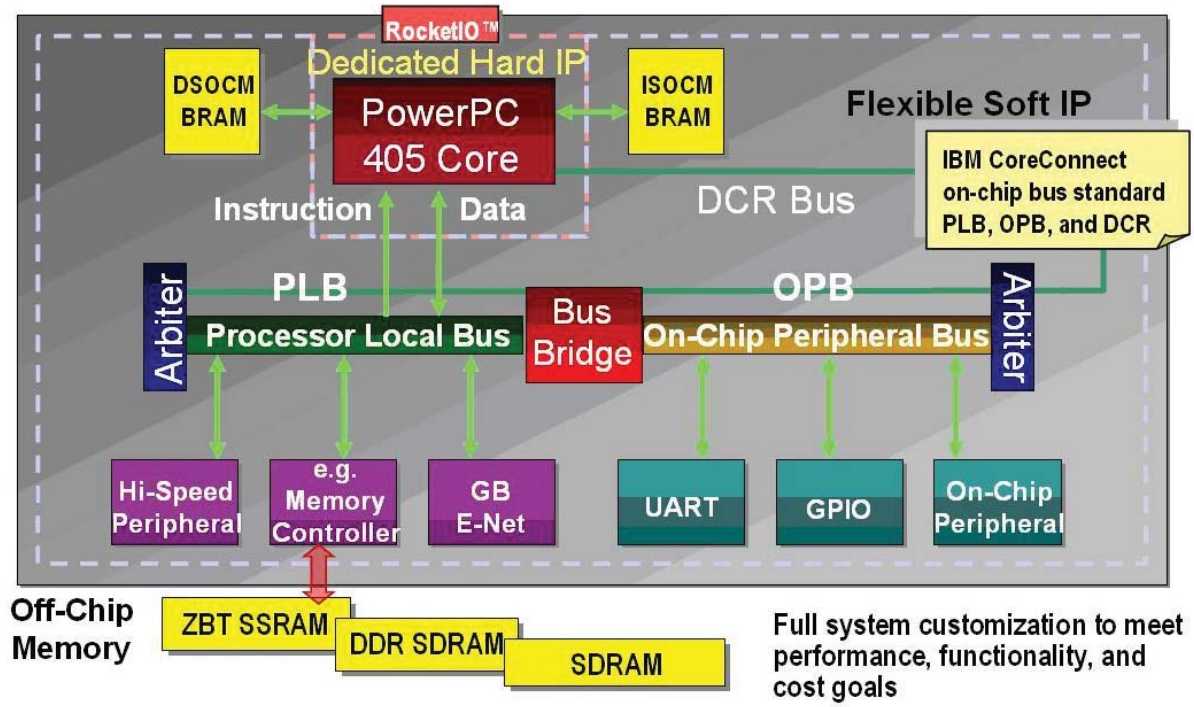

Figure 4.3: MicroBlaze embedded system design [133]

Memory devices outside (or inside) the FPGA are used to store the events and intermediate data during processing. The FPGA is connected to PC through universal serial bus (USB) micro controller and high speed USB 2.0 cable.

4.2.2 Convolution Core Implementation on FPGA Development Board

The FPGA development board adopted in this design is Digilent Nexys 3 Spartan-6 FPGA board. The FPGA board features a Xilinx Spartan 6 FPGA (XC6LX16-CS324), a 16MB cellular RAM, a 16MB parallel PCM and Digilent Adept USB port for power, programming and data transfer. This test implemented 4 convolution maps with this FPGA. The implementation result is shown in Table 4.1:

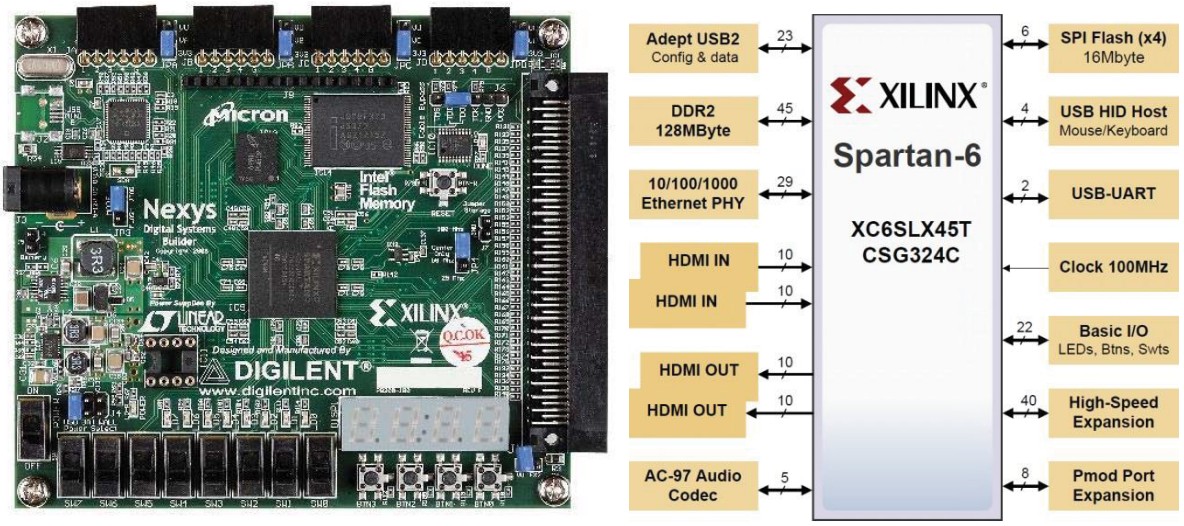

Figure 4.4: Digilent Nexys3 FPGA Dev Board. [134]

Table 4.1: Resource Summery of Convolution Core FPGA Implementation

Device Resource	Number of Utilization	Percentage
Number of Slice Registers	551 out of 18224	3%
Number of Slice LUTs	296 out of 9112	3%
Number of fully used LUT-FF pairs	206 out of 847	32%
Number of bonded IOBs	113 out of 232	48%
Specific Feature Utilization	1 out of 16	16%

4.3 Summary on Hardware Implementation

This chapter has presented a feedforward categorization system to process data from AER motion sensors. In order to fully utilize the power of asynchronous, sparse, event-driven representation of the sensor data, the concept of event-based processing is adopted at every stage. Feature is extracted by hierarchical maps of leaky integrate type neurons, inspired by a model of object categorization in the primate visual cortex. Due to the use of MAX operation, features are encoded into a limited number

of spikes. A virtually connected neuron network efficiently discriminates the spatiotemporal spike patterns. Two types of on-the-fly co-processing are also explored, namely motion symbol detector and centroid calculation. The overall system has been evaluated by extensive simulation. Promising results have been obtained.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This thesis presents an event-driven AER classification system with an extreme online learning speed. Sparse cortex-like, spike-based features are extracted by a neuromorphic MAX-like convolution network with leaky integration neurons. Those features are then classified by an online sequential regularized extreme learning machine with lookup table, resulting in a very fast training speed. Experimental results show that the proposed system decreases the training time enormously while still maintaining a competitive accuracy.

The Dynamic Vision Sensor can automatically remove still background in the scene through focal plane processing; it captures only the motion contour of moving object and generates a stream of binary motion events as output. This sensor avoids the transmission of the massive raw data, largely saving the channel bandwidth. In addition, since the outputs of this sensor are only some binary motion events, the privacy of the target person can thus be protected.

The motion events generated by the temporal difference image sensor are sent to the bio-inspired feature extraction unit, where each image is represented as a set of vectorial line segments. The proposed feature extraction approach is inspired by several feedforward models of primate visual cortex, such as

HMAX model and Serre's model. We use a bank of DoG filters (four scales) to model the $S1$ units which correspond to simplex cells in primary visual cortex. LocMAX operation is then performed on $S1$ layer maps to generate $C1$ maps, where the position, size and orientation of the line features can be easily extracted. One major difference between the proposed approach and previous feedforward models is the MAX operation. Previous models widens the maximum response in $S1$ maps to achieve blurred general features; while our approach suppresses non-maximum responses to obtain thinned line features.

Finally this thesis proposed a seamless combination of a bio-inspired event driven object feature extraction and recognition system. The system is tailored for frame-free asynchronous Address Event Representation (AER) vision sensor to receive and process the absolute binary event stream. The system fully utilizes the precise timing information in the output of DVS. The asynchronous nature of this system frees computation and communication from the adamant clock timing in typical systems. Neuromorphic processing is used to extract cortex-like spike-based features through an event-driven MAX-like convolution network. Every address event is sent to a batch of Difference of Gaussian (DoG) filters and convoluted in parallel. Modern neuron model, leaky and integration model is used to model dynamic membrane potential and fire status. Each neuron competes with others within its receptive field. The extracted spike patterns are then classified by an online sequential extreme learning machine with lookup table. Using a Lookup Table, the system can be made virtually fully connected by physically activating only a very small subset of the classification network. Experimental results prove that the proposed recognition system has a very fast training speed while still maintaining a competitive level of accuracy.

5.2 Future Work

The combination of event-based image sensors with bio-inspired object categorization system has not been fully implemented yet. We will study how to optimally encode data to remove undesired noise

image information and at the same time increase the efficiency of data-processing circuits. Future work includes research and development in the following areas:

- **To advance and improve the recognition algorithm.** At present, our recognition algorithm applies regularized online sequential ELM. Fast as it is, its classification ability is somehow limited. In the future, we might try implement multi-layer ELM to increase the model's ability to classify. In addition, new classification method such as deep learning can also be considered for evaluation.

- **To explore the possibility to track and recognize fast moving objects.** Since our sensor possesses the ability to capture object in high speed and the advantage of using less data. In future work, this character is able to be used to implement a system that can track objects in high speed motion might help applications like UAV or collision avoidance by a great deal.

- **To study event-based hardware architecture and implement the whole system.** Once the algorithm is fixed, the task will be to implement all parts into hardware, finally providing an energy-efficient real-time posture recognition solution for lightweight embedded platforms.

Publications

Conference Papers

(i) **Ruoxi Ding**, Bo Zhao and Shounshun Chen, "A Neuromorphic Categorization System with Online Sequential Extreme Learning," in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, EPFL, Lausanne, Switzerland, Oct 2014.

(ii) Bo Zhao, Qiang Yu, **Ruoxi Ding**, Shoushun Chen and Huajin Tang, "Event-Driven Simulation of the Tempotron Spiking Neuron," in *IEEE Biomedical Circuits and Systems Conference (BioCAS)*, EPFL, Lausanne, Switzerland, Oct 2014.

Journal Papers

(i) **Ruoxi Ding**, Bo Zhao, Shoushun Chen, Bernabe and Linares-Barranco, "An Event-Driven Classification System with Online Sequential Regularized Extreme Learning," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, submitted.

(ii) Bo Zhao, **Ruoxi Ding**, Shoushun Chen, Bernabe Linares-Barranco and Huajin Tang, "Feedforward Categorization on AER Motion Events Using Cortex-like Features in a Spiking Neural Network," *IEEE Transactions on Neural Networks and Learning Systems (TNNLS)*, 2014, vol.PP, no.99, pp.1,1

Bibliography

[1] Vision Research, "Phantom Miro 3 High Speed Camera." URL: http://www.visionresearch.com/blog/tag/phantom-miro-3/, 2014.

[2] A. Fish, L. Sudakov-Boresha, and O. Yadid-Pecht, "Low-power tracking image sensor based on biological models of attention," *International Journal on Information Theory and Applications*, vol. 14, no. 2, pp. 103–114, 2006.

[3] 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 Journal of Solid-State Circuits*, vol. 42, pp. 2978–2989, Dec 2007.

[4] P. Turaga, R. Chellappa, V. Subrahmanian, and O. Udrea, "Machine recognition of human activities: A survey," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 1473 –1488, nov. 2008.

[5] T. Serre, *Learning a Dictionary of Shape-Components in Visual Cortex: Comparison with Neurons, Humans and Machines*. PhD thesis, MIT, Apr. 2006.

[6] N. Robertson and I. Reid, "A general method for human activity recognition in video," *Computer Vision and Image Understanding*, vol. 104, no. 2-3, pp. 232 – 248, 2006. Special Issue on Modeling People: Vision-based understanding of a person's shape, appearance, movement and behaviour.

[7] X. Yuan and X. Yang, "A robust human action recognition system using single camera," in *International Conference on Computational Intelligence and Software Engineering. CiSE 2009.*, pp. 1 –4, dec. 2009.

[8] Z. Zhou, X. Chen, Y.-C. Chung, Z. He, T. Han, and J. Keller, "Activity analysis, summarization, and visualization for indoor human activity monitoring," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 1489 –1498, nov. 2008.

[9] X. Ji and H. Liu, "Advances in view-invariant human motion analysis: A review," *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, vol. 40, pp. 13 –24, jan. 2010.

[10] M. Yang, F. Lv, W. Xu, K. Yu, and Y. Gong, "Human action detection by boosting efficient motion features," in *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops)*, pp. 522 –529, oct. 2009.

[11] S. Ali and M. Shah, "Human action recognition in videos using kinematic features and multiple instance learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 288 –303, feb. 2010.

[12] M. Singh, A. Basu, and M. Mandal, "Human activity recognition based on silhouette directionality," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, pp. 1280 –1292, sept. 2008.

[13] H. Foroughi, B. Aski, and H. Pourreza, "Intelligent video surveillance for monitoring fall detection of elderly in home environments," in *11th International Conference on Computer and Information Technology (ICCIT 2008)*, pp. 219 –224, dec. 2008.

[14] N. Kiryati, T. Raviv, Y. Ivanchenko, and S. Rochel, "Real-time abnormal motion detection in surveillance video," in *19th International Conference on Pattern Recognition (ICPR 2008)*, pp. 1 –4, dec. 2008.

[15] A. Efros, A. Berg, G. Mori, and J. Malik, "Recognizing action at a distance," in *IEEE International Conference on Computer Vision (ICCV 2003)*, vol. 2, pp. 726 –733, oct. 2003.

[16] A. Nasution, P. Zhang, and S. Emmanuel, "Video surveillance for elderly monitoring and safety," in *2009 IEEE Region 10 Conference (TENCON 2009)*, pp. 1 –6, jan. 2009.

[17] E. Herrero-Jaraba, C. Orrite-Urunuela, F. Monzon, and D. Buldain, "Video-based human posture recognition," in *Proceedings of the 2004 IEEE International Conference on Computational Intelligence for Homeland Security and Personal Safety (CIHSPS 2004)*, pp. 19 – 22, july 2004.

[18] M. Ahad, J. Tan, H. Kim, and S. Ishikawa, "Human activity recognition: Various paradigms," in *International Conference on Control, Automation and Systems. ICCAS 2008.*, pp. 1896 –1901, oct. 2008.

[19] Q.-C. Pham, L. Gond, J. Begard, N. Allezard, and P. Sayd, "Real-time posture analysis in a crowd using thermal imaging," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR '07)*, pp. 1 –8, june 2007.

[20] J. Han and B. Bhanu, "Human activity recognition in thermal infrared imagery," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops. CVPR Workshops 2005.*, p. 17, june 2005.

[21] Flir News, "Infrared technology flir news." URL: http://www.flirnews.com/tag/infrared-technology, 2014.

[22] MESA Imaging, "Swissranger sr4000." URL: https://acroname.com/products/MESA-IMAGING-SR4000/, 2014.

[23] Microsoft Singapore, "Kinect for Xbox 360." URL: http://www.xbox.com/en-sg/Kinect, 2014.

[24] G. Diraco, A. Leone, and P. Siciliano, "An active vision system for fall detection and posture recognition in elderly healthcare," in *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE '10, pp. 1536–1541, 2010.

[25] Y. Zhu, B. Dariush, and K. Fujimura, "Controlled human pose estimation from depth image streams," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. CVPRW '08.*, pp. 1 –8, june 2008.

[26] V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun, "Real time motion capture using a single time-of-flight camera," in *CVPR '10*, pp. 755–762, 2010.

[27] M. Amoretti, F. Wientapper, F. Furfari, S. Lenzi, and S. Chessa, "Sensor data fusion for activity monitoring in ambient assisted living environments," in *First International ICST Conference on Sensor Systems and Software, S-CUBE 2009.*, sept. 2009.

[28] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from a single depth image," in *CVPR '11*, june 2011.

[29] Texas Instruments, "3D Time of Flight Imaging Solutions." URL: http://www.ti.com/ww/en/analog/3dtof/, 2015.

[30] J. A. Lenero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6μs asynchronous frame-free event-driven dynamic-vision-sensor," *IEEE Journal of Solid-State Circuits*, vol. 46, pp. 1443–1455, june 2011.

[31] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 $\times$ 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 43, pp. 566–576, Feb. 2008.

[32] 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 Journal of Solid-State Circuits*, vol. 38, pp. 1072–1075, Jun 2003.

[33] M. Gottardi, N. Massari, and S. Jawed, "A 100 μW 128 $\times$ 64 pixels contrast-based asynchronous binary vision sensor for sensor networks applications," *IEEE Journal of Solid-State Circuits*, vol. 44, pp. 1582–1592, May 2009.

[34] S. Chen, W. Tang, and E. Culurciello, "A 64 $\times$ 64 pixels UWB wireless temporal-difference digital image sensor," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1404–1407, Jun 2010.

[35] M. Litzenberger, C. Posch, D. Bauer, A. Belbachir, P. Schon, B. Kohn, and H. Garn, "Embedded vision system for real-time object tracking using an asynchronous transient vision sensor," in *Digital Signal Processing Workshop*, pp. 173–178, Sep 2006.

[36] M. Litzenberger, A. Belbachir, N. Donath, G. Gritsch, H. Garn, B. Kohn, C. Posch, and S. Schraml, "Estimation of vehicle speed based on asynchronous data from a silicon retina optical sensor," in *IEEE Intelligent Transportation Systems Conference (ITSC)*, pp. 653–658, Sep 2006.

[37] 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. Ballcels, 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 Transactions on Neural Networks*, vol. 20, pp. 1417 –1438, sept. 2009.

[38] J. Leero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 μs latency asynchronous frame-free event-driven dynamic-vision-sensor," *Solid-State Circuits, IEEE Journal of*, vol. 46, pp. 1443–1455, Jun 2011.

[39] I. Biederman, "Human image understanding: Recent research and a theory," *Computer Vision, Graphics, and Image Processing*, pp. 29–73, 1985.

[40] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, pp. 11–32, 1991.

[41] K. U. Barthel and I. Medieninformatik, "Color histogram 3d." URL: http://rsbweb.nih.gov/ij/plugins/color-inspector.html.

[42] S. Wang, "A robust cbir approach using local color histograms," Master's thesis, University of Alberta, 2001.

[43] X.-Y. Wang, J.-F. Wu, and H.-Y. Yang, "Robust image retrieval based on color histogram of local feature regions," *Multimedia Tools Appl.*, vol. 49, pp. 323–345, August 2010.

[44] J. Han and K.-K. Ma, "Fuzzy color histogram and its use in color image retrieval," *IEEE Transactions on Image Processing*, vol. 11, no. 8, pp. 944–952, 2002.

[45] J. Huang, S. Kumar, M. Mitra, W.-J. Zhu, and R. Zabih, "Image indexing using color correlograms," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR '97)*, pp. 762 –768, jun 1997.

[46] Q. Zhao and H. Tao, "Object tracking using color correlogram," in *Proceedings of the 14th International Conference on Computer Communications and Networks*, (Washington, DC, USA), pp. 263–270, IEEE Computer Society, 2005.

[47] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 3, pp. 610 –621, nov. 1973.

[48] K. Laws, "Rapid texture identification," in *Proceedings of the Seminar on Image Processing for missile guidance*, pp. 376–380, 1980.

[49] B. Manjunath and W. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 837 –842, aug 1996.

[50] R. S. Choras, "Image feature extraction techniques and their applications for CBIR and biometrics systems," *International Journal of Biology and Biomedical Engineering*, vol. 7, pp. 6–16, 2007.

[51] C.-L. Liu, M. Koga, and H. Fujisawa, "Gabor feature extraction for character recognition: comparison with gradient feature," in *Proceedings. Eighth International Conference on Document Analysis and Recognition (ICDAR '05)*, pp. 121 – 125 Vol. 1, 2005.

[52] R. Ramanathan, A. Nair, L. Thaneshwaran, S. Ponmathavan, N. Valliappan, and K. Soman, "Robust feature extraction technique for optical character recognition," in *International Conference on Advances in Computing, Control, Telecommunication Technologies*, pp. 573 –575, dec. 2009.

[53] Open Source Graphics, "Visualizing the 3D point of cloud of RGB colors." URL: http://opensource.graphics/visualizing-the-3d-point-cloud-of-rgb-colors/, 2015.

[54] S. Newsam and C. Kamath, "Comparing shape and texture features for pattern recognition in simulation data," in *SPIE's Annual Symposium on Electronic Imaging*, (San Jose, CA), Jan 2005.

[55] Y. Mingqiang, K. Kidiyo, and R. Joseph, "A survey of shape feature extraction techniques," in *Pattern Recognition Techniques, Technology and Applications*, pp. 43–90, 2008.

[56] E. Persoon and K.-S. Fu, "Shape discrimination using fourier descriptors," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 7, pp. 170 –179, march 1977.

[57] E. Persoon and K.-S. Fu, "Shape discrimination using fourier descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 388 –397, may 1986.

[58] D. Zhang and G. Lu, "A comparative study on shape retrieval using fourier descriptors with different shape signatures," *Journal of Visual Communication and Image Representation*, no. 14 (1), pp. 41–60, 2003.

[59] J.-S. Hu, T.-M. Su, and P.-C. Lin, "3-d human posture recognition system using 2-d shape features," in *2007 IEEE International Conference on Robotics and Automation*, pp. 3933 –3938, april 2007.

[60] J. Ricard, D. Coeurjolly, and A. Baskurt, "Generalization of angular radial transform," in *International Conference on Image Processing*, pp. 2211–2214, 2004.

[61] J. Ricard, D. Coeurjolly, and A. Baskurt, "Generalizations of angular radial transform for 2d and 3d shape retrieval," *Pattern Recogn. Lett.*, vol. 26, pp. 2174–2186, October 2005.

[62] H. Fujiyoshi and A. Lipton, "Real-time human motion analysis by image skeletonization," in *Proceedings., Fourth IEEE Workshop on Applications of Computer Vision*, pp. 15 –21, oct 1998.

[63] C.-H. Chuang, J.-W. Hsieh, L.-W. Tsai, and K.-C. Fan, "Human action recognition using star templates and delaunay triangulation," in *International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIHMSP '08)*, pp. 179 –182, aug. 2008.

[64] E. Yu and J. Aggarwal, "Human action recognition with extremities as semantic posture representation," in *CVPR Workshops 2009*, pp. 1–8, june 2009.

[65] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, pp. 509–522, apr 2002.

[66] P. Guo and Z. Miao, "A home environment posture and behavior recognition system," in *International Conference on Convergence Information Technology (ICCIT '07)*, pp. 175 –180, nov. 2007.

[67] M. H. Galuberman, "Character recognition for business machines," *Electornics*, pp. 132–136, Feb. 1956.

[68] P. Guo and Z. Miao, "Projection histogram based human posture recognition," in *2006 8th International Conference on Signal Processing*, vol. 2, 16-20 2006.

[69] O. D. Trier, A. K. Jain, and T. Taxt, "Feature extraction methods for character recognition - a survey," *Pattern Recognition*, vol. 29, no. 4, pp. 641–662, 1996.

[70] G. Vamvakas, B. Gatos, S. Petridis, and N. Stamatopoulos, "An efficient feature extraction and dimensionality reduction scheme for isolated greek handwritten character recognition," in *Ninth International Conference on Document Analysis and Recognition (ICDAR '07)*, vol. 2, pp. 1073 –1077, sept. 2007.

[71] M. Blumenstein, B. Verma, and H. Basli, "A novel feature extraction technique for the recognition of segmented handwritten characters," in *Proceedings of Seventh International Conference on Document Analysis and Recognition*, vol. 1 of *ICDAR'03*, pp. 137–141, aug. 2003.

[72] S. Rajashekararadhya and P. Vanaja Ranjan, "Zone-based hybrid feature extraction algorithm for handwritten numeral recognition of two popular indian scripts," in *2009 World Congress on Nature Biologically Inspired Computing (NaBIC '09)*, pp. 526–530, dec. 2009.

[73] G. Bo, D. Caviglia, and M. Valle, "An analog vlsi implementation of a feature extractor for real time optical character recognition," *IEEE Journal of Solid-State Circuits*, vol. 33, pp. 556–564, apr 1998.

[74] F. Niu and M. Abdel-Mottaleb, "View-invariant human activity recognition based on shape and motion features," in *Proceedings of the IEEE Sixth International Symposium on Multimedia Software Engineering*, ISMSE '04, pp. 546–556, 2004.

[75] M. Turk and A. Pentland, "Face recognition using eigenfaces," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '91)*, pp. 586–591, jun 1991.

[76] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, vol. 3, no. 1, pp. 71–86, 1991.

[77] A. F. Bobick and J. W. Davis, "The recognition of human movement using temporal templates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, pp. 257–267, March 2001.

[78] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.

[79] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt, "Performance of optical flow techniques," in *1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '92)*, pp. 236 –242, jun 1992.

[80] W. Yang, Y. Wang, and G. Mori, "Human action recognition from a single clip per action," in *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV Workshops '09)*, pp. 482 –489, Oct 2009.

[81] J. Y. Angela, M. A. Giese, and T. A. Poggio, "Biophysiologically plausible implementations of the maximum operation," *Neural Computation*, vol. 14, no. 12, pp. 2857–2881, 2002.

[82] T. Serre, *Learning a Dictionary of Shape-Components in Visual Cortex: Comparison with Neurons, Humans and Machines*. PhD thesis, MIT, Apr. 2006.

[83] "Cerebral cortex, wikipedia." URL: http://en.wikipedia.org/wiki/Cerebral_cortex, 2011.

[84] "Visual cortex, wikipedia." URL: http://en.wikipedia.org/wiki/Visual_cortex, 2011.

[85] L. G. Ungerleider and J. V. Haxby, "'what' and 'where' in the human brain," *Current Opinion in Neurobiology*, pp. 157–65, 1994.

[86] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architectures in cats visual cortex," *Journal of Physiology*, vol. 160, p. 106154, 1962.

[87] D. H. Hubel and T. N. Wiesel, "Sequence regularity and geometry of orientation columns in the monkey striate cortex," *J. Comp. Neur.*, vol. 158, pp. 267–294, 1974.

[88] C. G. Gross, C. E. Rocha-Miranda, and D. B. Bender, "Visual properties of neurons in the inferotemporal cortex of the macaque," *J. Neurophysiol.*, no. 35, pp. 96–111, 1972.

[89] K. Tanaka, "Inferotemporal cortex and object vision," *Annu. Rev. Neurosci.*, no. 19, pp. 109–139, 1996.

[90] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 3, pp. 411–426, 2007.

[91] M. Riesenhuber and T. Poggio, "Computational models of object recognition in cortex: A review," tech. rep., Massachusetts Institute of Technology, sept. 2003.

[92] N. K. Logothetis, J. Pauls, and T. Poggio, "Shape representation in the inferior temporal cortex of monkeys," *Current Biology*, vol. 5, pp. 552–563, 1995.

[93] S. Thorpe, D. Fize, and C. Marlot, "Speed of processing in the human visual system," *Nature*, vol. 381, pp. 520–522, june 1996.

[94] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological Cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[95] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, and L. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural Comp.*, vol. 1, no. 4, pp. 541–551, 1989.

[96] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, pp. 2278–2324, Nov 1998.

[97] Y. LeCun, K. Kavukcuoglu, and C. Farabet, "Convolutional networks and applications in vision," in *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS'10)*, pp. 253 –256, june 2010.

[98] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.

[99] T. Serre, L. Wolf, and T. Poggio, "Object recognition with features inspired by visual cortex," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'05)*, pp. 994–1000, june 2005.

[100] P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, pp. 21–27, 1967.

[101] L. Fausett, *Fundamentals of Neural Networks*. New York: Prentice Hall, 1994.

[102] StatSoft, "Neural networks." http://www.statsoft.com/textbook/neural-networks/.

[103] V. N. Vapnik, *Statistical Learning Theory*. Wiley-Interscience, Sept. 1998.

[104] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst. Man Cybern, B Cybern.*, vol. 42, pp. 513–529, April 2012.

[105] E. Cambria, G.-B. Huang, L. L. C. Kasun, H. Zhou, C. M. Vong, J. Lin, J. Yin, Z. Cai, Q. Liu, K. Li, V. C. Leung, L. Feng, Y.-S. Ong, M.-H. Lim, A. Akusok, A. Lendasse, F. Corona, R. Nian, Y. Miche, P. Gastaldo, R. Zunino, S. Decherchi, X. Yang, K. Mao, B.-S. Oh, J. Jeon, K.-A. Toh, A. B. J. Teoh, J. Kim, H. Yu, Y. Chen, and J. Liu, "Extreme learning machines [trends controversies]," *Intelligent Systems, IEEE*, vol. 28, pp. 30–59, Nov 2013.

[106] R. Minhas, A. Mohammed, and Q. Wu, "Incremental learning in human action recognition based on snippets," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, pp. 1529–1541, Nov 2012.

[107] StatSoft, "Classification trees." http://www.statsoft.com/textbook/classification-trees/.

[108] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neuroscience*, vol. 2, pp. 1019–1025, 1999.

[109] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio, "Robust object recognition with cortex-like mechanisms," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 29, no. 3, pp. 411–426, 2007.

[110] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Networks*, vol. 17, pp. 1411–1423, 2006.

[111] X. Zhang and S. Chen, "A hybrid-readout and dynamic-resolution motion detection image sensor for object tracking," in *2012 IEEE International Symposium on Circuits and Systems*, pp. 1628–1631, May 2012.

[112] T. D. Brndli Christian, Luca Longinotti, "jAER Open Source Project." http://sourceforge.net/p/jaer/wiki/Home/, 2014.

[113] T. Serrano-Gotarredona and B. Linares-Barranco, "MNIST-DVS Database." http://www2.imse-cnm.csic.es/caviar/MNISTDVS.html, 2014.

[114] S. Chen, P. Akselrod, B. Zhao, J. Perez-Carrasco, B. Linares-Barranco, and E. Culurciello, "Efficient feedforward categorization of objects and human postures with address-event image sensors," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 34, pp. 302–314, Feb. 2012.

[115] B. Zhao and S. Chen, "Realtime feature extraction using max-like convolutional network for human posture recognition," in *IEEE ISCAS'11*, pp. 2673–2676, May 2011.

[116] S. Chen, P. Akselrod, and E. Culurciello, "A biologically inspired system for human posture recognition," in *2009 IEEE Biomedical Circuits and Systems Conference*, pp. 113–116, Nov. 2009.

[117] X. Zhang and S. Chen, "Live demonstration: A high-speed-pass asynchronous motion detection sensor," *Circuits and Systems (ISCAS), 2013 IEEE International Symposium on*, pp. 671–671, 2013.

[118] Q. Yu, H. Tang, K. Tan, and H. Li, "Rapid feedforward computation by temporal encoding and learning with spiking neurons," *Neural Networks and Learning Systems, IEEE Transactions on*, vol. 24, pp. 1539–1552, Oct 2013.

[119] C. Enroth-Cugell and J. G. Robson, "The contrast sensitivity of retinal ganglion cells of the cat," *The Journal of physiology*, vol. 187, no. 3, pp. 517–552, 1966.

[120] M. J. McMahon, O. S. Packer, and D. M. Dacey, "The classical receptive field surround of primate parasol ganglion cells is mediated primarily by a non-gabaergic pathway," *The Journal of neuroscience*, vol. 24, no. 15, pp. 3736–3745, 2004.

[121] Q. Yu, H. Tang, K. C. Tan, and H. Li, "Rapid feedforward computation by temporal encoding and learning with spiking neurons," *IEEE Trans. Neural Netw. Learning Syst*, in press 2013.

[122] I. Lampl, D. Ferster, T. Poggio, and M. Riesenhuber, "Intracellular measurements of spatial integration and the max operation in complex cells of the cat primary visual cortex," *Journal of neurophysiology*, vol. 92, no. 5, pp. 2704–2713, 2004.

[123] S. J. Luck, L. Chelazzi, S. A. Hillyard, and R. Desimone, "Neural mechanisms of spatial selective attention in areas v1, v2, and v4 of macaque visual cortex," *Journal of neurophysiology*, vol. 77, no. 1, pp. 24–42, 1997.

[124] B. Zhao, Q. Yu, H. Yu, S. Chen, and H. Tang, "A bio-inspired feedforward system for categorization of AER motion events," in *IEEE BioCAS*, pp. 9–12, Oct 2013.

[125] S. Raspopovic, M. Capogrosso, J. Badia, X. Navarro, and S. Micera, "Experimental validation of a hybrid computational model for selective stimulation using transverse intrafascicular multichannel electrodes," *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 20, pp. 395–404, May 2012.

[126] Y. Peng, S. Wang, X. Long, and B.-L. Lu, "Discriminative graph regularized extreme learning machine and its application to face recognition," *Neurocomputing*, vol. 149, pp. 340–353, 2015.

[127] B. Zhao, R. Ding, S. Chen, B. Linares-Barranco, and H. Tang, "Feedforward categorization on aer motion events using cortex-like features in a spiking neural network," *IEEE Transaction on Neural Networks and Learning Systems*, 2014.

[128] R. Ding, B. Zhao, and S. Chen, "Proposed algorithm matlab webpage." URL:http://1drv.ms/1z8xr3C, 2014.

[129] T. Serrano-Gotarredona and B. Linares-Barranco, "A 128 $\times$ 128 1.5% contrast sensitivity 0.9% fpn 3 μs latency 4 mw asynchronous frame-free dynamic vision sensor using transimpedance preamplifiers," *IEEE Journal of Solid-State Circuits*, vol. 48, pp. 827–838, Mar. 2013.

[130] C. J. B. Yann LeCun, Corinna Cortes, "The mnist database of handwritten digits." http://yann.lecun.com/exdb/mnist/, 2014.

[131] T. Delbruck and P. Lichtsteiner, "Fast sensory motor control based on event-based hybrid neuromorphic-procedural system," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 845–848, May 2007.

[132] Z. Fu, T. Delbruck, P. Lichsteiner, and E. Culurciello, "An address-event fall detector for assisted living applications," *IEEE Transactions on Biomedical Circuits and Systems TBCAS*, vol. 2, pp. 88–96, Jun 2008.

[133] "Xilinx MicroBlaze." URL: http://www.xilinx.com/tools/microblaze.htm, 2008.

[134] "Digilent Nexys3." URL: http://www.digilentinc.com/, 2014.