Real-Time Recognition of Dynamic Hand Postures on a Neuromorphic System

Qian Liu, and Steve Furber, *Fellow, IEEE*

Abstract—

Keywords—spiking neural network (SNN), convolutional neural network (CNN), posture recognition, neuromorphic system.

I. Introduction

A Pattern or an object in a two-dimensional image can be described with four properties [1]: position, geometry (size, area and shape), colour and texture, and trajectory. Appearance-based methods are the most direct approaches to perform pattern recognition. The test image is compared with all the templates to find the best match on one particular or a combination of properties. However, the 2D projection of an object changes under various illuminations, viewing angles, relative positions and distances (sizes), so it is impossible to represent all appearances of an object in different conditions. Robust matching, such as edge matching [2], the divide-and-conquer approach [3], gradient matching [4], etc., and feature based methods [5], [6] are used to improve reliability, robustness and classification efficiency, However, to find a proper feature for a specific object still remains an open question and there is not any process as accurate, general and effective as the brain. It is not a new idea to turn to nature for inspiration. Riesenhuber and Poggio [7], e.g., presented a biologically-inspired model following the organisation of the visual cortex which has the ability to represent relative position- and scale-invariant features. Integrating a rich set of visual features became available using a feed-forward hierarchical pathway.

The biologically-inspired Dynamic Video Sensor (DVS) silicon retina [8] with its event-driven and redundancy-reducing style of computation is a good example such an approach to low-cost visual processing. With the DVS we take a step forward towards modelling the biological vision mechanisms of dynamic hand posture recognition on a neuromorphic system. SpiNNaker [9], as the back-end of the system, provides a flexible, event-driven mechanism for real-time simulation of SNNs, and is where the posture recogniser locates. With their instinctive temporal processing, SNNs have the advantage to deliver dynamic hand posture recognition.

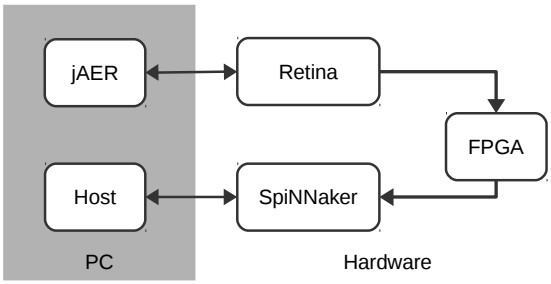
Nowadays, more and more attention has been drawn into the investigation of SNNs for vision processing. Pattern information can be encoded in the delays between the pre- and post-synaptic spikes since the spiking neurons are capable of computing radial basis functions (RBFs) [10]. A further study [11] has stated that spatio-temporal information can be also stored in the exact firing time instead of the relative delay. Maass [12] has proved mathematically that: 1) networks of spiking neurons are computationally more powerful than the first and second generation of neural network models; 2) a concrete biologically relevant function can be computed by a single spiking neuron, replacing hundreds of hidden units in a sigmoidal neural net; 3) any function that can be computed by a small sigmoidal neural net can also be computed by a small network of spiking neurons. Applications of SNN-based vision processing have been successfully carried out. A two-layered SNN has been trained using spike time dependent plasticity (STDP) and employed for a character recognition task [13]. Lee and co-authors [14] have implemented the direction selective filters in real time using spiking neurons. The direction selective filters here are considered as a layer of convolution module in the model of so called convolution neural network [15]. Different features, such as Gabor filter features (scale, orientation and frequency) and shape can be modelled as layers of feature maps. Rank order coding, as an alternative to conventional rate-based coding, treats the first spike the most important and has well applied to an orientation detection training process [16]. Nengo [17] is a graphical and scripting based software package for simulating large-scale neural systems and has been used to build the world's largest functional brain model, Spaun [18]. An FPGA implementation of a Nengo model for digit recognition has been reported [19]. Deep Belief Networks (DBNs), the 4th generation of artificial neural network, has shown a strong ability in solving classification problems. A recent study [20] has resoundingly mapped an offline-trained DBN onto an efficient event-driven spiking neural network for a digit recognition task.

Section II presents the details of the hardware of the neuromorphic system, including the silicon retina and the SpiNNaker machine. The neural network models are proposed and tested on Matlab, and the model structures and experiments results are stated in Section III. In the following section, the rate-based models are converted to spiking neurons, and real-time live recognition as well as experiments with recorded data are carried out. The work is summarised and the future work is planned in Section V.

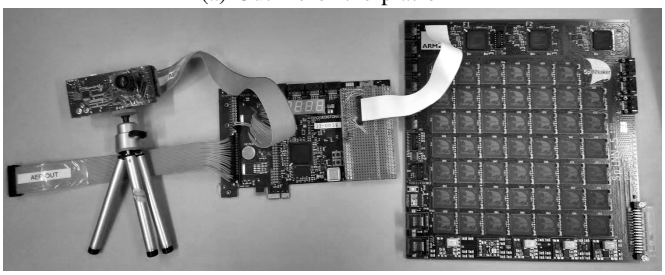
II. The Neuromorphic Platform

The system outline of the platform is illustrated in Figure 1a, where the hardware system is configured, controlled and monitored by the PC. The jAER [21] event-based processing software running on the PC configures the retina and displays the output spikes through a USB link. The host communicates to the SpiNNaker board via Ethernet to set up its runtime parameters and to download the neural network model off-line

The authors are with the School of Computer Science, University of Manchester, Manchester M13 9PL, U.K. (e-mail:qian.liu-3@manchester.ac.uk; steve.furber@manchester.ac.uk).



(a) Outline of the platform



(b) Picture of the hardware platform

Fig. 1: System overview of the dynamipc hand posture recognition platform. The silicon retina connects to the SpiNNaker system through an FPGA board. Spikes from the retina are streamed to the SpiNNaker system through this Spartan-6 FPGA board. The jAER software configures the retina and displays its outgoing spikes through the USB connection. The host sets up the runtime parameters off-line and downloads the network model to the SpiNNaker system.

and uses a visualiser [22] to show the spiking activities in real-time. From the picture of the hardware platform, Figure 1b, the silicon retina connects to the SpiNNaker 48-node board via a Spartan-6 FPGA board [23].

A. Silicon Retina

The visual input is captured by a DVS silicon retina, which is quite different from the conventional camera. A pixel only generates spikes to the connected neurons when its activity level reaches some threshold. Thus, instead of buffering video into frames, the activity of pixels is sent out and processed continually with time. So as the communication bandwidth is optimised by sending the activity only, which is encoded as events of address of pixels using address-event representation (AER [24]) protocol. The level of activity depends on the contrast change[4]; pixels generate spikes faster and more frequently when they are more active. The sensor is capable of capture very fast moving objects (up to 10 K rotations per second), which is equivalent to 100 K frames per second.

B. SpiNNaker System

The SpiNNaker project's architecture mimics the human brain's biological structure and functionality. This offers the possibility of utilizing massive parallelism and redundancy to provide resilience in an environment of unreliability and failure of individual components.

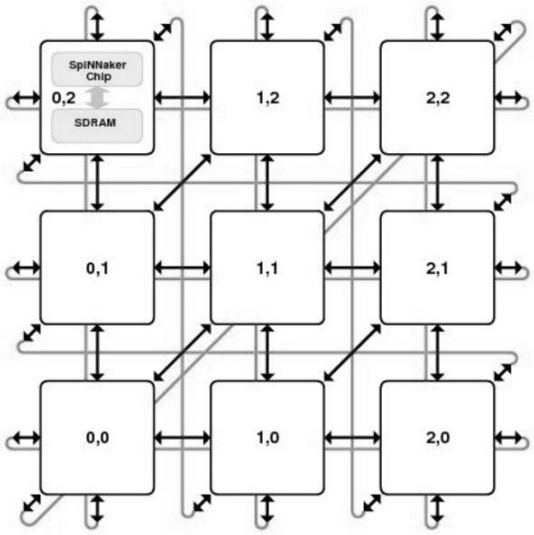


Fig. 2: System diagram. Each element represents one SpiNNaker chip with the local memory. Every chip connects to the other through the six bi-directional on-board links.

In the human brain, communication between its computing elements, or neurons, is achieved by the transmission of electrical 'spikes' along connecting axons. The biological processing of the neuron can be modelled by a digital processor and the axon connectivity can be represented by messages, or information packets, transmitted between a large number of processors which emulate the parallel operation of the billions of neurons comprising the brain.

The engineering of the SpiNNaker concept is illustrated in the Figure 2 where the hierarchy of components can be identified. Each element of the toroidal interconnection mesh is a multi-core processor known as the 'SpiNNaker Chip' comprising 18 processing cores. Each core is a complete processing sub-system with local memory and a DMA capability. It is connected to its local peers via a Network-on-Chip (NoC) which provides local high bandwidth communication and to other SpiNNaker chips via links between SpiNNaker chips. In this way the massive parallelism extending to thousands or millions of processors is possible.

The knowledge content and learning ability of the brain is embodied in its evolvable interconnection pattern; this routes a spike generated by one neuron to others which are interconnected with it by axons and these interconnections are modified and extended as a result of the learning and processes.

In SpiNNaker a packet Router within each multi-core processor controls the neural interconnection. Each transmitted packet representing a spike contains information which identifies its source neuron; this is used by a multi-core processor's Router to identify whether this packet should be routed to one of its contained application processors to respond, or should be routed on to one of the six adjacent multi-core processors connected to it as part of the overall SpiNNaker network.

The 103 machine is the 48-node board, see Figure 3,

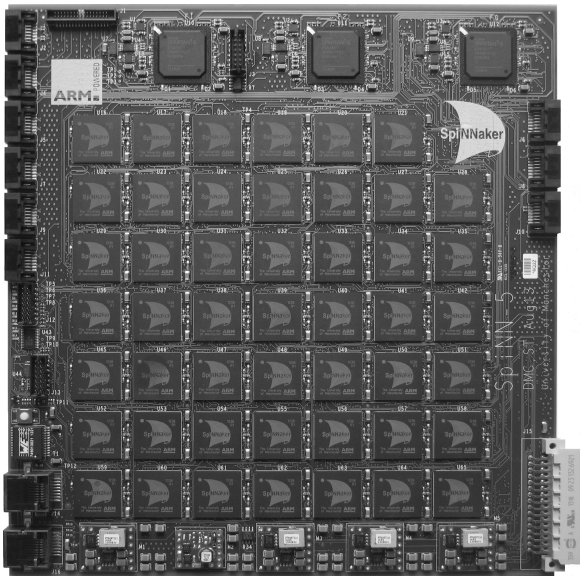


Fig. 3: 103 Machine PCB

and has 864 ARM processor cores, typically deployed as 768 application cores, 48 Monitor Processors and 48 spare cores. The 103 machine requires a 12V 6A supply. The control interface is two 100Mbps Ethernet connections, one for the Board Management Processor and the second for the SpiNNaker array. There are options to use the nine on-board 3.1Gbps high-speed serial interfaces (using SATA cables, but not necessarily the SATA protocol) for I/O; this will require suitable configuration of the on-board FPGAs that provide the high-speed serial interface support. 103 boards can be connected together to form larger systems using the high-speed serial interfaces.

C. Interfacing AER Sensors

Spikes from the silicon retina are injected to SpiNNaker through one of the six bi-directional on-board links by a SPARTAN-6 FPGA board that translates them into a SpiNNaker compatible AER format [25].

From the software point of view, interfacing the silicon retina can be done using pyNN. The retina is configured as a spike source population that resides on a virtual SpiNNaker chip, to which an AER sensor's spikes are directed, thus abstracting away the hardware details from the users[23].

III. Convolutional Neural Networks

The convolutional neural network (CNN) is well-known as an example of a biologically-inspired model. Figure 4 shows a typical convolutional connection between two layers of neurons. The repeated convolutional kernels are overlapped in the receptive fields of the input neurons.

A. Model Description

There are two CNNs proposed to accomplish the dynamic hand posture recognition task. A straight forward method of template matching was employed at first, and then a multi-layer perceptrons (MLP) was trained to improve the recognition performance.

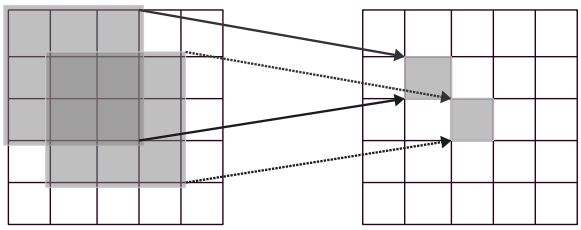


Fig. 4: Each individual neuron in the convolution layer (right matrix) connects to its receptive field using the same kernel. The value of the kernel can be seen as the synaptic weights between the connected neurons.

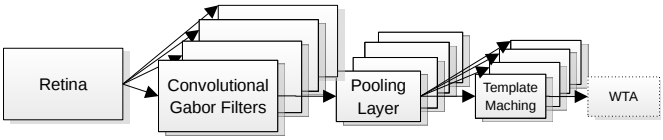


Fig. 5: Model 1. The retina input convolves with Gabor filters in the second layer, and then shrinks the sizes in the pooling layer. The templates are considered as convolution kernels in the last layer. The winner-take-all (WTA) circuit can be used as an option to show the template matching result more clearly.

1) Model 1. Template Matching: Shown in Figure 5 the first layer is the retina input, followed by the convolutional layer, where the kernels are Gabor filters responding to four orientations. The third layer is the pooling layer where the size of the populations shrinks. This down-sampling enables robust classification due to its tolerance to variations in the precise shape of the input. The fourth layer is another convolution layer where the output from the pooling layer is convolved with the templates. The optional layer of WTA neurons enables a clearer classification result due to the inhibition between the neurons. As to the Matlab simulation, the retina input spikes are buffered with 30 ms frames, and all the neurons are simple linear perceptrons. The templates are manually selected from the output of the pooling layer, see Figure 6.

Fig. 6: Templates of the five postures: 'Fist','Index Finger', 'Victory Sign', 'Full Hand' and 'Thumb up'.

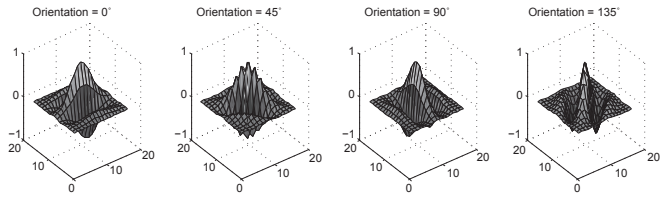


Fig. 7: Real parts of the Gabor filters orienting four directions.

$$\begin{aligned} \text{RealParts} &= \exp\left(\frac{-x'^2+y'^2}{2\sigma^2}\right)\cos\left(2\pi\frac{x'}{\lambda}\right) \\ \text{ImaginaryParts} &= \exp\left(\frac{-x'^2+y'^2}{2\sigma^2}\right)\sin\left(2\pi\frac{x'}{\lambda}\right) \end{aligned} \tag{1}$$

where :

$$x' = x\cos(\theta) + y\sin(\theta)$$

$$y' = -x\sin(\theta) + y\cos(\theta)$$

The Gabor filter is well-known as a linear filter for edge detection in image processing. A Gabor filter is a 2D convolution of a Gaussian kernel function and a sinusoidal plane wave; see Equation 1. θ represents the orientation of the filter, λ is the wavelength of the sine wave, and σ is the standard deviation of the Gaussian envelope. The frequency and orientation features are similar to the responses of V1 neurons in the human visual system. Only the real parts of the Gabor filters (see Figure 7) are used as the convolutional kernels to configure the weights between the input layer and the Gabor filter layer.

The output score of a convolution is determined by the matching degree between the input and the kernel. Regarding the template matching layer, one single neuron in a population responds to how closely its receptive field matches the specific template. And also, the position of moving gesture is naturally encoded in the address of template matching neuron. Thus, there are five populations of template matching neurons representing all the hand postures listed.

2) Model 2. Trained MLP: Inspired by the research of Lecun [26] , we designed a combined network model with MLP and CNN (Figure 8). The first three layers are exactly the same with Model 1. Since the training images for the 3-layered MLP is of same size and the posture is centred in the images. Therefore, a tracking layer plays an important role to finds the most active region and forwards the centred image to the next layer.

B. Experiments Set-up

In order to evaluate the cost and performance trade-offs in optimizing the number of neural components, both the convolutional models described above were tested at different sizes. Five videos of every posture were captured from the silicon retina in AER format. All the postures are of similar size and moving clock-wise in front of the retina. The videos are cut into frames (30 ms per frame) and push forward into the convolutional networks. The configurations of the networks are

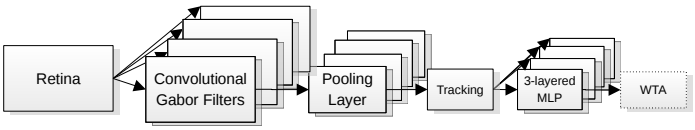


Fig. 8: Model 2. The retina input convolves with Gabor filters in the second layer, and then shrinks the sizes in the pooling layer. The following tracking layer finds the most active area of some fixed size, moves the posture to the centre and pushes the image to the trained MLP. The winner-take-all (WTA) layer can be used as an option to show the template matching result more clearly.

TABLE I: Sizes of the convolutional neural networks.

(a) Model 1: Template matching

	Full Resolution		**Sub-sampled Resolution**	
	Neuron Number	Connections per Neuron	Neuron Number	Connections per Neuron
Retinal Input	128 × 128	1	32 × 32	4 × 4
Gabor Filter	112×112×4	17 × 17	28×28×4	5 × 5
Pooling Layer	36×36×4	5 × 5	null	null
Integration Layer	36 × 36	4	28 × 28	4
Template Matching	16×16×5	21 × 21	14×14×5	15 × 15
Total	74320	15216512	5925	318420

(b) Model 2: Trained MLP

	Full Resolution		**Sub-sampled Resolution**	
	Neuron Number	Connections per Neuron	Neuron Number	Connections per Neuron
Tracked Input	21 × 21	null	15 × 15	null
Hidden Layer	10	21×21×10	10	15×15×10
Recognition Layer	5	5×10	5	5×10
Total	456	4460	240	2300

listed in Table I (Model 1: template matching; Model 2: trained MLP). The integration layer is not necessary in a convolutional network, it is used here to decrease the number of synaptic connections.

C. Experiment Results

In Figure 9 the first two plots refer to Model 1, using template matching. Each colour represents one of the recognition populations. Each point in the plot is the highest neuronal response in the recognition population during the time of one frame (30 ms). The neuronal response, 'the spiking rate', is normalised to [-1, 1]. It can be seen that the higher resolution input makes the boundaries between the classes clearer. On the other hand, recognition only happens when the test image and template are similar enough. The templates are only selected from the frames where the gestures are moving towards the right, and the gestures are moving clockwise in the videos.

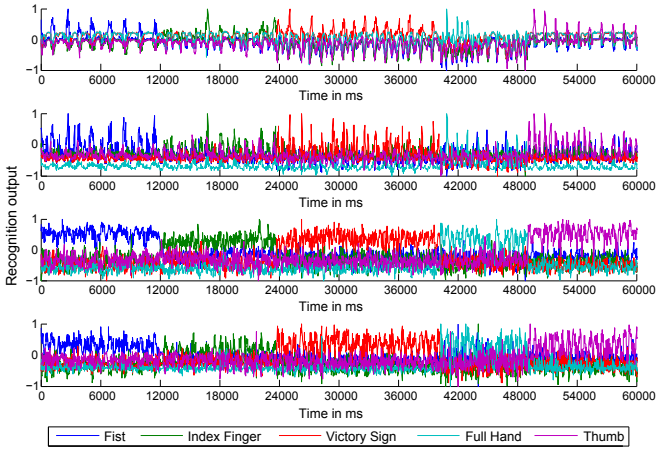


Fig. 9: Neural responses with time of four experiments to the same recorded moving postures. The recognition output is normalised to [-1, 1]. Every point represents the highest response in a specific population (different colour) for a 30 ms frame. The 1st plot refers to Model 1 with the full input resolution, and the 2nd plot Model 1 with the sub-sampled input resolution; and the 3rd and fourth plots both refer to Model 2, and with high and low input resolution respectively.

Thus, all the peaks in plot 1 signify that the direction of the gestures movement is right. It is notable that the higher resolution causes the recogniser to be more sensitive to the differences between the test data and the template, while the smaller neural network can recognize more generalized patterns. Therefore, a threshold is required to differentiate between data that is close enough and that which is not. Since the gestures are moving in four different directions during the clockwise movement, a rejection rate of 75% is to be expected.

The latter two plots refer to Model 2. The three-layer MLP network significantly improves the recognition rate and can generalize the pattern. There is no rejection rate for Model 2, since the MLP is trained with all the moving directions of the postures.

The detailed results are listed in Table II. The correct recognition rate is calculated from the non-rejected frames. The lower resolution of the 32×32 retina input is adequate for this gesture recognition task. The smaller network uses only 1/10 the number of neurons and 1/50 the number of synaptic connections compared to the full resolution network, while the recognition rate drops only around 10% with Model 1 and 15% with Model 2.

IV. Real-Time Recognition on SpiNNaker

A. Moving from Rate-based Artificial Neurons to Spiking Neurons

It remains a challenge to transfer a traditional artificial neural networks into the spiking ones. There are attempts [27] [28] to estimate the output firing rate of the Leaky integrate-and-fire (LIF) neurons (Equation 2) under certain conditions.

$$\frac{\mathrm{d}V(t)}{\mathrm{d}t} = -\frac{V(t)-V_{rest}}{\tau_m} + \frac{I(t)}{C_m} \tag{2}$$

TABLE II: Recognition results in %

		Model 1		Model 2	
		High Resolution	Low Resolution	High Resolution	Low Resolution
Fist	Correct	99.11	99.23	96.24	84.21
	Reject	71.93	67.42	Null	Null
Index Finger	Correct	92.98	80.00	94.39	71.69
(392 Frames)	Reject	70.92	75.77	Null	Null
Victory Sign	Correct	96.56	93.07	95.64	87.66
(551 Frames)	Reject	73.68	81.67	Null	Null
Full Hand	Correct	95.65	72.41	93.52	72.01
(293 Frames)	Reject	92.15	90.10	Null	Null
Thumb up	Correct	89.61	84.44	96.68	74.68
(391 Frames)	Reject	80.31	76.98	Null	Null

The membrane potential V evolves with the input current I starting from the resting membrane potential V_{rest}, where the membrane time constant $\tau_m = R_m C_m$, and R_m stands for the membrane resistance and C_m the membrane capacitance.

Given a fixed constant current injection I, the response function, firing rate, of the LIF neuron is

$$\lambda_{out} = \left[t_{ref} - \tau_m \ln\left(1 - \frac{V_{th}-V_{rest}}{IR_m}\right)\right]^{-1} \tag{3}$$

when $IR_m > V_{th} - V_{rest}$, otherwise the membrane potential cannot reach the threshold V_{th} and the output firing rate is zero. The absolute refractory period t_{ref} is included, for all the input during the period is invalid. In a more realistic scenario, the post-synaptic potentials (PSPs) are triggered by the spikes generated from the neuron's pre-synaptic neurons other than a constant current. Assume that the synaptic inputs are Poisson spike trains, the membrane potential of the LIF neuron is concerned as a diffusion process. Equation 2 can be modelled as a stochastic differential equation referring to Ornstein-Uhlenbeck process,

$$\tau_m \frac{\mathrm{d}V(t)}{\mathrm{d}t} = -[V(t)-V_{rest}] + \mu + \sigma\sqrt{2\tau_m}\xi(t) \tag{4}$$

where

$$\begin{aligned} \mu &= \tau_m(\mathbf{w_E}\cdot\lambda_\mathbf{E} - \mathbf{w_I}\cdot\lambda_\mathbf{I}) \\ \sigma^2 &= \tfrac{\tau_m}{2}\left(\mathbf{w_E^2}\cdot\lambda_\mathbf{E} + \mathbf{w_I^2}\cdot\lambda_\mathbf{I}\right) \end{aligned} \tag{5}$$

are the conditional mean and variance of the membrane potential. The delta-correlated process $\xi(t)$ is a Gaussian white noise with zero mean, $\mathbf{w_E} and \mathbf{w_I}$ stand for the weight vectors of the excitatory and the inhibitory synapses, and λ represents the vector of the input firing rate. The response function of the LIF neuron with Poisson input spike trains is given by Siegert function [29],

$$\lambda_{out} = \left(\tau_{ref} + \frac{\tau_Q}{\sigma_Q}\sqrt{\frac{\pi}{2}}\int_{V_{rest}}^{V_{th}} \mathrm{d}u \exp\left(\frac{u-\mu_Q}{\sqrt{2}\sigma_Q}\right)^2 \cdot\left[1+\mathrm{erf}\left(\frac{u-\mu_Q}{\sqrt{2}\sigma_Q}\right)\right]\right)^{-1} \tag{6}$$

where τ_Q, μ_Q, σ_Q are identified to τ_m, μ, σ in Equation 5, and erf is the error function.

Still there are some limitations on the response function. For the diffusion process, only small amplitude(weight) of the

PSPs generated by a large amount of input spikes(high spiking rate) works under this circumstances; plus, the delta function is required (the synaptic time constant is considered to be zero). Thus only a rough approximation of the output spike rate has been taken out. Secondly, given the input spike rates of the pre-synaptic neurons, the parameters of the LIF neuron and the output spiking rate, how to tune the every corresponding synaptic weight remains a hard task.

B. Live Recognition

We implemented the prototype of the dynamic posture recognition system on SpiNNaker with LIF neurons. The input retina layer consists of 128×128 neurons; each Gabor filter has 112×112 valid neurons, since the kernel size is 17×17; each pooling layer is as big as 36×36, convolving with five template kernels (21×21); thus, the recognition populations are 16×16 neurons each. Altogether 74 320 neurons and 15 216 512 synapses, see Table Ia , use up to 19 chips (290 cores) on a 48-node board. Regarding the lower resolution of 32×32 retinal input, see Table Ib, the network consists of 5 925 neurons and 318 420 synapses taking up only two chips (31 cores) of the board.

Figure 10 shows snapshots of neural responses of some populations during real-time recognition. Figure 10a is a snapshot of the Gabor population which prefers the horizontal direction, given the input posture of a 'Fist'; and Figure 10b shows the activity of the neurons in the integration layer, given a 'Victory Sign'. And the active neurons in the visualiser in Figure 10c are pointing out the position of the recognised pattern the 'Index finger'. All the videos can be found on Youtube [30], [31], [32].

C. Recognition of Recorded Data

To compare with the results of the experiments taken out in Section III-C with Matlab, the same recorded retinal data is conducted to SpiNNaker. The recorded data is presented as Spike Source Array in the system with 128×128 input, see Figure 12a, while the data is forwarded to a sub-sampling layer of 32×32 resolution , see Figure 13a, in the system of the smaller network. The output spikes generated from the recognition populations with time are shown in Figure 12 and 13 respectively for both the full resolution and the lower systems. More spikes generated during the period when the preferred input posture is shown.

Correspondingly, the spiking rates of each recognition population is sampled into frames (Figure 11) to make a comparison with the Matlab simulation. Each colour represents one recognition population, and the spike activity goes higher when the input posture matches the template. Firstly, the spike rates are sampled into 30 ms frames which is in accordance with the Matlab experiments. In the Matlab simulation, the templates are trained with cut frames and so as the test images are also fixed to the same length frames. Otherwise, the recogniser will not work properly because of the replications of the moving posture. On the contrast, the spiking rates can be sampled to various lengths of frames. Thus, the other two plots in the figure illustrate the classification in a wider

(a) Neural responses of the Gabor filter layer orienting to the horizontal direction [30]

(b) Neural responses of the integrate layer [31]

(c) Snapshot of the neuron responses of the template matching layer [32]

Fig. 10: Snapshots of the real-time dynamic posture recognition system on SpiNNaker.

TABLE III: Real-time recognition results on SpiNNaker in %

		30 ms per frame		300 ms per frame	
		High Resolution	Low Resolution	High Resolution	Low Resolution
Fist	Correct	91.78	78.02	100	92.31
	Reject	82.78	78.54	70.73	68.29
Index Finger	Correct	78.25	78.25	88.24	72.22
	Reject	80.46	73.56	57.50	55.00
Victory Sign	Correct	96.48	86.27	95.00	92.50
	Reject	64.46	72.68	28.57	28.57
Full Hand	Correct	85.29	60.78	90.00	75.00
	Reject	67.31	83.65	35.48	61.29
Thumb up	Correct	84.09	88.10	91.67	100
	Reject	87.54	73.81	66.67	66.67

window, 300 ms. From Table III, the recognition rates as well as the rejection rates are quantified in percentage. Regarding the latency between the retinal input and the recognition, we compared the spiking peak of the Matlab simulation and the real-time SpiNNaker test. The overall latency is about 1150 ms from a posture is shown to its recognition.

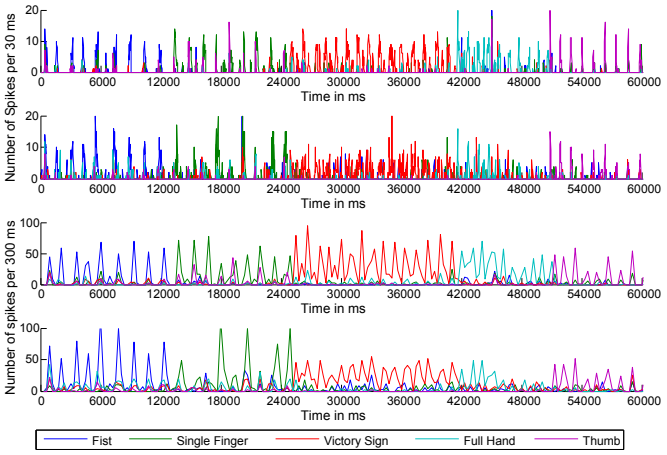
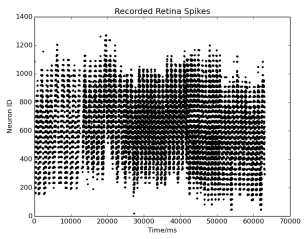
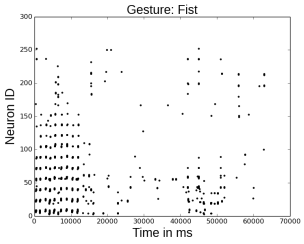


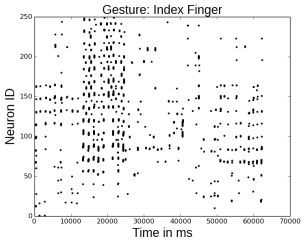
Fig. 11: Real-time neural responses of two experiments on SpiNNaker with time to the same recorded postures. These two experiments only differ on the input resolution. The result of the high input resolution test is plotted the first with a sample frame of 30 ms; while the 3rd plot shows the same result with a sample frame of 300 ms. The latter two plots refer to the smaller input resolution. Every point represents the over all number of spikes of a specific population (different colour) in a 'frame'.



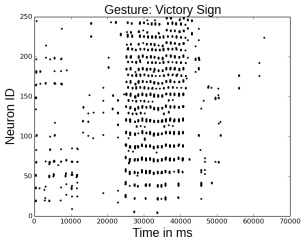
(a) Retinal input population



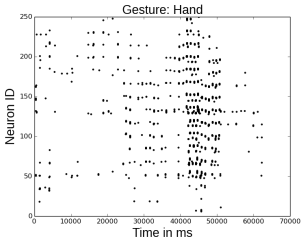
(b) Template matching population, 'Fist'



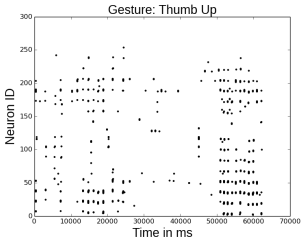
(c) Template matching population, 'Index Finger'



(d) Template matching population, 'Victory Sign'

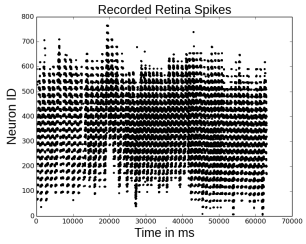


(e) Template matching population, 'Full Hand'

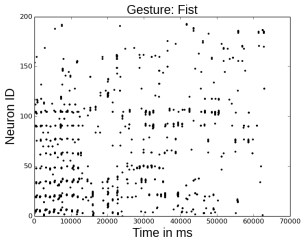


(f) Template matching population, 'Thumb Up'

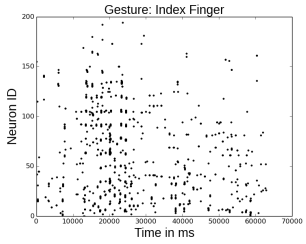
Fig. 12: Spikes captured during the live recognition of the recorded retinal input with the resolution of 128×128.



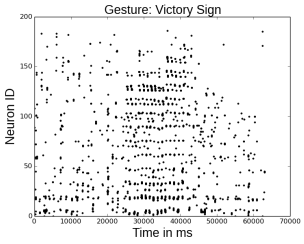
(a) Retinal input population



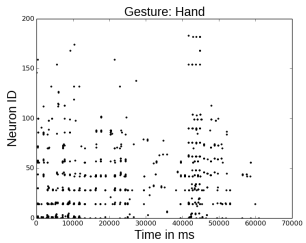
(b) Template matching population, 'Fist'



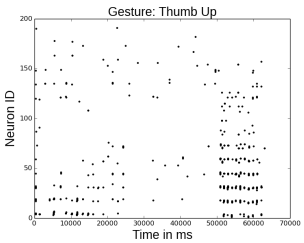
(c) Template matching population, 'Index Finger'



(d) Template matching population, 'Victory Sign'



(e) Template matching population, 'Full Hand'



(f) Template matching population, 'Thumb Up'

Fig. 13: Spikes captured during the live recognition of the recorded retinal input with the resolution of 32×32.

V. Conclusion and Future Work

The future work will include more collaboration with biology and work with neuroscientist on vision systems, especially on the orientation detection region. To equip the system with tracking is another important job where the recognition performance will be increased and the short-term memory of a gesture route can be stored. Using the idea of HMMs [33] to spiking neural networks may be a good approach.

VI. Acknowledgement

This research was supported by Samsung under their GRO programme. The authors appreciate the collaboration with Prof. Bernabé Linares-Barranco and Luis Camunas-Mesa on the silicon retina. The authors would like to thank the meaningful discussions with Evangelos Stromatias, Patrick Camilleri and Michael Hopkins.

References

[1] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Fast and adaptive network of spiking neurons for multi-view visual pattern recognition," *Neurocomputing*, vol. 71, no. 13, pp. 2563–2575, 2008.

[2] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, no. 6, pp. 679–698, 1986.

[3] Ö. Toygar and A. Acan, "Multiple classifier implementation of a divide-and-conquer approach using appearance-based statistical methods for face recognition," *Pattern Recognition Letters*, vol. 25, no. 12, pp. 1421–1430, 2004.
[4] S.-D. Wei and S.-H. Lai, "Robust and efficient image alignment based on relative gradient matching," *Image Processing, IEEE Transactions on*, vol. 15, no. 10, pp. 2936–2943, 2006.
[5] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.
[6] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-up robust features (surf)," *Computer vision and image understanding*, vol. 110, no. 3, pp. 346–359, 2008.
[7] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature neuroscience*, vol. 2, no. 11, pp. 1019–1025, 1999.
[8] J. A. Leñero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A 3.6 s latency asynchronous frame-free event-driven dynamic-vision-sensor," *Solid-State Circuits, IEEE Journal of*, vol. 46, no. 6, pp. 1443–1455, 2011.
[9] S. B. Furber, F. Galluppi, S. Temple, and L. A. Plana, "The spinnaker project," 2014.
[10] J. J. Hopfield, "Pattern recognition computation using action potential timing for stimulus representation," *Nature*, vol. 376, no. 6535, pp. 33–36, 1995.
[11] T. Natschläger and B. Ruf, "Spatial and temporal pattern analysis via spiking neurons," *Network: Computation in Neural Systems*, vol. 9, no. 3, pp. 319–332, 1998.
[12] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.
[13] A. Gupta and L. N. Long, "Character recognition using spiking neural networks," in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*, pp. 53–58, IEEE, 2007.
[14] J. H. Lee, P. Park, C.-W. Shin, H. Ryu, B. C. Kang, and T. Delbruck, "Touchless hand gesture ui with instantaneous responses," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pp. 1957–1960, Sept 2012.
[15] L. Camunas-Mesa, C. Zamarreno-Ramos, A. Linares-Barranco, A. J. Acosta-Jimenez, T. Serrano-Gotarredona, and B. Linares-Barranco, "An event-driven multi-kernel convolution processor module for event-driven vision sensors," *Solid-State Circuits, IEEE Journal of*, vol. 47, no. 2, pp. 504–517, 2012.
[16] A. Delorme, L. Perrinet, and S. J. Thorpe, "Networks of integrate-and-fire neurons using rank order coding b: spike timing dependent plasticity and emergence of orientation selectivity," *Neurocomputing*, vol. 38, pp. 539–545, 2001.
[17] C. Eliasmith and T. C. Stewart, "Nengo and the neural engineering framework: connecting cognitive theory to neuroscience," in *Proceedings of the 33rd annual meeting of the cognitive science society*, pp. 1–2, 2011.
[18] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain," *science*, vol. 338, no. 6111, pp. 1202–1205, 2012.
[19] M. Naylor, P. J. Fox, A. T. Markettos, and S. W. Moore, "Managing the fpga memory wall: Custom computing or vector processing?," in *Field Programmable Logic and Applications (FPL), 2013 23rd International Conference on*, pp. 1–6, IEEE, 2013.
[20] P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer, "Real-time classification and sensor fusion with a spiking deep belief network," *Frontiers in neuroscience*, vol. 7, 2013.
[21] T. Delbruck, "Frame-free dynamic digital vision," in *Proceedings of Intl. Symp. on Secure-Life Electronics, Advanced Electronics for Quality Life and Society*, pp. 21–26, 2008.
[22] C. Patterson, F. Galluppi, A. Rast, and S. Furber, "Visualising large-scale neural network models in real-time," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pp. 1–8, 2012.
[23] F. Galluppi, K. Brohan, S. Davidson, T. Serrano-Gotarredona, J.-A. P. Carrasco, B. Linares-Barranco, and S. Furber, "A real-time, event-driven neuromorphic system for goal-directed attentional selection," in *Neural Information Processing*, pp. 226–233, Springer, 2012.
[24] J. Lazzaro and J. Wawrzynek, "A multi-sender asynchronous extension to the aer protocol," in *Advanced Research in VLSI, Conference on*, pp. 158–158, IEEE Computer Society, 1995.
[25] L. A. Plana, "Appnote 8 - interfacing aer devices to spinnaker using an fpga." https://spinnaker.cs.man.ac.uk/tiki-download_wiki_attachment.php?attId=20, 4 2013.
[26] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
[27] G. La Camera, M. Giugliano, W. Senn, and S. Fusi, "The response of cortical neurons to in vivo-like input current: theory and experiment," *Biological cybernetics*, vol. 99, no. 4-5, pp. 279–301, 2008.
[28] A. N. Burkitt, "A review of the integrate-and-fire neuron model: I. homogeneous synaptic input," *Biological cybernetics*, vol. 95, no. 1, pp. 1–19, 2006.
[29] A. J. Siegert, "On the first passage time probability problem," *Physical Review*, vol. 81, no. 4, p. 617, 1951.
[30] Q. Liu, "A gabor filter prefers the horizontal lines running on spinnaker in real-time." https://www.youtube.com/watch?v=PvJy6RKAJhw&feature=youtu.be&list=PLxZ1W-Upr3eoQuLxq87qpUL-CwSphtEBJ, Sept. 2014.
[31] Q. Liu, "Feature extraction of live retinal input." http://youtu.be/FZJshPCJ1pg?list=PLxZ1W-Upr3eoQuLxq87qpUL-CwSphtEBJ, Sept. 2014.
[32] Q. Liu, "Live dynamic posture recognition on spinnaker." http://youtu.be/yxN90aGGKvg?list=PLxZ1W-Upr3eoQuLxq87qpUL-CwSphtEBJ, Sept. 2014.
[33] M. Elmezain, A. Al-Hamadi, J. Appenrodt, and B. Michaelis, "A hidden markov model-based isolated and meaningful hand gesture recognition," *International Journal of Electrical, Computer, and Systems Engineering*, vol. 3, no. 3, pp. 156–163, 2009.

Qian Liu received the B.Sc. degree in software engineering from Beijing University of Technology, Beijing, China, in 2008. She started the Ph.D. Study in The University of Manchester in 2014 working on visual and auditory processing with spiking neurons on neuromorphic system.

Steve Furber (Fellow, IEEE) was born in Manchester, U.K., in 1953. He received the B.A.degree in mathematics and the Ph.D. degree in aerodynamics from the University of Cambridge, Cambridge, U.K., in 1974 and 1980, respectively, and honorary doctorates from Edinburgh University, Edinburgh, U.K., in 2010 and Anglia Ruskin University, Cambridge, U.K., in 2012.

From 1978 to 1981, he was Rolls Royce Research Fellow in Aerodynamics at Emmanuel College, Cambridge, U.K., and from 1981 to 1990, he was at Acorn Computers Ltd., Cambridge, U.K., where he was a principal architect of the BBC Microcomputer, which introduced computing into most U.K. schools, and the ARM 32-bit RISC microprocessor, over 40 billion of which have been shipped by ARM Ltd.s partners. In 1990, he moved to the ICL Chair in Computer Engineering at the University of Manchester, Manchester, U.K., where his research interests include asynchronous digital design, low-power systems on chip, and neural systems engineering.

Prof. Furber is a Fellow of the Royal Society, the Royal Academy of Engineering, the British Computer Society, the Institution of Engineering and Technology and the Computer History Museum (Mountain View, CA). He was a Millennium Technology Prize Laureate (2010) and holds an IEEE Computer Society Computer Pioneer Award (2013).