Real-Time Implementation of Retinal Models

Garibaldi Pineda García
Supervisor: Steve Furber, Co-Supervisor: Dave Lester
School of Computer Science, University of Manchester, U.K.

***Abstract*—Converting frame-based video into spike-trains is a computationally intensive and time consuming task. There are very few examples of equipment that can produce spikes from video and they are either in development, use custom hardware or too expensive, thus every-day users are put behind a virtual wall, leaving them unable to experiment with visual input in their simulations or robotics applications. An efficient parallel implementation of a simple yet powerful retinal model would remove this wall by reducing the time it takes to compute a spike representation of a video frame. Furthermore if this is done on consumer graphics processing hardware, it will allow almost any user to generate their own spike-trains.**

I. Introduction

Vision systems in biological entities are among the most complex sensory inputs in nature. If we want to simulate them, it would require incredible amounts of computing power and, traditionally, different algorithms to perform each individual task. A parallel computation platform is the best way to go while attempting to solve this problem, since neural structures in the brain compute in this way.

In recent years neuromorphic (i.e. one that mimics the brain) hardware has risen attention as a different way of computing. Platforms such as *SpiNNaker* emulate the brain's parallelism [1]; furthermore it does so while maintaining low power consumption. SpiNNaker can also give neural simulations the flexibility of software models and biological real-time.

We hypothesize that a better understanding of vision in biology will lead to a unified computer vision framework. Using neural networks should translate in gaining an insight to the meaning of elements in a scene and, thus, a relation between different images of the same scenario. This relationships should prove to be a powerful tool to implement computer vision tasks such as registration, 3D perception or optical flow, to name a few. Bio-inspired vision algorithms using SpiNNaker hardware could be used on robotics, security or transportation applications. A key element for any neural network approach is learning, as far as we know, this remains an open problem for time or order based spike codes. The research on learning could lead into contributions on a theory of memory in the brain.

Converting conventional images or video into spike based representation is a must-do step for further studies, in section III we report on the work done so far towards this goal. We use off-the-shelf hardware to encode video, in particular we use Graphics Processing Units (GPU) due to their parallel architecture. In the final sections of this paper we present the plans we have to develop learning and classification algorithms.

II. Previous Work

In order to process visual input from frame based imaging devices on a spiking neural network (SNN) a transformation is needed. The most common way is to simply encode using Poisson spiking with a rate that is proportional to pixel intensity. This is an approximation to what the photoreceptor layer in the retina does, other cell layers react to changes in intensity and perform other computation before emitting actual spikes [2].

There has been multiple attempts at hardware based retina modelling, they've reported successful implementations with real-time performance, though require custom hardware [3]–[5]. So called *silicon retinas* where first described by Mead and Mahowald [6]. Similar devices have been developed and reported, they are splendid real-time, low-powered, high-dynamic-range event-based cameras [7], [8]. This great hardware is in early stages of production, thus it may be too expensive for some researchers' budget or might not even be available for purchase.

Software based models have been reported by many authors with different results. One of the most accurate retinal models was developed by Wohrer and Kornprobst, whose results display great levels of accuracy when compared to recorded data [9]. Gautrais and Thorpe proposed a functional retinal model that uses 16 different ganglion cells to encode images [10]. The model encodes images into *Rank-Ordered* spikes, following the ideas of Field on sparse coding an redundancy reduction [11]. Although they obtain good results, their model lacked biological plausibility. Starting with this model, Sen and Furber created a new one that takes into account the physical characteristics of the *foveal pit* in the retina [12]. We propose to implement this last model using parallel programming due to the nature of the problem.

III. Project Progress to Date

Our objective this year is to generate a video-to-spike train encoder using of-the-shelf components. For this, the first approach was to use a biologically plausible functional model (Algs. 1 and 2) that results in images being transformed into rank-ordered spikes [12].

The retina is a thin layer of neural cells located at the back of the eye (Fig. 1), it is responsible for the sensing,

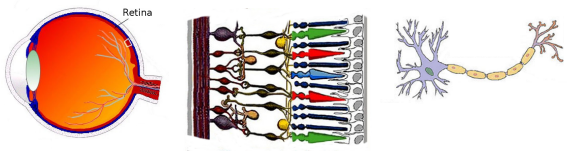
Algorithm 1 FoCal, Part 1

procedure CONVOLUTION(image I, kernels K)
  $C \leftarrow \emptyset$
  for all $k \in K$ **do**
    $C \leftarrow C \cup convolution(k, I)$
  end for
end procedure

Algorithm 2 FoCal, Part 2

procedure CORRECTION(coeffs C, correlations Q)
  $N \leftarrow \emptyset$ ▷ Corrected coefficients
  repeat
    $m \leftarrow max(C)$
    $M \leftarrow M \cup m$
    $C \leftarrow C \setminus m$
    for all $c \in C$ **do** ▷ Adjust all remaining c
      if $Q(m, c) \neq 0$ **then** ▷ Adjust only near
        $c \leftarrow c - m \times Q(m, c)$
      end if
    end for
  until $C = \emptyset$
  return M
end procedure

processing and transmission of visual input [2]. At its deepest layer, the retina has a millions of cells known as photoreceptors (right of Fig. 1b), they are in charge of transforming light into electrical signals. After this step there are three layers of neurons that perform different computations such as lateral inhibition or centre-surround behaviour (left of Fig. 1b) [2]. A small area near the centre of the retina has very few obstacles to obtain light and has the highest resolution, this area is known as the *foveal pit* (small depression on the right of Fig. 1a).



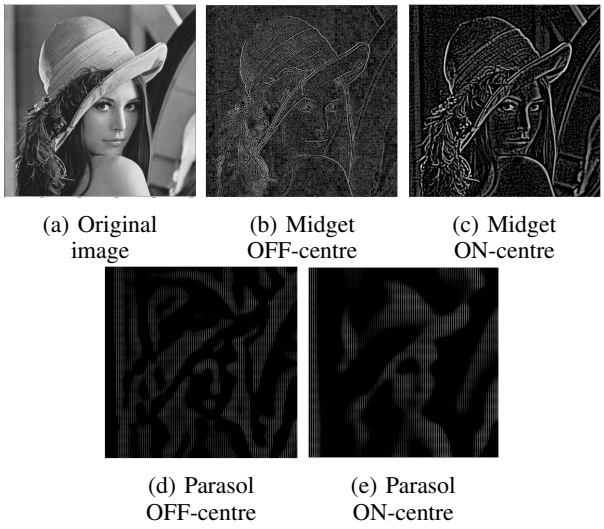
(a) Eye schematics (b) Retina (c) Neuron

Figure 1: Anatomy of the (human) eye [14]

Algorithms 1 and 2 model the foveal pit. On a first step (Alg. 1) we perform a two-dimensional *discrete convolution* of the current frame with four *kernels*. Each of the ganglion cells are modelled using a Difference of Gaussian (Eq. 1) using the parameters shown in Table I [11], [12].

$$DoG_w(x,y) = \pm\frac{1}{2\pi\sigma_{w,c}^2}e^{\frac{-(x^2+y^2)}{2\sigma_{w,c}^2}} \mp \frac{1}{2\pi\sigma_{w,s}^2}e^{\frac{-(x^2+y^2)}{2\sigma_{w,s}^2}} \quad (1)$$

where $\sigma_{w,c}$ and $\sigma_{w,s}$ are the standard deviation for the centre and surround components of the DoG at scale w (cell type). The signs will be $(-,+)$ if the ganglion cell is *off-centre* and $(+,-)$ if it is *on-centre*.



(a) Original image (b) Midget OFF-centre (c) Midget ON-centre

(d) Parasol OFF-centre (e) Parasol ON-centre

Figure 2: Results of simulating ganglion cells (convolved images are enhanced for better contrast)

Every pixel in the convolved images (Fig. 2) represents a spike emission time, the higher the pixel value, the sooner the spike will be sent out. Different ways of

Table I: Simulation parameters for ganglion cells

Cell type	Matrix width	Centre std. dev. (σ_c)	Surround std. dev. (σ_s)	Sampling resolution (cols, rows)
Midget Off-centre	3	0.8	$6.7 \times \sigma_c$	1, 1
Midget On-centre	11	1.04	$6.7 \times \sigma_c$	1, 1
Parasol Off-centre	61	8	$4.8 \times \sigma_c$	5, 3
Parasol On-centre	243	10.4	$4.8 \times \sigma_c$	5, 3

performing convolutions on a GPU where implemented, the naïve does a discrete convolution with the full kernels; for the biggest kernel (243×243 elements) we were unable to fit it into one of the fast memory locations of the GPU. We decompose the DoG into two horizontal and two vertical convolution kernels to perform a separated convolution. This method works best on kernels bigger than 3×3. The last approach, *Tiled Convolution* is reported by Advanced Micro Devices (AMD) [15]. They only present kernels of size 3×3, but we have an 11×11 convolution working; we are still developing solutions for the larger kernels. Convolution alone is a compute intensive task and we obtain about 12 frames-per-second (FPS) on videos with 640×360 8-bit grayscale pixel resolution. Encoding was carried out using a desktop computer running 64-bit GNU/Linux, with a Core i5-4570 4-core CPU @ 3.20 GHz processor with 8 GBytes of 64-bit DDR3 RAM @ 1600 MHz and a GeForce GT 720

Table II: Convolution performance comparison.

	Midget Off-centre	Midget On-centre	Parasol Off-centre	Parasol On-centre
Naïve	0.0009s	0.0031s	0.0587s	N/A[1,2]
Separated	0.0021s	0.0055s	0.0172s	0.0472s
Tiled	0.0009s	0.0044s	0.1643	N/A[2]

[1] Unable to fit convolution kernel into constant memory.
[2] Unable to compile OpenCL code.

(a) Original image (b) 100% of raw spikes (c) 100% of *corrected* spikes

Figure 3: Results of reconstruction procedure

GPU with 192 CUDA cores @ 797 MHz, 1 GBytes of 64-bit DDR3 RAM @ 1800 MHz.

In the retina, redundancy of information is reduced via lateral inhibition prior to any ganglion cell activity. In this algorithm, we perform a correction on the convolved images by adjusting the pixel values according to the correlation between convolution kernels (Alg. 2). The results of using correction (Fig. 3b) or not (Fig. 3c) show that the convolution stage can only provide redundant information. Furthermore, using only 30% of the corrected weights still provides enough visual information to reconstruct the original image [12].

Correcting the spikes for redundancy is a highly time consuming task which might be better suited for event-based programming, such as the one found on the SpiNNaker platform. We are still working on an implementation for this approach.

A second way of encoding is to simulate the early stages of the retina, which sense changes in intensity on the photoreceptors. This is quite similar to what real Dynamic Vision Sensors (DVS) do but with limited dynamic range and lower temporal resolution [7], [8]. The main advantage is that no specialized hardware is needed and the operation is so fast that any recent computer should be able to do it. For this type of encoding procedure we hypothesize that the bigger the change, the sooner a cell would spike and, thus, we can obtain a spike timings given the difference of two video frames. So far we can process about 20 and 25 FPS using a Numpy and an OpenCL back-end, respectively (using the same hardware set-up previously described). Although it's currently a good approximation, more research on this algorithm is needed to better approximate to biology.

IV. Thesis Outline

- Abstract
- **Chapter 1**. Introduction.
  - Neural networks.
  - Spike codes in vision.
  - Inhibition.
  - Spatio-temporal patterns and learning.
  - Research objectives.
- **Chapter 2**. Background.
  - SpiNNaker platform.
  - Real-time artificial neural computations.
  - Polychronization.
  - Classification.
- **Chapter 3**. Methodology.
  - Model visual input using time-based spike codes.
  - Hierarchical networks for robust classification.
  - Feature identification.
  - Sensor fusion and image registration.
- **Chapter 4**. Results.
  - Comparison with other methods.
  - Discussion.
- **Chapter 5**. Conclusions and Further Work.
  - Conclusions.
  - Future work.
  - Publications.
- **References**.

V. Conclusions and further work

A. Conclusions

With this work we obtained further knowledge about the anatomy of the eye from both a detailed and a functional approach. We also have come to appreciate the importance of inhibition circuits to enable high-efficiency, low-redundancy computing in the brain and how this brings robustness to neural structure.

Real-time spike encoding, although with low temporal resolution, is achievable with a common GPU and the right combination of mathematics and engineering. Memory reads and writes in a GPU are extremely important, it is one of the biggest bottlenecks of the presented algorithms. If each convolution was to be performed by a different GPU, we expect to see much better performance, though testing is needed on this regard. We are planning on developing an FPGA based solution to this problem.

We propose timing mechanisms to emit spikes from a rank-ordered source. Possible solutions for a faster mutual inhibition algorithm might be to do it in-line as we send spikes to neuromorphic hardware; or let the neural simulation deal with mutual inhibition.

B. Future work

We plan to explore learning on spiking neural networks using time-based codes or rank-ordered ones. This is an area where work has been made but remains an open problem. After a learning mechanism is proposed, we will use this to design networks that are able to robustly classify the

MNIST hand-written digit dataset we previously encoded (section III). Once some understanding of how features are interpreted in the network is gained, we shall proceed to make use of this knowledge to derive vision algorithms. In some cases we will use multiple sensors, so we'll apply techniques like sensor fusion.

VI. Publications

Part of the work carried during this year will be published as a paper on a **Frontiers in Neuroscience** journal in a special issue *Benchmarks and Challenges for Neuromorphic Engineering*. The article will present the MNIST database encoded using different types of spike codes and propose it as a standard way of testing learning and recognition tasks.

Acknowledgements

This research is funded by the National Council of Science and Technology (CONACyT) and the Secretariat of Public Education (SEP) of México.

References

[1] S. B. Furber, D. R. Lester, L. A. Plana, J. D. Garside, E. Painkras, S. Temple, and A. D. Brown, "Overview of the spinnaker system architecture", *Computers, IEEE Transactions on*, vol. 62, no. 12, pp. 2454–2467, 2013.

[2] H. Kolb. (2015). Simple anatomy of the retina. M. E. Center, Ed., Webvision, [Online]. Available: http://webvision.med.utah.edu/book/part-i-foundations/simple-anatomy-of-the-retina/.

[3] D. Balya and B. Roska, "Retina model with real time implementation", in *Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on*, 2005, 5222–5225 Vol. 5. DOI: 10.1109/ISCAS.2005.1465812.

[4] Z. Nagy, Z. Voroshazi, and P. Szolgay, "A real-time mammalian retina model implementation on fpga", in *Cellular Neural Networks and Their Applications, 2006. CNNA '06. 10th International Workshop on*, 2006, pp. 1–1. DOI: 10.1109/CNNA.2006.341593.

[5] R. J. Vogelstein, U. Mallik, E. Culurciello, G. Cauwenberghs, and R. Etienne-Cummings, *A multichip neuromorphic system for spike-based visual information processing*, 2007.

[6] C. A. Mead and M. Mahowald, "A silicon model of early visual processing", *Neural Networks*, vol. 1, no. 1, pp. 91 –97, 1988, ISSN: 0893-6080. DOI: 10.1016/0893-6080(88)90024-X.

[7] J. Leñero-Bardallo, T. Serrano-Gotarredona, and B. Linares-Barranco, "A five-decade dynamic-range ambient-light-independent calibrated signed-spatial-contrast aer retina with 0.1-ms latency and optional time-to-first-spike mode", *Circuits and Systems I: Regular Papers, IEEE Transactions on*, vol. 57, no. 10, pp. 2632–2643, 2010, ISSN: 1549-8328. DOI: 10.1109/TCSI.2010.2046971.

[8] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 x 128 120 db 15 us latency asynchronous temporal contrast vision sensor", *Solid-State Circuits, IEEE Journal of*, vol. 43, no. 2, pp. 566–576, 2008, ISSN: 0018-9200. DOI: 10.1109/JSSC.2007.914337.

[9] A. Wohrer and P. Kornprobst, "Virtual retina: a biological retina model and simulator, with contrast gain control", English, *Journal of Computational Neuroscience*, vol. 26, no. 2, pp. 219–249, 2009, ISSN: 0929-5313. DOI: 10.1007/s10827-008-0108-4.

[10] J. Gautrais and S. Thorpe, "Rate coding versus temporal order coding: a theoretical approach", *Biosystems*, vol. 48, no. 1–3, pp. 57 –65, 1998, ISSN: 0303-2647. DOI: 10.1016/S0303-2647(98)00050-1.

[11] D. J. Field, "What is the goal of sensory coding?", *Neural Comput.*, vol. 6, no. 4, pp. 559–601, Jul. 1994, ISSN: 0899-7667. DOI: 10.1162/neco.1994.6.4.559.

[12] B. Sen and S. Furber, "Evaluating rank-order code performance using a biologically-derived retinal model", in *Proceedings of the 2009 International Joint Conference on Neural Networks*, ser. IJCNN'09, Atlanta, Georgia, USA: IEEE Press, 2009, pp. 1835–1842, ISBN: 978-1-4244-3549-4.

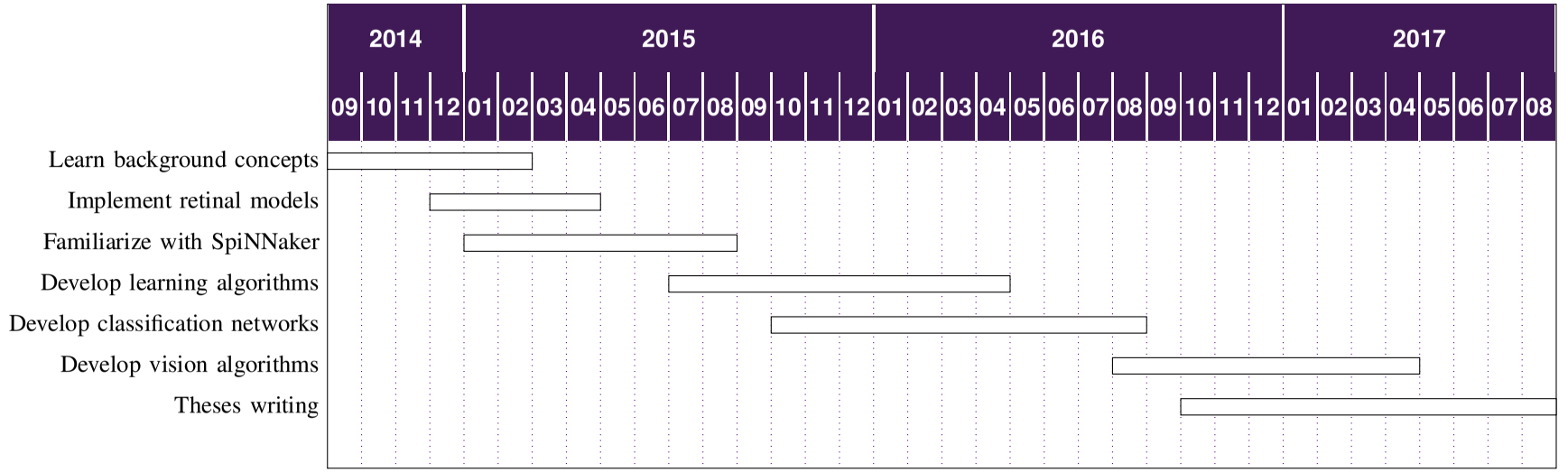
[13] S. Thorpe, A. Delorme, and R. V. Rullen, "Spike-based strategies for rapid processing", *Neural Networks*, vol. 14, no. 6–7, pp. 715 –725, 2001, ISSN: 0893-6080. DOI: 10.1016/S0893-6080(01)00083-1.

[14] (2015). Anatomical retina imagery source, modified to fit paper., [Online]. Available: http://webvision.med.utah.edu/.

[15] (2015). Tiled convolution: fast image filtering, [Online]. Available: http://developer.amd.com/resources/documentation-articles/articles-whitepapers/tiled-convolution-fast-image-filtering/.

Appendix

The project plan is presented on the following Gantt chart.


2014
2015
2016
2017
09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02 03 04 05 06 07 08 09 10 11 12 01 02 03 04 05 06 07 08
Learn background concepts
Implement retinal models
Familiarize with SpiNNaker
Develop learning algorithms
Develop classification networks
Develop vision algorithms
Theses writing