A Neuromorphic Approach for Tracking using Dynamic Neural Fields on a Programmable Vision-chip

Julien N.P. Martel
Institute of Neuroinformatics
University of Zurich and ETH Zurich
8057 Zurich, Switzerland
jmartel@ini.ethz.ch

Yulia Sandamirskaya
Institute of Neuroinformatics
University of Zurich and ETH Zurich
8057 Zurich, Switzerland
ysandamirskaya@ini.uzh.ch

ABSTRACT

In artificial vision applications, such as tracking, a large amount of data captured by sensors is transferred to processors to extract information relevant for the task at hand. Smart vision sensors offer a means to reduce the computational burden of visual processing pipelines by placing more processing capabilities next to the sensor. In this work, we use a vision-chip in which a small processor with memory is located next to each photosensitive element. The architecture of this device is optimized to perform local operations. To perform a task like tracking, we implement a neuromorphic approach using a Dynamic Neural Field, which allows to segregate, memorize, and track objects. Our system, consisting of the vision-chip running the DNF, outputs only the activity that corresponds to the tracked objects. These outputs reduce the bandwidth needed to transfer information as well as further post-processing, since computation happens at the pixel level.

CCS Concepts

•**Computing methodologies** → **Object detection; Tracking;** *Massively parallel and high-performance simulations;* •**Computer systems organization** → **Embedded systems; Cellular architectures;** •**Hardware** → *Cellular neural networks; Sensors and actuators;* •**Mathematics of computing** → Differential equations;

Keywords

Dynamic Neural Fields; Vision chip; Tracking; Cellular Processor Array; Cellular Neural Network; Artificial vision; Local Computation; Smart Sensor

1. INTRODUCTION

Video and imaging systems are ubiquitous and applications, in which visual sensing devices are employed, are diverse: imaging and recognizing defects in industrial products or pathologies in medical images; visual navigation systems for mobile platforms; 3D object scanning or environment mapping; as well as surveillance scenarios. In many of these applications, tracking of objects or features holds a prominent place. Such applications require systems to extract relevant information and to deal with the collection and processing of a massive amount of data coming from, in some cases, a large number of sensors. A way to address this challenge is to equip these sensors with more processing capabilities, thus avoiding a communication and processing bottleneck. This can be done either in the form of smart sensors featuring one or more processing units sitting next to the sensing chip, or in a more radical approach with architectures which integrate sensing and processing even tighter, directly on-chip in Application Specific Integrated Circuit (ASIC). Such a system must then realize a difficult trade-off between several contradicting constraints: be low-power, present a small form-factor, feature interesting computational properties, and run at a low latency.

In this work we propose an approach using an algorithm running on such a vision-chip that embeds an array of processors [2]. The vision-chip consists of processing elements with memory organized on a grid; each of them is placed

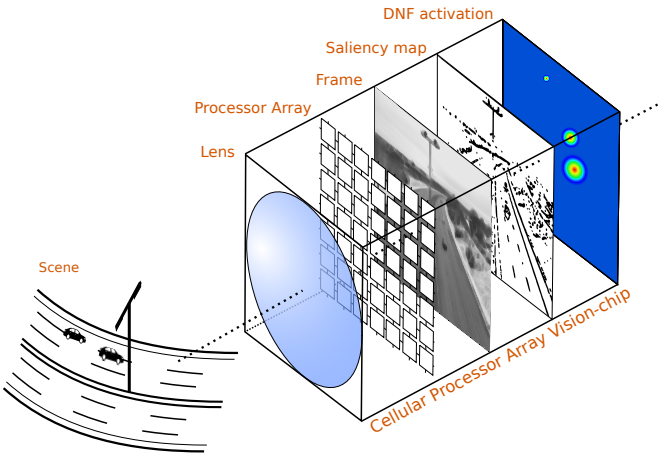

Figure 1: An illustration of the system implemented in this work: A vision-chip is directed at a scene and captures a stream of images processed on-chip in a saliency map. This map is fed as an input to a Dynamic Neural Field (DNF), also run on-chip and used for tracking objects. The system either outputs frames of the 2D activation of the DNF or the sole (x, y) centers of the bumps of activity on the DNF.

ICDSC '16, September 12 - 15, 2016, Paris, France

ISBN 978-1-4503-4786-0/16/09...$15.00

DOI: http://dx.doi.org/10.1145/2967413.2967444

next to a photosensitive element. Each pixel is thus able to perform local simple "arithmetic" operations. In this vision-chip, all processors can operate simultaneously on their own local piece of data captured at each pixel. This fine-grained massively parallel processing substrate can therefore achieve array computation very efficiently without having to incur a cost for transferring data from and to an external memory. Such array computations are common in image processing and computer vision tasks, where the same operation is repeated on all the pixels.

Tracking objects based on visual input is a computationally challenging task that might require processing at high frame rate and thus rely on a substantial bandwidth between the sensor and the unit processing the data. Deporting part of the tracking to the sensor avoids this bottleneck by offloading the computation to the periphery.

To profit from the many advantages of this device, specifically its intrinsic ability to perform local computation, we propose a local approach to tracking that makes use of a neuromorphic algorithm in the form of a dynamical system known as a Dynamic Neural Field (DNF) [25]. Even though tracking might not appear as a problem that can be solved with simple local computations, we demonstrate that the DNF behaviours allow us to do so. Furthermore, we show that DNFs can be naturally mapped on the device and that the hardware features present on-chip provide an efficient way to simulate their dynamics.

DNFs originate as a mathematical description of activity patterns in neuronal populations in both visual and motor brain cortices [22, 24]. Conceptually, a DNF is an attractor dynamics of a continuous activation function spanned over a behavioural dimension, e.g. the visual space. As a dynamical system, DNFs have a peculiar stable solution–a localised bump of increased activity. These localised activity bumps transform the continuous representation of the behavioural space (e.g. space of visual locations) to a discrete, categorical, and sparse representation of, e.g., perceived objects in space. The DNFs have been used in the past both to account for psychophysical data [25, 7] and to control cognitive robots, in particular realising object recognition with fast, one-shot learning [8], scene representation for human-robot interaction [1], organisation of robotic behaviour [18], or action recognition [11].

However, application of the DNFs in real-world technological systems, e.g. to solve problems in artificial vision, is limited by the computational complexity associated with solving the high-dimensional (in theory, continuous) dynamical system equation in real time. Our implementation of the DNF dynamics in vision-chip is fast and scalable, potentially allowing to implement large DNF architectures that can run in real time and be used in real-world applications.

In this paper, we present the first implementation of a DNF in the computing array of a vision-chip and demonstrate its efficiency in a tracking task. We start with a brief overview of the theory and functional properties of Dynamic Neural Fields that enable their application in vision tasks, in particular, tracking (Sec. 2). Then, we present the details of the vision-chip hardware used in this work and explain how the DNF equation can be efficiently implemented in this hardware (Sec. 3). Finally, we present results of experiments that demonstrate functionality of the developed system (Sec. 4) and conclude with an analysis of directions for future research (Sec. 5).

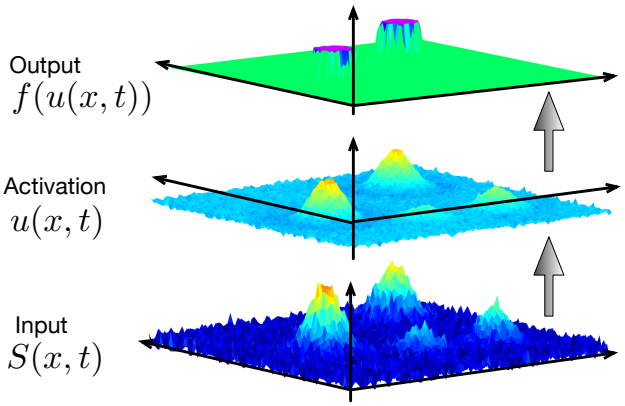

Figure 2: Activation of a simulated 2-dimensional dynamic neural field. Bottom: input to the DNF is a noisy activity pattern with four regions of increased activity ($S(x,t)$). Middle: in the activation of the DNF, $u(x,t)$, two of the locations with higher activity are above activation threshold and are strengthen by the lateral interactions. Top: the output of the DNF ($f(x,t)$) contains only two bumps that represent the locations of the most salient objects.

2. TRACKING WITH DYNAMIC NEURAL FIELDS

The Dynamic Neural Fields theory

The activation of a Dynamic Neural Field (DNF) follows the dynamical system equation, Eq. (1):

$$\tau \dot{u}(x,t) = -u(x,t) + h + \int f\big(u(x',t)\big)\omega(x,x')dx' + S(x,t). \quad (1)$$

Here, $u(x,t)$ is the activation function at time t of a DNF, defined over a parameter space x that describes the state of the system (in this work, x is the image space, $u(x,t)$ corresponds to the saliency, observed for each position in the image). $-h$ is a negative resting level that sets values of $u(x,t)$ to be below zero (the output threshold) in absence of an external input. $S(x,t)$ is the external input (e.g., from the sensor). $f(u)$ is a sigmoidal non-linearity (Eq. (2)) that shapes the output of the DNF: the output is zero for values of $u(x,t)$ that are negative and the output is positive for positive $u(x,t)$, saturating for larger values of $u(x,t)$:

$$f\big(u(x,t)\big) = (1 + e^{-\beta u(x,t)})^{-1}. \quad (2)$$

$\omega(x,x')$ is the interaction kernel that determines connectivity between positions x and x' on the DNF (i.e., lateral interactions). Typically, the interaction kernel has a "Mexican hat" shape with a short-range excitation and a long-range inhibition (Eq. (3)), leading to segregation of localised bumps in the DNF.

$$\omega(x,x') = c_{\text{exc}} e^{-\frac{(x-x')^2}{2\sigma_{\text{exc}}^2}} - c_{\text{inh}} e^{-\frac{(x-x')^2}{2\sigma_{\text{inh}}^2}}. \quad (3)$$

Fig. 2 shows a simulated 2-dimensional DNF. The DNF receives a noisy input with four regions of higher input strength.

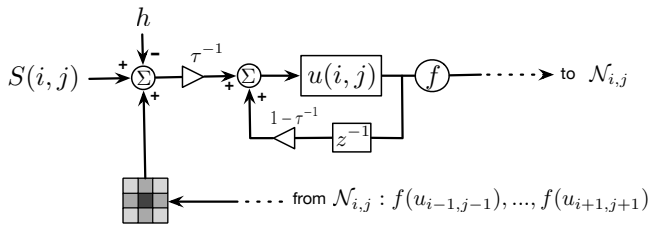

Figure 3: A block diagram that illustrates how the computational scheme of a DNF equation can be cast in the framework of Cellular Neural Networks: $\mathcal{N}_{i,j}$ is a neighbourhood of the cell (i, j), τ^{-1} is multiplication with the time-constant factor, f is a sigmoid non-linearity, z^{-1} stands for a delay of a unit of time.

The DNF dynamics and in particular the lateral interactions lead to strengthening of two of the input bumps, which reach the activation threshold and suppress the other bumps and the noise–the DNF "selects" the stronger of the inputs and stabilises their representation. Output of the DNF, $f(u(x, t))$, thus contains two stabilised activity peaks that correspond to locations of the two more salient objects.

Functional properties of DNFs

From a functional perspective, the DNF dynamics has the following properties that may be exploited in vision applications: (1) the low-pass filter characteristics of the DNF dynamics suppresses noise of the sensory input; (2) the local lateral excitation leads to stabilisation of a localised-peak attractor of the DNF, leading to a "detection instability" which marks detection of a sensory object in the input and stabilising a place-coded representation of this object or its attributes; (3) the lateral inhibition suppresses input from other locations in the DNF and stabilizes the "selection decision", thus avoiding oscillations between competing strong inputs; (4) at sufficiently strong lateral interactions, the DNF can sustained a memory: the DNF activity at the location of a strong input is kept positive even if input disappears or goes below activation threshold.

Exploiting these properties, DNF architectures have been used in robotic applications [18, 1]. Tracking with DNFs has been demonstrated in a robotic table-top scenario [8] along with object recognition and object pose estimation. However, application of the DNFs in robotic or computer vision tasks has been demonstrated in simplified lab settings. Discretization of the DNF dynamics both in time and in space on a conventional computer is a computationally demanding task. For instance, a two-dimensional DNF, discretized to an 100x100 array with a truncated 5x5 kernel, one iteration of the Eq. (1) takes several milliseconds on an average workstation. Parallellisation of the DNF dynamics on, e.g., a GPU is possible, but does not lead to computational breakthrough, since DNF dynamics relies on continual integration of new sensory input and exchange between field locations.

From a computational perspective, DNFs can be cast in the framework of Cellular Neural Networks (CNNs) [6, 5]. We represent a block diagram of the operation of a cell of our DNF in Fig. 3, whose similarities with block diagrams in the CNN literature [16] can be trivially established. This interpretation of the DNF is useful to realise it in hardware in this work, as it will become clear in the following section.

3. HARDWARE IMPLEMENTATION OF A DNF ARCHITECTURE FOR TRACKING

Cellular processor arrays and vision-chips

We consider smart vision sensor that combines both sensing and processing on the same substrate. It consists of an array of cells–that we also refer to as Processing Elements (PEs),–each comprising an Arithmetic Logic Unit (ALU) with some memory (registers) and including a photosensitive element to capture light. The architecture we consider also provides a way to communicate data from adjacent cells and is referred to as a Cellular Processor Array (CPA) [17, 9, 10]. Such a vision-chip architecture presents several advantages: first, by collocating a sensor with a processor array one benefits from very low latencies to access information. Data to be processed lies where it is collected, contrary to more conventional massively parallel processor array architectures in which tremendous amounts of data transfers between memory and processing units constitute a classical bottleneck. This principle consisting in processing luminance data where it is integrated is commonly referred to as near focal-plane processing [23]. Secondly, sensing and processing can be performed in parallel; in other words, it is possible to carry out computation as light is being integrated, thus "masking" the computation time of the ALU. Since the photosensitive element is accessible directly from the ALU, it can be considered to act as a light-sensitive register, whose value in time varies depending on the amount of light falling onto the pixel. This allows us to write algorithms in which sensing can affect the computation in a tight sense-compute loop.

This approach has been successfully demonstrated to address computationally expensive problems such as in High Dynamic Range imaging [12, 13], in message passing algorithms for visual inference [14] or as in the case of DNFs: for the simulation and resolution of Partial Differential Equations [15].

Architecture of the SCAMP-5 vision-chip

In this work we use a SCAMP-5 vision-chip [3] for the implementation of the DNFs. A peculiarity of this device is the design of its PEs featuring a mixed-signal circuitry [2]: each cell includes seven analog registers and thirteen single-bit digital registers. This design allows a reasonable trade-off between the silicon area required to implement the ALU and its registers, its power consumption, and the "computational primitives" it provides. Precisely, SCAMP-5 features an array of 256×256 PEs achieving up to 655 Gop/s (operations per second) for a power consumption of less than 1.2W.

Using current-mode computation, the analog part of the ALU can efficiently implement addition, negation (and therefore subtraction), and multiplication by constants at almost no silicon cost. An analog squarer allows us to compute the normalized square value of an analog register and can therefore be used as an analog multiplier to modulate two quantities. Finally a comparator allows to check the sign of a quantity. The digital part of the ALU incorporates circuitries to "OR" and negate the digital registers.

PEs can be masked using an activity flag. Program instructions received by the set of masked processing elements are not executed. This mechanism realizes conditional branching. The activity flag can be loaded from the analog comparator or from the digital logic.

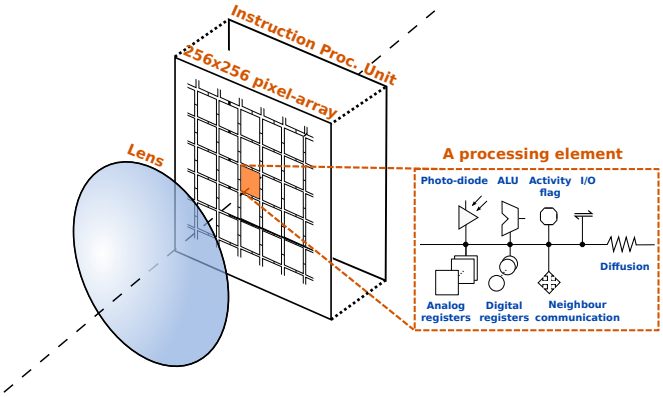

Figure 4: An illustration of SCAMP-5 showing a Processing Element and the features it includes.

In line with other CPA designs, a communication network enables each PE to write in a local register that can be read by the four adjacent neighbours hence allowing to shift data from one PE to another. In addition, a resistive network between the analog buses of the PEs can be used to diffuse the value of an analog quantity between PEs, practically allowing us to carry out operation of convolution with a Gaussian kernel.

The instructions are fed into each PE by an Instruction Processing Unit (IPU) implemented on a Field Programmable Gate Array (FPGA) physically lying behind the vision-chip. The same instruction is dispatched by the IPU to all the vision-chip processing elements that execute it on their own local piece of data. In this Single Instruction Multiple Data (SIMD) paradigm, it is particularly easy to perform array computations for which the same operation needs to be carried out element-wise.

To output data from the device, both digital readouts from the single-bit register planes as well as analog readouts after an Analog-to-Digital Conversion (ADC) stage can be achieved. Another interesting feature is the ability to produce Address Event Representations (AER), in which only a very limited set of PEs can be read and output. We use this representation to output only the centres of the detected and tracked objects in this work. As a consequence of the drastic bandwidth reduction it induces, it is possible to output AER at a much faster rate, eventually as fast as [4].

The system is programmable in software using a Domain Specific Language (DSL) and a toolchain including a programming environment with a compiler, debugger, an interface to harness the SCAMP-5 vision-chip, eventually post-process and visualize multiple outputs from the device.

A match between DNFs and SCAMP-5

Because SCAMP-5 is a software-programmable device, we can realise a DNF implementation and exploit the features of the hardware such as its massive parallelism, its built-in computational primitives–in particular, the diffusion–and its low latency in sensing/processing for tracking.

Specifically, the numerical simulation of the differential equation governing the behaviour of the DNF can be implemented in discretized time-steps. According to Eq. (1) and Fig. 3, every cell can update its state based on its state in the previous iteration and other signals, some of which vary through time. Since all cells obey the same differential equation with their own local input and only depend on neighbours at previous time steps, the DNF model naturally benefits from the parallelism of the SIMD device.

Algorithm 1 Pseudocode illustrating the DNF tracking algorithm run on the CPA vision-chip

Require: $u_0 = [\mathbf{0}]$ and $R_{now} = R_{pre} = R_{pix} = [\mathbf{0}]$
Require: N: number of iterations for the relaxation of the DNF for each incoming frame
$V \leftarrow R_{\text{now}} - R_{\text{pre}}$ ▷ Compute temporal gradient
$|\vec{G}| \leftarrow |\text{shift}_{west}(R_{\text{now}}) - R_{\text{now}}|$
$\quad + |\text{shift}_{north}(R_{\text{now}}) - R_{\text{now}}|$ ▷ Spatial gradient
$S \leftarrow |V| + |\vec{G}|$ ▷ Compute saliency map
$R_{pre} \leftarrow R_{now}$ ▷ Save previous frame
$R_{now} \leftarrow R^t_{pix}$ ▷ Copy current frame
$R^t_{\text{pix}} \leftarrow [\mathbf{0}]$ ▷ Reset pixel integration
for $i \in \{1, ..., \text{N}\}$ **do** ▷ Start relaxing for the frame
$\quad u_t \leftarrow$ updated according to Eq. (4)
$\quad t \leftarrow t + 1$ ▷ Keep u_t across iterations and frames
end for ▷ u_t has relaxed N steps for the frame
(Post-process to extract the centers of activity bumps)
Send AER centers of activity bumps or Readout $f(u_t)$

Computations running on a neighbourhood present the advantage to be easily implementable on a CPA like SCAMP-5 exploiting the direct communication between adjacent neighbours and a diffusion network. Also, the DNF does not require computationally convoluted functions such as exponential, trigonometric, or hyperbolic functions–not even for the non-linearity, as we will describe in the following–but rather simple arithmetic operations making it a natural candidate to use the analog ALUs.

The external input signals (h and $S(x)$ in Eq. (1)) are present on-chip, $S(x)$ amounts to the visual data captured and pre-processed on the array. An additional advantage is the ability to let the photo-sensitive element integrate light as the iterations of the DNF are being performed (realising relaxation towards an attractor) and use the visual information captured meanwhile for the next batch of iterations. We summarize this process in the pseudocode in Alg. (1).

Finally, in a tracking setup, one can read out the whole register containing $f(u_t)$, i.e the output of the DNF at the current time point, and post-process the bumps of activation it holds to extract their centres, sizes, etc. One might also consider an AER mode, in which the chip directly extracts the centres of the bumps as a post-processing step of the DNF and solely outputs them.

Formalisation of the hardware implementation

The DNF equation can be cast in a CNN framework as illustrated in Fig. 3. Particularly we show in the following how we reformulate the equation such that it maps on the device and exploit its intrinsic properties.

Let us consider a discretized version in time and space of Eq.(1):

$$u_{t+1}(i,j) = (1 - \frac{1}{\tau}) \cdot u_t(i,j) + \frac{1}{\tau} \cdot \Big(-h + S_{t+1}(i,j) + \big[f(u_t) * \omega\big](i,j)\Big), \tag{4}$$

where (i, j) is the spatial position of a cell in the array and t indexes time steps of the numerical update scheme.

First, let us note that the update of the state u_{t+1} of a cell results from an additive blending between its old state u_t and a term adding an offset, the input, and the interaction with other cells.

The offset term $-h$ is implemented in hardware filling the registers with a constant analog input.

The input $S(x)$ we consider for tracking is a basic "saliency map" $S = |\vec{G}| + V$ that combines a spatial gradient map $|\vec{G}| = |I_t(i+1,j) - I_t(i,j)| + |I_t(i,j+1) - I_t(i,j)|$ and a temporal gradient map $V = I_{t+1}(i,j) - I_t(i,j)$, where I_t is the light intensity captured during the previous imaging loop of the algorithm as seen in the pseudo-code of Alg. 1.

The cell interactions in the DNF (the kernel ω) is the sum of a local (proximal) excitation and a broader (distal) inhibition: $\omega = \omega^+ - \omega^-$. As a consequence, and thanks to the linearity of the convolution operator, the interaction with other cells can be rewritten as:

$$\big[f(u_t) * \omega\big](i,j) = \big[f(u_t) * \omega^+\big](i,j) - \big[f(u_t) * \omega^-\big](i,j) \tag{5}$$

This decomposition shows that we can simply implement two convolutions passes with the positive kernels ω^+ and ω^- – the last one having a broader span – on $f(u_t)$ and subtract them to obtain the desired cell interactions required for the DNF. These convolutions can be performed in parallel on all pixels, in a very few number of clock-cycles using the diffusion network on-chip.

The non-linearity needs not be a sigmoid function such as presented in Eq. (2); in fact, the desired DNF functional properties as described in Sec. 2 can be obtained with the function $f : \mathbb{R} \longrightarrow \mathbb{R}_+$ satisfying the following conditions:

$$f : x \longrightarrow \begin{cases} f(x) > 0, \text{ for } x \in]-\infty; 0[& : \text{rectification} \\ f(x) \sim x, \text{ for } x \in [0; \Gamma] & : \text{linearity} \\ f(x) = \Gamma, \text{ for } x \in]\Gamma, +\infty[& : \text{saturation} \end{cases} \tag{6}$$

In practice, operations on the registers we use on SCAMP-5 saturate, therefore, a non-linear f can be implemented by a simple rectifier.

Implementations on analog computing devices are usually tricky due to various sources of "noise" degrading their performance: first during computation, in which the outputs of operations are not "exact" due to mismatch between the components and secondly during handling in registers, particularly when storage exceeds long periods of time (analog registers can be thought as capacitors whose values decay through time due to leaks). However, DNFs from a dynamical system point of view are robust to a certain level of noise and thus they are excellent candidates to profit from analog computation.

In addition, devices have a limited range: in the case of SCAMP-5, analog registers are equivalent to holding 8-9 bits values that are converted at the output stage at readout in 8-bits digital values. Consequently, a special care must be taken with the operations performed on-chip. One can remark that in the form of Eq. (4), the blending preserves the range of the state u_t since the update respectively scales quantities by τ and $1 - \frac{1}{\tau}$. In the term associated to the scaling by $1 - \frac{1}{\tau}$, in which several quantities are added, one can remark by introducing Eq. (5) in (4), that operations with signed quantities can be alternated. This helps to "condition" the computation in the 8-bit range on-chip by avoiding the saturation of the registers when it is not desired.

4. EXPERIMENTAL SET-UP AND RESULTS

To demonstrate the performance of our system in tracking moving objects, we presented video sequences on a computer screen showing different road scenes as input to the vision-chip. These videos were downloaded from a free online resource (https://pixabay.com), had a resolution of 360x460 pixel and duration of 60 seconds. The implemented DNF was able to form localised activity bumps over moving objects in the videos ("detecting" them) and track these objects until they disappeared from the image frame. This result is illustrated in Fig. 5. In the figure, blue color shows negative levels of activation of a DNF (middle), red color shows regions with positive activation. In the output of the DNF (right), green regions have zero value, red regions have a positive value. Note how the implemented DNF can segment and track moving objects at different speed, size, and spatial gradient of sensed light intensity.

5. CONCLUSIONS

The system we presented is a very first implementation of a dynamic neural field architecture on a vision-chip. We have shown here that the system can perform tracking based on local computation and with high efficiency in terms of computing time and energy. The results look very promising and we will present the more detailed analysis of the system's performance and comparison with the state of the art in an extended version of the paper. The main focus of our future research will be on increasing robustness of the system to perturbations and integration of a more sophisticated saliency computation.

6. ACKNOWLEDGEMENTS

We would like to thank G. Indiveri and M. Cook from the Institute of Neuroinformatics, University of Zurich and ETH Zurich for their supervision. This work was financially supported by the EU H2020-MSCA-IF-2015 grant 707373 ECogNet, the SNF grant 143947, and the CapoCaccia Cognitive Neuromorphic Engineering Workshop.

7. REFERENCES

[1] E. Bicho, W. Erlhagen, L. Louro, and E. Costa e Silva. Neuro-cognitive mechanisms of decision making in joint action: A human-robot interaction study. *Human Movement Science*, 30(5):846–868, 2011.

[2] S. J. Carey, D. R. W. Barr, B. Wang, A. Lopich, and P. Dudek. Mixed signal SIMD processor array vision chip for real-time image processing. *Analog Integrated Circuits and Signal Processing*, 77(3):385–399, 2013.

[3] S. J. Carey, D. R. W. Barr, A. Lopich, and P. Dudek. A 100'000 fps vision sensor with embedded 535 GOPS/W 256×256 SIMD processor array. In *Proc. of the VLSI Circuits Symp.'13*, pages C182–C183, 2013.

[4] S. J. Carey, D. R. W. Barr, B. Wang, A. Lopich, and P. Dudek. Live demonstration: A sensor-processor array integrated circuit for high-speed real-time machine vision. In *Proc. of the IEEE Internat. Symp. on Circuits and Systems, ISCAS'14*, page 447, 2014.

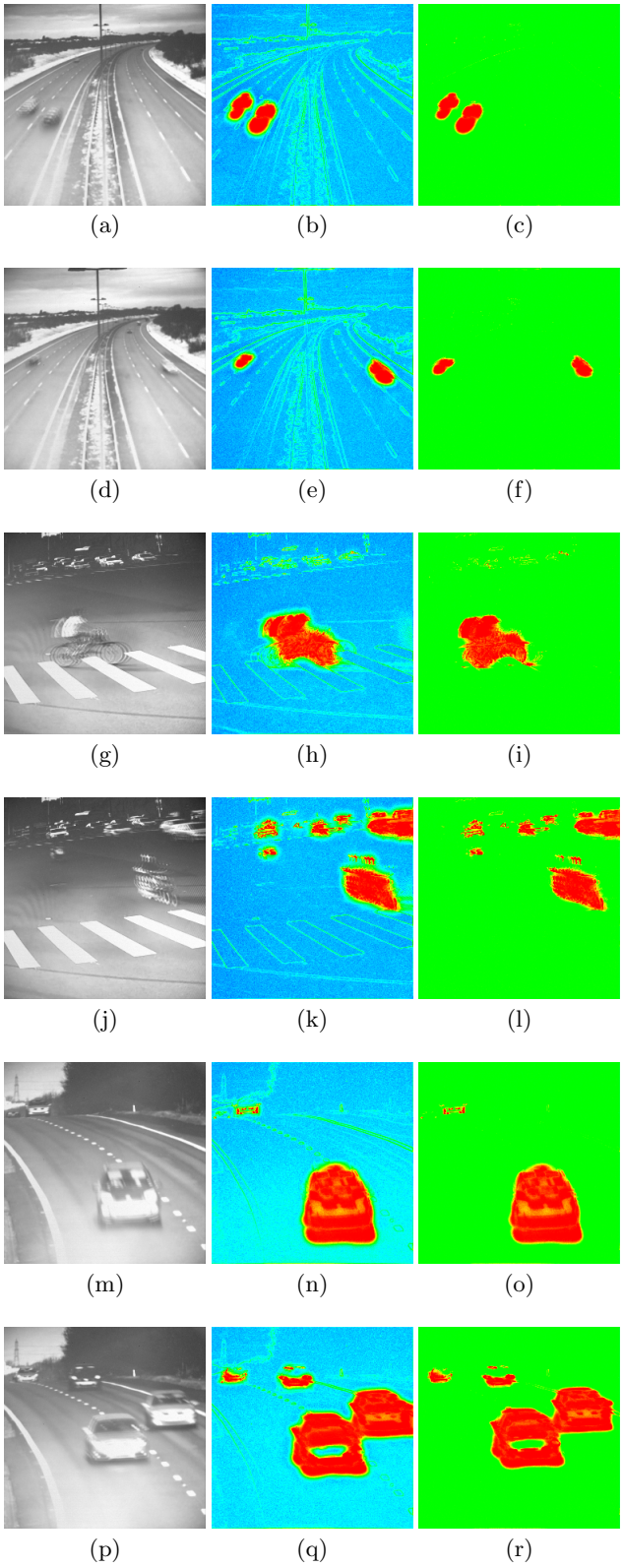

Figure 5: Frames of the video sequences (left), activity in the DNF on the vision-chip (middle), and output of the DNF (right) for three videos with road scenes.

[5] L. Chua and L. Yang. Cellular neural networks: theory. *IEEE Transactions on Circuits and Systems*, 35(10):1257–1272, 1988.

[6] L. O. Chua and T. Roska. The CNN Paradigm. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 40(3):147–156, 1993.

[7] P. Cisek and J. F. Kalaska. Neural mechanisms for interacting with a world full of action choices. *Annual review of neuroscience*, 33:269–98, jan 2010.

[8] C. Faubel and G. Schöner. Learning to recognize objects on the fly: A neurally based dynamic field approach. *Neural Networks*, 21(4):562–576, 2008.

[9] M. Ishikawa, K. Ogawa, T.Komuro, and I. Ishii. A CMOS vision chip with SIMD processing element array for 1ms image processing. In *Digest of Technical Papers, IEEE Internat. Solid-State Circuits Conf., ISSCC'99*, pp. 206–207, 1999.

[10] M. Laiho, J. Poikonen, and A. Paasio. *Focal-plane Sensor-Processing Chips*, chapter MIPA4k: Mixed-Mode Cellular Processor Array, pages 45–71. Springer, 2011.

[11] D. Lobato, Y. Sandamirskaya, M. Richter, and G. Schöner. Parsing of action sequences: A neural dynamics approach. *Paladyn, Journal of Behavioral Robotics*, 6(1):119–135, 2015.

[12] J.N.P. Martel, L. K. Muller, S. J. Carey, and P. Dudek, "Parallel hdr tone mapping and auto-focus on a cellular processor array vision chip," in *Proc. of the IEEE Internat. Symp. on Circuits and Systems, ISCAS'16*, 2016.

[13] J.N.P. Martel, L. K. Muller, S. J. Carey, and P. Dudek, "A real-time high dynamic range vision system with tone mapping for automotive applications," in *Proc. of the Internat. Workshop on Cellular Nanoscale Networks and Applications, CNNA'16*, 2016.

[14] J.N.P. Martel, M. Chau, P. Dudek, and M. Cook, "Toward joint approximate inference of visual quantities on cellular processor arrays," in *Proc. of the IEEE Internat. Symp. on Circuits and Systems, ISCAS'15*, 2015, pp. 2061–2064.

[15] J.N.P. Martel, M. Chau, M. Cook, and P. Dudek, "Pixel interlacing to trade-off the resolution of a cellular processor array against more registers," in *Proc. of the IEEE European Conf. on Circuits Theory and Design, ECCTD'15*, 2015.

[16] R. Matei. *Advanced Technologies*, chapter Chapter 25: New Model and Applications of Cellular Neural Networks in Image Processing. InTech, 2009.

[17] W. Miao, Q. Lin, W. Zhang, and N.-J. Wu. A programmable SIMD vision chip for real-time vision applications. *"IEEE Journal of Solid-State Circuits"*, 43(6):1470–1479, 2008.

[18] M. Richter, Y. Sandamirskaya, and G. Schöner. A robotic architecture for action selection and behavioral organization inspired by human cognition. In *Proceedings of IEEE/RSJ IROS*, 2012.

[19] Y. Sandamirskaya. Dynamic Neural Fields as a Step Towards Cognitive Neuromorphic Architectures. *Frontiers in Neuroscience*, 7:276, 2013.

[20] S. Schneegans, J. P. Spencer, G. Schöner, and A. Hollingworth. Dynamic interactions between visual working memory and saccade target selection. 14(2014):1–23, 2014.

[21] G. Schöner. *Dynamical systems approaches to cognition.* Cambridge University Press, 2008.

[22] H. R. Wilson and J. D. Cowan. A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue. *Kybernetik*, 13(2):55–80, 1973.

[23] A. Zarandy, editor. *Focal-plane Sensor-Processing Chips.* Springer, 2011.

[24] Rougier, Nicolas P. Dynamic neural field with local inhibition. *Biological cybernetics, 94(3):169-179, 2006*

[25] G. Schöner, and J. Spencer. Dynamic thinking: A primer on dynamic field theory. *Oxford University Press, 2015.*