Cellular automata and non-static image processing for embodied robot systems on a massively parallel processor array*

Martin Hülse[1], David R. W. Barr[2], Piotr Dudek[2]

[1] Aberystwyth University, UK
msh@aber.ac.uk
[2] University of Manchester, UK
d.barr@postgrad.manchester.ac.uk
p.dudek@manchester.ac.uk

Abstract. A massively parallel processor array which combines image sensing and processing is utilized for the implementation of a simple cellular automata (CA). This CA is essential part of an image processing task supporting object detection in real-time for an autonomous robot system. Experiments are presented, which demonstrate that objects will be detected only if they move below a specific velocity in the visual scene. Based on these experiments we will discuss the role of configurations changes if a CA is seen as a parallel processing computer. This leads us to the conclusion that if CA are performing non-static data processing tasks they might be better approached as sensor-driven parameterized dynamical system rather than as parallel computers operating on initial configurations only.

1 Motivation

Recent progress in chip design provides massively parallel processor arrays where each processor is linked to its neighbor processors. Such systems are ideally suited to perform low-level pixel-parallel image processing tasks. The SCAMP (SIMD Current-mode Analogue Matrix Processor) vision chip is one example of such a parallel processor array that integrates image sensing and processing on a single silicon die providing low-cost, low-power and high-performance vision system for autonomous robot systems [5].

Unfortunately, the majority of current low-level pixel-based image processing approaches are focused on 2-dimensional convolution operators and the analysis of static features in static images. However, image processing in autonomous robots is rather confronted with permanently changing signals and noise.

On the other hand, architecture and data processing of the SCAMP vision system are able to instantiate 2-D cellular automata (CA). It is well known that

* This article appeared in "AUTOMATA-2008: Theory and Applications of Cellular Automata", pp.504-513, Luniver Press, 2008. The version of the article published in the book contained errors, which have been corrected in this document.

CA provide a rich reservoir of dynamical properties and behavior on a global scale [7]. Our assumption is that nonlinear dynamical properties generated by CA are a promising substrate for the processing of complex and changing image data for autonomous robot systems. However, methods and frameworks are missing which allow a targeted design of CA supporting specific processing tasks on image data. The following investigation is a first step towards CA-based processes with high update rates (100 kHz) performing image processing in real-time in embodied autonomous robot systems.

This contribution summarizes first simple experiments that demonstrate how a CA can support image processing for a robot platform in a changing environment (including an active vision system and a manipulator). In particular we will present an image processing task performed on different time scales. Based on this example we will discuss the utilization of CA as data processing devices. This discussion emphasizes the importance of changes in CA configurations during the evolution of the system (here caused by visual input data). This view might be contrary to traditional approaches, where CA are seen as discrete dynamical system working as a parallel processing computer, "where data is considered to be the initial CA configuration" [7]. Our hypothesis is that an application of CA for "non-static" / robotic related image processing has to consider CA as sensor-driven parameterized dynamical systems [6]. In consequence, the data the system is operating on are the parameter values of the CA and their changes over time caused by the visual input.

2 The SCAMP vision system

The SCAMP vision chip is a sensor/processor device, including a 128x128 array of processor cells, arranged in a 4-connected neighborhood. Each processor also integrates image sensor, for pixel-parallel image input. The processors operate in a single-instruction multiple-data (SIMD) mode, i.e. all processors execute the same instruction, issued by a central controller. The instruction set includes arithmetic operations and neighbor transfers, which allows implementation of a variety of pixel-parallel local computations, characteristic of early vision applications. SCAMP executes instructions at 1MHz rate, so that even fairly complex operations can be easily implemented at video frame rates.

The general-purpose, software-programmable nature of the device enables the execution of a variety of algorithms. In particular, CA can be easily implemented, by writing simple programs to execute state update rules. The neighborhood needs not be restricted to 4-connectivity, as data can be easily passed between processors in several steps over larger distances.

The processors in the SCAMP chip operate on analogue data values, i.e. unlike a conventional digital processor it natively operates on real-valued numbers, albeit with a limited signal/noise performance and constrained dynamic range. The out-of-range operation results in a soft saturation (sigmoid function), which can be exploited to implement nonlinearities in the system [2]. Logic operations, and discrete state-space CA can be implemented using thresholds (comparison

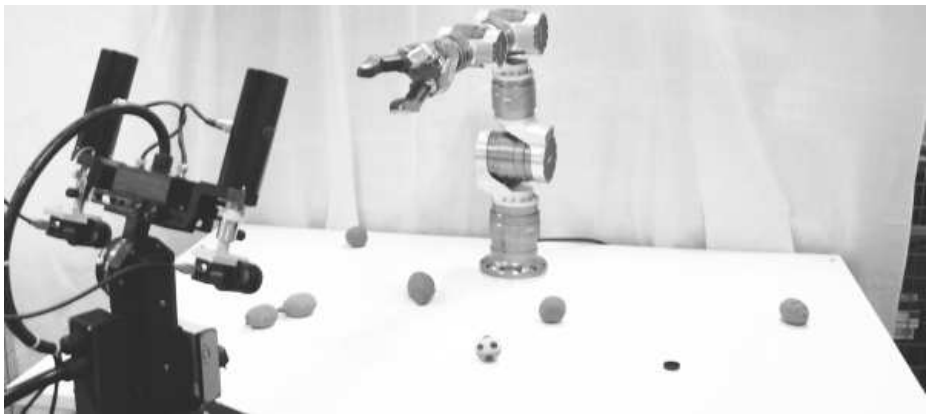

Fig. 1. Robotic arm and manipulator system mounted on a table and the active vision system, scanning the scenario with two cameras and a SCAMP vision system.

operations), however the native mode of operation of the chip is continuous-value.

3 The use of SCAMP in a robotic setup

The robotic scenario in which SCAMP is used consists of an active vision system and a robot manipulator with 7 degrees of freedom (DOF). As one can see in Fig. 1, the robot manipulator is mounted on a table. Sensory input driving reaching and grasping is delivered from the active-vision system scanning the table. Apart from SCAMP two additional cameras deliver visual inputs that can drive the pan, tilt and verge motors of the active vision system. In the following we present experiments where only SCAMP is used for visual input and the pan-tilt-verge configuration is fixed. In other words the SCAMP camera doesn't move while scanning the table.

Confronted with such a robot system the first step is the implementation of control systems performing goal-directed and robust gaze and reaching behavior. However, for autonomous robots it is important that the system itself, i.e. without human interaction, is able to trigger action sequences for reaching or gazing. In consequence, the system needs to measure not only where an object is located on the table but also whether or not it is moving, and if so, does it move slow enough in order to reach it with its manipulator.

Due to the physical reality of our robot hardware, a simple pan-tilt-system, obviously, can operate in a much faster way than an 7 DOF robot arm. Successful reach movements can only be performed with object much slower than it is necessary for a visual based object tracking with our pan-tilt system. Therefore, we have implemented a process that detects objects in real-time depending on the speed the objects move in the visual field. If an object is faster it becomes

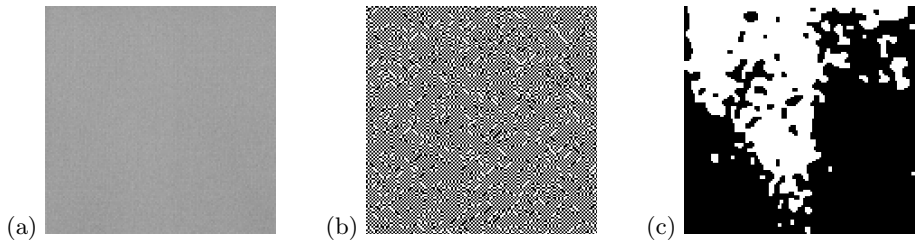

Fig. 2. Resulting processor values staring from an random initial configuration (a). The grey values representing values around zero. The patterns in (b) emerge from the inhibitory coupling ($w = -0.25$), while the pattern in (c) is generated by the excitatory linking ($w = 0.25$).

invisible to the system. If an object emerges in the visual field then it is implicitly given that it doesn't move at all or at least slow enough to reach for it. The sensitivity to changes in the visual input depends on a single parameter and therefore the system can easily be adapted to different time scales of the different actuators in the robot system (here we have a manipulator and a pan-tilt-verge system). The core application of this image data processing task is provided by a CA. The CA is instantiated by the SCAMP system, which we will explain in the following section.

4 Embedding cellular automata into a robotic scenario

The structure of the applied CA is very simple and straightforward to implement with SCAMP. Each processor P_n receives input from its four direct neighbor processors $P_{n,N}$, $P_{n,S}$, $P_{n,E}$ and $P_{n,W}$. Due to the analogue character of SCAMP operating on currents, we have continuous values for all P_n. As we have already mentioned, the lower and upper saturation domain resulting from out-of-range operations match with a sigmoid function f [2]. The system can be formally written as:

$$P_n(t+1) = f\left(P_n(t) + w \cdot \left(P_{n,N}(t) + P_{n,S}(t) + P_{n,E}(t) + P_{n,W}(t)\right)\right),$$

where $w > 0$ (here $w = 0.25$). In fact this CA implementation on SCAMP is continuous-value CA with a nonlinear update rule, or as it is called in the literature a discrete-time cellular neural network (DT-CNN) [4].

Since the values of the processors can be positive or negative this excitatory linking scheme generates an interplay of excitation and inhibition. Without visual input and random initialization, the processor array evolves to a state where the value of a processor is very likely the same of its direct and indirect neighbors. This is represented by large regions and patches colored either black or white, as shown in Fig. 2(c). These black and withe regions indicate the emergence of clusters and closed regions containing processors whose values are driven into the

Fig. 3. Sequence of CA configuration (time step: 0,..., 9, 50, 70 and 90) without resetting and continuous image data input.

same saturation domain, either in the upper or lower saturation. It is interesting to see, that the opposite is the case for negative links between the processors (i.e. $w < 0$). Such an inhibitory coupling generates almost a chess-board pattern, i.e. each direct neighbor processor is driven into the opposite saturation domain, Fig. 2(b).

Considering the positive coupling between the processors it seems that the growing black patches are perfect to indicate salience region within an image. Therefore, we implemented the CA with positive couplings on SCAMP and further on, used the visual data provided by SCAMP (128x128 grey value image) as permanent input for this specific CA. Hence, the CA configuration at time step $(t + 1)$ is determined by the configuration at time t and the current value of the corresponding pixel in the image ($PIX_n(t)$):

$$P_n(t+1) = f\left(PIX_n(t) + P_n(t) + w \cdot (P_{n,N}(t) + P_{n,S}(t) + P_{n,E}(t) + P_{n,W}(t))\right),$$

where $w = 0.25$. In Fig. 3 an example of the resulting sequence of images is shown. The black circular regions in the image which represent the objects on the table are growing, even if initially they have very low grey values. This is the result of adding the current visual input in each time step.

As the sequence in Fig. 3 is indicating, such a setup seems rather inapplicable for non-static image data, i.e. moving objects or a moving camera. After a number of time steps the whole image is black, because the black regions never disappear again and the system becomes "blind". In order to stay sensitive to new visual stimuli we setup an additional reset mechanism, which resets the CA configuration every n time steps. The CA configuration before the reset is the output of our image processing. In this way the image processing is actually

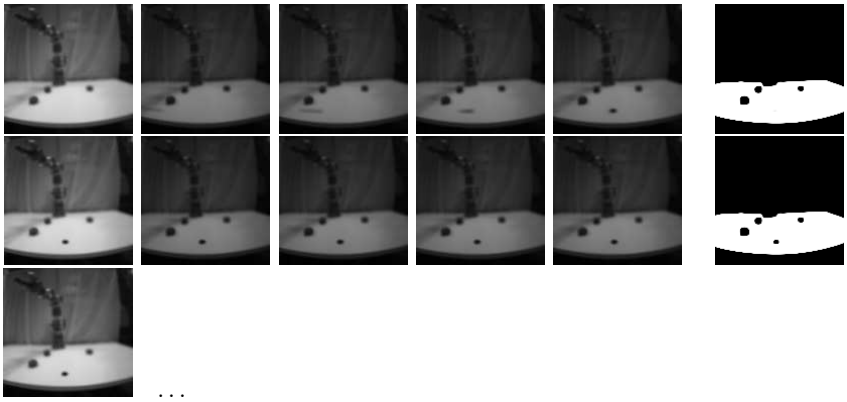

Fig. 4. Sequence of images representing the visual input PIX and the CA configuration between the resets, done every 6th time step. The six images in each line represent one period starting with the first state after the reset and ending with the last state before reset. The first five images in each line represent the visual input PIX at the first five time steps after reset. The last image represents the CA configuration before it is reset, i.e. the actual output data of the process

established by two processes: the inner loop that updates the CA involving the current visual input of SCAMP, and the outer loop reading the CA configuration every n steps and resetting the CA configuration. As we will show in the next section the output stream generated by the outer loop provides the properties we are aiming for: objects on the table only emerge in the output image, if their speed in the visual image is below a certain value.

5 Experiments with moving objects

In the following experiments we have reset the system every six time steps ($n = 6$). In order to give the reader an impression of the relation between output and visual scene, we have plotted six images between each reset. The first five images in this sequence represent the visual input data PIX for the first five steps after each reset. The last image in this line the actual output / result of the process. It is the CA configuration its reset.

In the first example shown in Fig. 4 we see at the front part of the table a small object sliding into the scene. The first five images in the first line show this event. The faster the object moves the more blurred it appears in these images. Notice, these images represent the visual input data PIX which are fed into the CA. The last image in this line shows the CA configuration representing the actual output of this process. One can see, that the new object doesn't emerge in this image, despite the fact that it is clearly visible in four out of five PIX images. Once the object has stopped on the table (second line of images) it

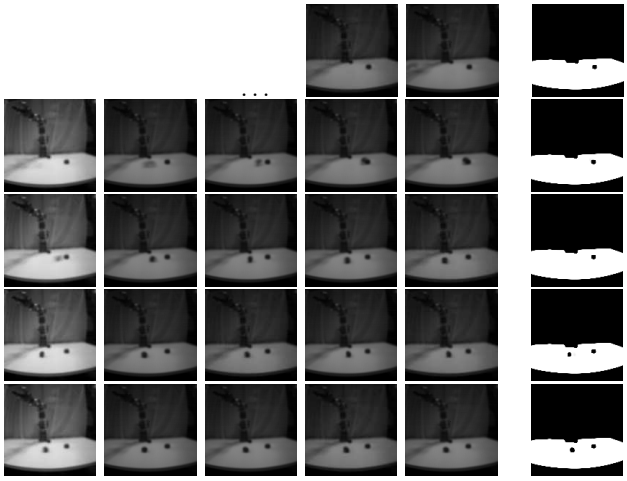

Fig. 5. The representation of image data processing as in Fig. 4. See text for explanation.

emerges in the output image. In other words, only after it has stopped it becomes visible for the robot system.

Another example of object detection is shown in Fig. 5. There the new object remains invisible for more than 12 time steps because it is moving to fast for this configuration.

6 Conclusion

We have shown that a CA can provide image processing for an autonomous robot system. Objects are only detected by the vision system if they move below a certain speed. This quality of object detection simplifies the triggering of reaching actions because if a stimulus emerges then it is guaranteed that it isn't moving to fast for the arm system and the robot can directly perform the related action sequence without any further data processing.

The "insensitivity" to fast moving objects is determined by the frame rate SCAMP is providing visual input. Within certain boundaries this frame rate is a free parameter of the SCAMP vision system and therefore, this process can easily be tuned to different time scales, even online.

The crucial element for detecting slow moving objects only is the continuous input of new visual data into the CA at each time step. As the image sequences in Fig. 4 and 5 clearly show occasional and limited appearance of low activities in the visual data (low grey values) doesn't immediately drive the output values into the lower saturation. Only if a low activation in a region of high activations is continuously measured then the corresponding CA units change and indicate a object in the visual scene. This mechanism can be seen as accumulation of

evidence and is frequently discussed for action-selection processes in biological and artificial systems [3].

However, CA are usually referred to as computational devices that operate only on initial configurations. The setup here could be seen as a counterpart of this approach and we believe, our experiments give evidence that it is worth investigating CA operating with continuously changing inputs. Nevertheless, there is a fundamental difference between a CA operating on initial configurations only and a CA driven by permanent changing inputs. With respect to the update rule describe above we can describe three principal ways CA can operate on image data. The first possibility is that image data are only used for determining the initial state of the CA, i.e. each processor P_n is initialized with the value of the corresponding pixel in the image:

$$P_n(0) = PIX_n(0).$$

After this initialization the system is updated according to the update rule which doesn't involve any image data:

$$P_n(t+1) = f\left(P_n(t) + w \cdot (P_{n,N}(t) + P_{n,S}(t) + P_{n,E}(t) + P_{n,W}(t))\right).$$

In fact we have a dynamical system without free parameter. Hence, its behavior is determined by its initialization, a single image, only.

Another way of combining visual data and CA uses the image data for the definition of a constant offset:

$$O_n = PIX_n(0)$$

for each unit in the CA. This offset is a parameter that now determines the update rule as follows:

$$P_n(t+1) = f\left(O_n + P_n(t) + w \cdot (P_{n,N}(t) + P_{n,S}(t) + P_{n,E}(t) + P_{n,W}(t))\right),$$

where $P_n(0)$ is arbitrary chosen. In this case we have a parameterized dynamical system [1]. The system behavior of the CA is determined by the initialization *and* the parameter O_n for each processor.

Comparing these two cases with the original update rule applied for our robotic experiments:

$$P(t+1) = f\left(PIX_n(t) + P_n(t) + w \cdot (P_{n,N}(t) + P_{n,S}(t) + P_{n,E}(t) + P_{n,W}(t))\right),$$

we see that our setup is the only process that can deal with changing image data. In contrast to the other systems, where there is no way of feeding data changes into the running process.

To sum up, the here introduced CA is, precisely speaking, an implementation of a parameterized dynamical system, whose parameters change according to the image data. This is, what we call a sensor-driven parameterized dynamical system, because parameter changes are driven by the image data which is the sensory input of an autonomous robot system. Sensor-driven parameterized

dynamical systems seem to be a promising approach to deal with changing inputs, since the two other alternatives of combining CA and image data are only operating on static data. This observation let us conclude that CA should be approached as sensor-driven parameterized dynamical systems, if one is interested in applications and novel computational paradigms for embodied robot systems.

Further on, as we have outlined above our continuous-value CA can also be seen as discrete-time cellular neural networks. Research on DT-CNN has already emphasized and exemplified the need of continuous input for visual data processing [4]. However, as it is often the case in the domain of complex systems, the phenomena are well know, but the challenge is to develop related applications as well as to derive general mechanisms that might lead to alternative paradigms for information processing. In this paper we have demonstrated that the SCAMP system is able to create complex dynamics based on CA in real-time. Therefore, embodied robot systems equipped with a SCAMP vision chip seem to be an efficient framework for bringing together the complex dynamics of CA *and* complex image processing in real-time. In addition, autonomous robot systems together with the application of self-organized adaptation processes (e.g. evolutionary algorithms) provide a context where the use of complex dynamics for real-world scenarios can be systematically explored and analyzed. This approach was demonstrated successfully within an evolutionary robotics framework for small discrete-time dynamical systems (small with respect to the state space) [6]. We believe SCAMP is a promising tool for scaling up this approach massively.

Acknowledgement: The first author gratefully acknowledges the support of EPSRC through grant EP/C516303/1.

References

1. V. I. Arnold, *Geometrical methods in the theory of ordinary differential equations*, (Springer,1983).
2. D. R. W. Barr, P. Dudek, J. Chambers and K. Gurney, "Implementation of Multilayer Leaky Integrator Networks on a Cellular Processor Array" (in International Joint Conference on Neural Networks, IJCNN 2007, Orlando, Florida, 2007).
3. R. Bogacz, K. Gurney, "The basal ganglia and cortex implement optimal decision making between alternative actions" (in Neural Computation, 19, 442-477, 2007).
4. L. O. Chua, T. Roska, *Cellular neural networks and visual computing - Foundations and applications*, (Cambridge University Press, 2001)
5. P. Dudek, S.J.Carey, "A General-Purpose 128x128 SIMD Processor Array with Integrated Image Sensor" (in Electronics Letters, 42(12), 678-679, 2006).
6. M. Hülse, S. Wischmann, P. Manoonpong, A.v. Twickel, F. Pasemann, "Dynamical Systems in the sensorimotor loop: On the interrelation between internal and external mechanisms of evolved robot behavior" (in M. Lungarella et al. (Eds.) 50 Years of Artificial Intelligence, LNCS 4850, Springer, 186-195, 2007).
7. A. Wuensche, M. Lesser, *The global dynamics of Cellular Automata* (Addison Wesley, 1992).