

Cellular Automata Simulation on FPGA for Training Neural Networks with Virtual World Imagery

Olivier Van Acker and Oded Lachish

Department of Computer Science and Information Systems
Birkbeck, University of London
London, United Kingdom
olivier@dcs.bbk.ac.uk, oded@dcs.bbk.ac.uk

Graeme Burnett

Enhyper Ltd.
London, United Kingdom
graeme.burnett@ieee.org

Abstract— We present ongoing work on a tool that consists of two parts: (i) A raw micro-level abstract world simulator with an interface to (ii) a 3D game engine, translator of raw abstract simulator data to photorealistic graphics. Part (i) implements a dedicated cellular automata (CA) on reconfigurable hardware (FPGA) and part (ii) interfaces with a deep learning framework for training neural networks. The bottleneck of such an architecture usually lies in the fact that transferring the state of the whole CA significantly slows down the simulation. We bypass this by sending only a small subset of the general state, which we call a 'locus of visibility', akin to a torchlight in a darkened 3D space, into the simulation. The torchlight concept exists in many games but these games generally only simulate what is in or near the locus. Our chosen architecture will enable us to simulate on a micro level outside the locus. This will give us the advantage of being able to create a larger and more fine-grained simulation which can be used to train neural networks for use in games.

Keywords— *Cellular Automata; FPGA; Simulation; Machine learning; Neural networks; Unreal Engine*

I. BACKGROUND

There have been exciting new results of training neural networks with photorealistic imagery from virtual worlds [1]. The training of these neural networks uses rendered images from virtual worlds instead of real world data, the two biggest advantages of this approach being, firstly, fewer limitations in executing potentially difficult or dangerous scenarios, and secondly, the ability to accelerate the speed of the simulation means faster training of the neural network.

In recent years, the game industry has spent a lot of effort on creating game engines which can output near-photorealistic imagery in real time, making it possible to train neural networks for real world scenarios using this output. Several projects are being developed to make it easier for neural network frameworks to interface with these engines [2], [3].

Cellular automata (CA) is an effective technique for simulating, on a micro level, complex behavior such as pedestrian traffic, moving agents [4] or, as in our proof of concept, the traffic of narrowboats¹ on a system of canals. With a simple set of rules, contained in each cell, describing the

behavior of passing agents, it is possible to get an extraordinarily complex macroscopic view of the flow of traffic [4].

CA, because of its inherent massive spatial parallelism, locality and discrete nature, is a perfect candidate for implementation on programmable FPGA (field-programmable gate array) technology [5]. FPGAs are reconfigurable hardware devices, where the set of rules contained in each cell can be described in hardware via lookup tables and flip-flops (for storing state), and with every clock cycle the state of all cells can be updated in parallel.

The remainder of this paper describes our current work, in which we are implementing a CA for simulating traffic on an FPGS, to train neural networks. These trained neural networks can be used to create game environments with 'real world'-like behavior.

II. IMPLEMENTATION OF CA MICROSIMULATION ON FPGA

We are developing a tool which microsimulates traffic in a virtual world and gives a game engine a limited view of certain areas of the world – a 'locus of visibility'. By limiting the amount of data made available for rendering and subsequent learning of a neural network, we can increase the size and granularity of the simulation, which will make the macro view more realistic.

The data of this locus of view, akin to a torchlight in a darkened 3D space, or a traffic camera used to monitor traffic on a busy crossing in a city, will be passed on to a game engine which will generate a photorealistic video feed of the exposed area. This video feed will be used for training a neural network. The simulation will run several times faster than it would normally do when a game is played, to speed up the training of the neural network.

The simulation uses the cellular automata (CA) method and a 'locus of visibility controller' extracts the localized data from the simulation and exposes it over the network to one or more consumers.

The CA will be implemented on a FPGA and communicate over PCIe to a network card (NIC), exposing it over the network. The Unreal Engine captures this data and a neural network interfaces with the game engine via UnrealCV [2], an

¹ Narrowboats were the main transportation system for goods at the start of the industrial revolution in the UK

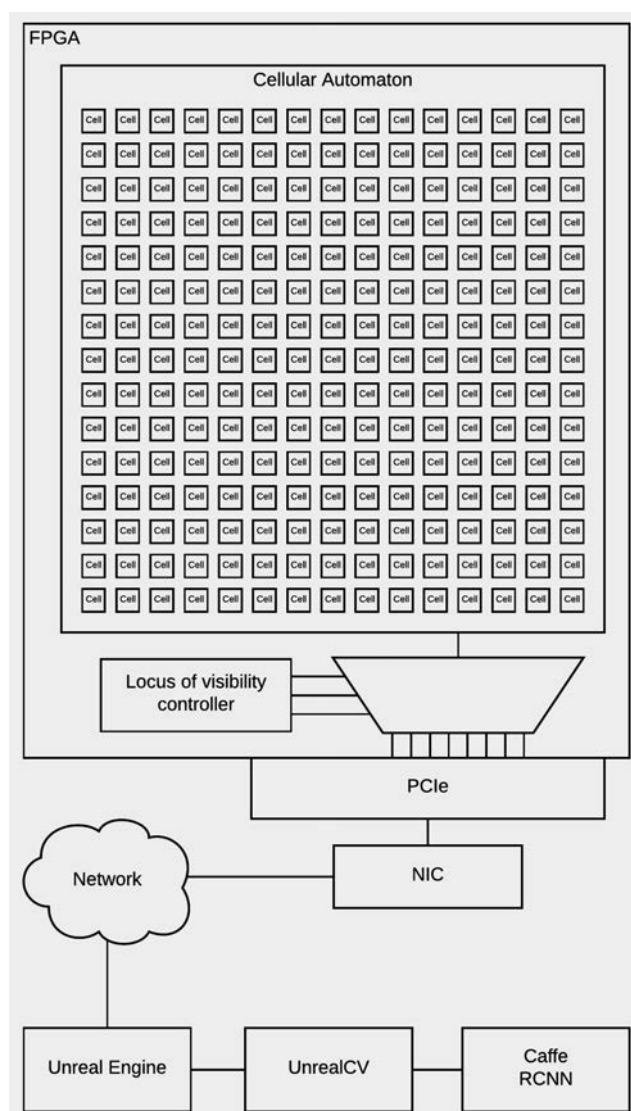


Figure 1. Simulation tool architecture

Unreal Engine plugin for interfacing with deep learning frameworks such as Caffe [6].

The simulated world is split up into hexagons and each individual area has a fixed locality to adjacent hexagons and can hold multiple agents. Each hexagon is represented by a cell in the CA. The movement of agents and the state of the cell is determined by the rule set contained in every individual cell.

A. Proof of concept: Narrowboat Simulator

In our first proof of concept we will simulate the traffic of a large number of narrowboats on an intricate canal system, transporting resources (for example, coal or grain) between supply points and delivery points in competing economic areas. The flow and density of traffic can be regulated via locks (chambered gates used to raise or lower water level, allowing boats to move to higher or lower levels of a canal), distributed throughout the canal system.

The neural network first trains itself by observing the traffic at several locks. Once trained, the neural network will be used to play a game in which it competes against a (human) opponent, to bring as many resources as possible to its own delivery points, manipulating the traffic by operating the locks.

FUTURE WORK

The implementation of the narrowboat simulator will be a starting point from which to build more complex simulations of growing cities with different transportation systems, interacting with each other. This city transport simulator can then be used to train neural networks to operate different aspects of the simulation, for example, resource management. Both trained neural networks and simulator can be used to create games in which the environments and elements within them exhibit more complex, 'real world'-like behavior.

REFERENCES

- [1] "Artificial intelligence: Why AI researchers like video games | The Economist," *The Economist*, 11-May-2017. Available: <http://www.economist.com/news/science-and-technology/21721890-games-help-them-understand-reality-why-ai-researchers-video-games>. [Accessed: 15-May-2017].
- [2] W. Qiu and A. Yuille, "UnrealCV: Connecting Computer Vision to Unreal Engine," in *Computer Vision – ECCV 2016 Workshops*, 2016, pp. 909–916.
- [3] H. Kinsley, *pygta5: Explorations of Using Python to play Grand Theft Auto 5*. 2017.
- [4] V. J. Blue and J. L. Adler, "Cellular automata microsimulation for modeling bi-directional pedestrian walkways," *Transp. Res. Part B Methodol.*, vol. 35, no. 3, pp. 293–312, Mar. 2001.
- [5] M. Halbach and R. Hoffmann, "Implementing cellular automata in FPGA logic," in *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, 2004, p. 258-.
- [6] Y. Jia *et al.*, "Caffe: Convolutional Architecture for Fast Feature Embedding," in *Proceedings of the 22nd ACM International Conference on Multimedia*, New York, NY, USA, 2014, pp. 675–678.