Hardware speculation for robotic plants through cellular automata principle

Marilia Lyra Bergamo[1] and Sandro Benigno[2]

[1]University of Newcastle, Newcastle, NSW 2300
[2]Federal University of Minas Gerais, Belo Horizonte, MG 31270-901
*marilia.lyrabergamo@newcastle.edu.au

Abstract

Assuming that plants behave like colonies with many independent but connected parts, this text describes a board design to emulate a cellular automata principle as a solution to biomimetic those organisms. The developed hardware speculates the behaviour of nodes as a result of the conditions of their neighbours, simulating the concept of a small world. Each designed board easily accommodates one to four neighbours connected by cable wiring. A firmware was also produced. It helped to read the states of the neighbouring sensors and actuators before a node made a local decision. Using the board, we built a robotic plant composed of five flowers/nodes whose individual behaviour is codependent on the neighbours' behaviour. This extended abstract presents the preliminary results of this experiment.

Introduction

The field named neurobiology recognises that plants evolved as sessile organisms. Hence, their bodies are modular without individual organs, and this distinction from the animal world has created obstacles to realising them as intelligent organisms Mancuso (2018). In terms of decision-making, plants frequently reveal opposite strategies of those produced by animals. Still, they can also mimic and negotiate, and their intelligence resembles networks. Still, according to Mancuso, approaches in design and engineering are generally based on animal functions, with some exceptions like Mazzolai et al. (2014); Mazzolai (2016) or FloraRobotica (2018). In several situations, it may not be a technological advance.

Focusing on intelligence as a network, we imagine that developing a robotic plant could take advantage of the cellular automata concept used recently in ecology and evolutionary computing research, such as Jiaxin et al. (2023) and Plantec et al. (2022). The concept of cellular automata is typically attributed to Stanisław Ulam and John von Neumann. According to Shiffman (2012), Ulam was studying the growth of crystals, and von Neumann imagined a world of self-replicating robots. Those ideas were the main reason the concept of cellular automata was used to envision robotic plants, due to this idea of self-replication units as an evolutionary behaviour adapted to a sessile life.

Board Design

The hardware designed is an octagonal robotic cell (Fig. 1a) and it is an open-source project by Sandro (2021a). Its creation is oriented to the interconnection of multiple units to form a network with interdependent entities. Each cell has local sensing capability, and the collected data can be shared among its neighbours, as in cellular automata models. Radial positioning of the physical ports of the cell characterises the board. There is a door on each edge, according to the cardinal directions. The primary system resources are distributed in these eight ports: power supply, communication, sensing and external activation.

The four neighbours are addressed on the four sides (Fig. 1b), corresponding to the four communication and power ports (terminals with screws): North, East, South and West. The sensing and activation ports (pin bars) are on the diagonal edges: Northeast, Southeast, Southwest and Northwest. Interconnect ports ensure data and power sharing. Its power rails are peripheral and were sized to conduct higher currents at higher voltages than the inside of the board. This design makes it possible to chain more cells per power supply. A power source connected to a periphery cell will distribute the power to all the others in the network. Each cell makes a local voltage conversion regulated to the internal circuitry. Cell communication occurs in two ways (RX/TX), using only an asynchronous serial peripheral (UART). The transmission path (TX) is unique, the cell constantly makes its data available, and it is up to neighbouring cells to determine when they will capture it. The receive path (RX) is multiplexed into four input (listening) points. At each reading cycle, the cell erases its input buffer and commands the multiplexer to connect the RX to one of the four inputs: N, E, S or W. A state machine detects the data packet's header, iterating over the bytes in the input buffer. Upon seeing the header, the state machine captures all information about the neighbouring cell and its sensors. Finally, the data is verified before storing them in a representation instance corresponding to the neighbour.

The board's firmware has two main classes: $Being$ and $SerialSMV1$, can be accessed at Sandro (2021b). The

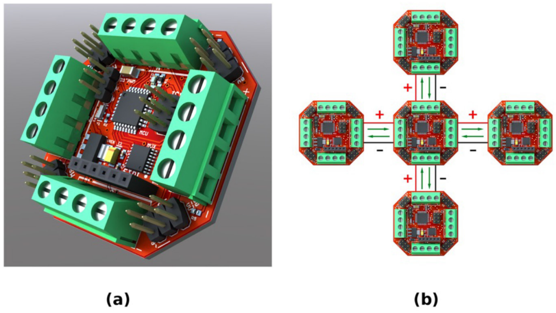

Figure 1: On the left (a) is the board's design, and on the right (b) is the representation of network architecture.

$Being$ class defines the attributes of a robotic cell. The $SerialSMV1$ class defines the state machine that performs the iterations of capturing and interpreting data in the contents of the serial input buffer. Each cell defines and instantiates itself and four simulacra, which are representations of its neighbors. Each of these instances is a repository of cell characteristics and states, as well as their respective sensing data. In addition to information on which types of sensors are active, the $Being$ class also has an internal structure called $SensingData$. It stores data from sensors according to their kind. Sensor types are enumerated, corresponding to their trigger bit position, in a 64-bit map (Activated Sensor BitMap or ASBM). This map is present in the header of communication data packets between cells. Each bit of the map indicates which sensor types are active in the current cell (one for active and $zero$ for inactive). As for the sensing and triggering ports, at the north end, the two diagonal ports have analog input (ADC) capability, one capable of communication via the i2C protocol. The two diagonal ports at the south end are digital, one of which has two channels capable of pulse width modulated (PWM) output. A dedicated PWM port inside the board, with a filtering capacitor, reduces the noise caused when driving micro-servos. Microcontroller programming can be straightforward via the ICSP port (which also gives access to external SPI devices) or via the bootloader via a UART connection port.

Robotic Plant experiment

Figure 2 presents a sculpture developed for EmMeio14 Exhibition MediaLab-UnB (2022) using the hardware described in the previous section. In this extended abstract, we will not deal with the aesthetic issues that involved the distribution and final form of the plant robot. Still, we will focus on the grid construction of cells that make up the object, the possible states and the neighbourhood condition of the nodes/flowers. The structure presented in figure 2 is a simple one-dimensional grid: a line of cells. Five nodes were created using the hardware; two small flowers were only extensions of one of the nodes without the capacity for local decision-make. The nodes had four states: generating energy (defined by purple lighting), being pollinated (designated by blue lighting and triggered by interruption of light capture by its sensors), critical state (characterised by red lighting, activation of vibration actuators) and finally, dead (defined by white lighting). A random initial state instantiates each node among the existing four. From this grid, each node/flower would follow the following logic: if my neighbour on the right is active (purple), I am also purple; if my neighbour on the right is being pollinated (blue), I remain in an active state (purple), if my two neighbours are being pollinated, I go into a critical condition (red), if I am pollinated, and one of my neighbours is in a critical state, I die (white).

Figure 2: Sculpture developed for EmMeio14 Exhibition MediaLab-UnB (2022) using five boards of the described hardware of Fig. 1

The preliminary results demonstrate that the network of nodes can grow as a grid: each node with four neighbours. Consequently, the system maintains a consistent internal logic and is conducive to developing patterns. Studying the patterns generated by the system was yet to be possible during this first experiment. However, establishing this logic creates conditions for unfolding robotic plants organized in a network through wiring and physical structures that can be discarded and repositioned at other points in the network. This metaphor relates to the replication and growth of similar structures in a sessile organism. This research follows: observe patterns in a one-dimensional grid of robotic cellular automaton and create a more complex grid of nodes of four neighbours for nodes to understand differences in pattern generation.

References

FloraRobotica (2018). About floratobotica. Project website available at https://www.florarobotica.eu.

Jiaxin, M., Huping, H., Shaoliang, Z., Yifei, H., Yongjun, Y., Yuanli, Z., and Zhongyi, D. (2023). Detecting long-term effects of mining-induced ground deformation on plant succession in semi-arid areas using a cellular automata model. *Ecological Indicators*, 151:110–290.

Mancuso, S. (2018). *The Revolutionary Genius of Plants: A New Understanding of Plant Intelligence and Behavior*. Atria Books, New York.

Mazzolai, B. (2016). Plant-inspired growing robots. In Laschi, C., Rossiter, J., Iida, F., Cianchetti, M., and Margheri, L., editors, *Soft Robotics: Trends, Applications and Challenges. Biosystems and Biorobotics*, page 57–63. Springer, Cham.

Mazzolai, B., Mattoli, V., Beccai, L., and Sinibaldi, E. (2014). Emerging technologies inspired by plants. In Cingolani, R., editor, *Bioinspired Approaches for Human-Centric Technologies*, page 111–132. Springer, Cham.

MediaLab-UnB (2022). 21.art - 21 encontro internacional de arte e tecnologia; ix siimi - ix simpósio internacional de inovação em mídias interativas; exposição emmeio 14 (galeria térreo). Available at http://medialab.unb.br/index.php/categorias/30-21-art.

Plantec, E., Hamon, G., Etcheverry, M., Oudeyer, P., Moulin-Frier, C., and Chan, B. (2022). Flow-lenia: Towards open-ended evolution in cellular automata through mass conservation and parameter localization. arXiv:2212.07906 available in https://arxiv.org/abs/2212.07906.

Sandro, B. (2021a). Windrose. The WindRose project available at https://github.com/sandrobenigno/WindRose.

Sandro, B. (2021b). Windrose's firmware class description and use. The WindRose Firmware available at https://sandrobenigno.github.io/WindRose/html/index.html.

Shiffman, D. (2012). *The Nature of Code*. Creative Commons Attribution-NonCommercial 3.0 Unported License, California.