CULET: Cubelet Lego Turing Machine

Ricardo Q. Figueroa  
Artificial Life Robotics Lab, Escuela Superior de Cómputo, Instituto Politécnico Nacional, México

Daniel A. Zamorano  
Artificial Life Robotics Lab, Escuela Superior de Cómputo, Instituto Politécnico Nacional, México

Genaro J. Martínez  
Artificial Life Robotics Lab, Escuela Superior de Cómputo, Instituto Politécnico Nacional, México, Unconventional Computing Laboratory, University of the West of England, Bristol, United Kingdom

Andrew Adamatzky  
Unconventional Computing Laboratory, University of the West of England, Bristol, United Kingdom

E-mail: qf7.ricardo@gmail.com, daz23ayala@gmail.com, genaro.martinez@uwe.ac.uk, andrew.adamatzky@uwe.ac.uk

Abstract

In this paper, we show the construction of CULET: a robot that simulates the execution of any 2-symbols Turing machine. CULET is constructed with Cubelets (small autonomous robot-cubes used for teaching basic robotics and programming for kids) and Lego bricks. Cubelets are a kind of high level programming where you do not use traditional code but rather achieves different robots trough the concatenation of blocks, i.e. robots assembled. CULET is designed to perform the read and write of a Turing machine.

Keywords: Turing machine, Robotics, Cubelets, Lego, Cellular Automata.

1. Introduction

In the history of Turing machines an ample number of physical mechanical devices have been developed. Indeed, during the Turing's 100th birthday, Lego built an autonomous mechanical Turing machine adding binary numbers in 2012 year². So we, inspired in the idea of machines self-reproducing from cellular automata theory², construct the robot CULET, a robot based in small robots-cube Cubelets and some pieces Lego. The main feature of this proposal is that it is the first machine constructed with robots and bricks. This way, this robotic machine is able to simulate the function of other machine: machines simulating machines. Cubelets, helped with pieces Lego, proof how a robotic machine is able to simulate an abstract computable machine. The brain of this robot is concentrated in a cube which can read and other two cubes that move the head and the arm (write). Of course, these cubes need be re-programed for each table of transitions.

For the implementations we programming two special Turing machines. The first one is a universal Turing machine able to simulate the behaviour of a complex elementary universal cellular automaton known as Rule 110³. The second Turing machine programmed in CULET doubles the number of ones in the tape.
The paper has the next structure. In Section 2 we introduce quickly the Cubelet robots and describe the Turing machines implemented. Section 3 describes the design (hardware and software) and finally in Section 4 we give some final remarks.

2. Preliminaries

2.1 The Cubelets

The Cubelets are small cubes designed to teach fundamentals of robotics and programming to kids. Each cube has a different functionality; the general idea is to construct robots only through the concatenation of different cubes, this way Cubelets can be represented as a formal language, where every valid construction is a high level of program, and been able of design non-von-Neumann-style computers. The cellular automata consist of a grid of square cells, each of which is a finite-state machine associated with a neighborhood of eight cells (Moore neighborhood). The popularity of this automata comes from his very complex and unpredictable behavior evolving in two dimensions. Von Neumann was developing a very complex automaton with 29 states and being able of design decentralized systems and been able of design non-von-Neumann-style computers. The cellular automata were invented by von Neumann in Ref. 2 as, ironically, a mean for study decentralized systems and been able of design non-von-Neumann-style computers. The cellular automata consist of a grid of square cells, each of which is a finite-state automaton (the same rules for all the cells simultaneously) and where the state of each cell in the next generation depends on his current state and symbol, and the last one indicates the direction of movement³.

One of the greatest achievements of Turing’s work is the conceptualization of the so called universal machines. A universal machine U can do everything that any machine can do. U receives as input two things, a description of a machine M, and the input for that machine M; and, as output U puts on the tape the same thing that the machine A would put when have as input M.

2.2 Universal Turing machines

The Turing machines was invented by Turing in 1936⁴ to solve problems about completeness and decidability. A Turing machine is a finite-state machine associated with an external memory medium, an infinite tape divided in

\[ (q_0, s_0, q_{i+1}, s_{i+1}, d) \]  

(1)

Where \( q \) is the set of states, \( s \) is the set of symbols and \( d \) indicates the direction of movement of the head in the tape (right or left); \( i \) is the time step. So, the first two elements of the quintuple indicate the current state and symbol, the next two elements indicate the following state and symbol, and the last one indicates the direction of movement⁵.

The Turing machine is a set of quintuples of the form:

\[ (q_0, s_0, q_{i+1}, s_{i+1}, d) \]  

(1)

© The 2019 International Conference on Artificial Life and Robotics (ICAROB2019), Jan. 10-13, B-Con Plaza, Beppu, Oita, Japan

---

³ Some other standard literature, like Ref. 8, defines the machine in terms of quadruples instead of quintuples.

⁴ The 2019 International Conference on Artificial Life and Robotics (ICAROB2019), Jan. 10-13, B-Con Plaza, Beppu, Oita, Japan
Wolfram hypothesis was that if we cannot understand such minimal models, then we could never understand bigger ones. The interesting part comes when even in those small automata are still, in some of them, very complex behavior.

2.4 Universal ECA: rule 110

In Ref. 12 Cook proofed that the elementary cellular automaton Rule 110 is universal. The proof consists in showing how, with particle interactions typical of the rule, emulate the operation of a cyclic tag system. This one is the smallest cellular automaton proofed computationally universal simulating a very sophisticated sequential machine by particle collisions in one dimension. This way, a new machine was invented to simulate a computable function known as cyclic tag system in this automaton, for details see Ref. 12 and 14. This way, Rule 110 can simulate a cyclic tag system, cyclic tag systems can simulate tag systems, and a Turing machine (developed by Eppstein in Ref. 12) can simulate Rule 110 and therefore such a Turing machine is universal. Finally, CULET can simulate the universal Turing machine which simulate the behavior of Rule 110, complex behavior, and collisions of particles.

Table 1. 7x2 Rule 110 Turing Machine.

|   | 0       | 1       |
|---|---------|---------|
| S00 | (0, T00, R) | (1, T01, R) |
| S01 | (1, T00, R) | (1, T10, R) |
| S11 | (1, T00, R) | (0, T11, R) |
| S12 | (0, T01, L) | (1, T00, L) |
| T00 | (1, S00, R) | (0, S11, L) |
| T01 | (1, S01, R) | X |
| T11 | (1, S11, R) | X |

The Table 1 shows the transition table of the machine implemented. Every generation in which the head of the machine reaches a right end, a new generation of the rule 110 is computed. There are some considerations with respect to this machine: the initial condition and the representation of the cellular automata symbols. The initial condition of the machine must be, from left to right, an infinite string of zeros, the encoded version of the initial that the machine is going to emulate, and an infinite repetition of the string “10”. A “1” of the cellular automata is represented like “11” in the machine, and a “0” like “00”; the symbol “10” is used like a blank symbol.

2.5 A duplicator Turing machine

A duplicator Turing machine was used by Rendell as part of the Game of Life cellular automaton, for details see Ref. 17. We implement this machine in CULET. In Fig. 1, we show a 3-symbols version: it starts advancing to the right, for each one scanned, it put another one in the left until there is no more ones to the right. For the implementation in the CULET we used a 2-symbols version of this machine.

3. CULET Design

3.1 General description

In the Fig. 2.1 we show a snapshot of CULET. The tape is represented as an array of Lego blocks over rectangles of black paper; the position of the block (front or back) indicates the symbol on the cell (zero or one respectively). The Cubelet car moves through the tape reading and writing in the tape; the read process is achieved with a distance sensor that test the position of the Lego block under scanning; the writing is done with a lever of Lego and a rotate cube: depending on the desired symbol, the leaver push or pull the block of the tape. The movement across the tape is done with a couple of wheels that move in right or left direction. In the backwards of the car is attached a rail that helps to maintain the alienation of the car with respect to the tape.

3.2 Hardware

In the Fig 2.2 we show a picture of the cubes needed by the CULET: a couple of batteries, a couple of drives, two distance sensors, one rotator, one passive and one light. In figure 2.3 are shown the Lego components needed (from left to right): a rail of 4 blocks width; a stabilizer car (it must go over the rail); a leaver; the symbols of the

© The 2019 International Conference on Artificial Life and Robotics (ICAROB2019), Jan. 10-13, Beppu Plaza, Beppu, Oita, Japan
cells.

3.3 Software

As we point out in the previous section, we reprogram some of the cubes for obtain the desired behavior. The central cube, the one that has encoded the production rules of the Turing machine, is the rotator. In the state diagram of the Fig. 3 we show the stages of the program of this cube. The scan operation is achieving reading the value of the sensor that is pointing to the cell; the other sensor, the one that is pointing up, control the movement of the leaver: it establishes when stop moving the rotator motor; both sensors have the original program. The other two cubes that have custom programs are the motors; they receive instructions from the rotator: if they receive a 1, they move to the right; if they receive a 2, they move to the left; a 0 indicates stop moving.

The car starts with a cell below the reading sensor; it scans the position of the block and determine the next action: write a one, write a zero, or doesn’t write anything; then, it moves left or right and start over again. The decisions of this automaton (the symbols to write, the directions to move) are determined accordingly to the automaton of the machine that is being executed; thus, in any moment, there are two descriptions of the program working at different levels: the low level one, that determines the behavior of the hardware components (move left, move right, scan, write); and the high level one, which is given by the rules of the Turing Machine.

4. Final Remarks

We develop a robotics Turing machine named CULET, able to simulate any 2-symbols Turing machine. CULET is constructed with autonomous cube robots Cubelets and Lego bricks. A video showing the function of the Turing machine simulating Rule 110 with CULET is available in Ref. 18 and the duplicator Turing machine in Ref 19. The full code to reprogram Cubelets is available from Ref. 20. In the future work, we will develop machines reading more than two states.

References

1. Lego Turing Machine. [https://www.ecalpemos.nl/filmmaking/lego-turing-machine/], consulted in 04/10/2018.
2. J. von Neumann, Theory of Self-Reproducing Automata, edited and completed by A.W. Burks (University of Illinois Press, London, 1966).
3. Rule 110 repository. [http://uncomp.usw.ac.uk/genaro/Rule110.html], Consulted in 05/10/2018.
4. The Cubelets. [https://www.modrobotics.com/cubelets/], Consulted in 3/10/2018.
5. Cubelets API. [http://api.modrobotics.com/cubelets/], Consulted in 3/10/2018.
6. A.M. Turing, On Computable Numbers with an Application to the Entscheidungsproblem, in Proc. of the London Mathematical Society 2(1) (1936), pp. 230-265.
7. M. Minsky, Computation: finite and infinite machines (Prentice-Hall, 1967).
8. M. Davis, Computability and Unsolvability (McGraw-Hill, 1958).
9. M. Gardner, Mathematical Games: The Fantastic Combinations of John H. Conway’s New Solitaire Game ‘Life’, in Scientific American (224) (1970), pp. 120-123.
10. E.R. Berlekamp, J.H. Conway, R.K. Guy, Winning Ways for Your Mathematical Plays, 2nd ed. (Academic Press, London, 1982).
11. S. Wolfram, Cellular Automata and Complexity (Addison-Wesley Publishing Company, 1994)
12. M. Cook, Universality in Elementary Cellular Automata in Complex Systems 15(1) (2004), pp. 1-40.
13. A. Adamatzky, Collision-Based Computing (Springer, 2002).
14. S. Wolfram, A New Kind of Science (Wolfram Media, Champaign Illinois, 2002).
15. G.J. Martínez, J.C.S.T. Mora, H. Zenil, Computation and Universality: Class IV versus Class III Cellular Automata 7(5-6) (2013) pp. 393-430.
16. G.J. Martínez, A. Adamatzky, H.V. McIntosh, A Computation in a Cellular Automaton Collider Rule 110, In Advances in Unconventional Computing: Volume I Theory, edited by A. Adamatzky (Springer, 2017), pp. 391-428.
17. P. Rendell, Turing Machine Universality of the Game of Life (Springer, 2016).
18. CULET: Cubelets Lego Universal Turing Machine ECA Rule 110, [https://www.youtube.com/watch?v=GIQDA5GnYsu], Consulted in 05/10/2018.
19. CULET: Cubelets Lego Doubling Turing Machine, [https://www.youtube.com/watch?v=CrUkKjwz34w], Consulted in 05/10/2018.
20. Code developed for CULET, [http://www.comunidad.escom.ipn.mx/Alirob/CULET/], Consulted in 03/12/2018.

Figure 3. State diagram of the main program.