Thermal bead 3D printing: easy generation and realization of artistic 3D models for primary school pupils

Marco Scirea
Maersk Mc-Kinney Moller Institute Institute,
Embodied Systems for Robotics and Learning
Campusvej 55
Odense, Denmark 5230
msc@mmmi.sdu.dk

Andrea Valente
Maersk Mc-Kinney Moller Institute Institute,
Embodied Systems for Robotics and Learning
Campusvej 55
Odense, Denmark 5230
anva@mmmi.sdu.dk

ABSTRACT

This paper describes a system for generating low resolution 3D models that can be "printed" through plastic beads. The final objective is creating a system to facilitate Computational Thinking education for primary school pupils.

KEYWORDS

3D model generation, Computational Thinking, Low-fi 3D printing

ACM Reference format:
Marco Scirea and Andrea Valente. 2016. Thermal bead 3D printing: easy generation and realization of artistic 3D models for primary school pupils. In *Proceedings of FDG, Malmö, Sweden, August 7-10, 2018 (FDG'18),* 2 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Computational Thinking is increasingly present in Denmark primary schools, and in some courses introduction to programming is coupled with robotics, to make it more concrete and tangible for the pupils. Virtual and mixed reality are also often experimented with, to engage learners and support more intuitive interaction with 3D worlds and artifacts. This work looks at constructionist (Papert [1]) and attempts to combine generation of 3D artifacts with simple and inexpensive forms of 3D printing.

The user scenario we envision is for pupils in class (or in a Danish afternoon *club*) to created a pixel-art image, then choose a procedural generation agent (from a palette) and obtain a 3D shape. The shape is exported to an interactive 3D environment to be explored, for example Minecraft or similar sandbox games. The 3D exploration program also exports tomographic images ("horizontal slices") of each shape that pupils can use as plastic bead templates; in this way even complex 3D artifacts can be manually recreated as physical objects. Plastic beads are usually placed on a pinboard, then melted into a single flat plastic object via a clothes iron; to create a 3D shape the process will have to be repeated for each horizontal slice. We see this as a low-resolution, inexpensive kind of 3D printing (Figure 2).

We expect that the aesthetic nature of the generation process should provide a more gender-neutral experience than is usually found with primary school programming courses; the activities described here are closer to design and tinkering with plastic beads interests a wider range of pupils than scientific or mathematical-related activities. In the future we would like to connect our procedural generator with an Arduino/LEGO machine capable of actually placing the beads on the board by itself, making the low-resolution 3D printing at least semi-automatic.

In our scenario Computational Thinking is supported by the fact that the procedural generation agents can be used as functions, composed together to create more complex behaviors. Moreover, in the next iteration of our prototype we will implement a GUI to breed new agents starting from the initial population, and a simple rule-based language to specify the behavior of agents. Pupils will then have access to both an organic and a programming way to create agents.

2 THE PROTOTYPE

A prototype has been developed to generate shapes to be printed using the plastic beads[1]. The system implements a deterministic algorithm that takes inspiration from the Diamond-Square algorithm [3] and Cellular Automata (CA) [4]. While this paper does not have the scope to properly describe these two methods, we think it's important to at least describe their basic ideas. Diamond-Square is an algorithm to create 2D fractals, if the reader is interested in learning more about the algorithm's history we suggest reading Fisher et al. [2].

CA is a discrete model (generally) applied to a 2D grid of cells, each of which can represent a finite number of states (often on/off). The grid is initialized with some values (time $t = 0$), after that each subsequent time-step some **rules** are applied to decide the state of each cell based on its neighbors' state. CAs are commonly known to create, as outlined by Wolfram [4], four classes of behavior: (i) patterns that stabilize into homogeneity, (ii) patterns which evolve into stable or oscillating structures, (iii) patterns that evolve in chaotic fashion, and (iv) patterns that become extremely long and complex.

Our algorithm takes inspiration from these two techniques by implementing a 3D version of the Diamond-Square algorithm and by substituting the averaging of corners with CA-like rules. The steps of the algorithm are:

(1) Initialize the 3D matrix used in the generation, as with the classical Diamond-Square, the sides of the cube must be $2^n + 1$.

[1] Download the Windows build at http://msci.itu.dk/Pearler.zip

FDG'18, Malmö, Sweden

DOI: 10.1145/nnnnnnn.nnnnnnn

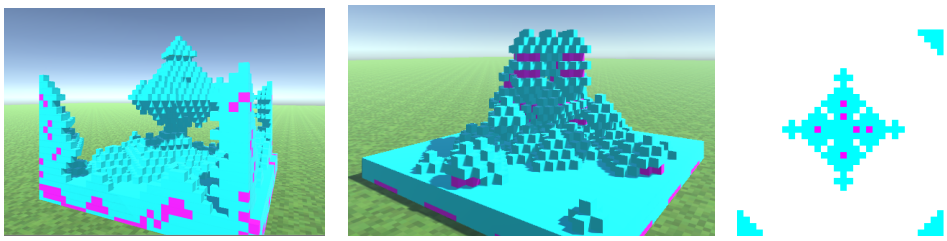


Figure 1: Two examples of generated structures based on a cube of side 33 and a horizontal slice.

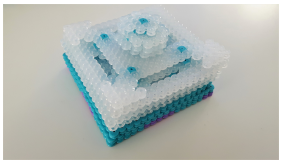


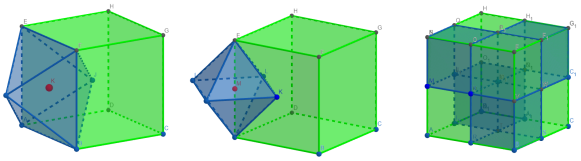
Figure 2: A physical model of a generated structure.

(2) Perform the first Octahedron step (Figure 3a): this corresponds to the diamond step, we calculate the midpoint of the cube, which splits it in 6 half-octahedrons (regular 4-sided base pyramid).
(3) Perform the second Octahedron step (Figure 3b): we calculate the midpoints of the previously generated octahedrons, these points fall in the center of the faces of the cube(s). This step is not enough to go back to cubes, since we are still missing the midpoints lying on the edges of the cube. Still by now possessing the faces midpoints we can see how now a set of 16 octahedrons can be constructed on all of the cube's edges, which leads to…
(4) The final Cube step (Figure 3c), for each previously created octahedron, we calculate its midpoint. This way we obtain all the midpoints of the cube's edges and finally we have split each of the original cubes into 4 new cubes with side $side_n = side_{n-1}/2$.

The CA inspired part comes into play whenever we have to calculate the value of the midpoints of a polyhedron based on the its vertices. In this first prototype, the rules are generated randomly, but in the future the user will be able to influence them either directly (through an editor) or indirectly (e.g. through a genetic recombination of two set of rules). The system currently supports three states: *empty, cyan bead, and magenta bead.*

An important difference with the classic implementation of Diamond-Square is that we decided to execute the algorithm starting from a specific side of the cube. This choice is motivated by our intention of giving the user the option to either generate or provide a starting image 2D image (placed as the bottom level of the cube) as a "seed" for the generative algorithm.

A final step we should discuss is the introduction of a post-processing "filter". This filter has been introduced to increase the connectivity of the generated structures and limit "disconnected elements". With the final objective of creating printable 3D models (Figure 2), it is important that the generated structure can be constructed in practice and that it has a reasonable structural integrity. This filter works in the same way as a cellular automaton, with these simple rule: *if there are more than 2 neighbors and the current cell is empty, fill in the cell using onthe same rules used by the 3D Diamond-Square.*



(a) (b) (c)

Figure 3: In *(a)* you can observe the construction of octahedrons on a face. Once the midpoint (in red) is calculated, then we can construct the edge octahedrons *(b)*. Finally once we have the midpoint of the edge octahedron we have split the original cube in eight *(c)*.

3 CONCLUSIONS

While the system described is still in very early stages, and is lacking in much of the interactivity necessary for pupils to really engage with it, we believe the prototype supports our scenario where a customizable algorithm may be used to generate interesting physical 3D models, and finally export a template to build them with plastic beads. One of the key features of the algorithm we described is that, while not being particularly novel, it supports a high degree in customization (through the CA rules) while also being able to create diverse and interesting shapes. Finally, some issues remain with creating structures that render properly in the physical medium, which would require additional study. Future works involves iterative co-design of the complete system in Danish schools and afternoon school-clubs, using focus groups.

REFERENCES

[1] E Ackermann. 2001. Piagetś constructivism, Papertś constructionism: Whatś the difference? *Future of learning group publication* (2001), 85–94.
[2] Yuval Fisher, Michael McGuire, Richard F Voss, Michael F Barnsley, Robert L Devaney, and Benoit B Mandelbrot. 2012. *The science of fractal images.* Springer Science & Business Media.
[3] Alain Fournier, Don Fussell, and Loren Carpenter. 1982. Computer rendering of stochastic models. *Commun. ACM* 25, 6 (1982), 371–384.
[4] Stephen Wolfram. 1983. Statistical mechanics of cellular automata. *Reviews of modern physics* 55, 3 (1983), 601.