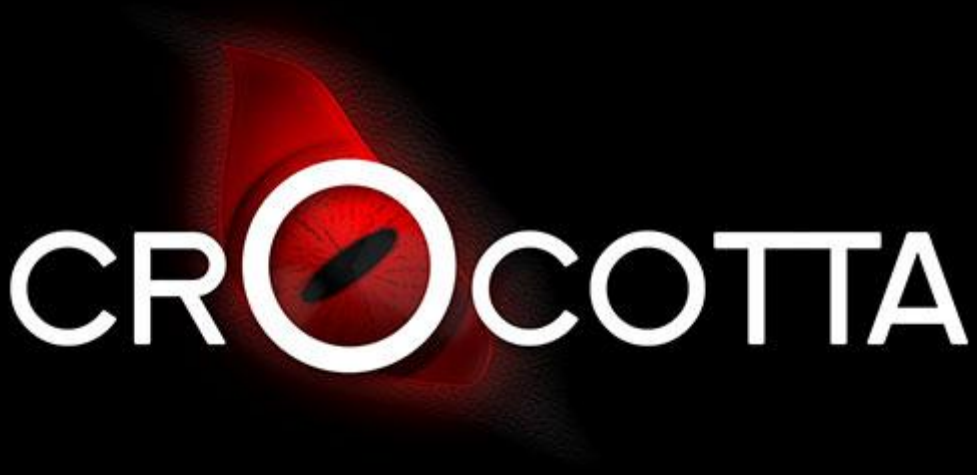

Crocotta R&D Limited
10 Wellington Street, Cambridge CB1 1HW, United Kingdom

“Be ambitious of climbing up to the difficult, in a manner inaccessible...”

We are a small team of international researchers with the aim of conducting leaps in technologies like visualization, virtual reality, artificial intelligence, and robotics.

www.crocotta.co.uk
crocotta@crocotta.co.uk
+44 20 3239 7007

Crocotta R&D Limited
10 Wellington Street, Cambridge CB1 1HW, United Kingdom
Registered number 8451298
www.crocotta.co.uk

VIRTUAL SYNTHESIS
GENERATING VIRTUAL (IN-) ORGANIC MATERIAL PATTERNS

By Crocotta R&D

Robert Sugar[1]

FOREWORD

Procedural generation is a widely used term in the production of media: it refers to content generated algorithmically rather than manually by humans. We present a novel approach in which we synthetize virtual (in-) organic patterns as well as the design and the material structure of complex virtual objects. These virtual objects contain a high degree of complexity from a topological aspect and from a materials distribution aspect as well. Our aim is to create complex forms beyond the capacity of the human mind.

INTRODUCTION

Procedural Generation refers to content that has been generated by an algorithm instead of manually designing it. The content is generated at run-time usually when an event occurs such as - the player reaches the end of the visible scene and more needs to be generated.

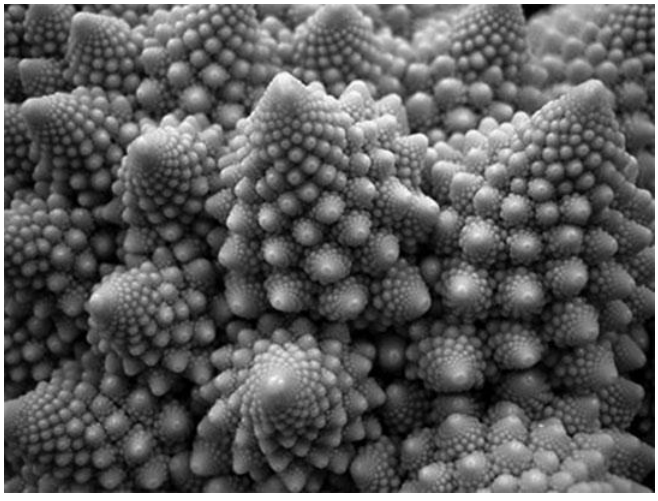

Figure 1:
Using Lindenmayer System to generate the structure of a broccoli plant.

Procedural generation was implemented in early computer graphics applications as a solution for the limited memory storage of computers at the time. It has a wide field of applications in computer graphics but also comes with several issues and constraints which are mentioned in this paper.

With today's technology and a trend towards the constantly increasing amount of computational power & storage of CPU's/GPU's, users expect a higher level of detail in graphics, more uniquely distinguishable objects, high definition resolution and exceptional visual effects in computer games. This is where procedural generation reaches its limits - It might be easy to create one single tree with procedural generation using a rule based system - but the generation of a realistic forest, where the different trees adapt to each other and this eventually leads to visible changes in the trees structure, is not at all a trivial problem. Another issue with procedural generation is that the results usually aren't realistic enough and are very unpredictable due to the pseudo random nature of the functions being used. These are some of the reasons why we need a new method for procedural generation of structures.

METHODS

Current methods for procedurally generated content include fractals, image analysis and synthesis, and content generated using (pseudo) randomized functions. This paper will go over the aforementioned methods and outline issues which render them unusable for more general use.

Fractals & Noise

Fractals are used to describe a broad set of shapes. What makes them interesting is the fact that they have a non-integer or fractal dimension. Each part of the fractal has the same characteristics as the whole. They are frequently used in procedural content generation with the main field of application being terrain generation. The three most famous and widely used algorithms in PCG being:

- Diamond-Square (a.k.a. Midpoint Displacement)
- fractional Brownian Noise (ex. using Perlin noise)
- Lindenmayer system (or L-system)

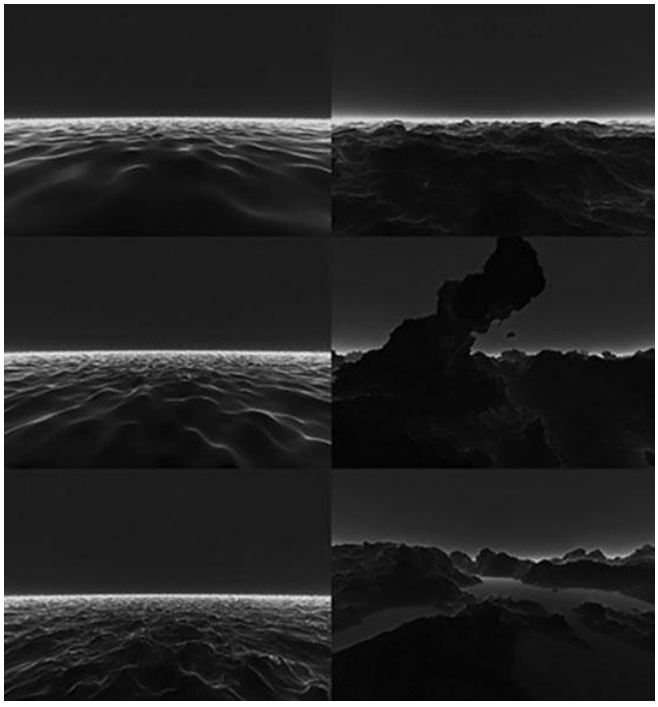

Figure 2:
Terrains based on fBM Noise.

All the previous methods are capable to draw realistic natural objects - but mostly limited to landscapes, trees, and some plants - on a computer screen.

The issue here is that fractals as well as noise are not applicable to more general structures other than the previously mentioned and if the structure needs to be locally modified this may lead to global changes which cannot be predicted

Less useful for representing structures where a change in local attributes might be necessary. While this might be acceptable for demonstrations of computer graphics applications or computer games this is not well suited for any type of accurate simulation.

Image Analysis and Synthesis

This method involves implementing a simple AI to perform learning techniques - based on an input set of images it classifies and groups them. The first part of the method being the analysis which involves training the AI and the second is the synthesis. The synthesis part is just basically generating images which correspond to certain parameters of an appropriate set of images determined by the analysis procedure. The images are generated using interpolation and sampling.

The method as a whole is slow and the results in most cases do not correspond to the original image at all as the structural details of the image are discarded in the synthesis procedure. There are solutions to this problem which include other complex methods such as Gaussian Markov random field models. While these solve most of the issue of a seamlessly repeating pattern, though still ignore structural details, for image and texture synthesis they also introduce complex computations and further slowdown the process.

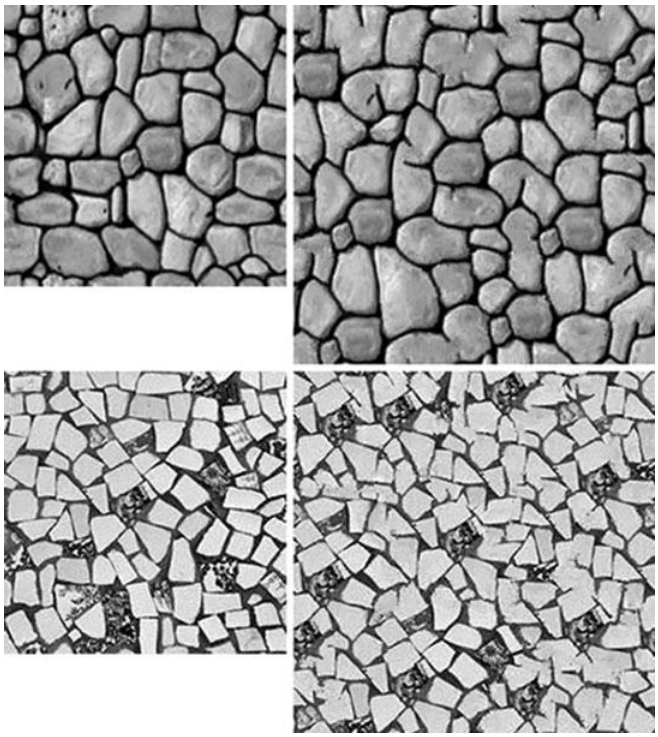

Figure 3:
Texture synthesis via a nonparametric Markov random field model.
(left) Original texture.
(right) Synthetized texture.

Genetic algorithms

Genetic algorithms (also known as evolutionary algorithms) are another approach of randomized content generation to make the newly synthesized structures or objects appear visually distinguishable from each other while still having a same set of features. Genetic algorithms are essentially modifying the parameters of an existing structure or object using random or noise functions which lead to a minimal difference between the newly synthesized structure and the original one (or set of structures).

There are two methods which genetic algorithms use for the generation of new structures - Crossover or Mutation. Crossover is just the recombination of certain parameters from the initial population and the Mutation involves a probability that some parameters in the new structure will be randomly changed.

The reason why genetic or evolutionary algorithms aren't used for real-time applications is that the results are too random and do not correspond to the users expectations, nevertheless they are extremely slow due to the enormous number of required mutations.

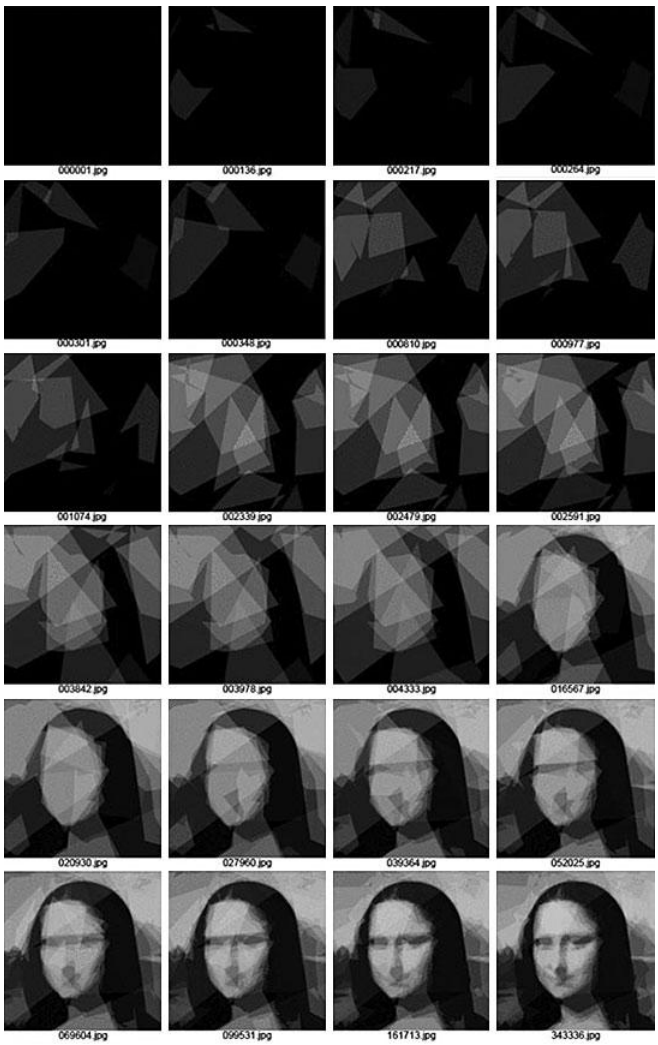

Figure 4:
A replica of the Mona Lisa using only 50 semi transparent polygons.
The procedure is simple:
0) Setup a random DNA string
1) Copy the current DNA sequence and mutate it
2) Use the new DNA to render polygons onto a canvas
3) Compare the canvas to the source image
4) If the new painting looks more like the source image than the previous painting did, then overwrite the current DNA with the new DNA
5) repeat from 1

SUMMARY

Using procedural content generation is becoming a very popular method in computer graphics and in the gaming industry. There are a lot of benefits to using this approach including:

- Enhanching the details of graphics;
- Less time & money spent on manually designing terrain & landscapes, vegetation, etc.;
- Storing less textures, structures and objects on disk.

Procedural content generation also comes with a lot of issues mentioned above and solutions to these problems are often very complex to implement or totally diminish the benefits of using PCG. Another issue with all of the aforementioned methods is that they are very good at what they do but they are also very limited in their field of application. This is where PCG hits a wall. Our goal is to solve these issues and propose a generalized approach for PCG and synthesis which is applicable to a more general set of structures and objects - and with the current trend of the continuously increasing computational power we want all of this to be able to happen in real time.

REFERENCES

Robert Sugar[1] is a scientist, researcher and IT entrepreneur the same time. He has been starting companies since 1996, ranging from software companies, media companies, computer game developer companies and internet companies.

He was born in 1978, and grew up in Hungary. He graduated in physics at the Lorand Eotvos University (Budapest). First software engineering was just his hobby and later it has become his full time profession. His first development project was about artificial intelligence and graphical visualization for computer games back in 1996. He founded his own game developing studio in 2001 - called Mithis Entertainment - in the heart of Budapest for the purposes of "AAA" game development. From a small group of enthusiastic people, Mithis Entertainment has become the biggest developer studio in Hungary by 2005 and completed four big game titles which were distributed world-wide by well-known multi-national publishers.

Since his departure from the gaming industry in 2006 he has been focusing on the researching of cutting edge technologies.

By Crocotta R&D, July 2012