GDC | March 21-25, 2022
San Francisco, CA

Turing Patterns and Fractals: Program Generation of Natural Textures

Yiheng Zhou
LeoCool

Qidan Chen
Thunderfire UX

#GDC22

Content

Background & Introduction

Main Problem

With the continuous development of hardware technology, high-definition and large screens have gradually become a vital part of the immersive experience of gamers. Reflected in game design, it means higher and higher resolution requirements, which in a sense also represents higher and higher texture resolution requirements. It makes that in many current mainstream games, **most of the content that occupies the volume of the game installation file are these high-resolution texture materials.**

Game Graph Promotion

The long history of *Tomb Raider* is a good example of the game's graph enhancement.

Today's Game Capacity

- The capacity of *Titanfall* exceeds **50GB**.

- When *GTA 5* launched on the PC for the first time, the high-definition material made it easily exceed **70GB**.

- The capacity of *Assassin's Creed: Odyssey* reaches more than **100GB**.

- The capacity of *Final Fantasy 15* is between 60GB and 70GB, but the alternative high-definition material package can be close to **100GB**.

High Definition Texture

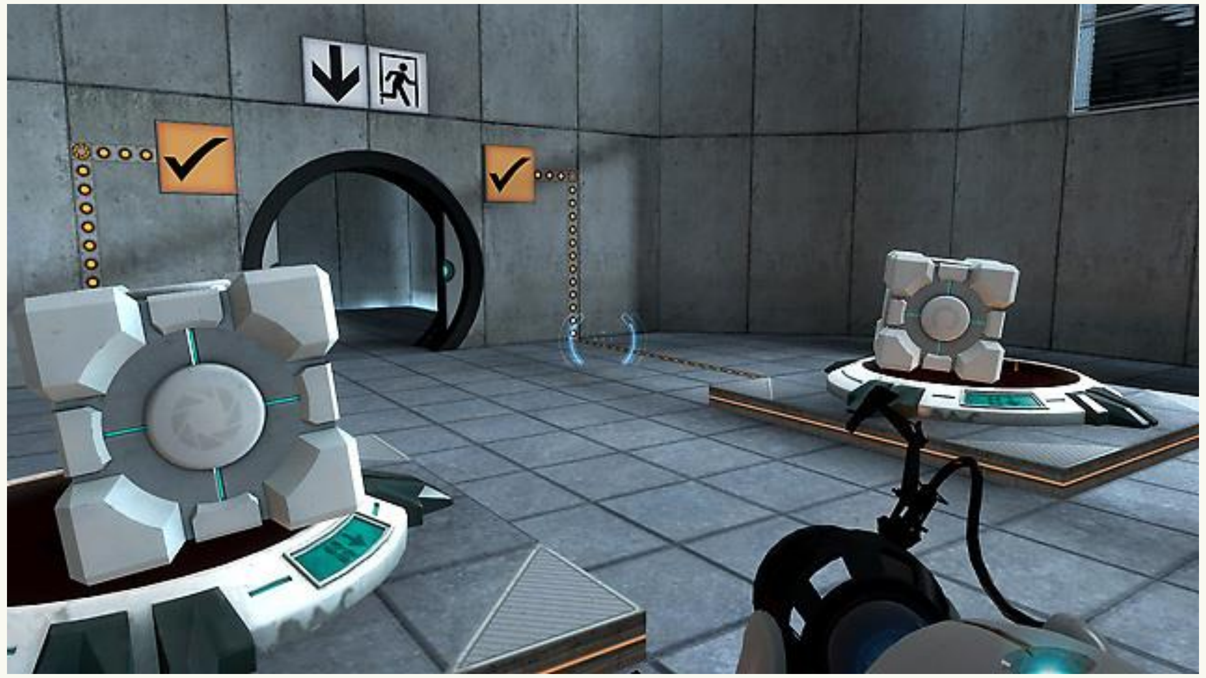
more realistic

larger capacity

An example of high-definition texture, we can see that the more realistic textures, the larger the capacity.

Does Human-eye-limit Help?

Because of the limitation of the visual resolution of human eye, the need for material resolution is not unlimited. But the game industry's pursuit of building a colorful world is limitless. For example, with the development of technology, games can create a wider and larger world.



Indoor scene, regular shape

Outdoor scenes, plants and terrain

Texture, too big to be loaded at once.

In order to ensure the smoothest gaming experience, sometimes textures have to be loaded step by step. When an object appears in the game, the program first uses a low-resolution texture to make it displayable, and load the high-definition texture later.

Load HD texture

What We Can Do?

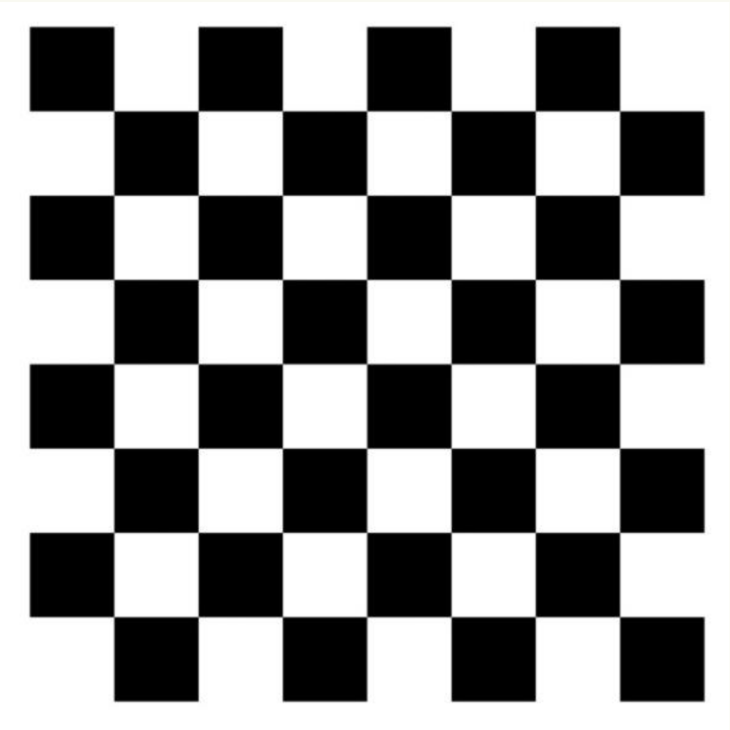
On the one hand, the High-Resolution trend of game world is inevitable, but on the other hand, some texture maps with a certain regularity have a lot of space for optimization.

with less patterns and rules

with more patterns and rules

Simple Example

If we want to create a chess board or similar texture, we don't need to save a whole image file, but just create the image when it was needed according to the rules. That means, pictures with certain patterns and rules can be compressed into a mathematical generation, this compression ratio can be theoretically almost infinite.



a image file

Compressed capacity

"A small piece of code"

Real World Example

The above example is too simple, more common are real world creatures with their texture have patterns and rules.

There are many biological textures in the real world with patterns and rules.

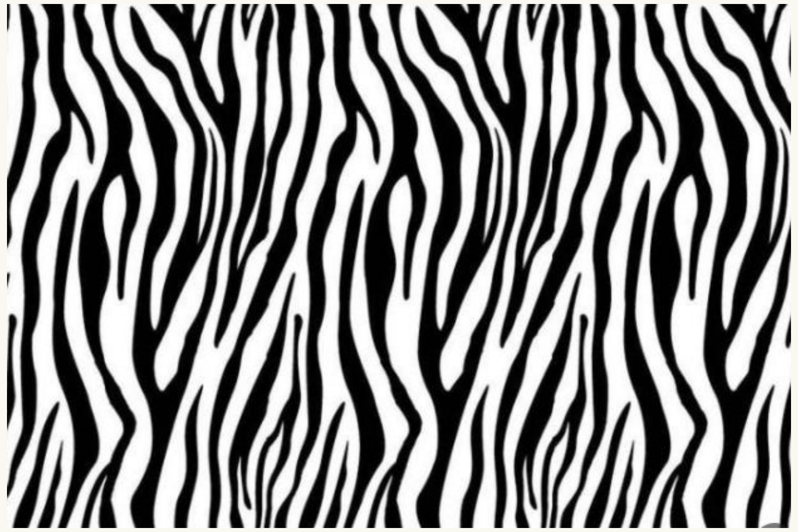
Details to consider

The zebra's skin is black and white stripes, but if one directly uses a black and white texture map to repeat, then this mechanical extension will cause the overall effect to be extremely distorted.

Pattern, Rule, but not Repeat.

Real zebra

repeat the black and white stripes



Zebra texture currently used

Direction

We know that there is a behavioral pattern, but using this pattern to achieve compression of textures cannot be repeated simply, but a mathematical tool needs to be sought.

Pattern

Rule

Correct math and algorithm

Experiment and parameter optimization

Natural Textures Program Generator

Solution

Fortunately, in the field of mathematics, **Turing patterns and fractals** are very good methods for generating natural textures.

Combined with some cutting-edge machine learning methods, a very efficient natural texture generation program can be constructed.

We will show how these technologies can quickly generate authentic and believable natural textures to greatly save the high-definition texture space in the game.

Turing pattern & Gray-Scott model

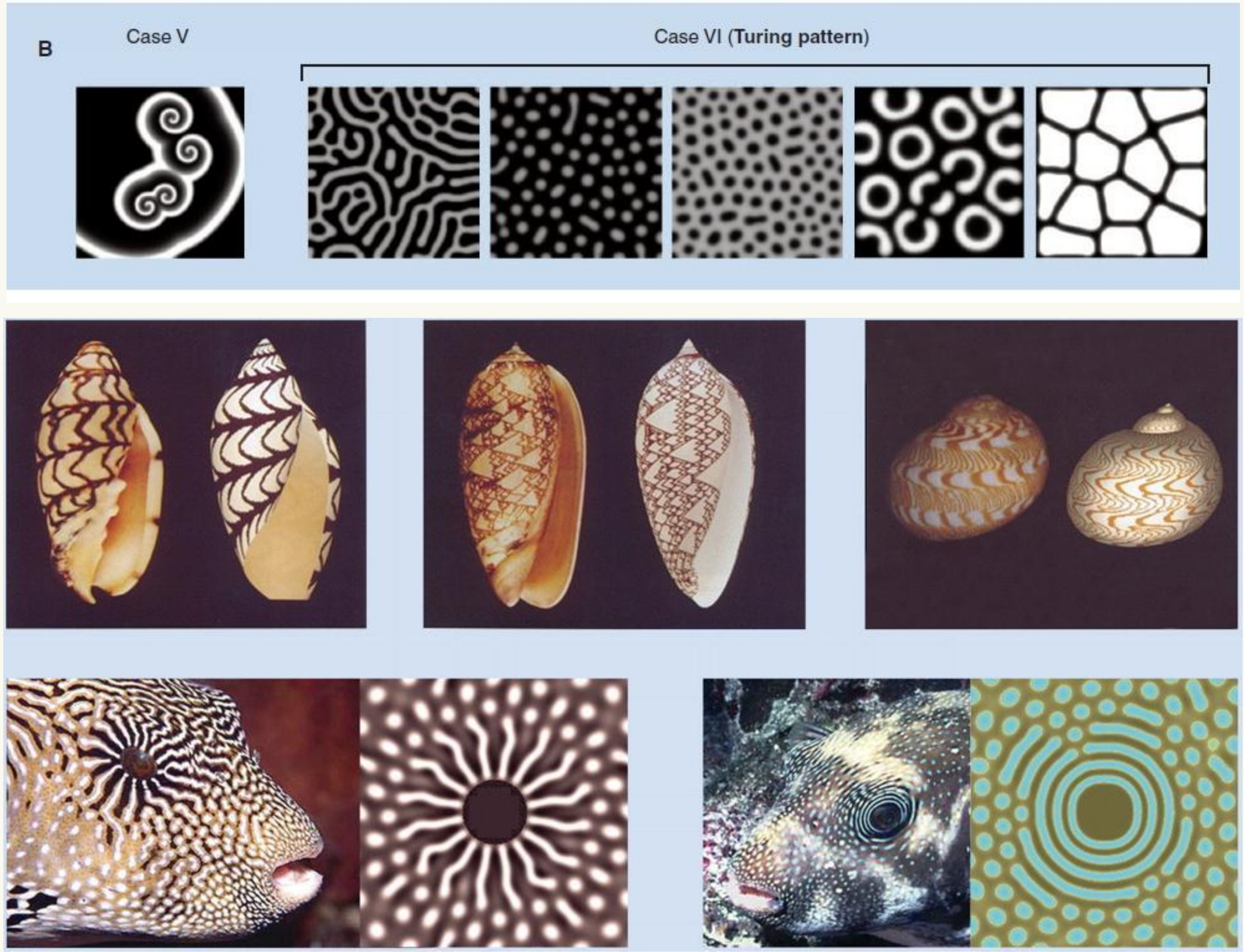
Turing pattern

Turing creatively used the mathematical model of the reaction-diffusion system to **describe patterns in nature**, such as tiger stripes and leopard spots. This kind of pattern that can be described by the reaction-diffusion equation is called **Turing pattern**.

Turing pattern

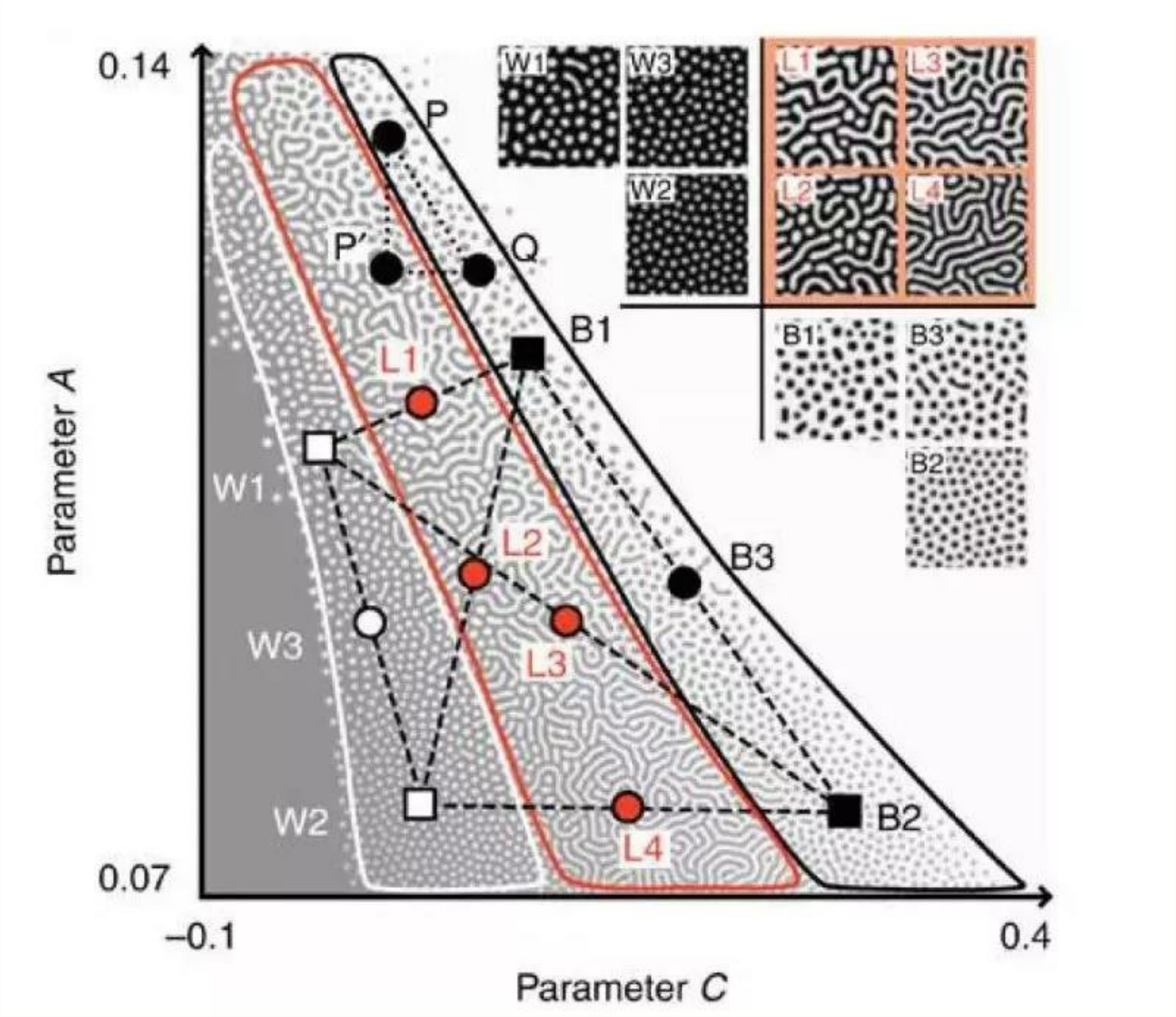
Turing believes that when the interaction of chemical substances is well mixed (without spatial heterogeneity), it will show a stable equilibrium state. In other words, the reaction kinetics makes any disturbance from the equilibrium disappear over time, thereby returning the system to its original equilibrium state.

But when spatial heterogeneity appears, the system is not well mixed, and proliferation may occur at this time. Diffusion drives the equilibrium state to become unstable, creating new spatial patterns. As shown next page:



DOI: 10.1126/science.1179047

Reaction-Diffusion Model as a Framework for Understanding Biological Pattern Formation



DOI：10.1090/noti865

Why Are There No 3-Headed Monsters? Mathematical Modeling in Biology

Concrete model

There are many ways to realize the automatic generation of the above models. Here, **Walgraef-Aifantis equations** and **Gray-Scott model** are taken as two examples to describe the simulation and drawing of the pattern of the reaction diffusion process.

Walgraef-Aifantis equations: describes a reaction-diffusion process about grain growth and dislocation interaction;

Gray-Scott model: models a chemical process that consists of a reaction and diffusion.

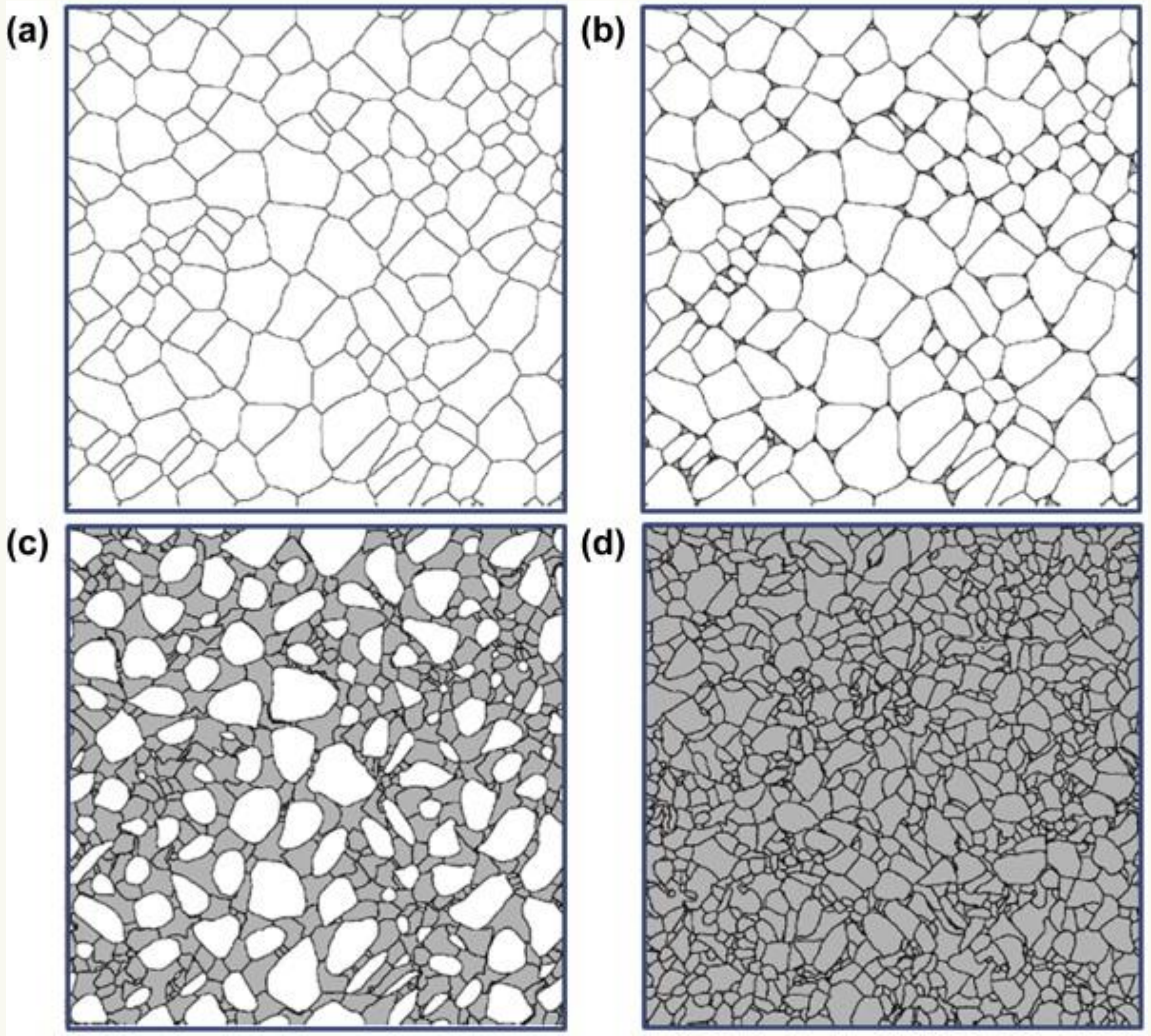
Reaction diffusion equation

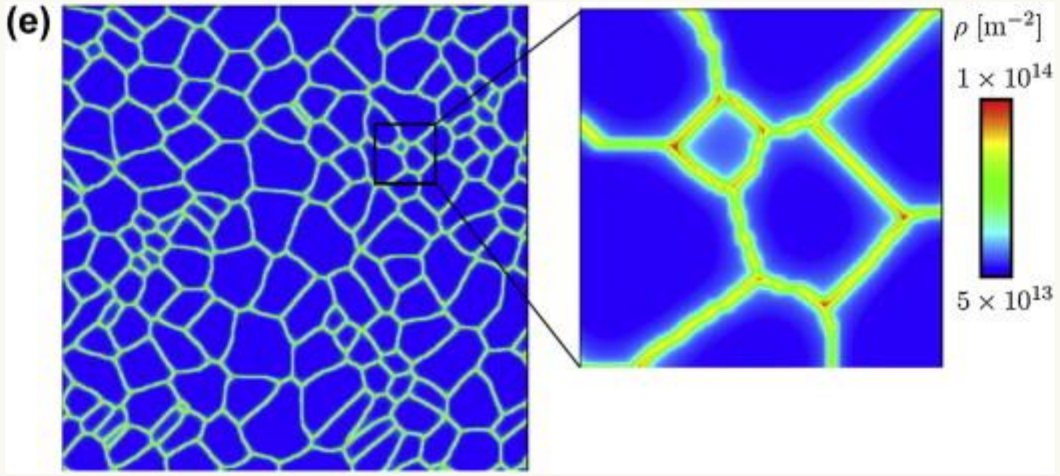
The following equations are the general form of the reaction diffusion equations:

$$\frac{dU(x,y)}{dt} = D_U \nabla^2 U + f(U,V)$$
$$\frac{dV(x,y)}{dt} = D_V \nabla^2 V + g(U,V)$$

Among them, D_U and D_V are the diffusion coefficients of U and V, respectively, $f(U,V)$ and $g(U,V)$ are the generation rates of U and V, respectively. Turing believes that the two terms are quadratic polynomials.

Walgraef-Aifantis equations





Results of modelling dynamic recrystallisation using reaction–diffusion equations

DOI: 10.1016/j.commatsci.2012.09.016

Microstructure evolution influenced by dislocation density gradients modeled in a reaction–diffusion system

Walgraef-Aifantis equations

The following equations are the basic form of the Walgraef-Aifantis equations:

$$\frac{\partial \rho_m}{\partial t} = D_m \frac{\partial^2 \rho_m}{\partial x^2} - \left[k_2 \rho_m + k_3 \sqrt{\rho_i} - k_1 \frac{\rho_i}{\rho_m} \right] \dot{\varepsilon}^{plastic}_{effective}$$

$$\frac{\partial \rho_i}{\partial t} = D_i \frac{\partial^2 \rho_i}{\partial x^2} + \left[k_2 \rho_m + k_3 \sqrt{\rho_i} - k_4 \rho_i \right] \dot{\varepsilon}^{plastic}_{effective}$$

D_i and D_m are the diffusion coefficients for relatively immobile and mobile dislocations, ρ_i and ρ_m are the respective dislocation densities, and $k_i (i = 1, 2, 3, 4)$ are constants and $\dot{\varepsilon}^{plastic}_{effective}$ is the effective plastic strain rate.

Gray-Scott model

Gray-Scott model describes the chemical reaction:

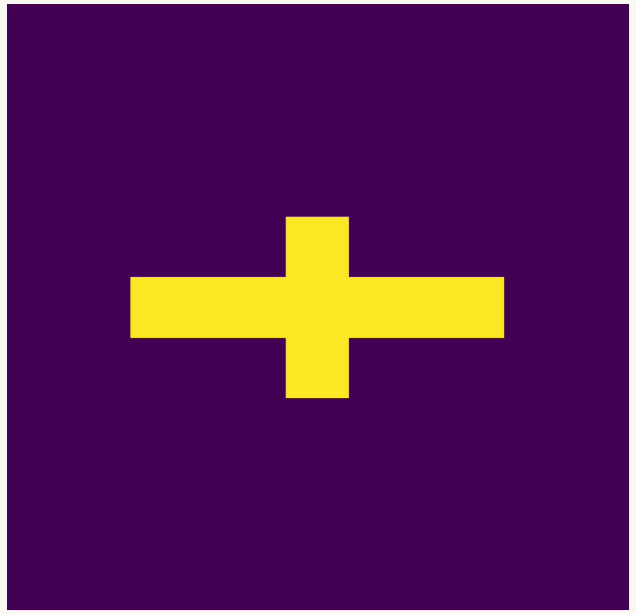
$$U + 2V \rightarrow 3V$$

The reaction consumes U and produces V. Because V appears on both sides of the reaction, it acts as a catalyst for its own production. To maintain the reaction, the amount of U and V needs to be controlled, it is done by adding U at a “feed rate”(F) and removing V at a “kill rate”(k). U and V diffuse at the rates D_u and D_v.

The overall behavior of the system is described by the following equations.

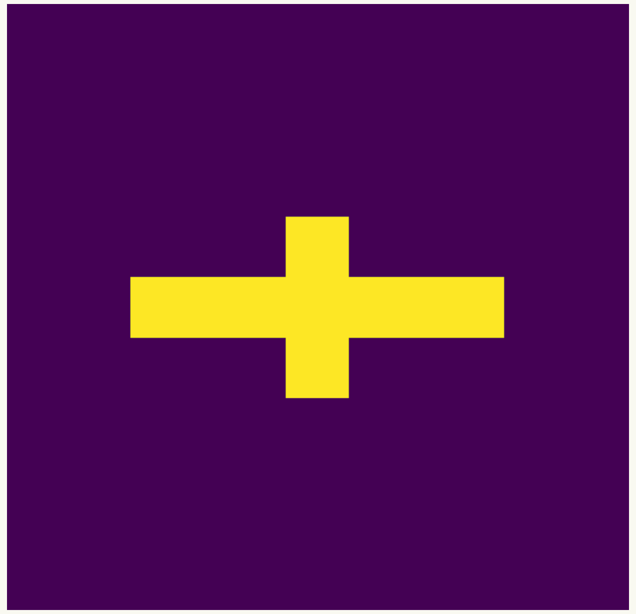
$$\frac{du}{dt} = D_u \nabla^2 u - uv^2 + F(1-u)$$
$$\frac{dv}{dt} = D_v \nabla^2 v + uv^2 - (F+k)v$$

Gray-Scott model



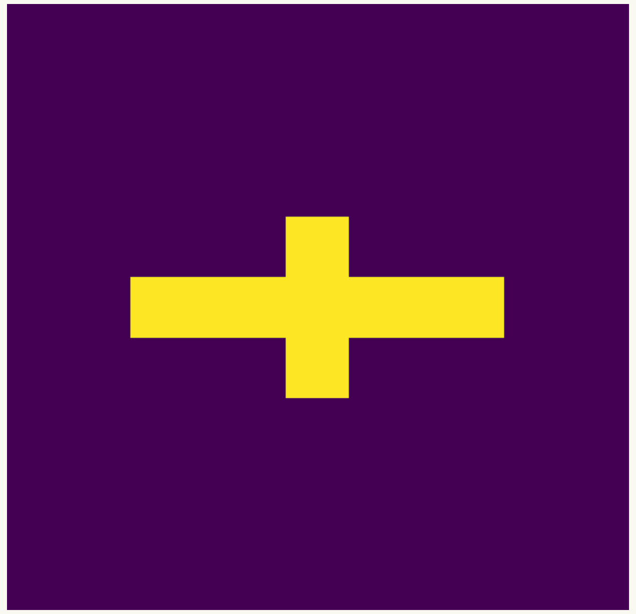
feed rate:0.055

kill rate:0.062



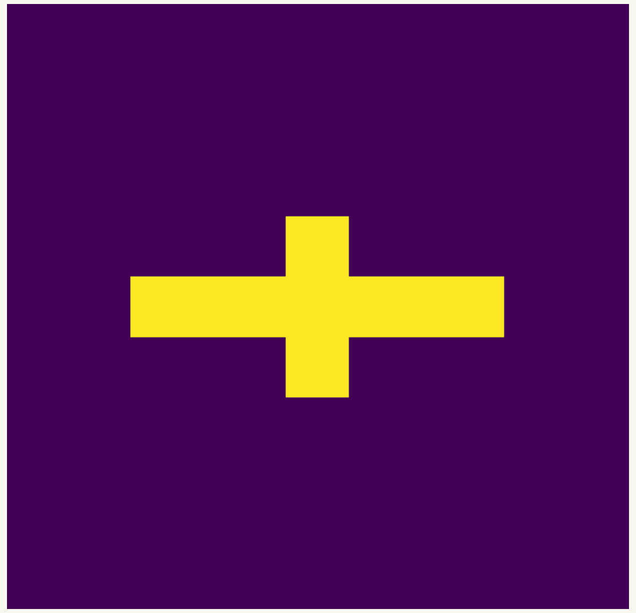
feed rate:0.024

kill rate:0.055



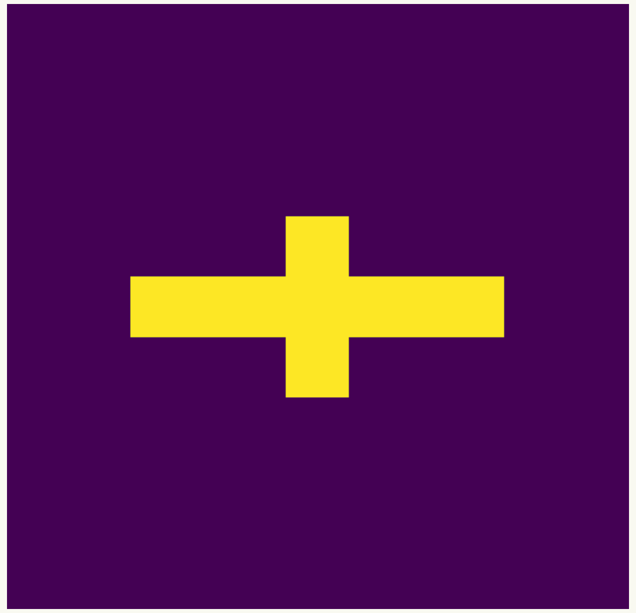
feed rate:0.025

kill rate:0.06



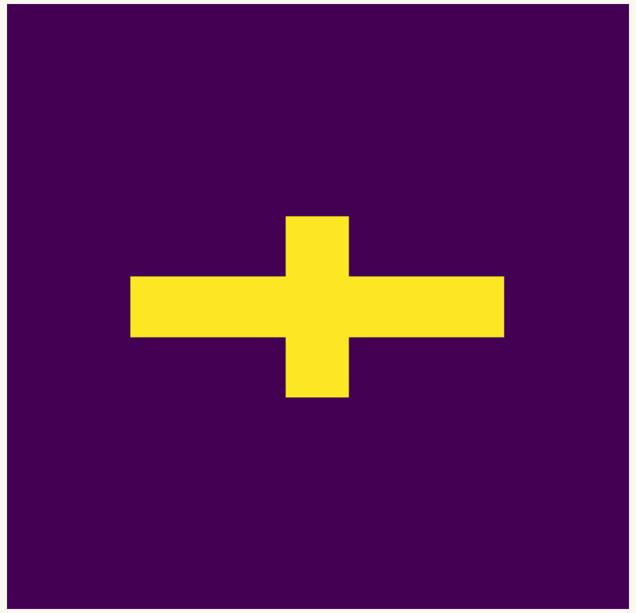
feed rate:0.04

kill rate:0.06



feed rate:0.012

kill rate:0.05

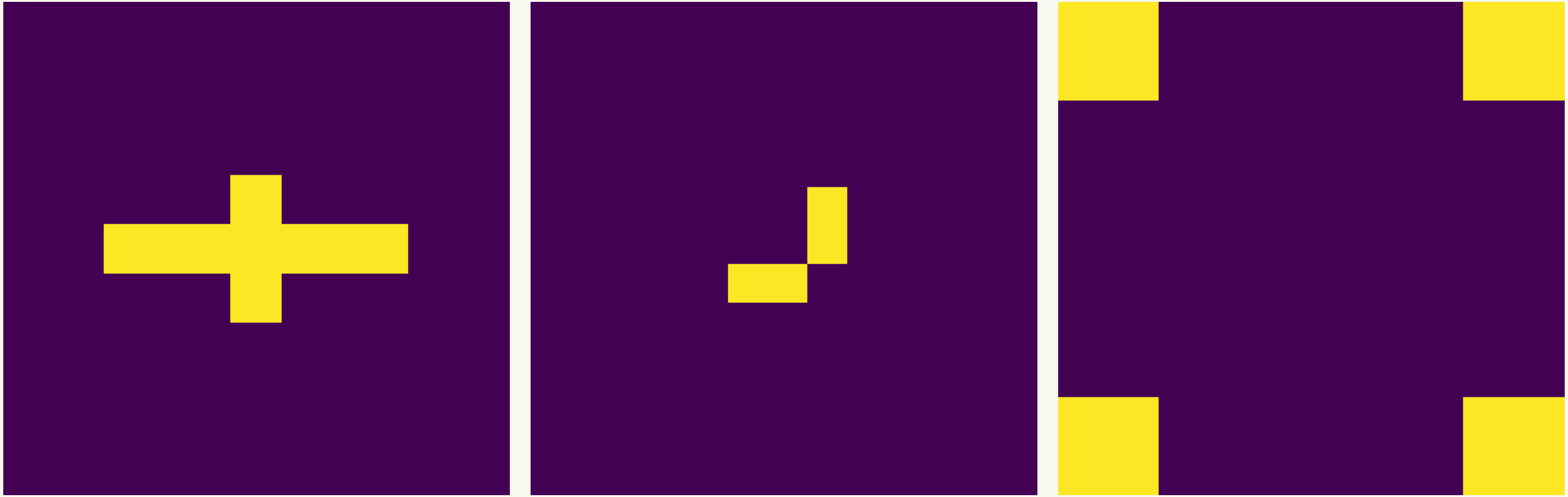


feed rate:0.03

kill rate:0.06

Here are several program examples of Turing pattern, which are generated over time.

Gray-Scott model practice



Turing patterns with different initial positions generated over time.

Image Quilting

Image Quilting

Image Quilting is a method of generating a new image by stitching together small patches of the existing image.

Combining image quilting and turing pattern helps decrease time for generating natural textures.

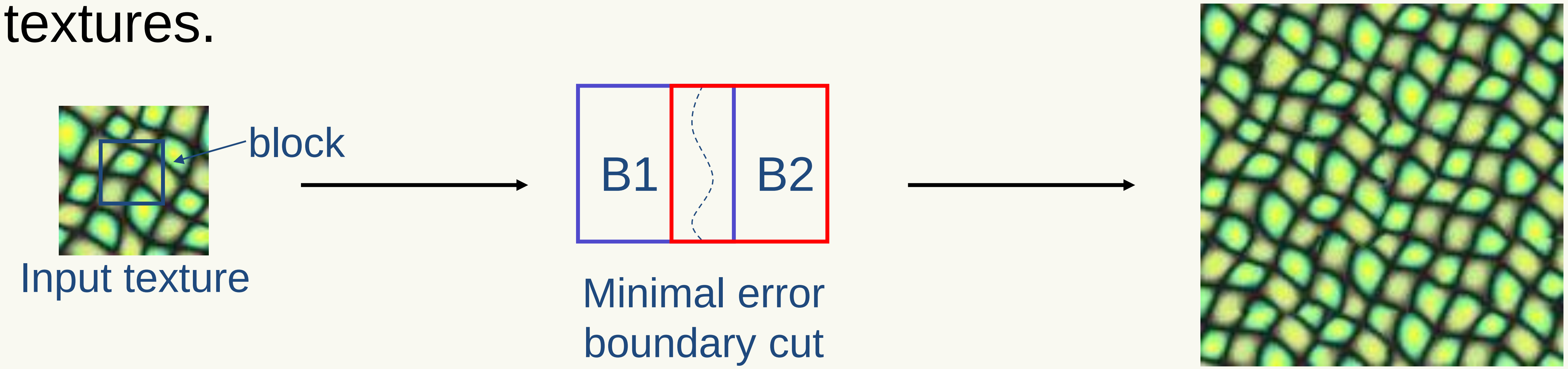


Image Quilting Process

DOI: 10.1145/383259.383296

Image quilting for texture synthesis and transfer

Image Quilting

Main idea: Find minimal error consecutive boundary cut

$$cut^* : min\{X_{m,j}\}$$

For each element at overlap error mat:

$$X_{i,j} = x_{i,j} + \underbrace{\boxed{min(X_{i,max(j-1,1)}, X_{i,j}, X_{i,min(j+1,n)})}}_{\text{consecutive boundary cut}}$$
$$j = 1, 2, \ldots, n$$

$X_{i,j} \sim$ accumulative error

$x_{i,j} \sim$ point error

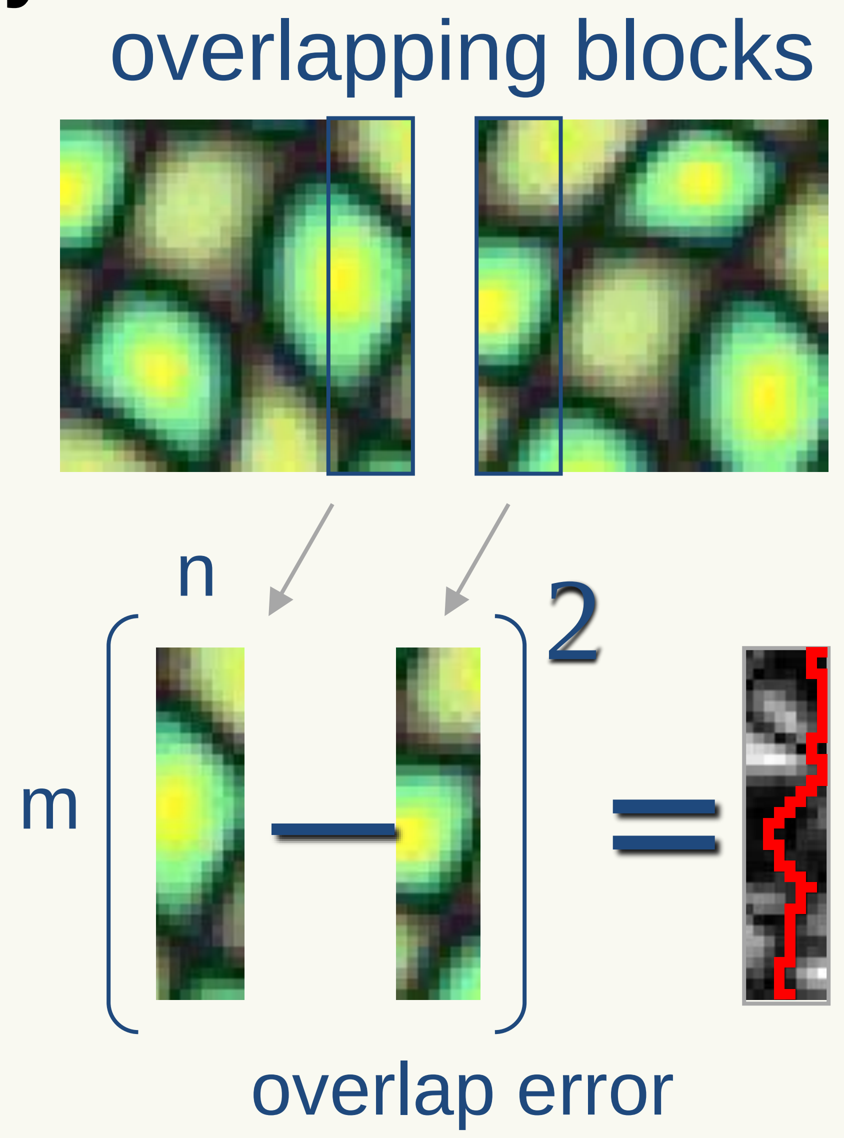
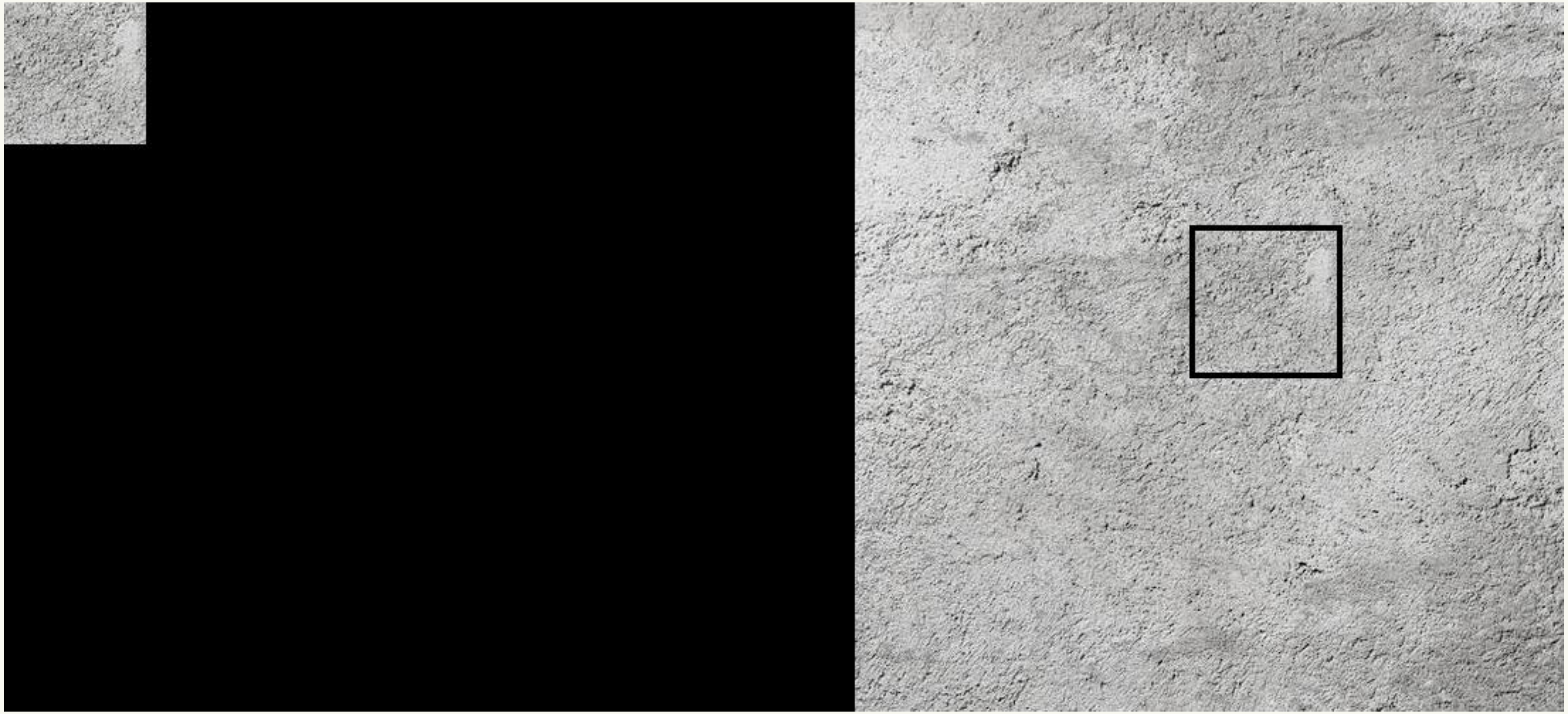


Image Quilting



New texture created by Image Quilting

Image Quilting

Natural texture created by Image Quilting

Image Quilting

neighboring overlap

Image Quilting

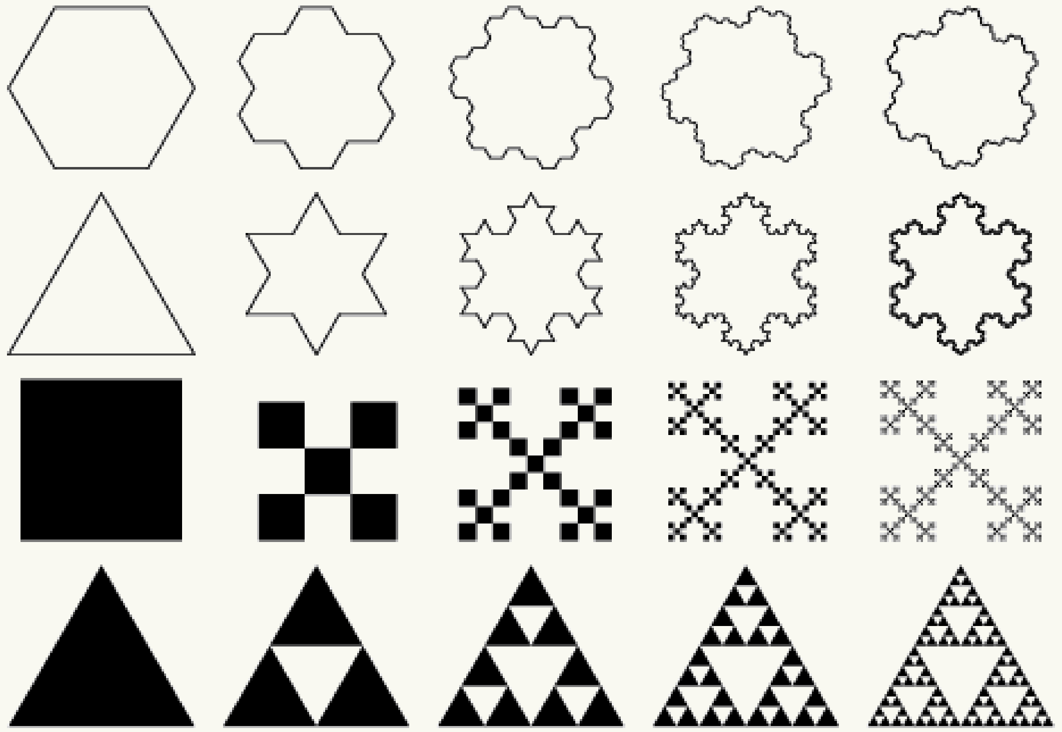
Fractal: L-system & IFS

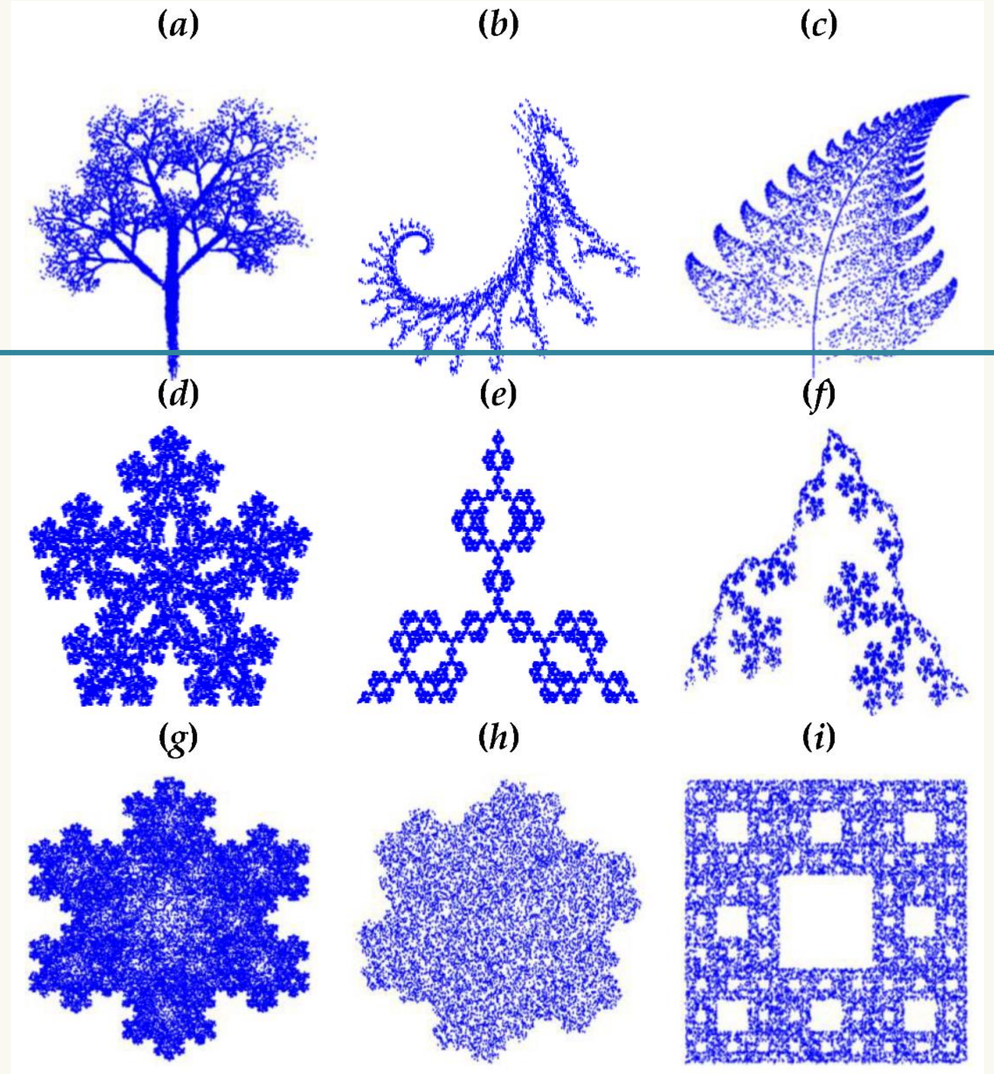
Fractal

In mathematics, a fractal is a subset of Euclidean space with a fractal dimension that strictly exceeds its topological dimension. Fractals appear the same at different scales, as illustrated in successive magnifications of the Mandelbrot set. **Fractals often exhibit similar patterns at increasingly smaller scales, a property called self-similarity**, also known as expanding symmetry or unfolding symmetry.

The above characteristics allow certain fractals to be used to **generate natural structure with self-similarity.**

Fractal – some example





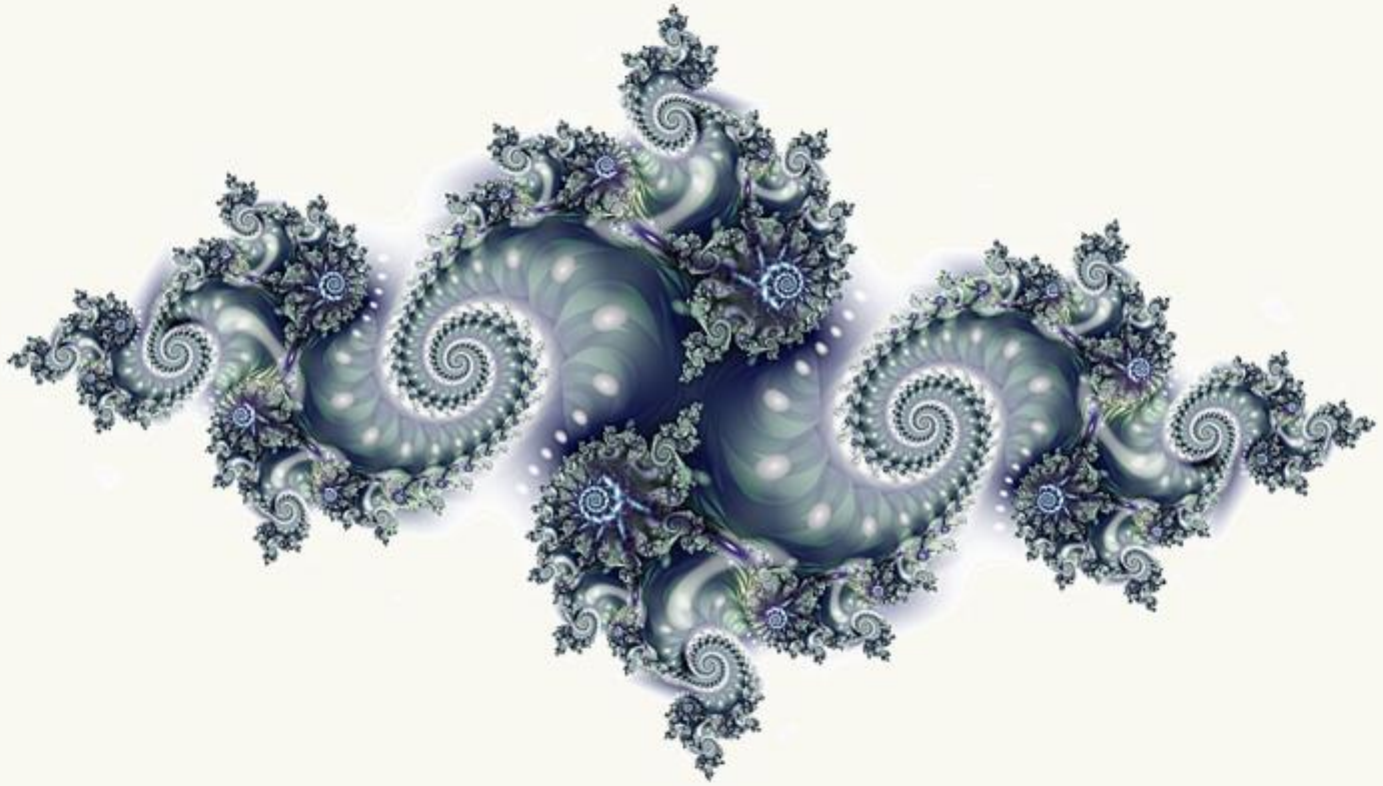
can be used to generate plant shapes

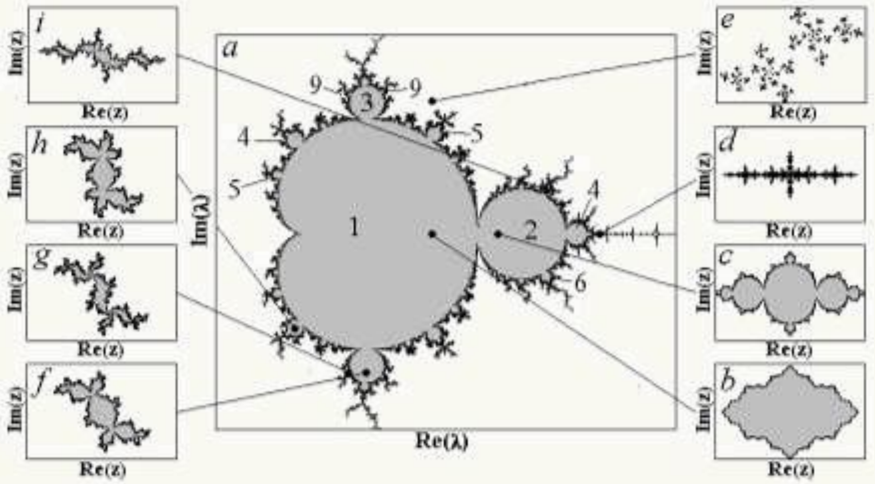
Fractal – some example – Julia set

Julia set consists of values such that an arbitrarily small perturbation can cause drastic changes in the sequence of iterated function values.

$$f_c(z) = z^2 + c$$

$$z_0, f_c(z_0), f_c(f_c(z_0)), f_c(f_c(f_c(z_0))), \ldots .$$





L-system

An **L-system** or *Linden Mayer* system is a parallel rewriting system and a type of formal grammar. An L-system consists of an alphabet of symbols that can be used to make strings, a collection of production rules that expand each symbol into some larger string of symbols, an initial "axiom" string from which to begin construction, and **a mechanism for translating the generated strings into geometric structures.**

L-system example

A simple example is to start from the string A, each iteration, replace A with AB, replace B with A, and so on:

n = 0 : A

n = 1 : AB

n = 2 : ABA

n = 3 : ABAAB

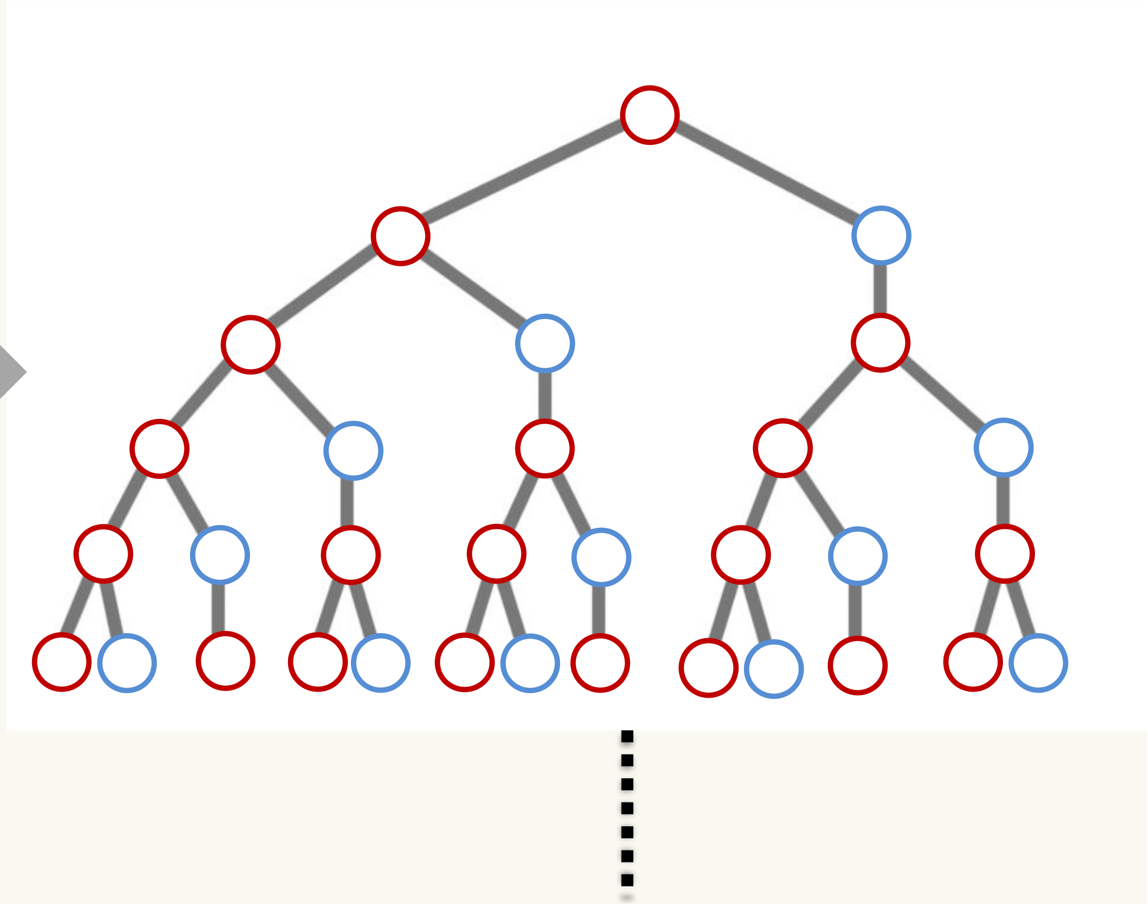
n = 4 : ABAABABA

n = 5 : ABAABABAABAAB

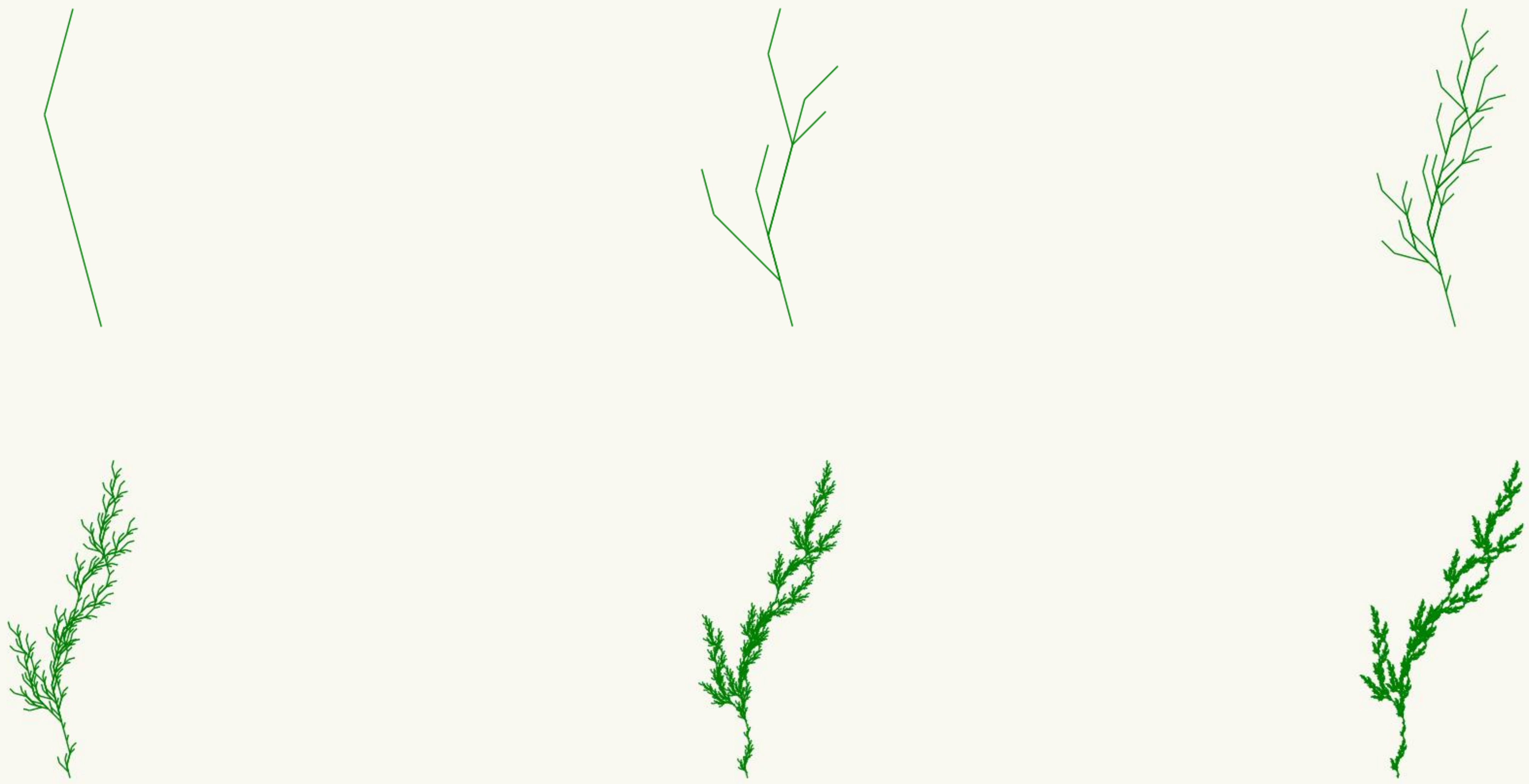
n = 6 : ABAABABAABAABABAABABA

n = 7 : ABAABABAABAABABAABABAABAABABAABAAB

Convert into fractal graph according to the rules



L-system plant



Example: L-system plant with multiple iterations

IFS Fractal

Iterated function systems (IFSs) is a method of constructing fractals, the resulting fractals are often self-similar. The fractal is made up of the union of several copies of itself, each copy being transformed by a function.

IFSs are a formalism for generating exactly self-similar fractals based on work of Hutchinson and Mandelbrot, and popularized by Barnsley.

$$X_n = aX_{n-1} + bY_{n-1} + c$$
$$Y_n = dX_{n-1} + eY_{n-1} + f$$

IFS Fractal

IFS is a finite set of contraction mappings on a complete metric space.

IFS in 2D: (X, Y) is the coordinate of point n

$$X_n = aX_{n-1} + bY_{n-1} + c$$
$$Y_n = dX_{n-1} + eY_{n-1} + f$$

Function sets: Randomly selected according to probability

$$F_1 : (a_1, b_1, c_1, d_1, e_1, f_1,\ prob_1)$$
$$F_2 : (a_2, b_2, c_2, d_2, e_2, f_2,\ prob_2)$$
$$F_3 : (a_3, b_3, c_3, d_3, e_3, f_3,\ prob_3)$$
$$F_i : (a_i, b_i, c_i, d_i, e_i, f_i,\ prob_i)$$
$$\ldots\ldots$$

IFS Fractal

tree

fern leaf

mountains

Reference

Kondo S, Miura T. Reaction-diffusion model as a framework for understanding biological pattern formation[J]. science, 2010, 329(5999): 1616-1620.

Murray J D. Why are there no 3-headed monsters? mathematical modeling in biology[J]. Notices of the American Mathematical Society, 2012, 59(6): 785-795.

Hallberg H, Ristinmaa M. Microstructure evolution influenced by dislocation density gradients modeled in a reaction–diffusion system[J]. Computational Materials Science, 2013, 67: 373-383.

Efros A A, Freeman W T. Image quilting for texture synthesis and transfer[C]//Proceedings of the 28th annual conference on Computer graphics and interactive techniques. 2001: 341-346.

Thank you