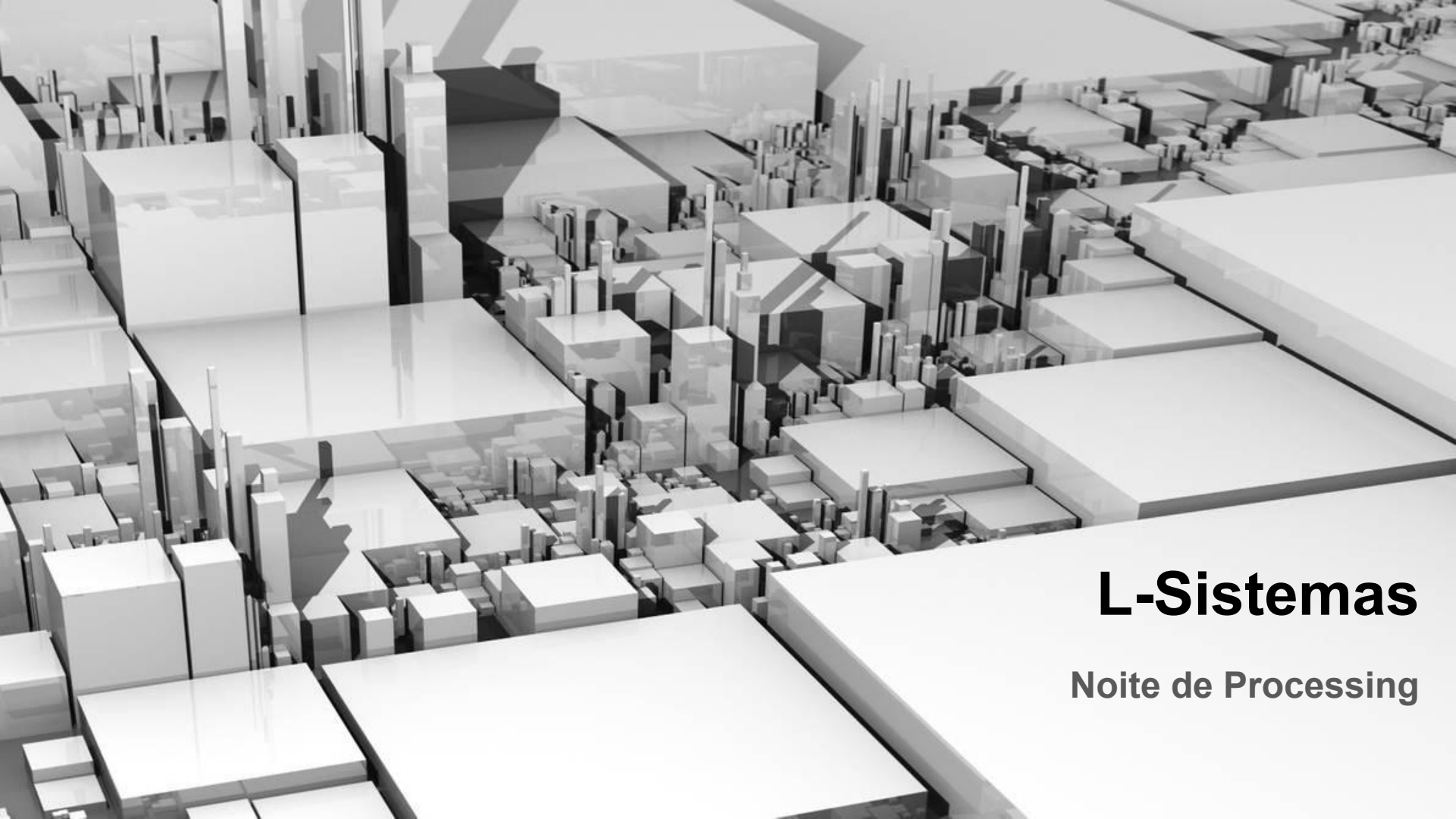
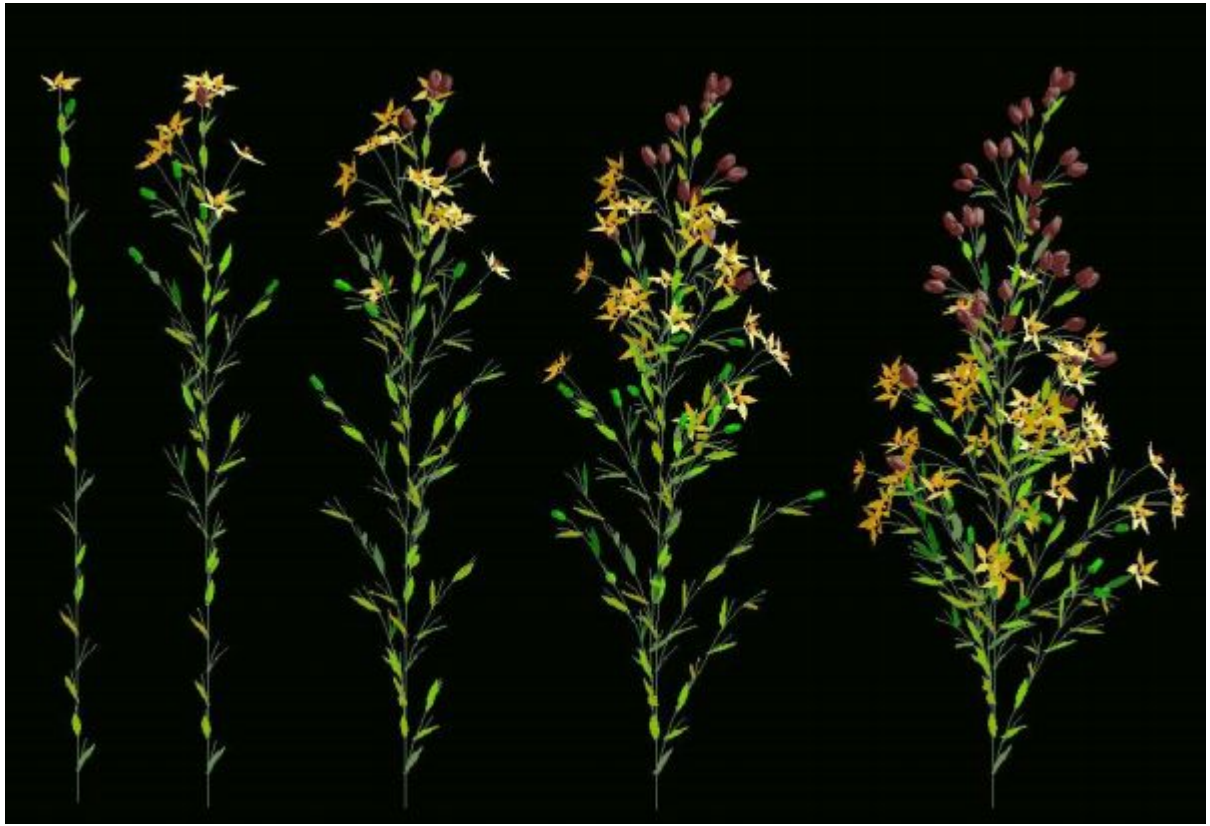


L-Sistemas

Noite de Processing

História

- 1968: Aristid Lindenmayer
- Biologia e botânica teórica
- Problema: descrever crescimento de fungos e bactérias
- Mais tarde: simulações mais complexas

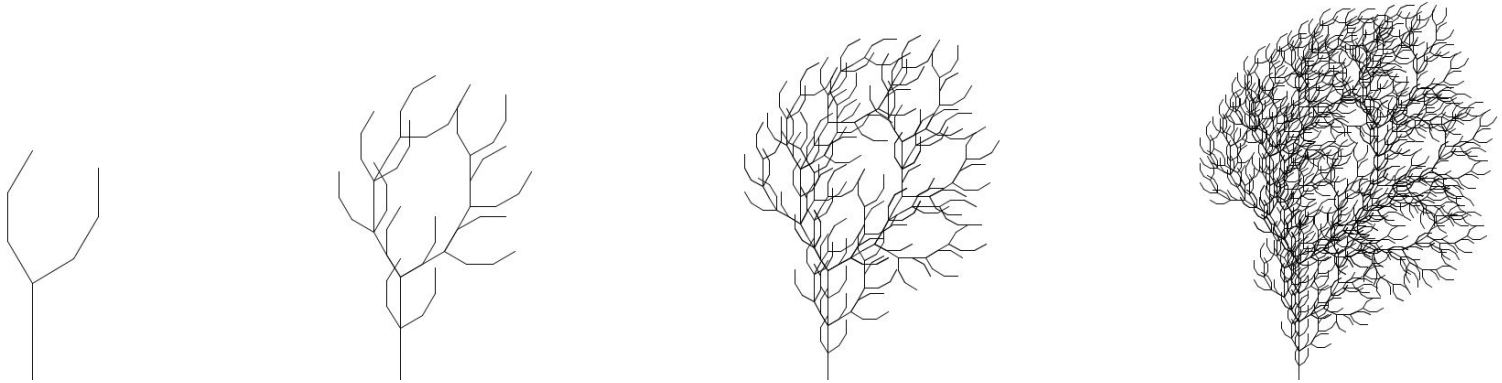


Estrutura

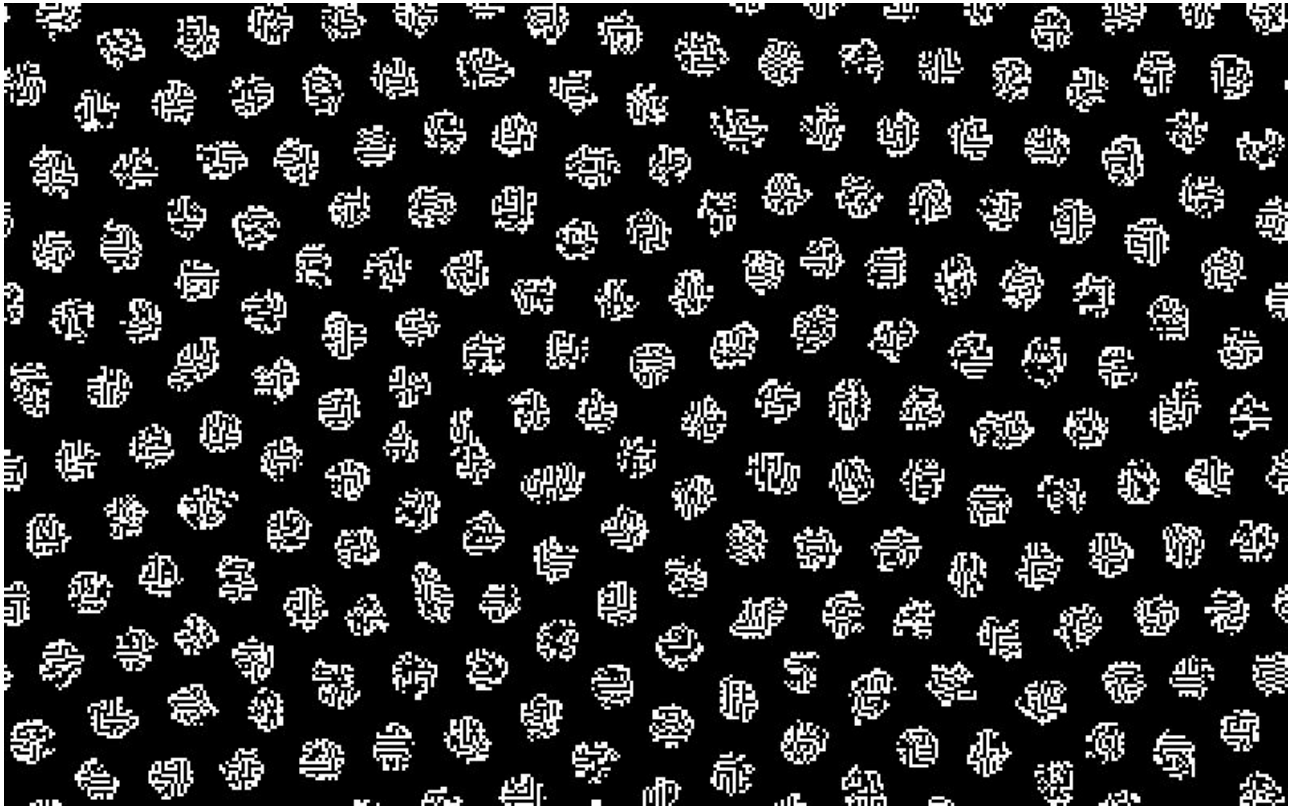
- Alfabeto
  - Elementos que podem ser substituídos por outros
  - Elementos terminais
- Axioma: estado inicial
- Regras: A → B
- Regras arbitrariamente complicadas
  - Uma única regra
  - Múltiplas regras
  - Regras estocásticas (um dos outputs é escolhido ao acaso).

Evolução

- Começamos com o axioma
- Aplicamos as regras iterativamente



Se for ver, parece um pouco autômato celular ;)

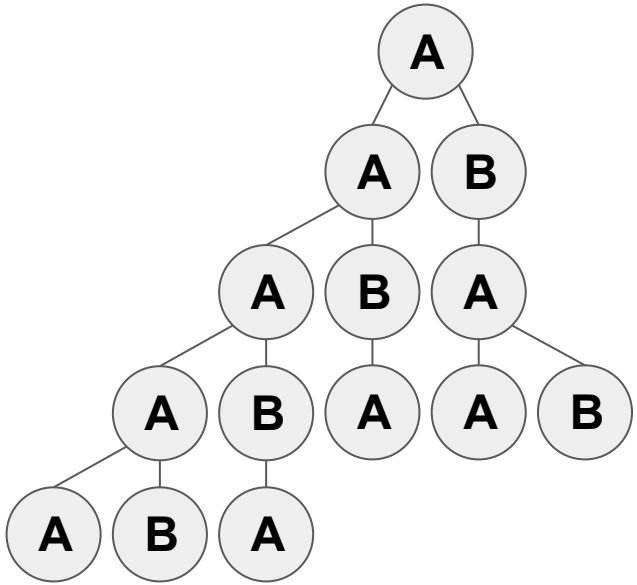


Mas e como isso funciona, afinal?

Algas

- Alfabeto: A, B
- Axioma: A
- Regras: A → AB, B → A

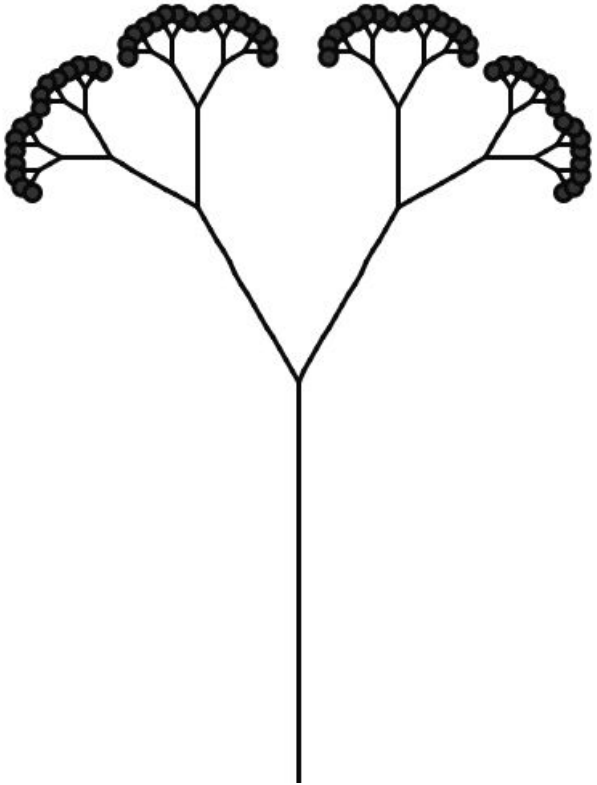
1. A
2. AB
3. ABA
4. ABAAB
5. ABAABABA



Árvore Fractal

Esse é um pouco mais complicado :)

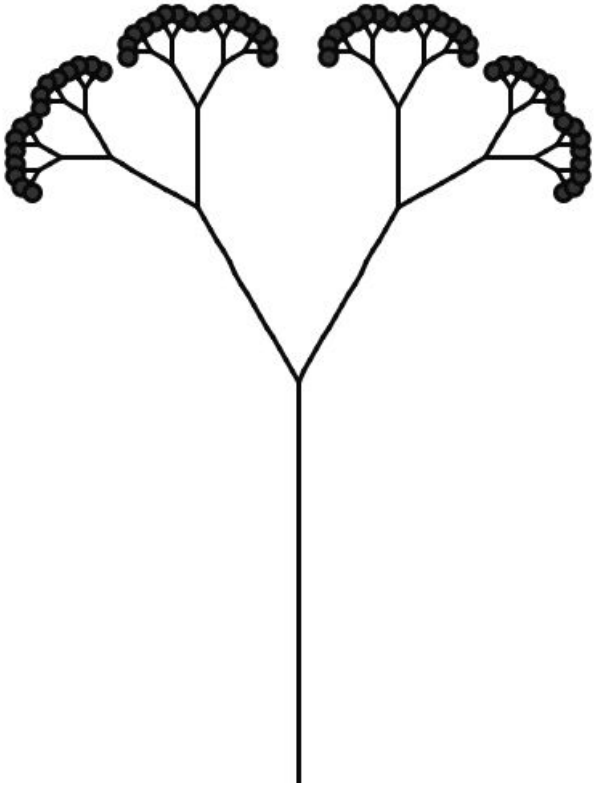
- Alfabeto: F, G
- Constantes: + - [ ]
- Axioma: F
- Regras:
  - F → G+[F]-[-F]
  - G → GG

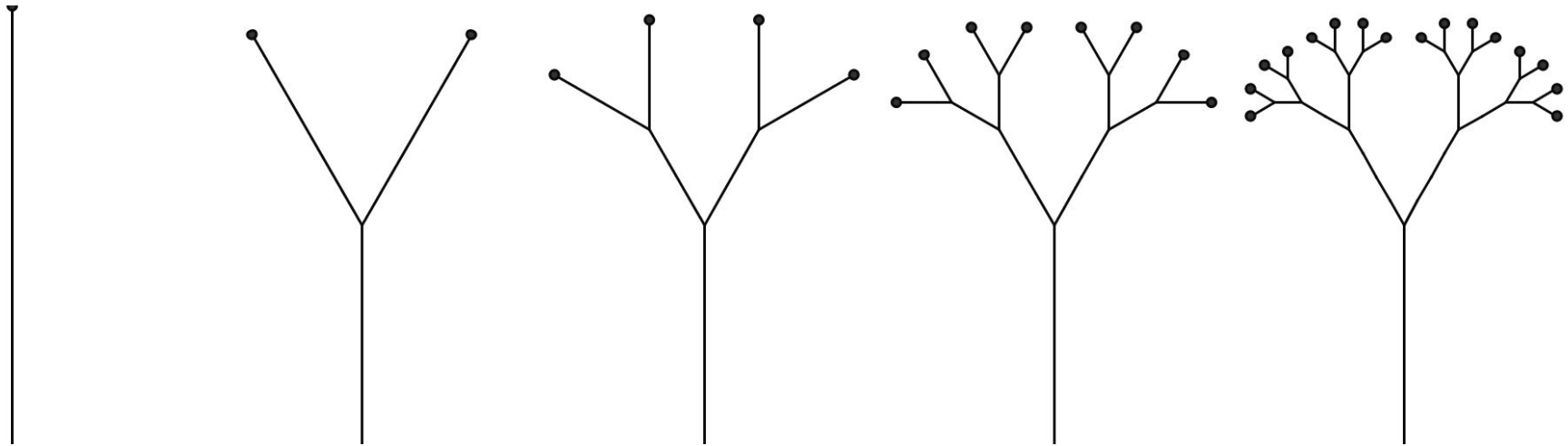


Árvore Fractal

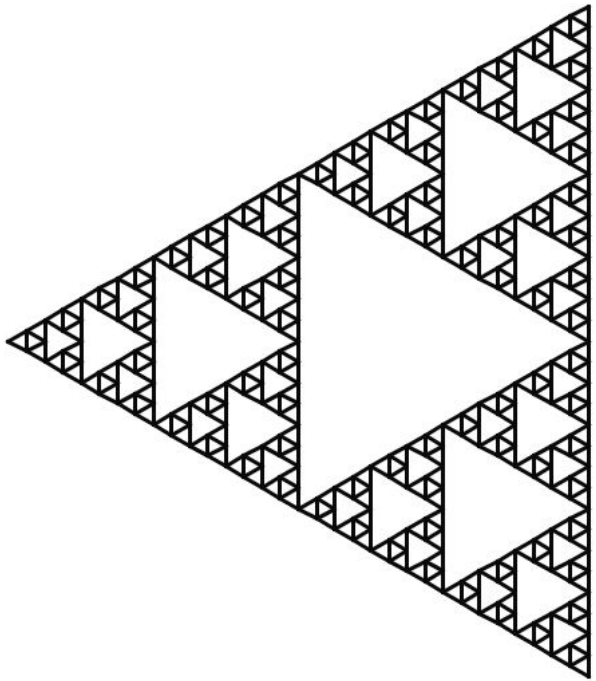
Mas o que são essas coisas todas?

- F: ramo que termina em folha
- G: ramo normal
- +: rotaciona 30 graus
- -: rotaciona -30 graus
- [: “push” - salva a posição atual
-]: “pop” - restora a posição atual



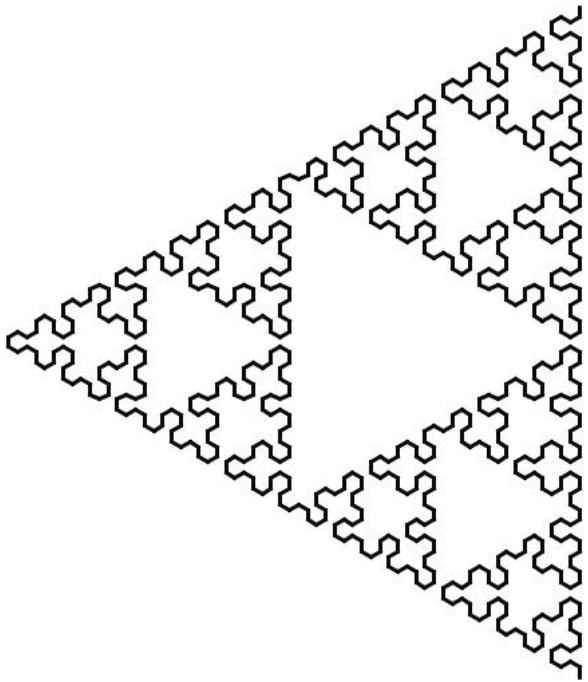


Triangulo de Sierpinski



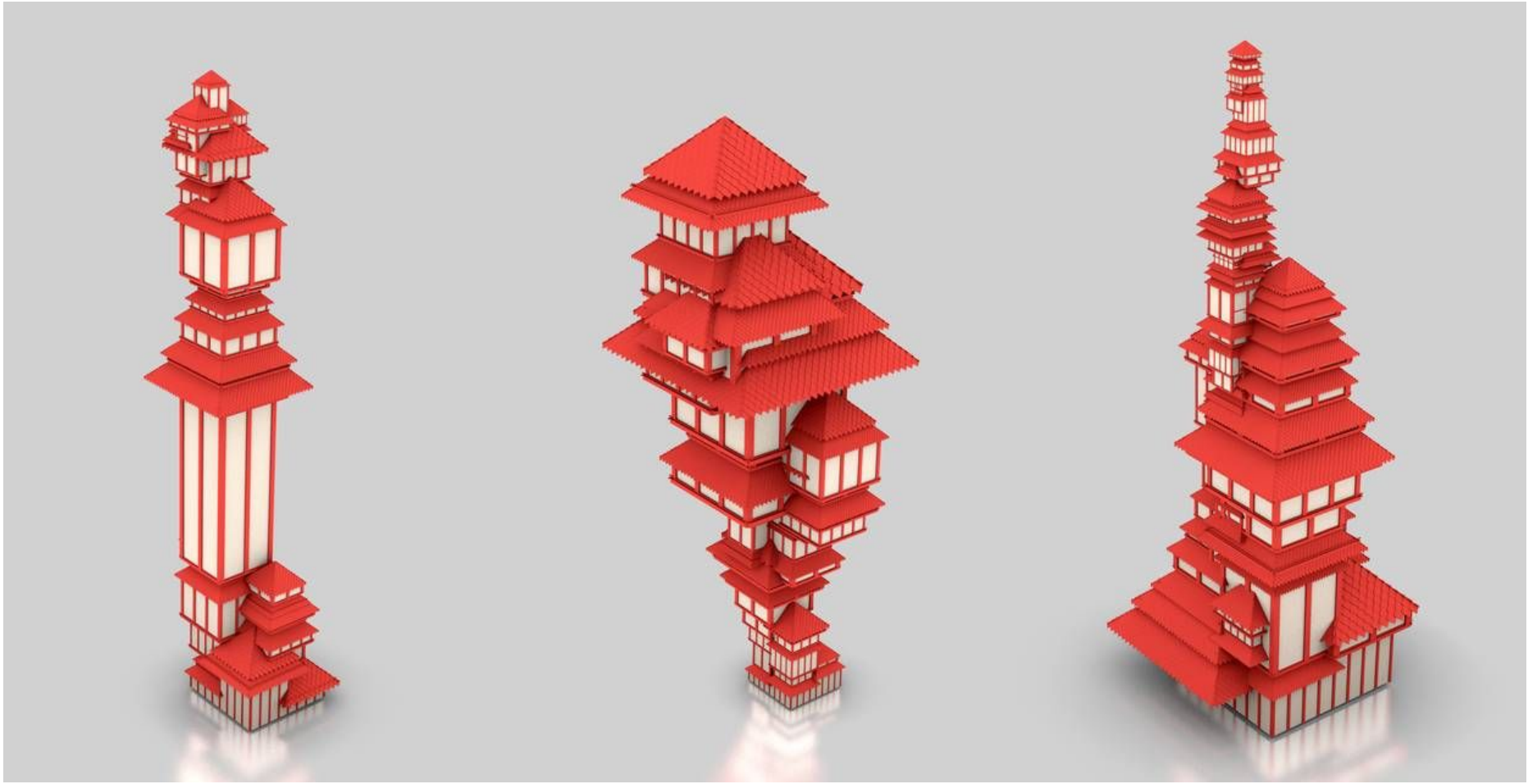
- Alfabeto: F, G
- Constantes: + -
- Axioma: FGG
- Regras:
  - F → F−G+F+G−F
  - G → GG
- A rotação neste caso é de 120 graus

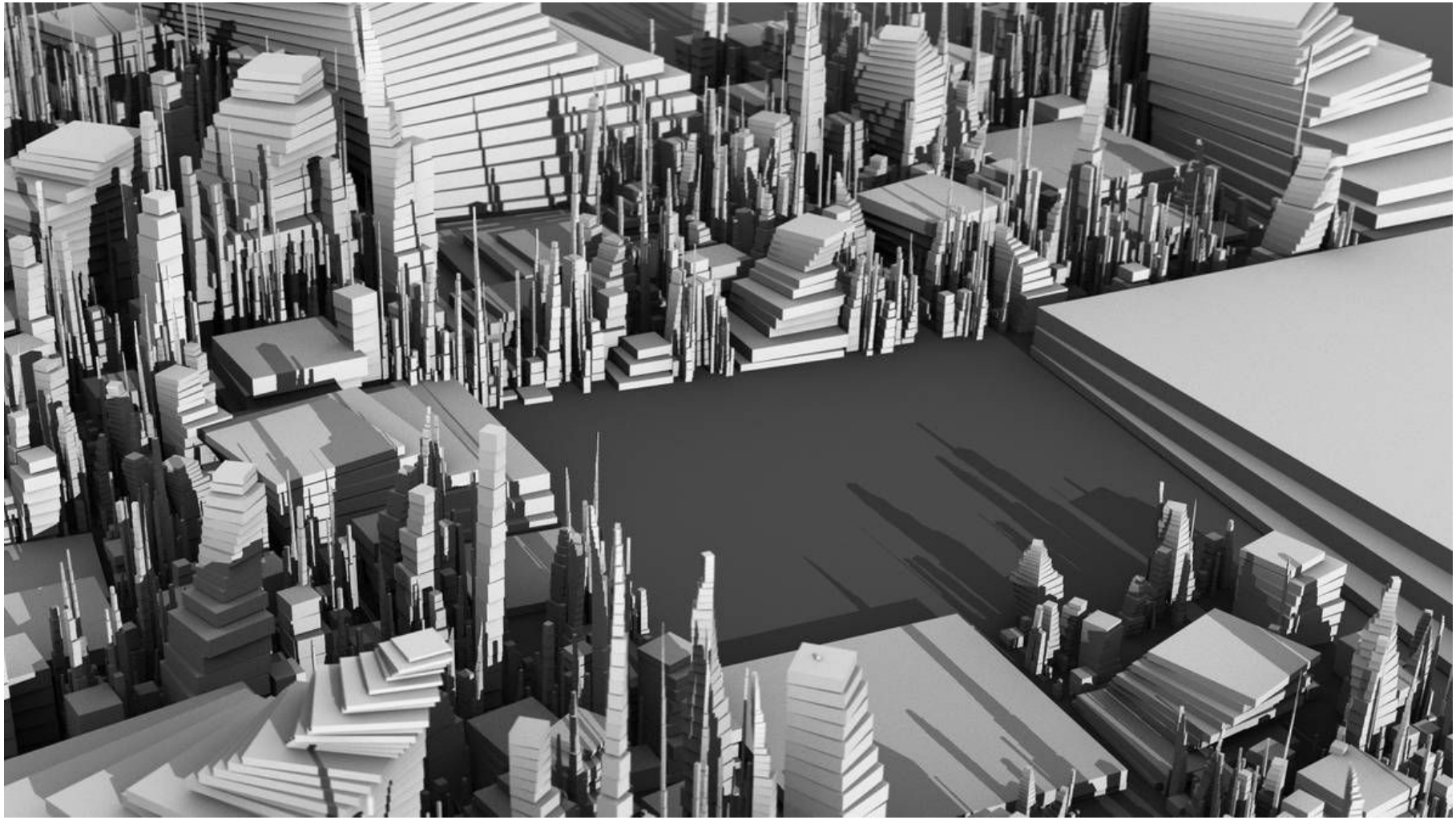
Triângulo de Sierpinski - de outro jeito ;)

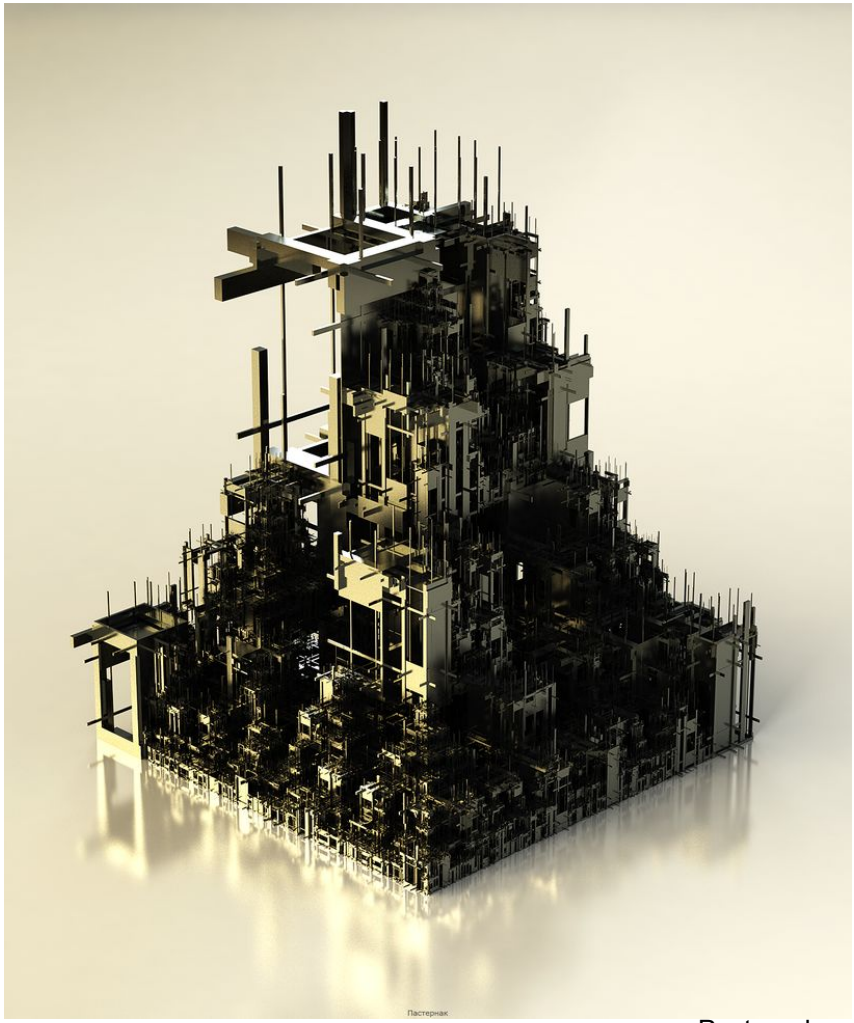


- Alfabeto: F, G
- Constantes: + -
- Axioma: F
- Regras:
  - F → G−F−G
  - G → F−G−F
- Rotação: 60 graus

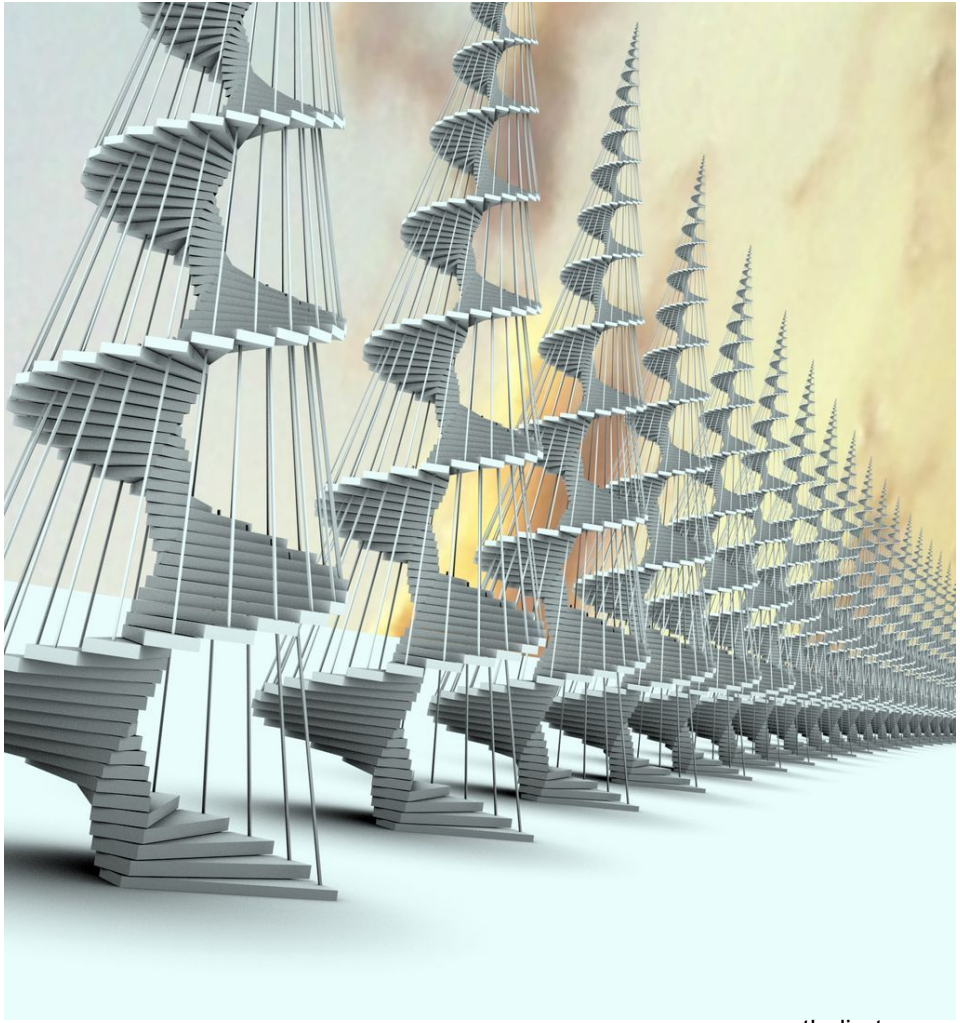
Exemplos de arte usando sistemas-l



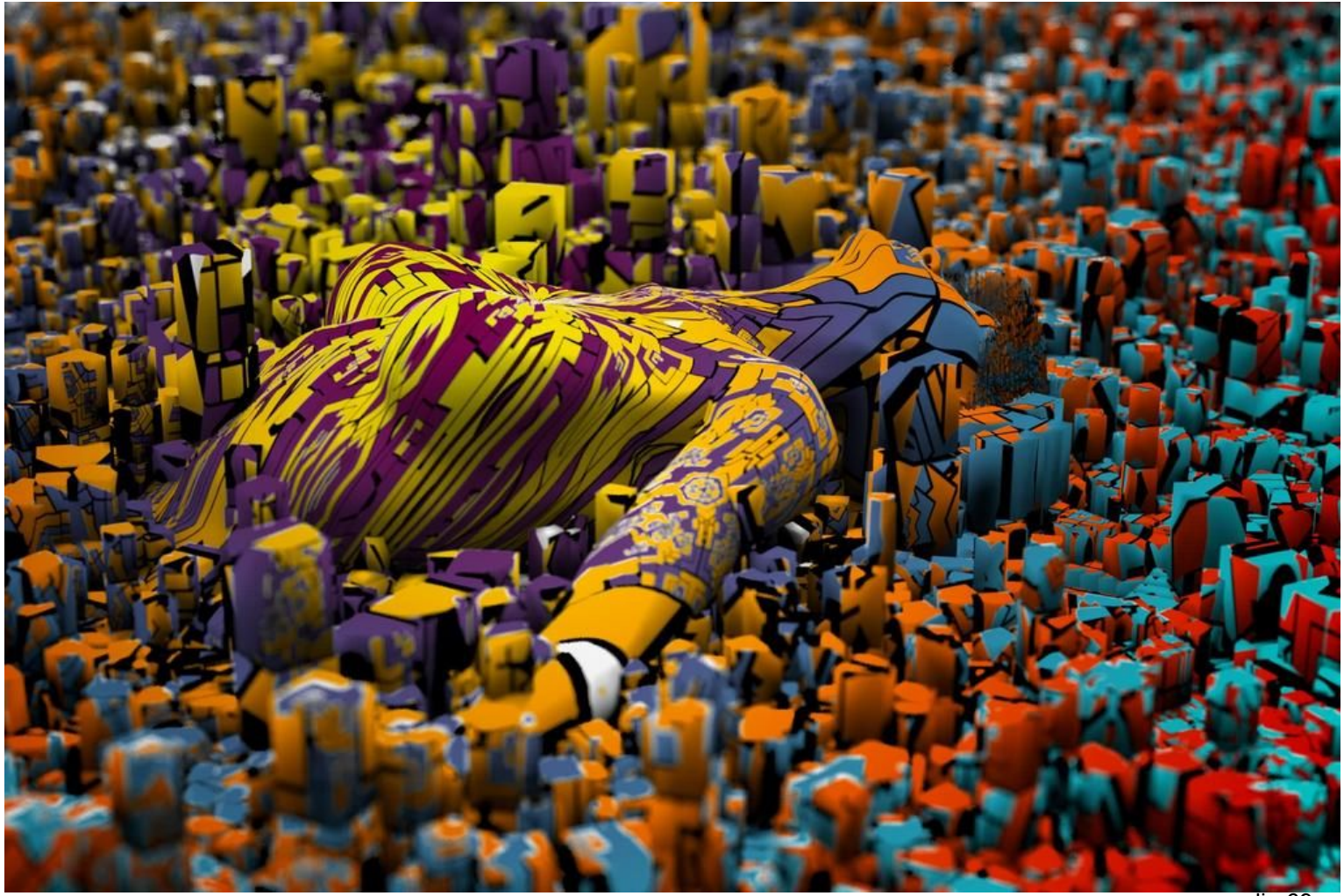




Pasternak

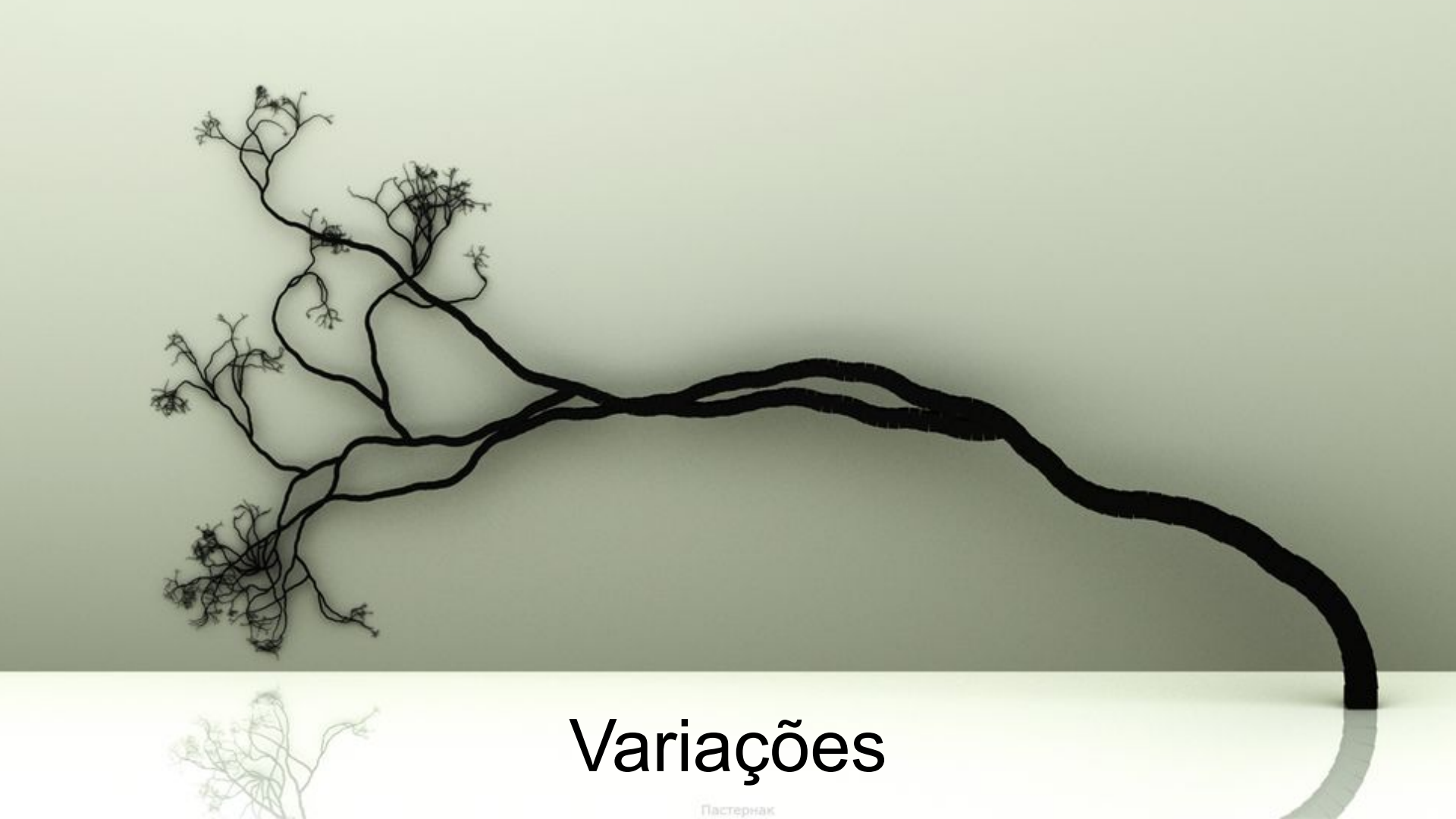


theli-at



scaulier69

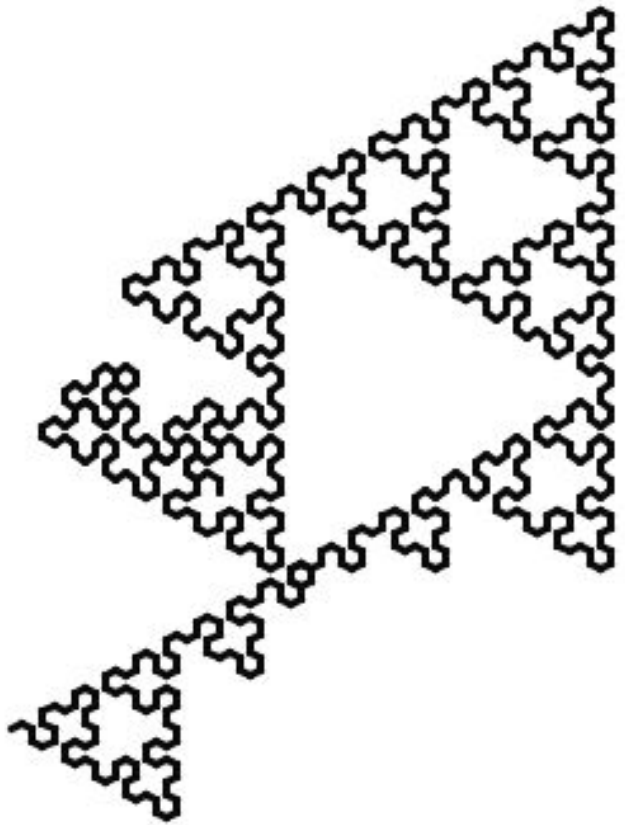
C-JR



Variações

Regras estocásticas

- Até agora, vimos regras determinísticas. E se botar uma aleatoriedade no meio?
- Exemplo: regra do triângulo de sierpinski F → G−F−G
- Alteramos para:
  - F (0.99) → G−F−G
  - F (0.01) → G+F+G



Outras ideias

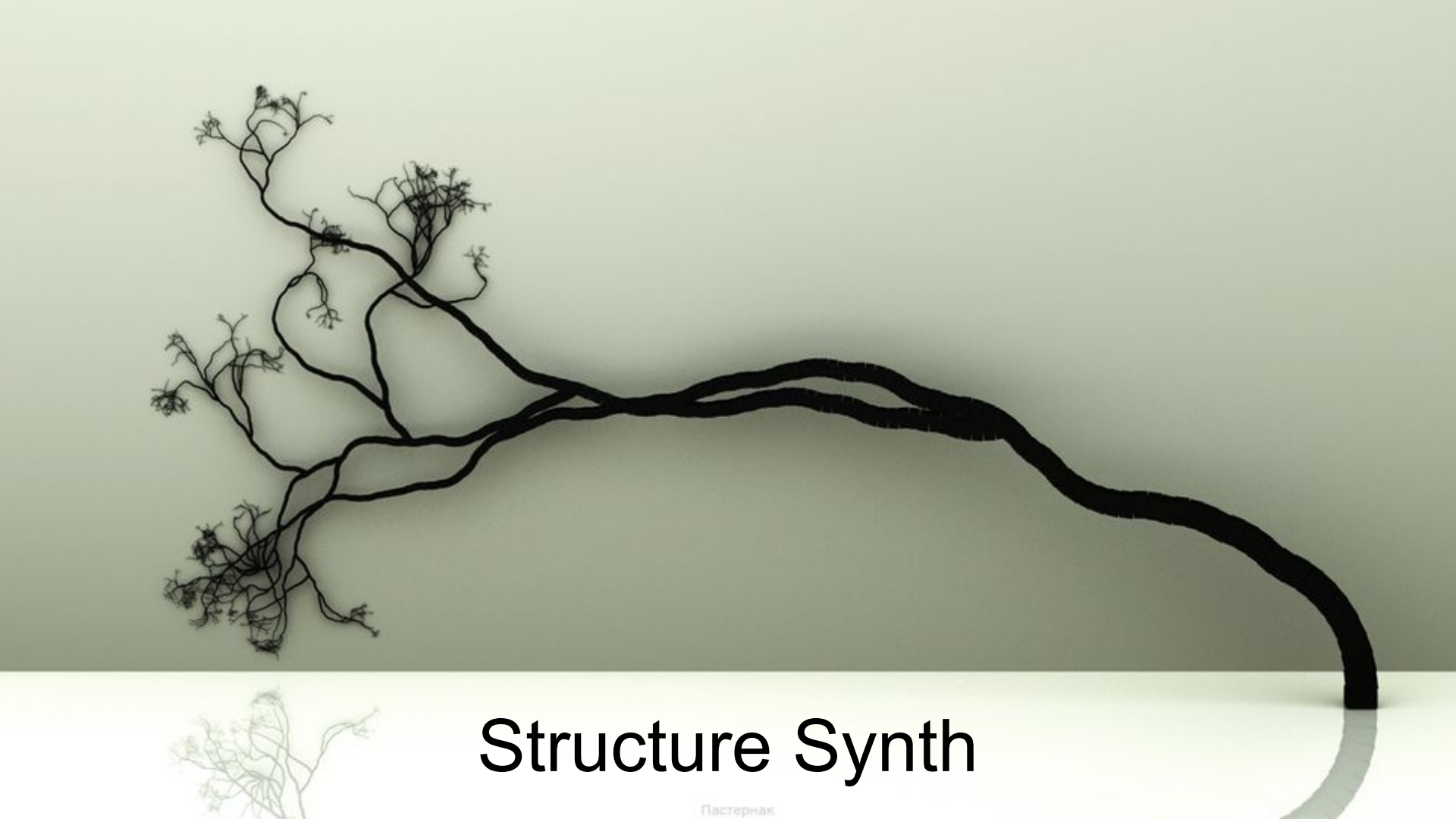
- Regras sensitivas ao contexto:
  - Resultado depende não só da letra, mas das letras vizinhas também
  - <F> F <G> → G−F−G: esta regra substitui F por G−F−G, mas só se F está entre F e G
- Alfabeto paramétrico:
  - Cada elemento tem parâmetros
  - Podem ser usados para desenhar ou para construir regras
  - Regras paramétricas permitem determinar os ângulos e comprimentos na hora, ou mudar o comportamento dependendo do número de gerações

n=5, δ=18°

ω : **plant**
p_1 : **plant** → **internode** + [**plant** + **flower**] − − //
[− − **leaf**] **internode** [+ + **leaf**] −
[**plant flower**] + + **plant flower**
p_2 : **internode** → F **seg** [// & & **leaf**] [// ∧ ∧ **leaf**] F **seg**
p_3 : **seg** → **seg** F **seg**
p_4 : **leaf** → [' { +f−ff−f+ | +f−ff−f }]
p_5 : **flower** → [& & & **pedicel** ‘ / **wedge** //// **wedge** ////
wedge //// **wedge** //// **wedge**]
p_6 : **pedicel** → FF
p_7 : **wedge** → [‘ ∧ F] [{ & & & & −f+f | −f+f }]

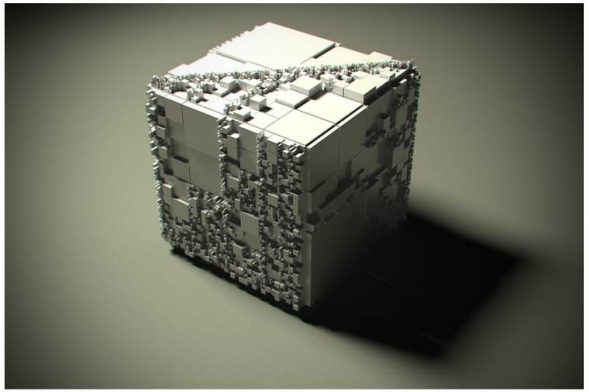
Outras ideias

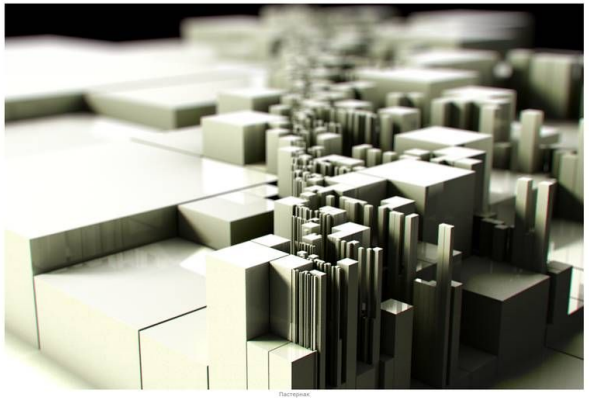
- Exemplo de geração de plantas
- “The Algorithmic Beauty of Plants”, 1990
- http://algorithmicbotany.org/papers/#abop
- Muitos exemplos legais
- Do básico ao avançado



Structure Synth

O que é e onde vive

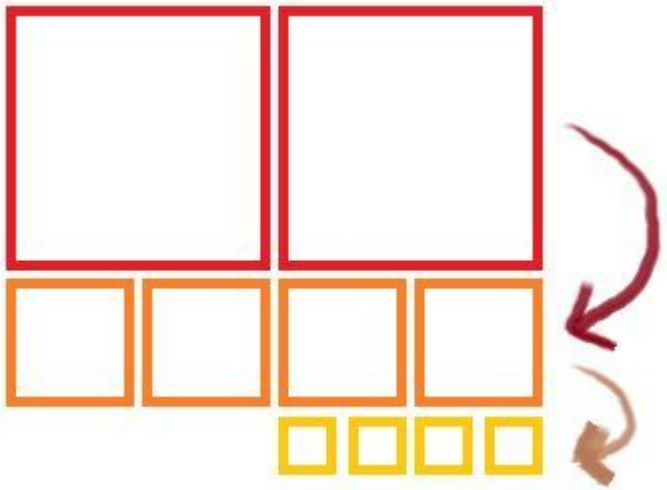




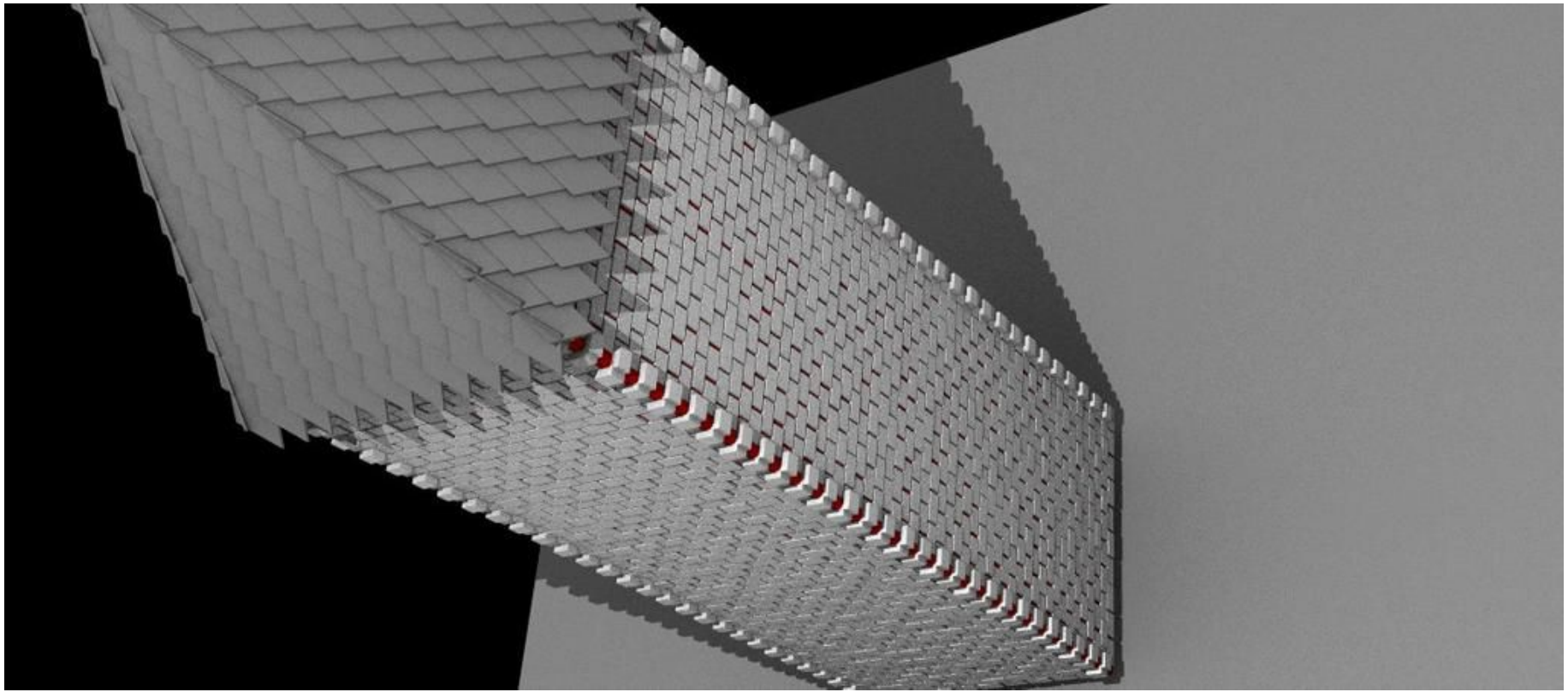
- Aplicativo para gerar L-Sistemas em 3D
- Gratuito
- Relativamente fácil de implementar coisas complexas
- http://structuresynth.sourceforge.net/

Lógica básica

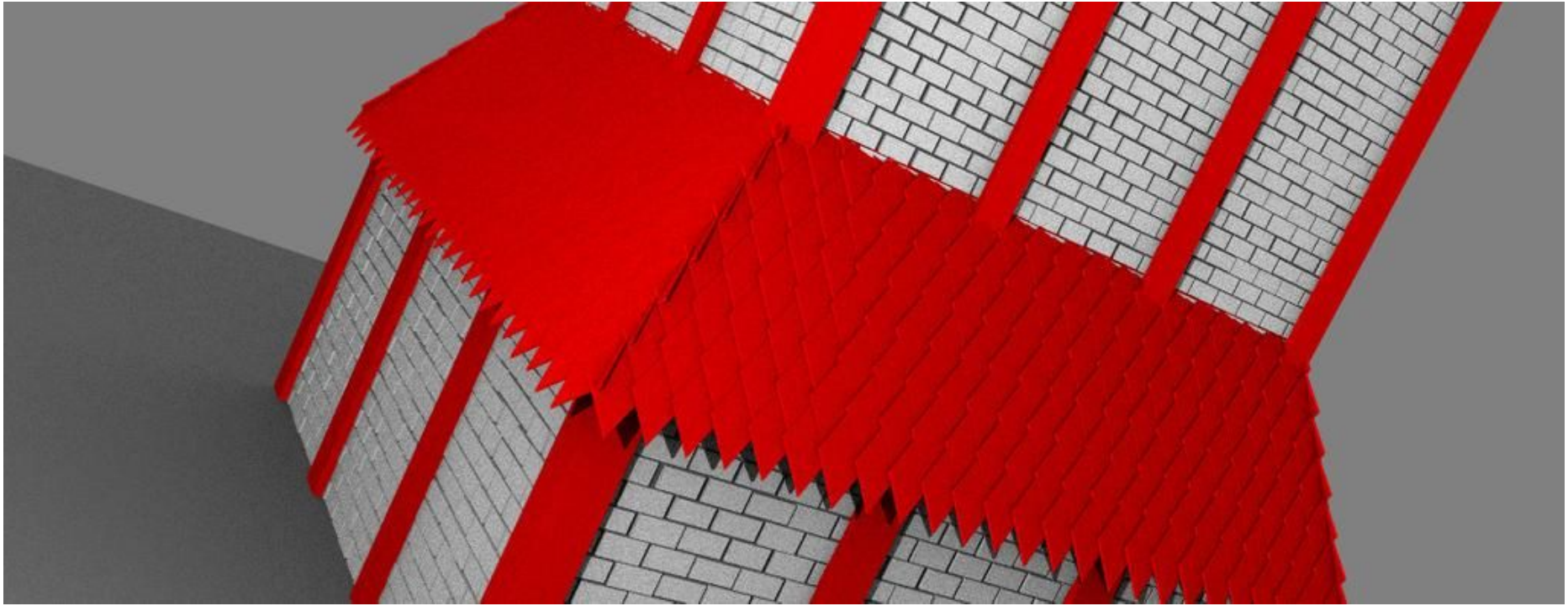
```
//executa regra
r
//definimos regra 1, com 7 iterações
rule r md 7 {
        { x 0.25 y -0.25 s 0.5 0.5 1} r
        { x -0.25 y -0.25 s 0.5 0.5 1} r
        base
}
//criamos uma base
rule base {
        {x 0.25 y 0.25 s 0.45 0.45 0.1}box
        {x -0.25 y 0.25 s 0.45 0.45 0.1}box
}
```



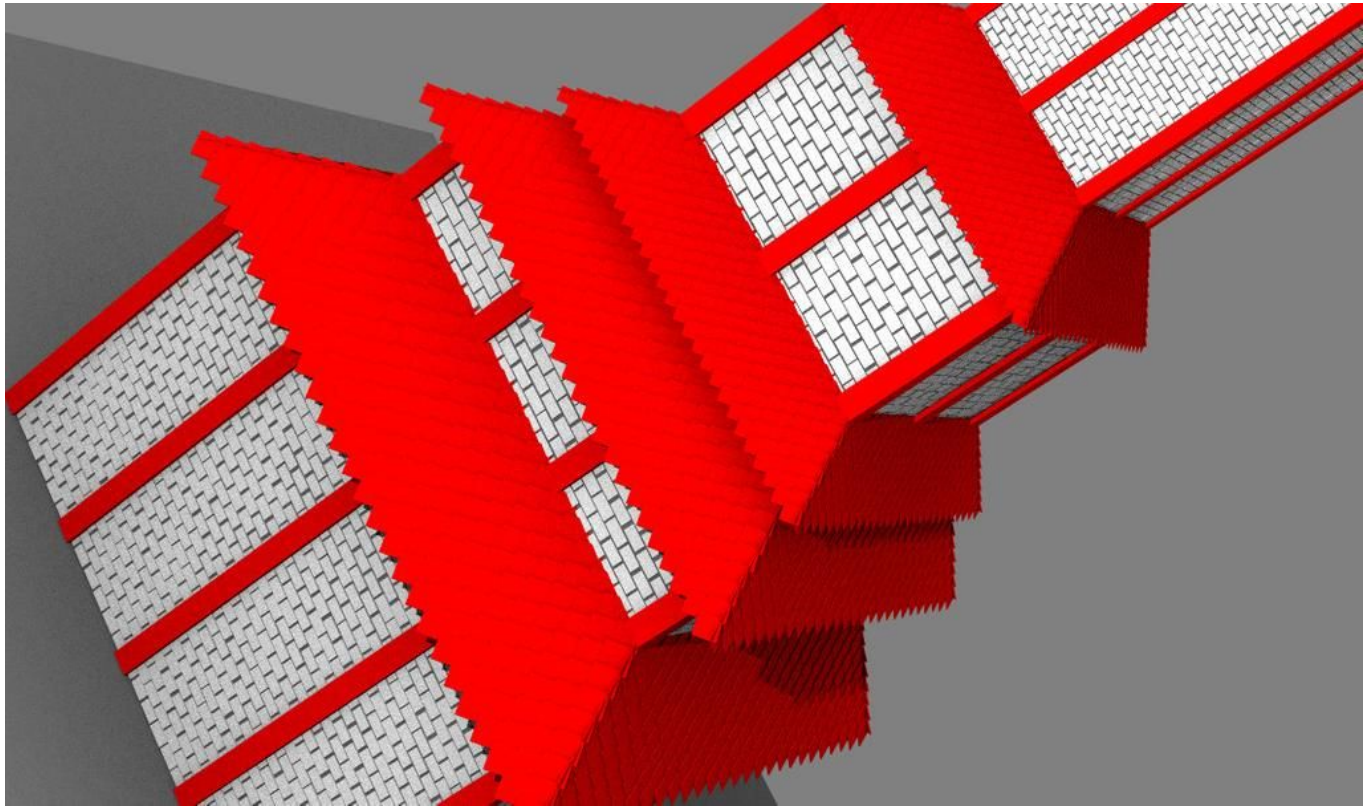
Evolução da torre



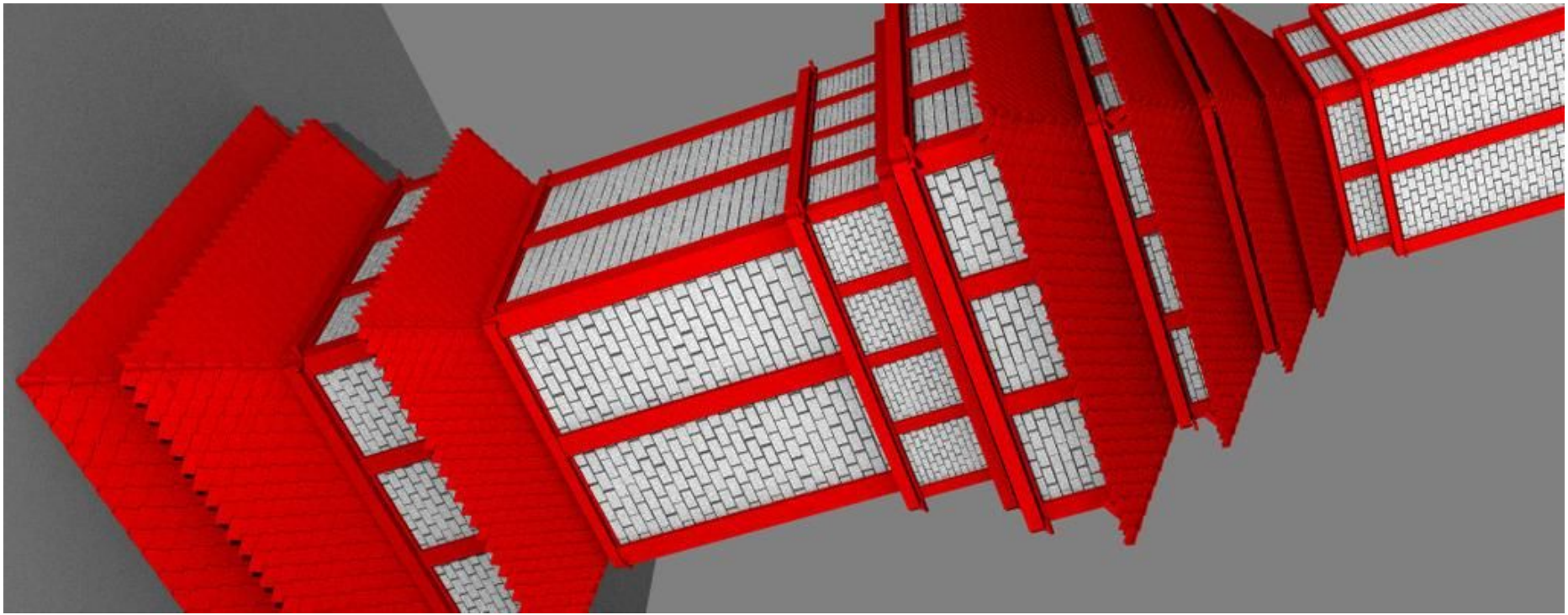
Evolução da torre



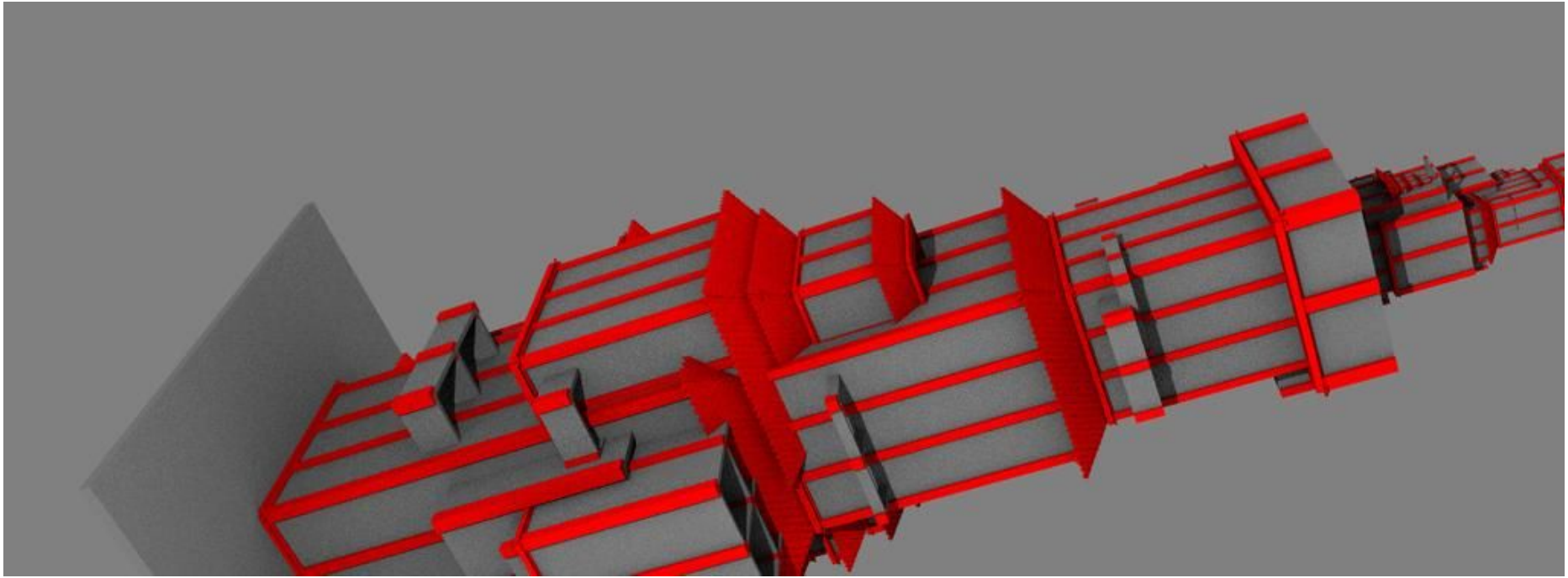
Evolução da torre



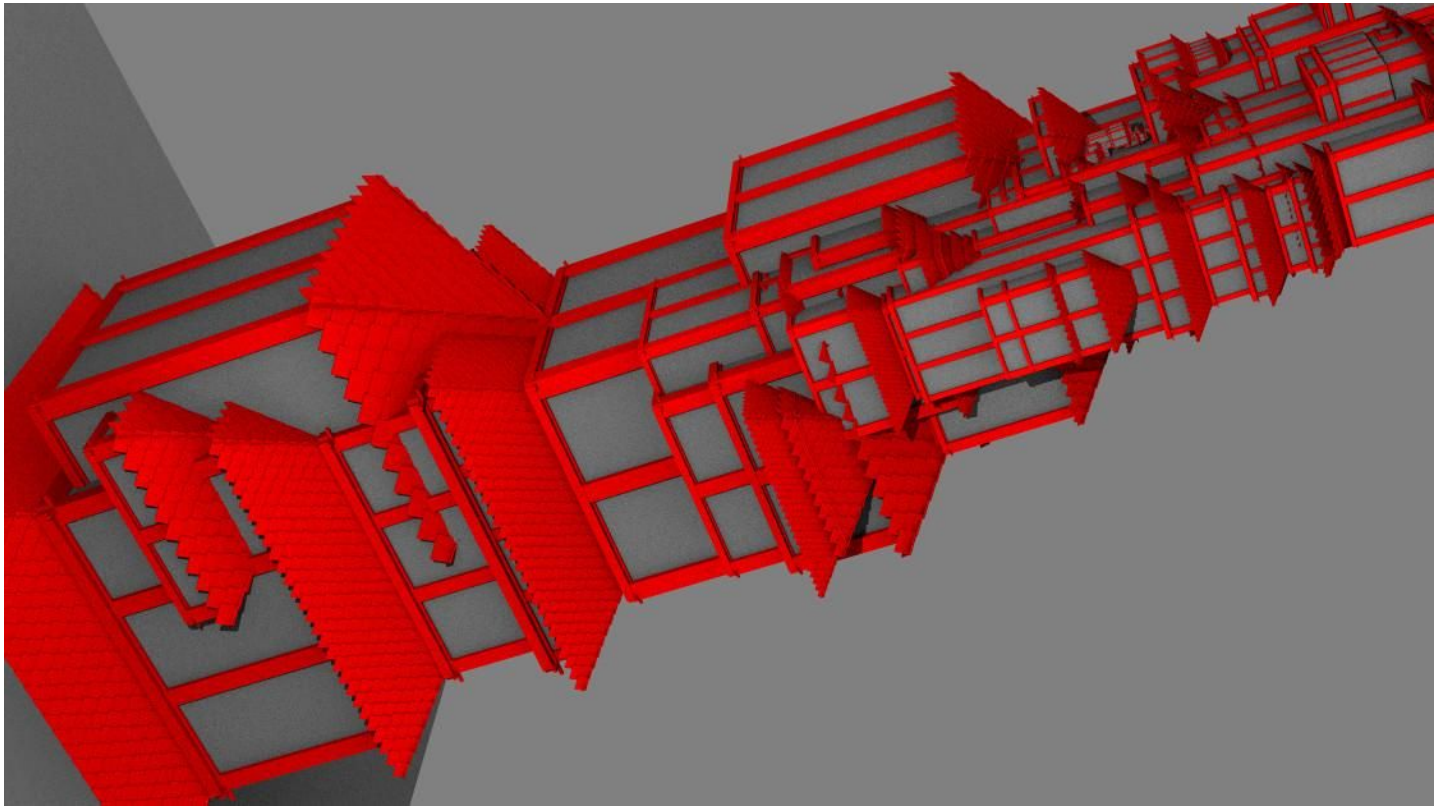
Evolução da torre



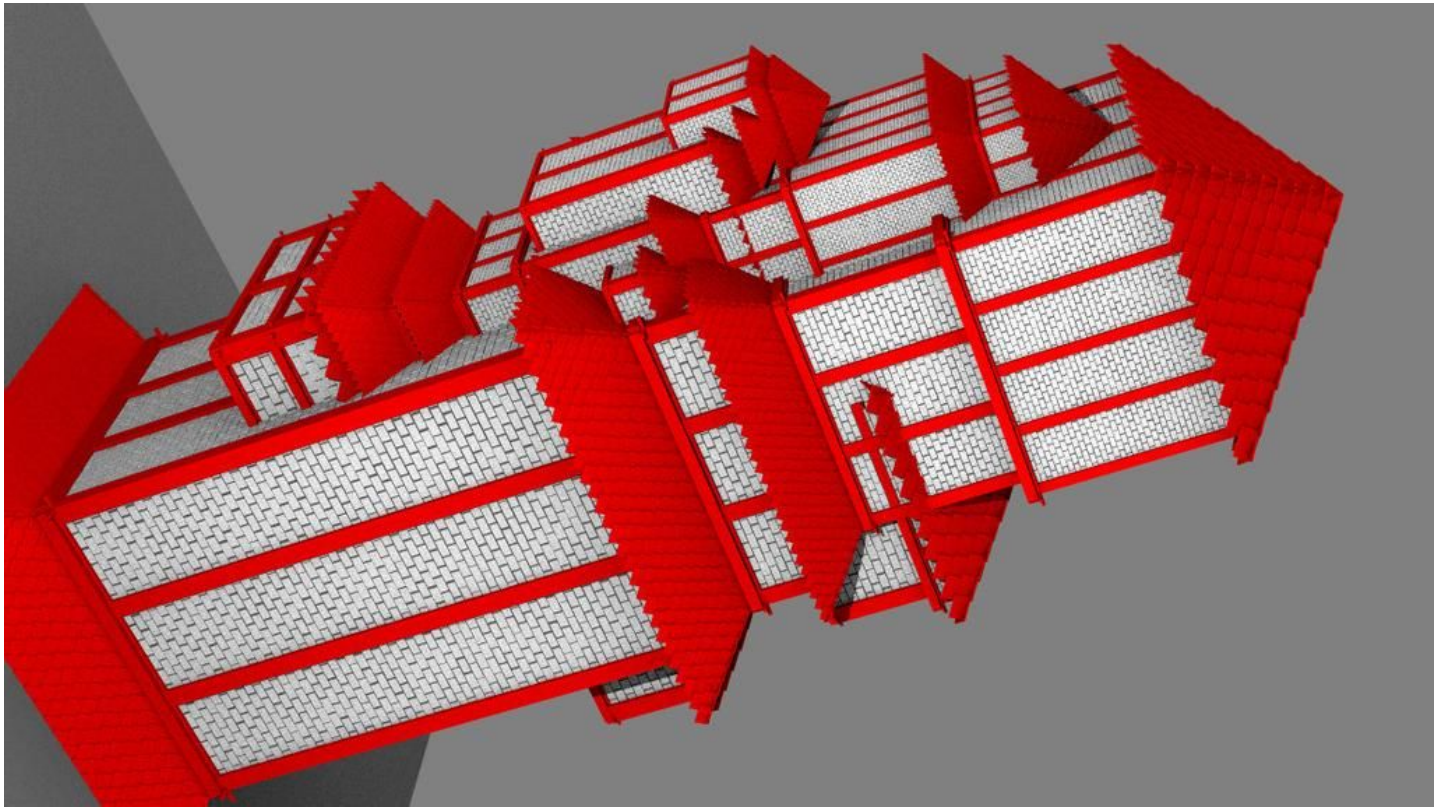
Evolução da torre



Evolução da torre



Evolução da torre



Evolução da torre