Université Lumière Lyon 2

Institut de la communication

Cours

Génération procédurale par l'exemple

Master 2 Programmation et Développement

Gilles Gesquière

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

Génération procédurale

Introduction par l'exemple

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Un premier exemple…

https://www.youtube.com/watch?v=-d2-PtK4F6Y

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Une introduction par Kate Compton

https://www.youtube.com/watch?v=WumyfLEa6bU

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

Exemple Ghost Recon Wildlands

https://www.gdcvault.com/play/1024029/-Ghost-Recon-Wildlands-Terrain

- GhostReconWildlands is the largest action adventure open world ever made at Ubisoft

The game is set in Bolivia
We have a lot of diversity in the game, 11 completely different biomes

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Réseaux d'eau et lac

Lakes, rivers and streams over a surface of 16km2

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Végétation et rochers

It’s mostly a natural environment with many forests
Millions of trees, bushes and rocks

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Génération de routes

Roads : over 600km of roads and even more paths

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Réseau ferré

A full railway network going across the world

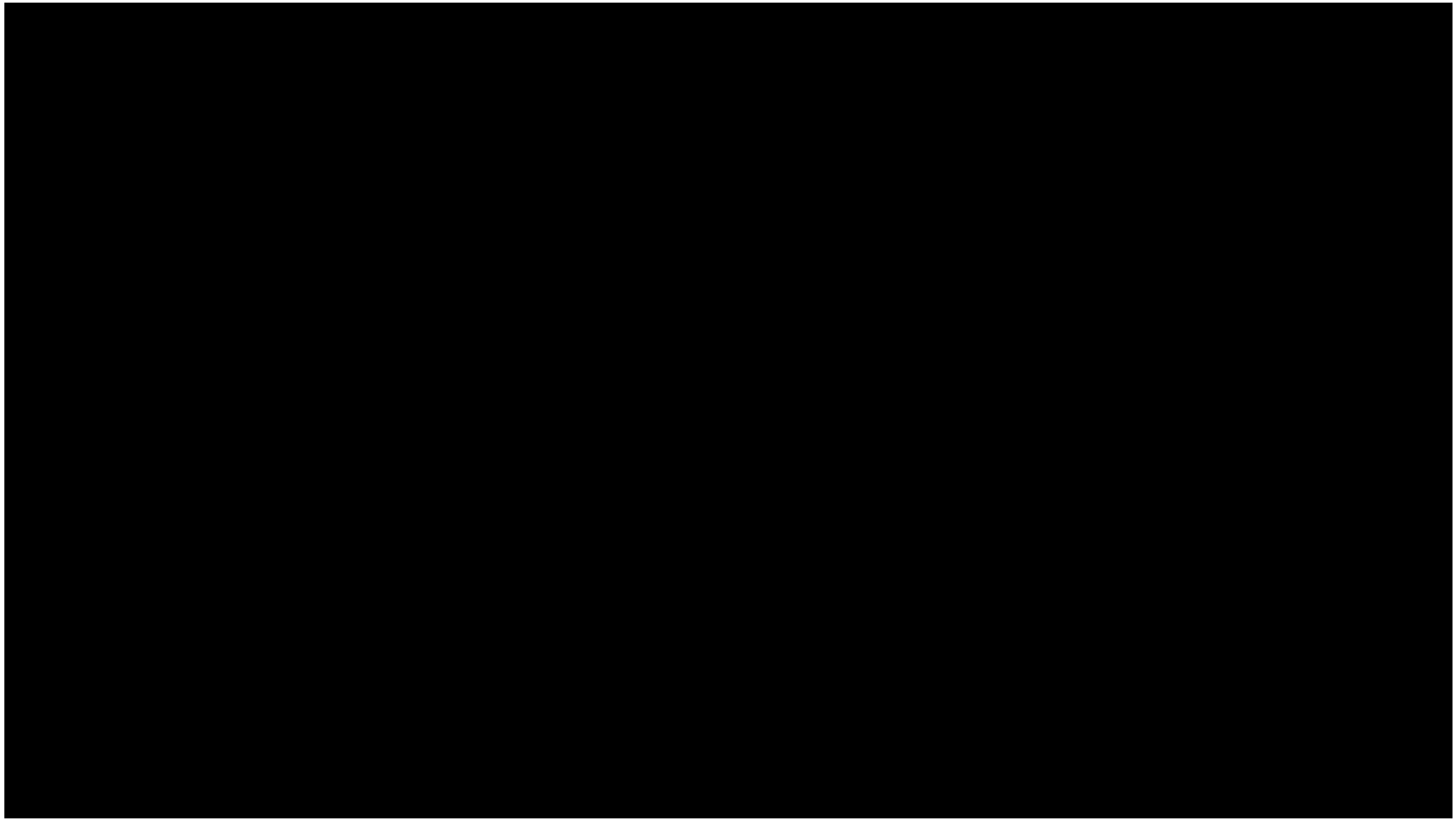
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Création de villages

Over 200 specific locations in the game: camps, landmarks, outposts and 58 fully procedurally built villages

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Modification de terrains

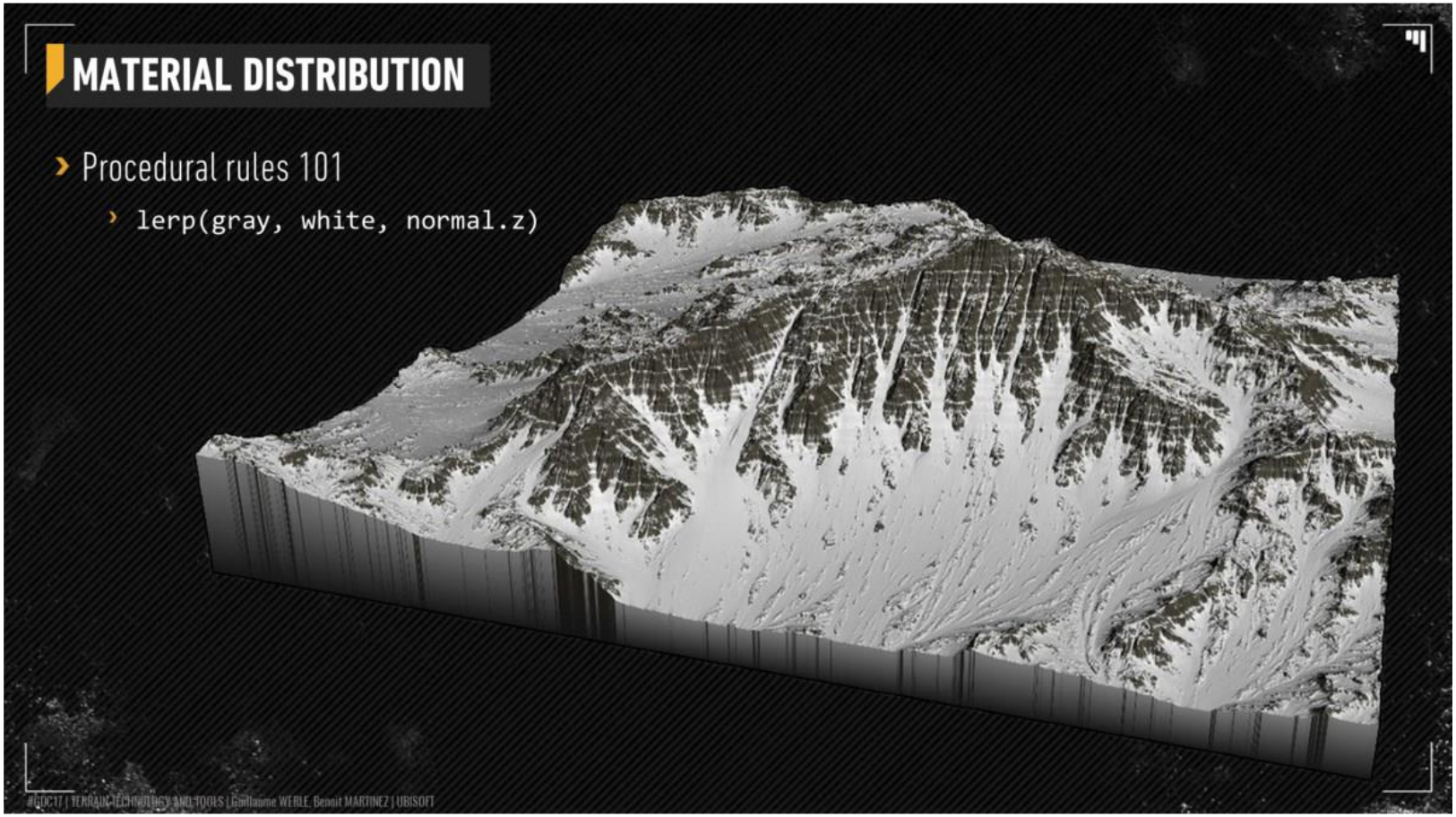


Base : real terrain
Editing terrain and sculpting tools; Possible to copy/ paste; Possible to erase

By looking at the size of the map led to the conclusion that painting everything manually was a terrible idea.
Need a semi automatic process.
=> Generate the material distribution procedurally

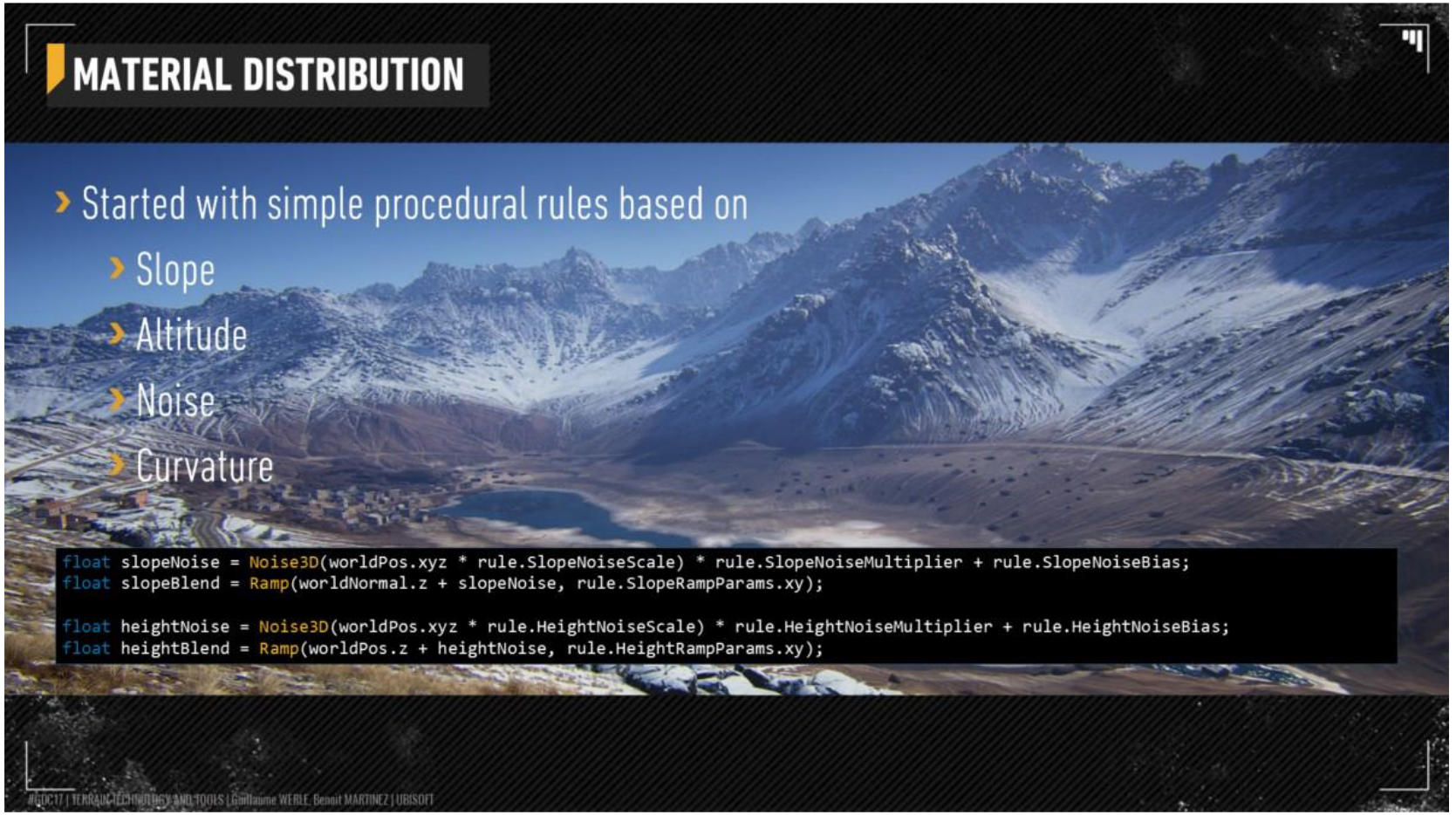
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Modifier les matériaux en fonction des propriétés du terrain



To give a better sense of the relief, the terrain editing tool was already tinting the heightmap depending on the normal direction.
When looking from far away, this simple rule is almost enough to generate a convincing mountain top

- Modifier les matériaux en fonction des propriétés du terrain

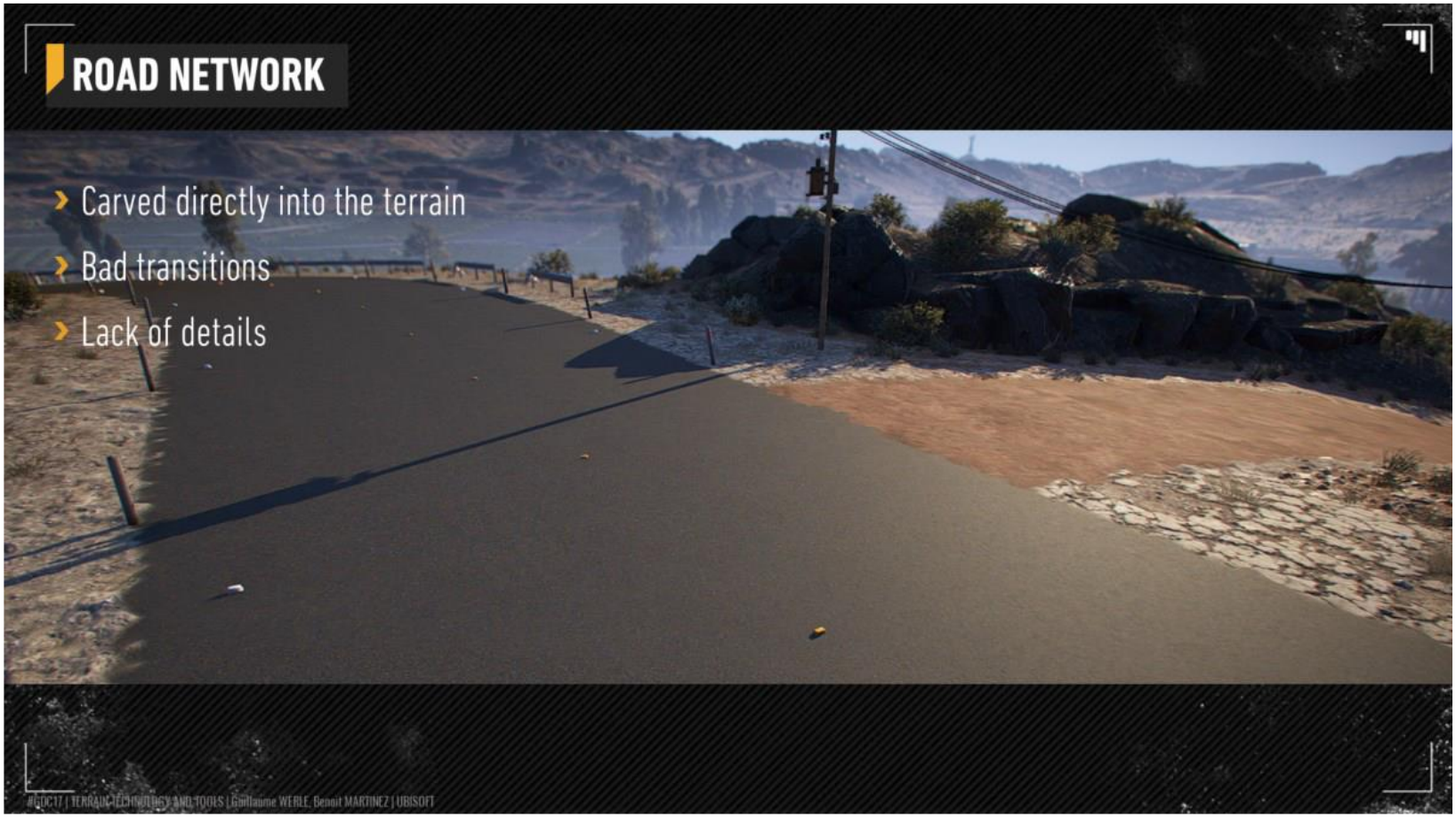


Add several conditions based on the topology, noise or simple kernels that could be evaluated in real time in a pixel shader

- Gestion du réseau routier

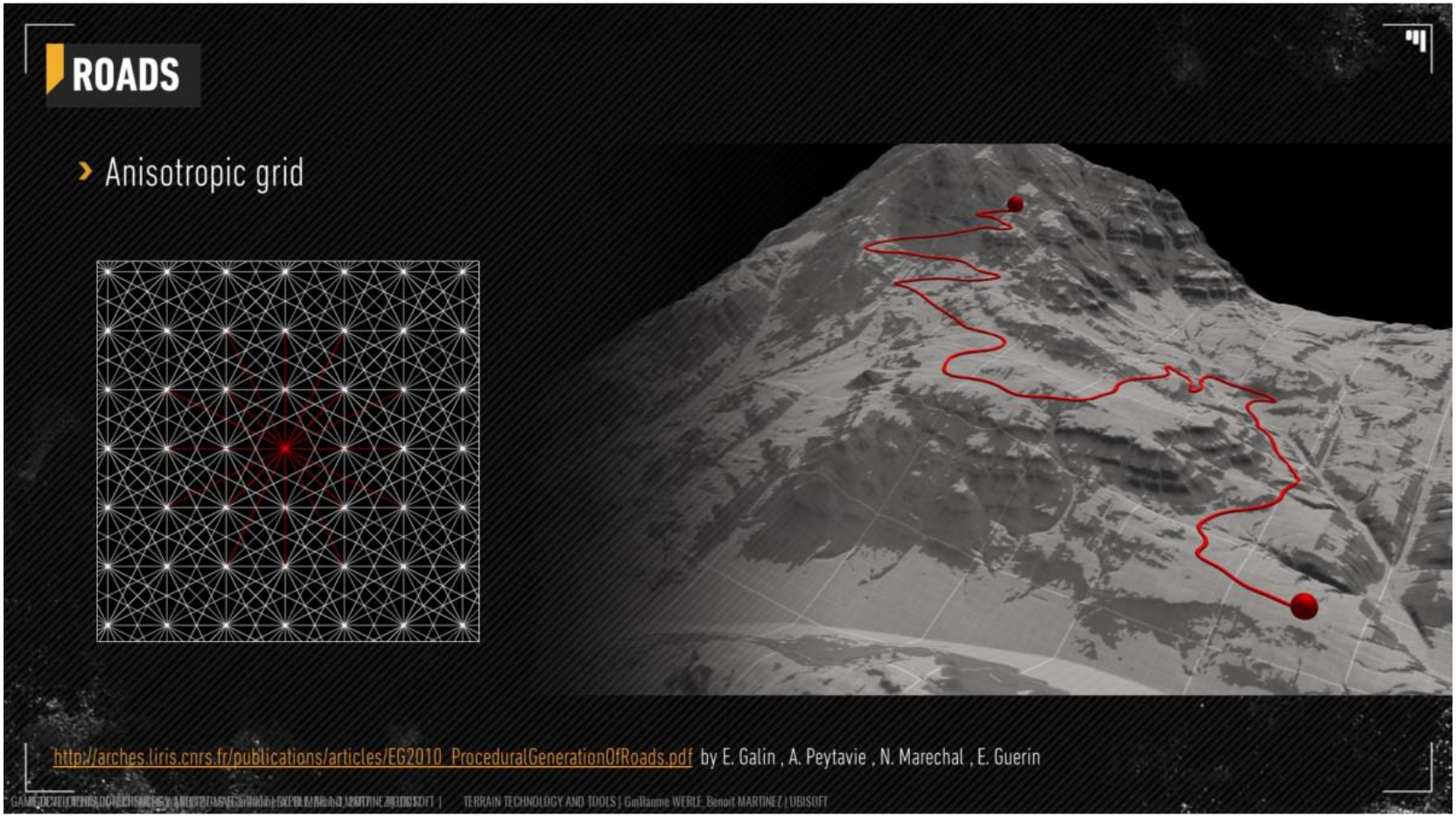
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Gestion du réseau routier



Carve/ sculpt the road directly into the terrain instead of representing it with traditional geometry.
But this technique would raised a few issue like cheap material transition and an obvious lack of details.

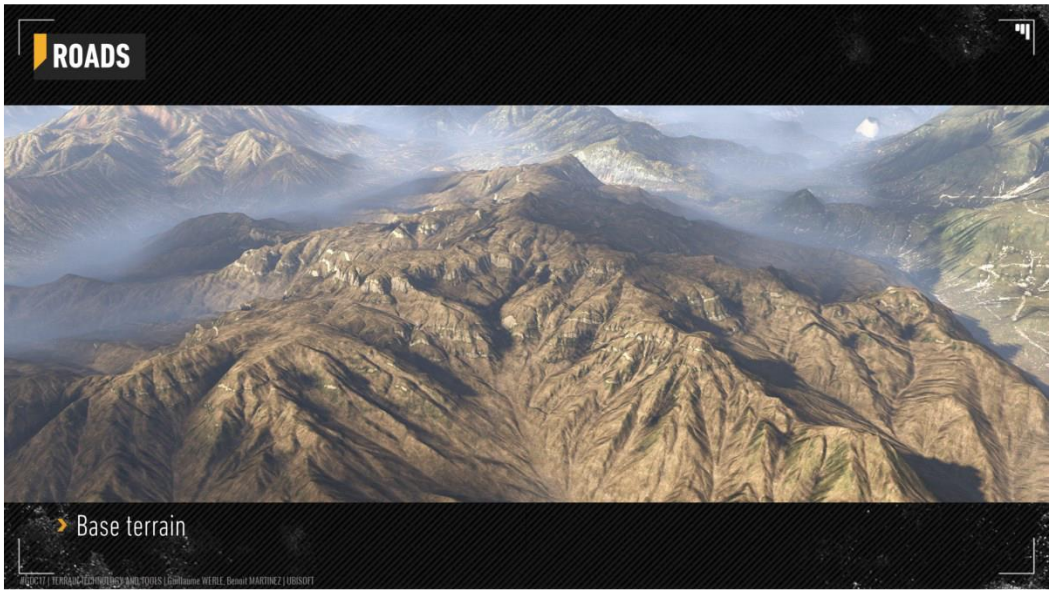
- Gestion du réseau routier



Using anisotropic shortest path algorithm.

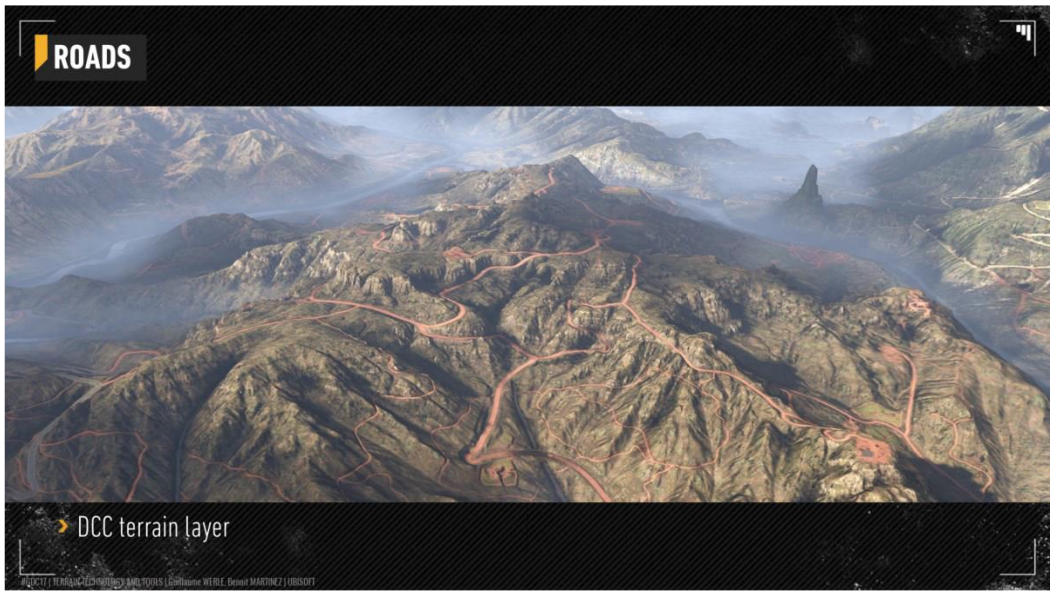
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Gestion du réseau routier. Placement de points sur le terrain

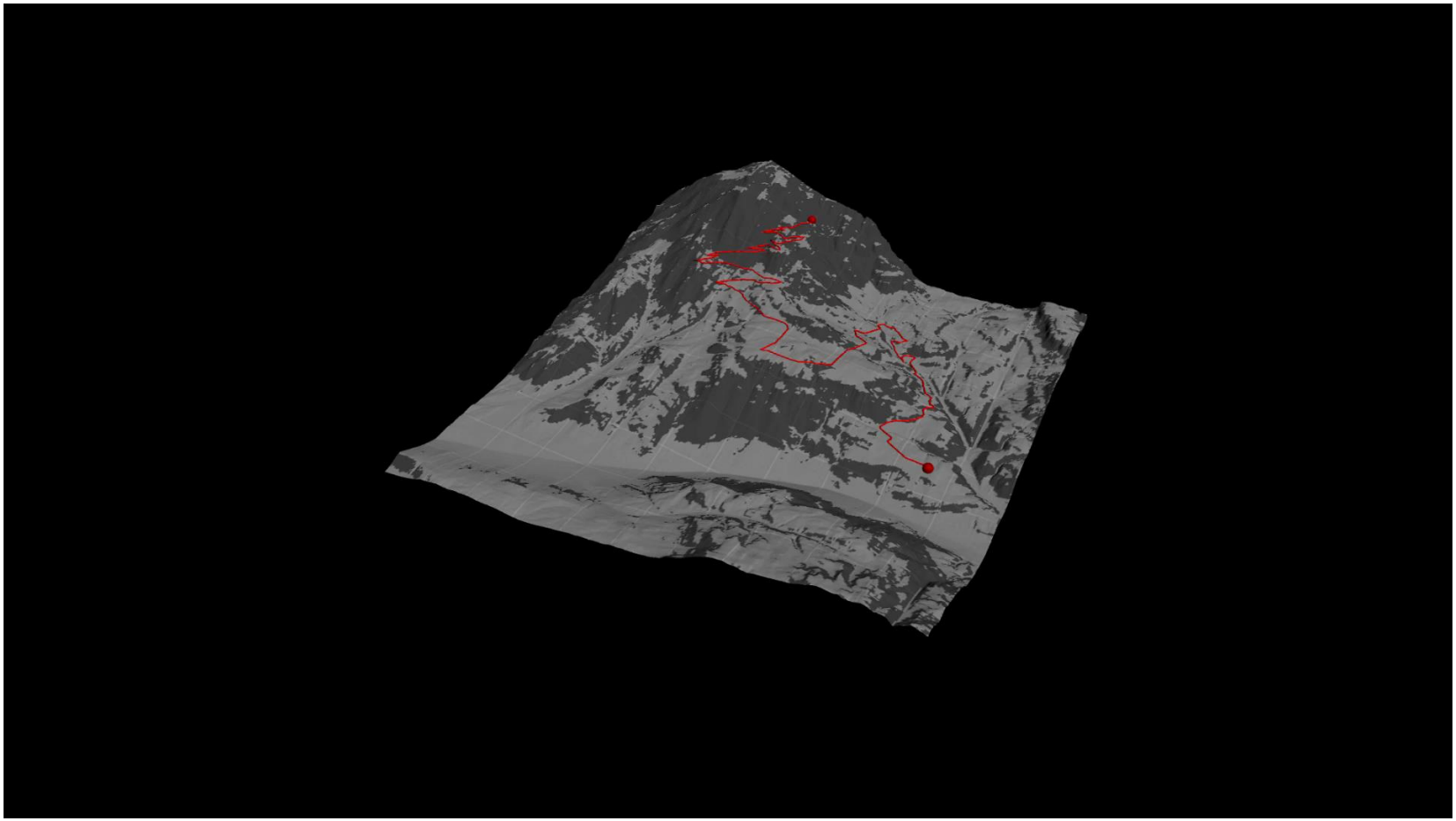


We obtain a consistent network with different types of roads and smallest paths.

Remember that it is all offline processed. Computing time is not (or not really) an issue (about 10minto compute a 2km*2km tile).



- Gestion du réseau routier; résultat



Video of the pathfindingover a tile of terrain and the full extent of road network (without small paths only main roads)

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Ajout de ponts et tunnels

https://vimeo.com/207479400

As previously mentioned some tools, like the bridge tool here, can be used in standalone mode to manually create bridges or can be embedded in a bigger tool to automatically create content.
Note: The bridges and tunnels are made of modules. No new or specific geometry is created, its all just instancing.

All the presentation of Ubisoft guys is available here
https://www.gdcvault.com/play/1024029/-Ghost-Recon-Wildlands-Terrain

- Une définition extraite de la thèse de JD Genevaux
  - La génération procédurale de contenu (Procedural Content Generation ou PCG) est une famille de méthodes où, au lieu de stocker des objets 3D, on stocke les paramètres d'un algorithme de construction. Ces méthodes ont été très utilisées dans les années 70-80, à l'époque où les ordinateurs avaient peu de mémoire. Pour créer des univers très étendus, on s'appuyait généralement sur ces méthodes qui utilisent beaucoup la notion d'aléatoire.

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

Génération procédurale

L-Systems
Procedural Terrain
Procedural Behavior

Based on the slides from 15-462 Spring 2007
http://graphics.cs.cmu.edu/nsp/course/15-462/Fall07/462/lectures/nov20a.ppt

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

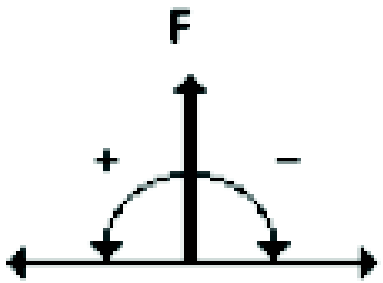
- Procedural content generation is attractive because it allows for significant database amplification
- Limited input data produces rich & varied output
  - ie: Perlin noise function + basic math gives fire, clouds, wood, etc.
- If it can be generated on the fly…
  - Artist doesn't have to design it
  - Don't need to store/transmit it

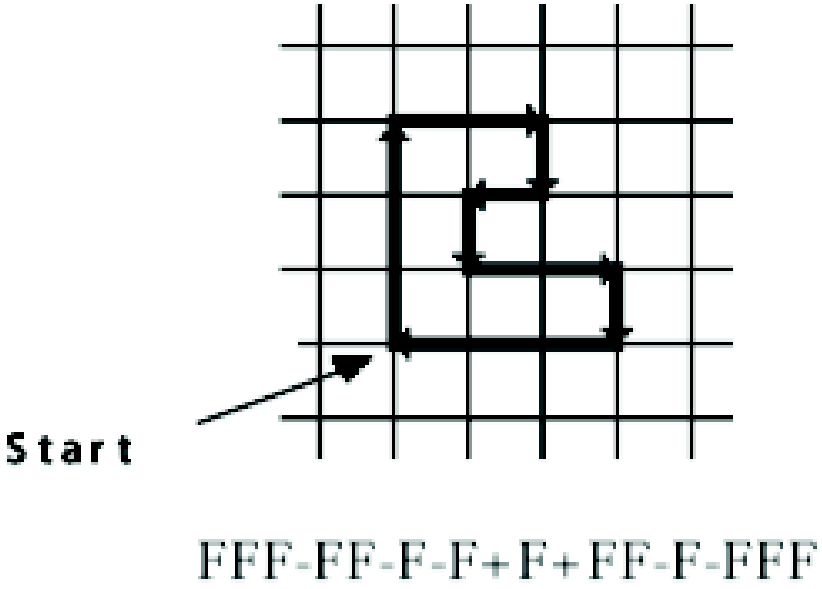
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Developed by Aristid Lindenmayer to model the development of plants
- Based on parallel string-rewriting rules
- Excellent for modelling organic objects and fractals
- Begin with a set of “productions” (replacement rules) and a “seed” axiom
- In parallel, all matching productions are replaced with their right-hand sides
- Ex:
  - Rules:
    - B -> ACA
    - A -> B
  - Axiom: AA
  - Sequence: AA, BB, ACAACA, BCBBCB, etc.
- Strings are converted to graphic representations via interpretation as turtle graphics commands

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

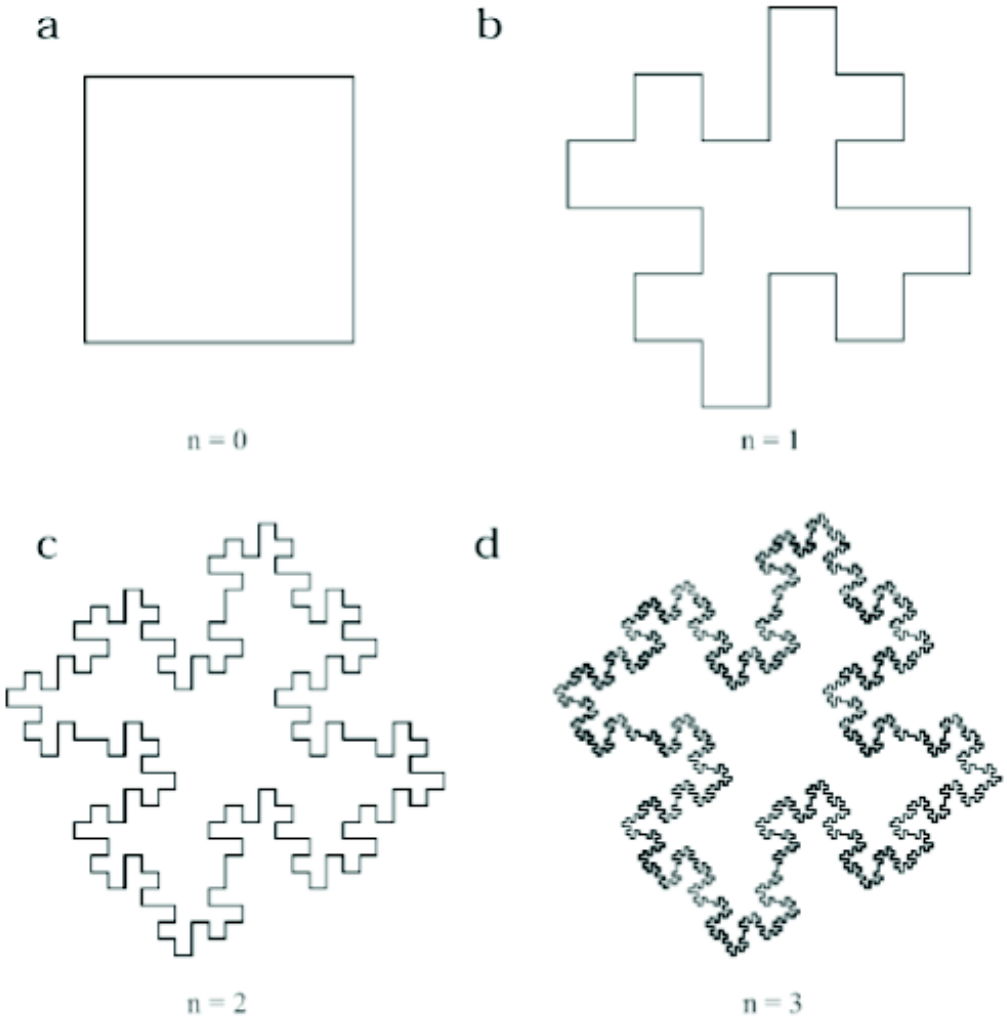
- Turtle Commands
  - F_x: move forward one step, drawing a line
  - f_x: move forward one step, without drawing a line
  - $+_x$: turn left by angle ∂
  - $-_x$: turn right by angle ∂





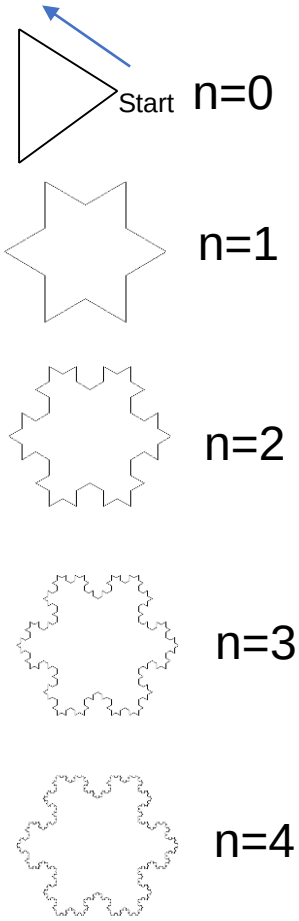
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Example: Koch Snowflake
  - Axiom: F-F-F-F ∂ :90 degrees
  - F -> F-F+F+FF-F-F+F



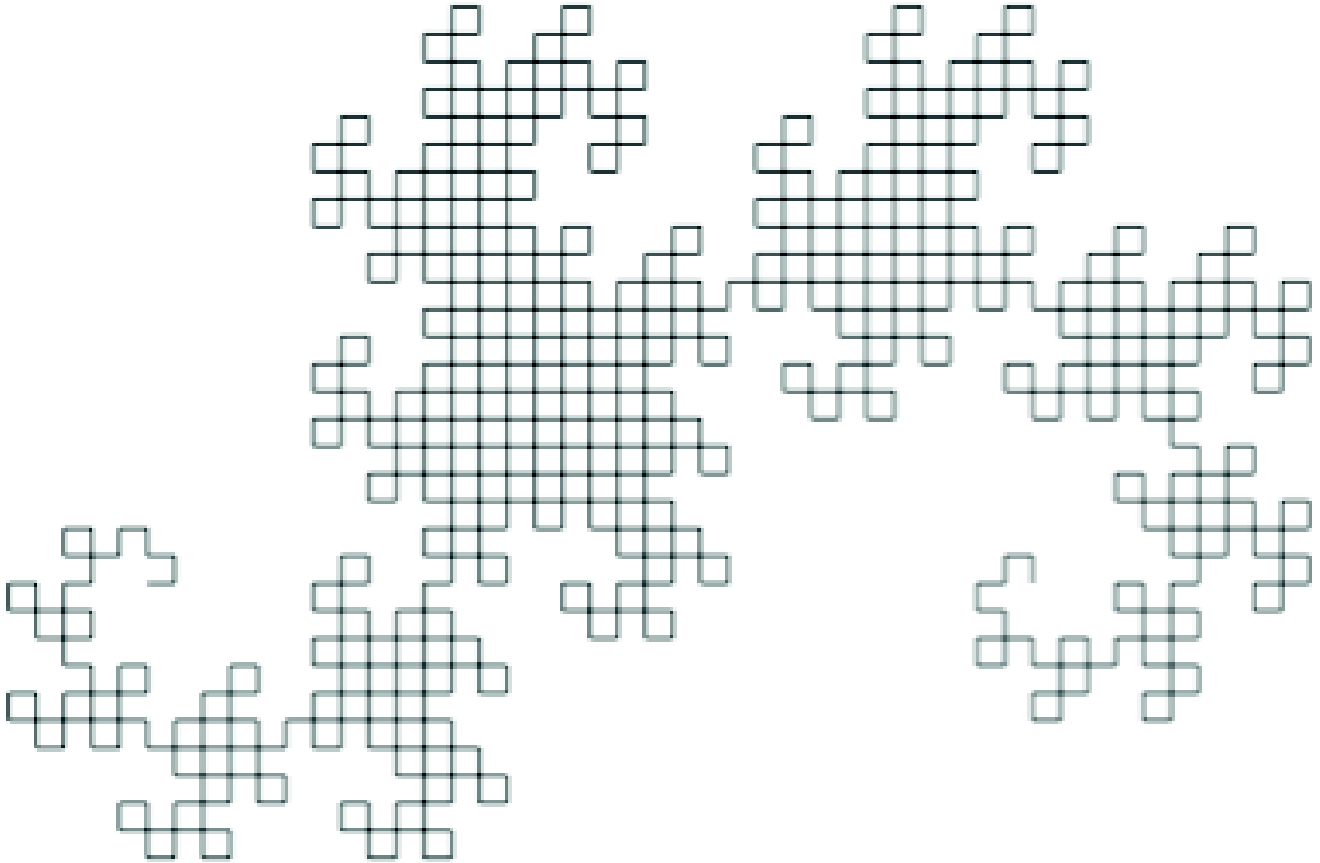
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Example : Koch Snowflakes
  - n=0 : f -- f -- f
  - n=1 : f + f -- f + f -- f + f -- f + f -- f + f -- f + f
  - n=2 : f + f -- f + f + f + f + f -- f + f -- f + f -- f + f + f + f -- f + f -- f + f -- f + f + f -- f + f -- f + f -- f + f + f + f + f -- f + f -- f + f -- f + f -- f + f + f + f + f -- f + f -- f + f -- f + f -- f + f + f + f -- f + f -- f + f + f + f -- f + f
  - Assign meaning to each component of the alphabet
    - f draw forwards
    - + turn 60 degrees right
    - -- turn 60 degrees left



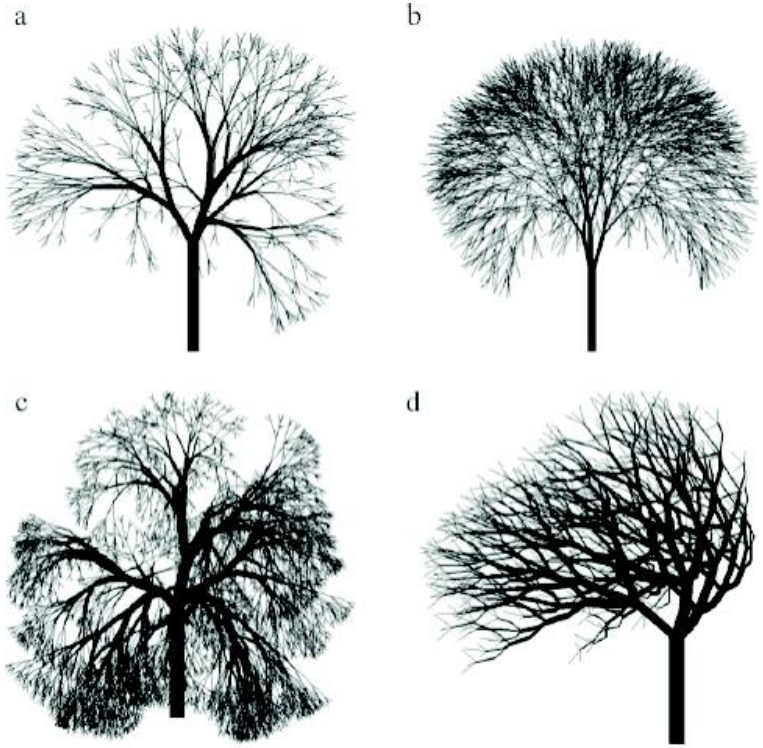
G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Example: Dragon Curve
  - Axiom:F_l ∂ :90 degrees n:10 iterations
  - F_l -> F_l+F_r+
  - F_r -> F_l-F_r-



G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Grammar: Extensions
  - Basic L-Systems have inspired a large number of variations
  - Context sensitive: productions look at neighboring symbols
  - Bracketed: save/restore state (for branches)
  - Stochastic: choose one of n matching productions randomly
  - Parametric: variables can be passed between productions



G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Pour aller plus loin
  - Algorithmic Botany
    - Covers many variants of L-Systems, formal derivations, and exhaustive coverage of different plant types.
    - http://algorithmicbotany.org/papers
  - PovTree
    - http://arbaro.sourceforge.net/
  - Construction de végétations (arbres) : Soeren Pirk :
    - https://www.youtube.com/watch?v=5HJSHJvlMFE

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Exemples

https://www.youtube.com/watch?v=E1B4UoSQMFw&list=PLQu8TxiWYLxID80vTywVWjaloCyytE2gB&index=2

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Création d'un réseau de rues dans une ville [Parish01]

- Start with a single street
- Branch & extend w/ parametric L-System
- Parameters of the string are tweaked by goals/constraints
- Goals control street direction, spacing
- Contraints allow for parks, bridges, road loops
- Once we have streets, we can form buildings with another L-System
- Building shapes are represented as CSG operations on simple shapes

Procedural Modeling of Cities, Parish, Müller, 2001
https://cgl.ethz.ch/Downloads/Publications/Papers/2001/p_Par01.pdf)

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Noise Functions
  - Seeded pseudo-random number generator
  - Over R^n
  - Approximation to gaussian filtered noise
  - Implemented as a pseudo-random spline
  - The trick is to make it fast

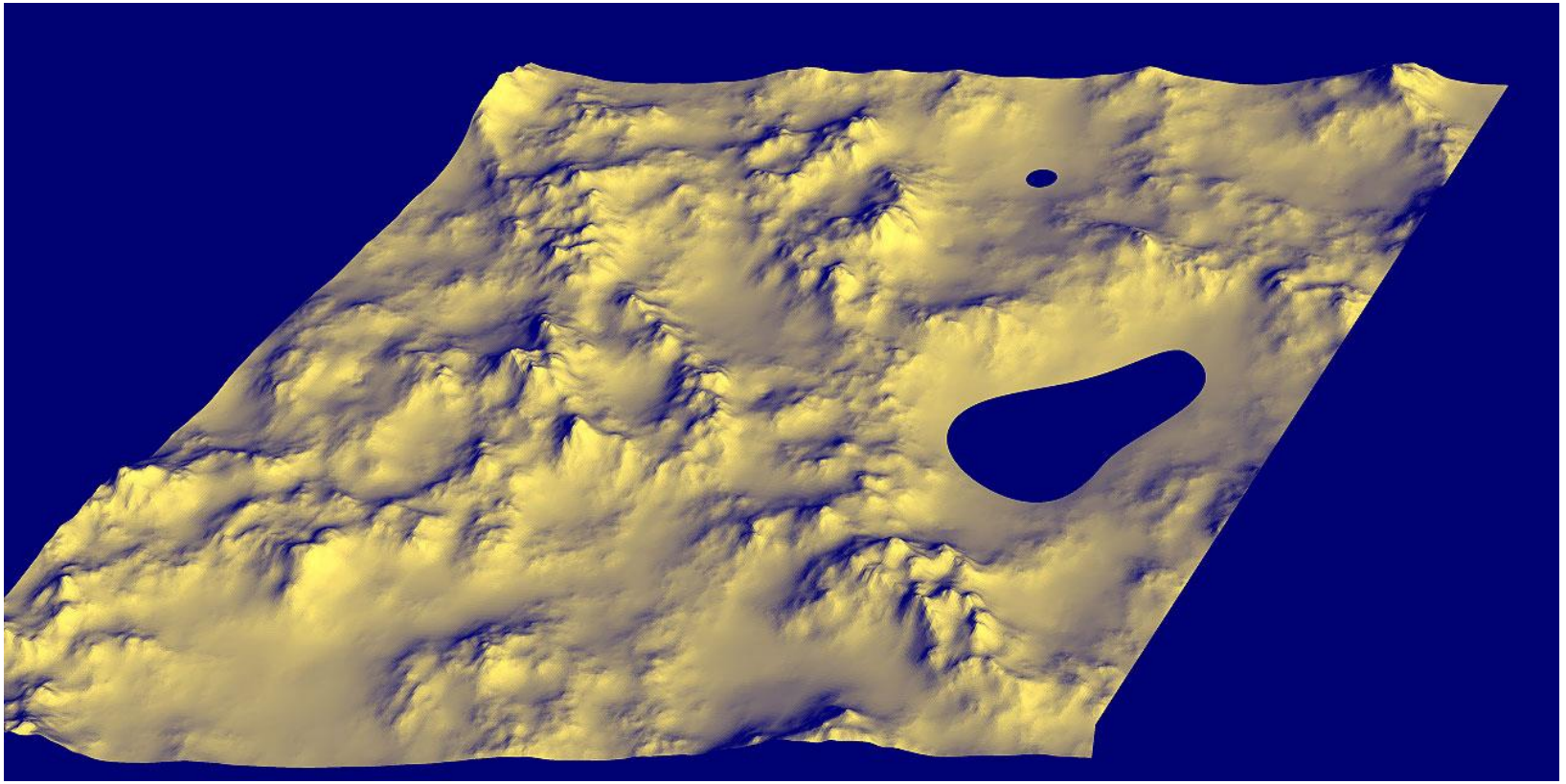
https://www.youtube.com/watch?v=wbpMiKiSKm8

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Using Perlin Noise in Unity
  - Noise : https://www.youtube.com/watch?v=WP-Bm65Q-1Y
  - Adding Octaves : https://www.youtube.com/watch?v=MRNFcywkUSA
  - Adding colors : https://www.youtube.com/watch?v=RDQK1_SWFuc
  - Mesh creation : https://www.youtube.com/watch?v=4RpVBYW1r5M
  - Level of details : https://www.youtube.com/watch?v=417kJGPKwDg
  - Endless terrain : https://www.youtube.com/watch?v=xlSkYjiE-Ck
  - …

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Adding water
  - Use an elevation threshold ($z < z_{water}$)



F.K. Musgrave

- Utilisation de produits comme Terragen

- Commercial product (free for personal use)
- Website: http://www.planetside.co.uk/terragen/

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- http://martindevans.me/game-development/2015/12/11/Procedural-Generation-For-Dummies-Roads/
  - Road (with grid or randomized)
  - Buildings (differents buildings)
  - Vegetation with L-System
  - Terrain generation

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

Some links

- https://www.youtube.com/watch?v=bkWU6SlhXm0 (procedural generation by implementing a building generator in Unity)
- https://www.youtube.com/watch?v=xGi0Hr-kNw0 (Procedurally Generated City And Creating A Grid System)
- https://www.youtube.com/watch?v=xkuniXI3SEE (Generating a Procedural City with Unity 5 Part 1)
- https://marian42.de/article/wfc/ (Infinite procedurally generated city with the Wave Function Collapse algorithm)

Procedural Content Generation in Games

A textbook and an overview of current research

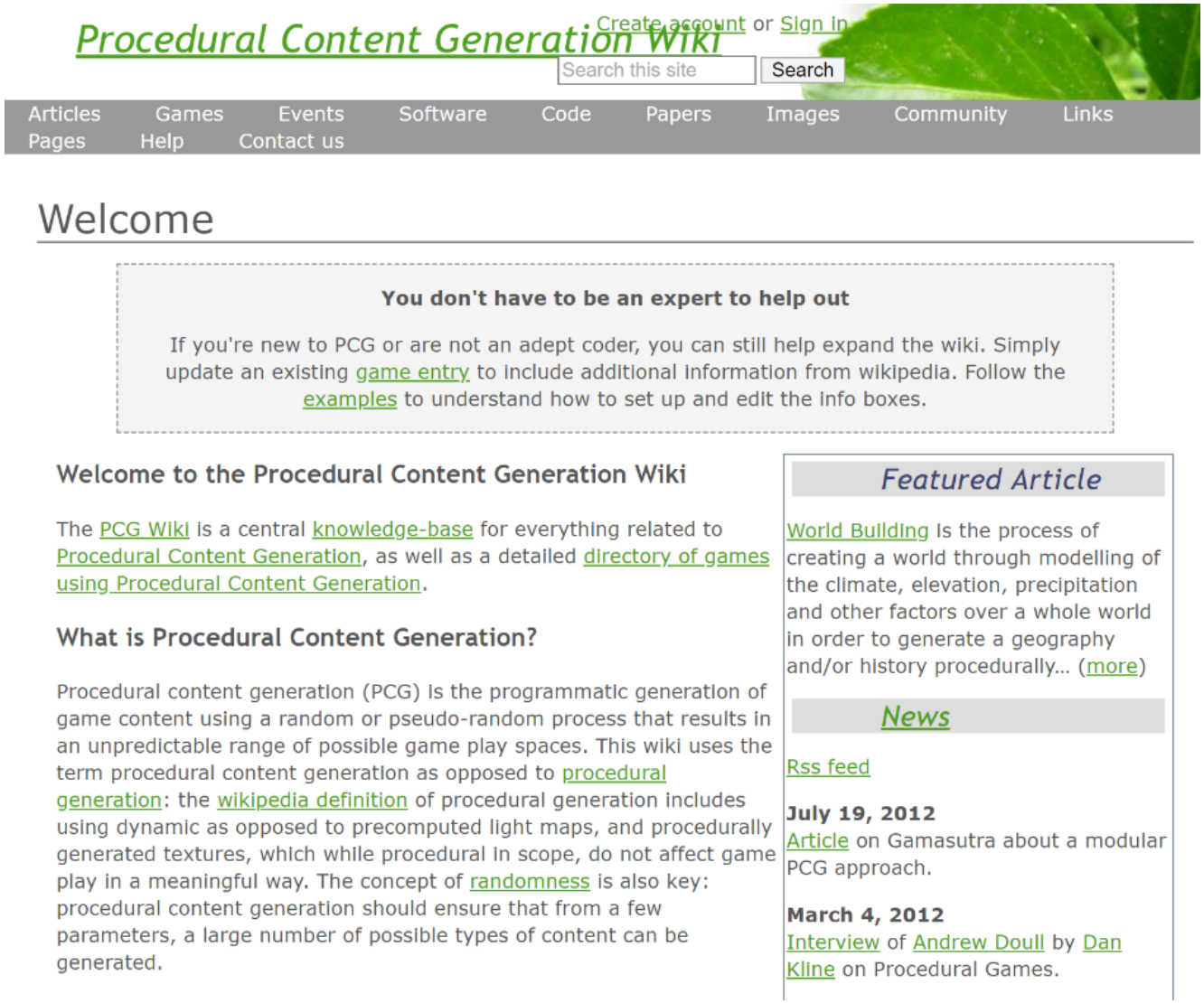
About the book | Authors | Citing the book | Supplementary material | *Table of Contents*

Table of Contents

Noor Shaker, Julian Togelius, and Mark J. Nelson

http://pcgbook.com/

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr



Procedural Content Generation Wiki

Create account or Sign in

Search this site | Search

Articles | Games | Events | Software | Code | Papers | Images | Community | Links
Pages | Help | Contact us

Welcome

You don't have to be an expert to help out

If you're new to PCG or are not an adept coder, you can still help expand the wiki. Simply update an existing game entry to include additional information from wikipedia. Follow the examples to understand how to set up and edit the info boxes.

Welcome to the Procedural Content Generation Wiki

The PCG Wiki is a central knowledge-base for everything related to Procedural Content Generation, as well as a detailed directory of games using Procedural Content Generation.

What is Procedural Content Generation?

Procedural content generation (PCG) is the programmatic generation of game content using a random or pseudo-random process that results in an unpredictable range of possible game play spaces. This wiki uses the term procedural content generation as opposed to procedural generation: the wikipedia definition of procedural generation includes using dynamic as opposed to precomputed light maps, and procedurally generated textures, which while procedural in scope, do not affect game play in a meaningful way. The concept of randomness is also key: procedural content generation should ensure that from a few parameters, a large number of possible types of content can be generated.

Featured Article

World Building is the process of creating a world through modelling of the climate, elevation, precipitation and other factors over a whole world in order to generate a geography and/or history procedurally... (more)

News

Rss feed

July 19, 2012
Article on Gamasutra about a modular PCG approach.

March 4, 2012
Interview of Andrew Doull by Dan Kline on Procedural Games.

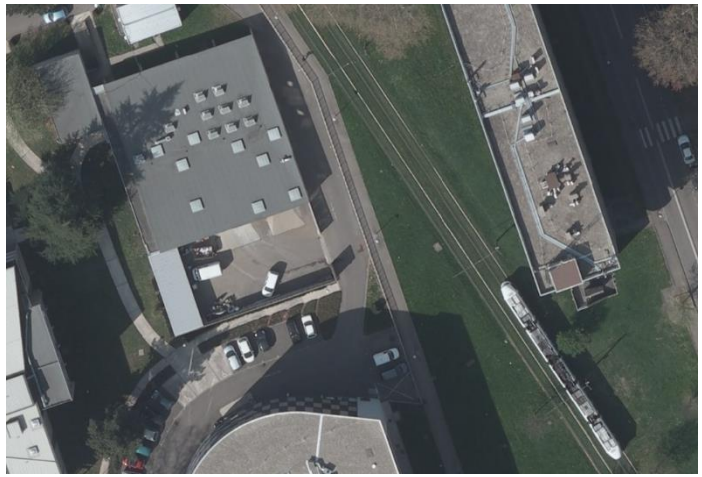
http://pcg.wikidot.com/

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr

- Génération d'objets à partir d'observation du monde réel…

http://arches.liris.cnrs.fr/

- Start with data as input



GRANDLYON communauté urbaine

G. Gesquière - Université Lumière Lyon 2- Laboratoire LIRIS
Gilles.gesquiere@univ-lyon2.fr