

Generacion Procedural de Contenido - PGC

Jorge Eduardo Ardila
[@datelligence](#)

Qué

Para qué

Quien

Como

Donde

PGC

Qué

Creacion algoritmica de
datos.

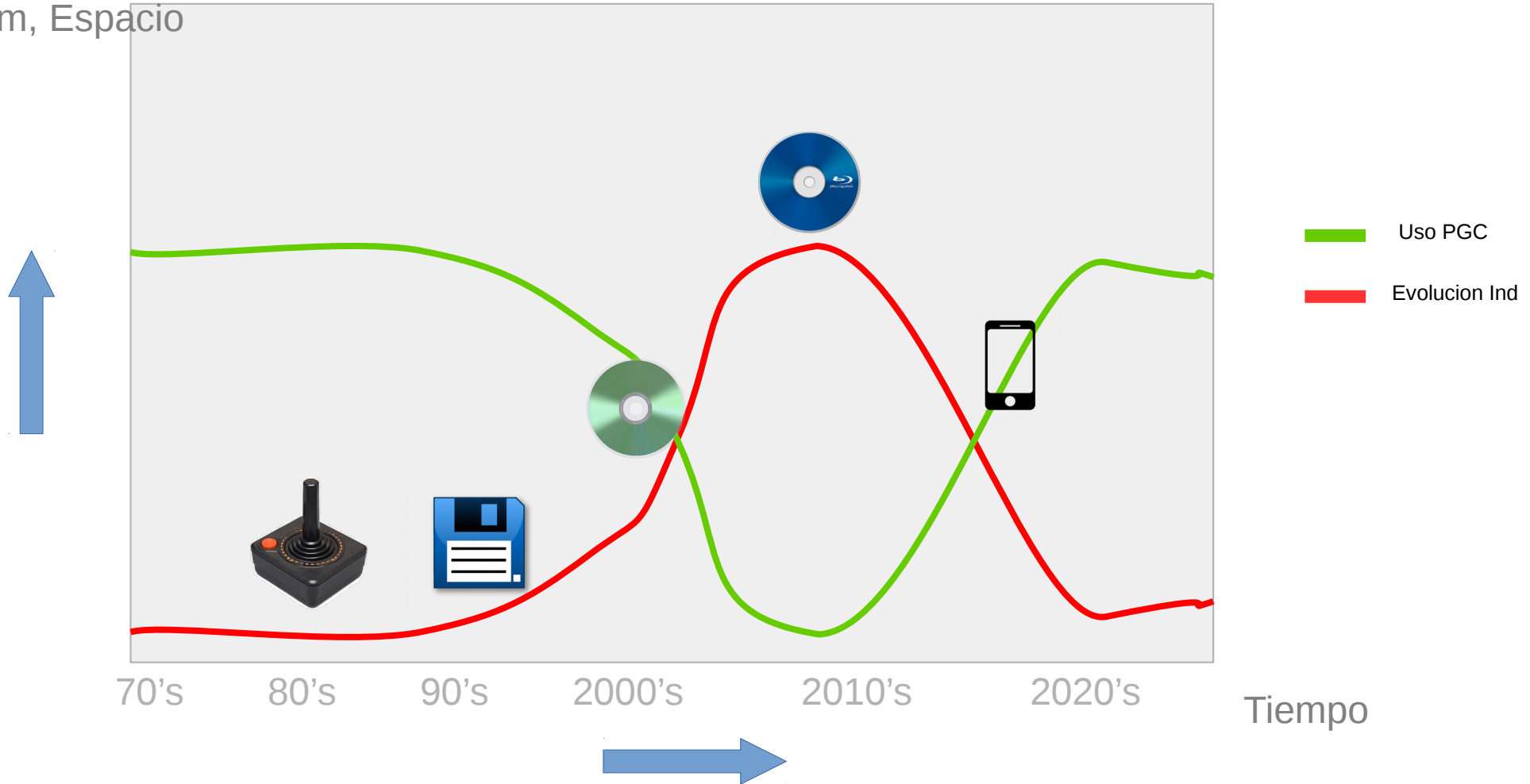
PGC

Para qué?

- Tamaño
- Diversidad

Para qué?

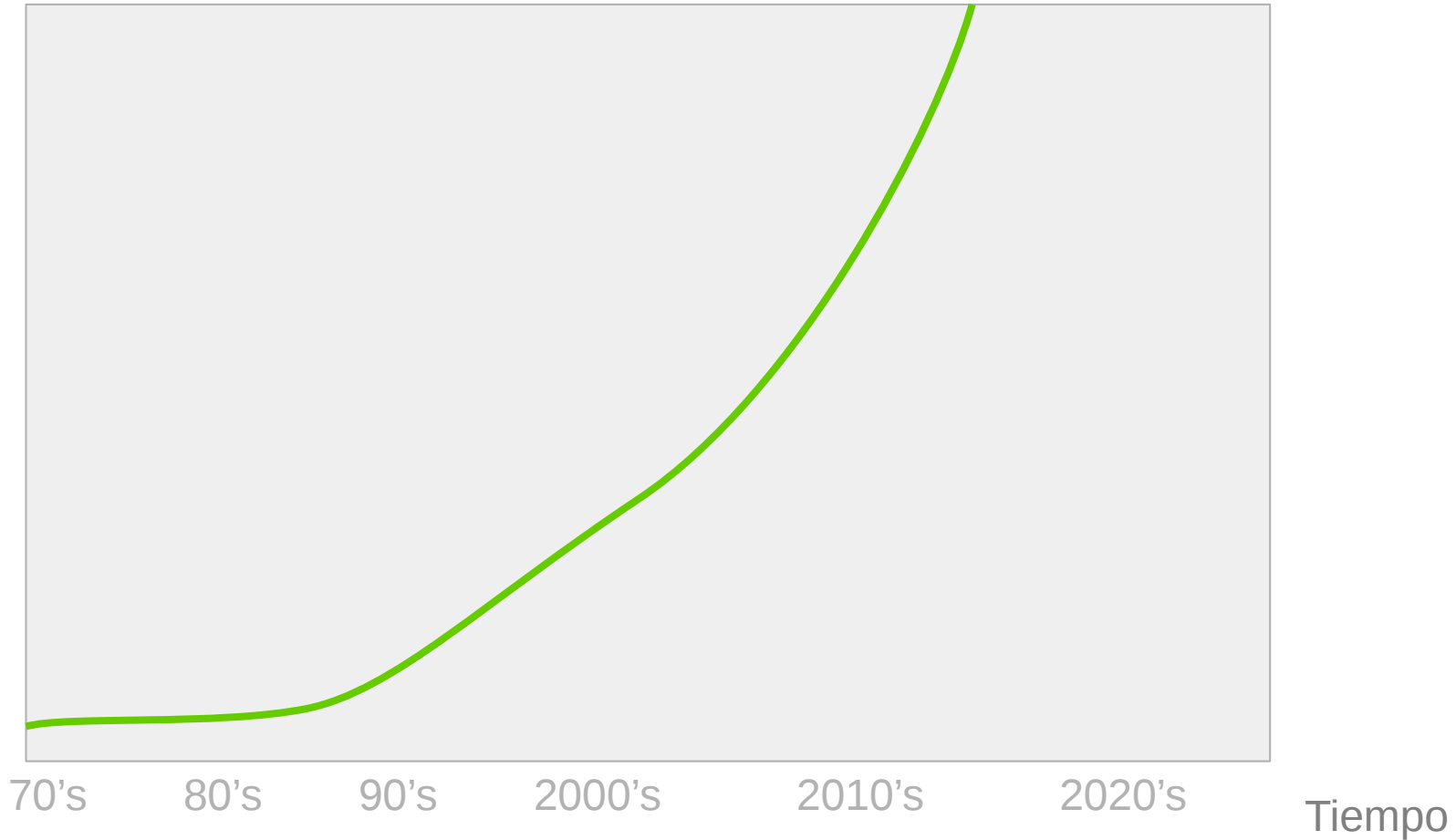
Recursos
Ram, Espacio



PGC

Para qué?

Expectativa
Sorpresa



PGC

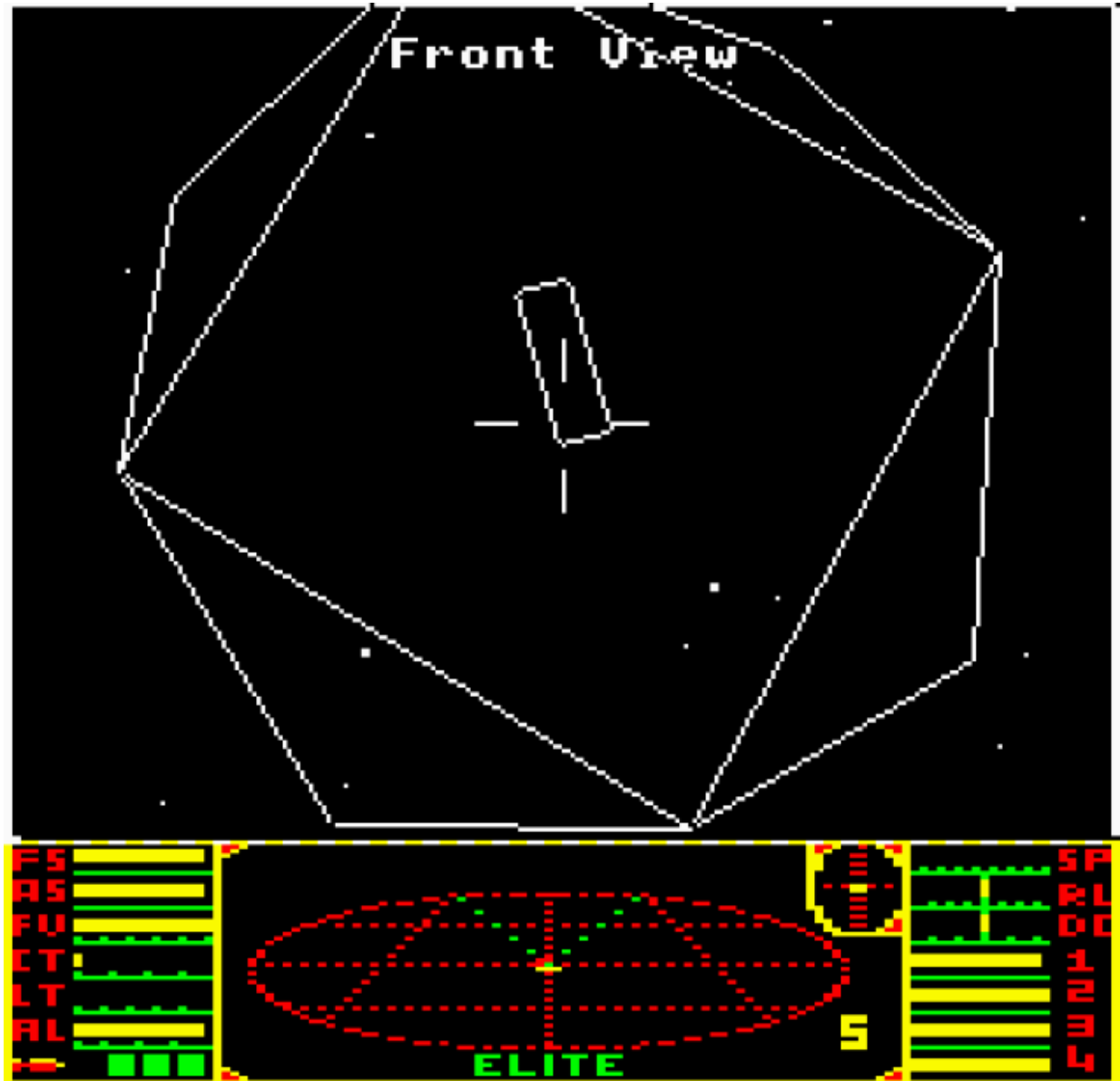
Quien?

- Juegos
- Cine/Video
- Arte
- Musica
- Mensajes

Quien?

- Simulaciones : flujos
Tráfico, caudal,
urbanización, distribución.
- Decision Support:
- AI
- VR, AR

Quien?



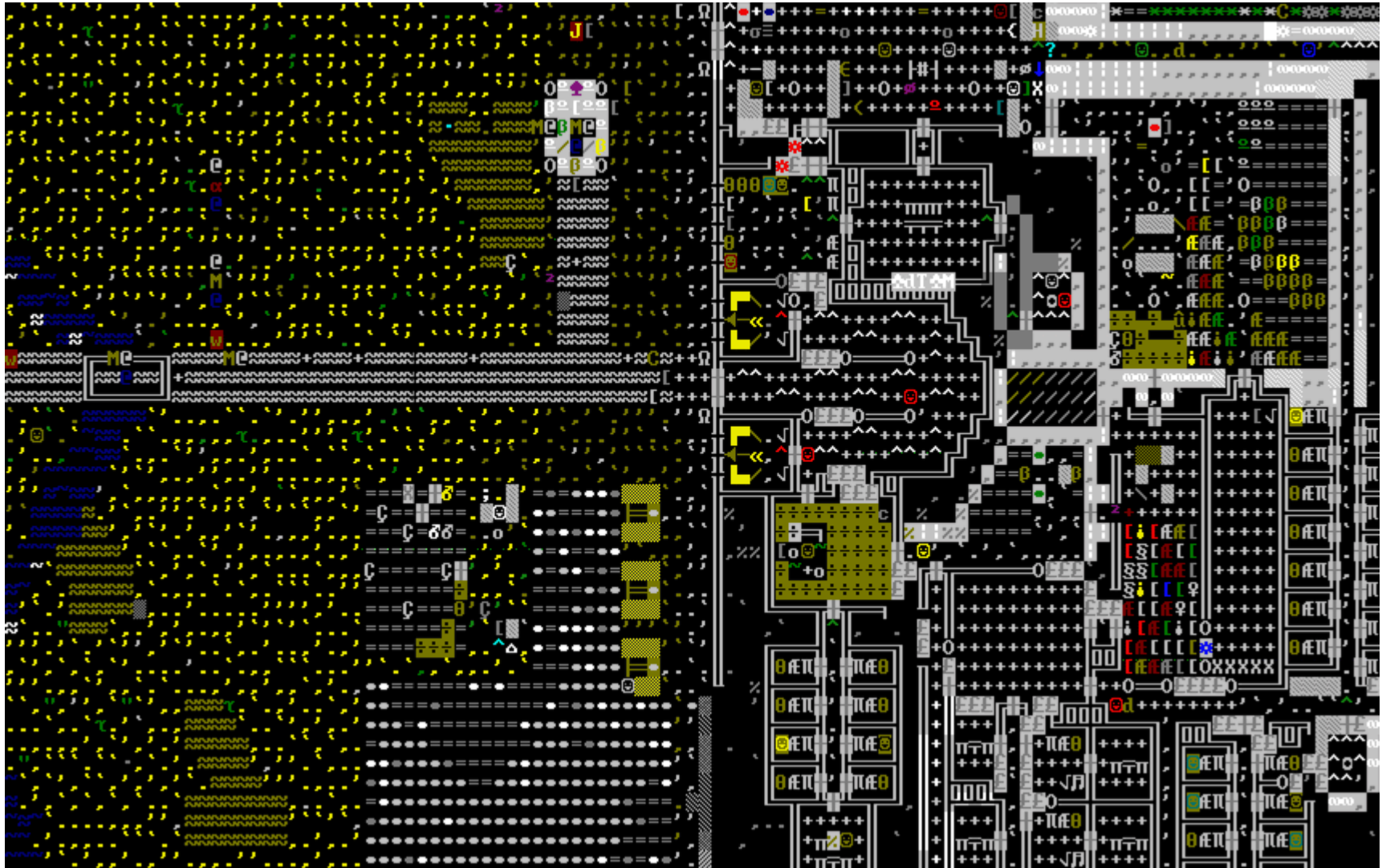
PGC

Quien?



PGC

Quien?



PGC

Quien?



PGC

Quien?



PGC

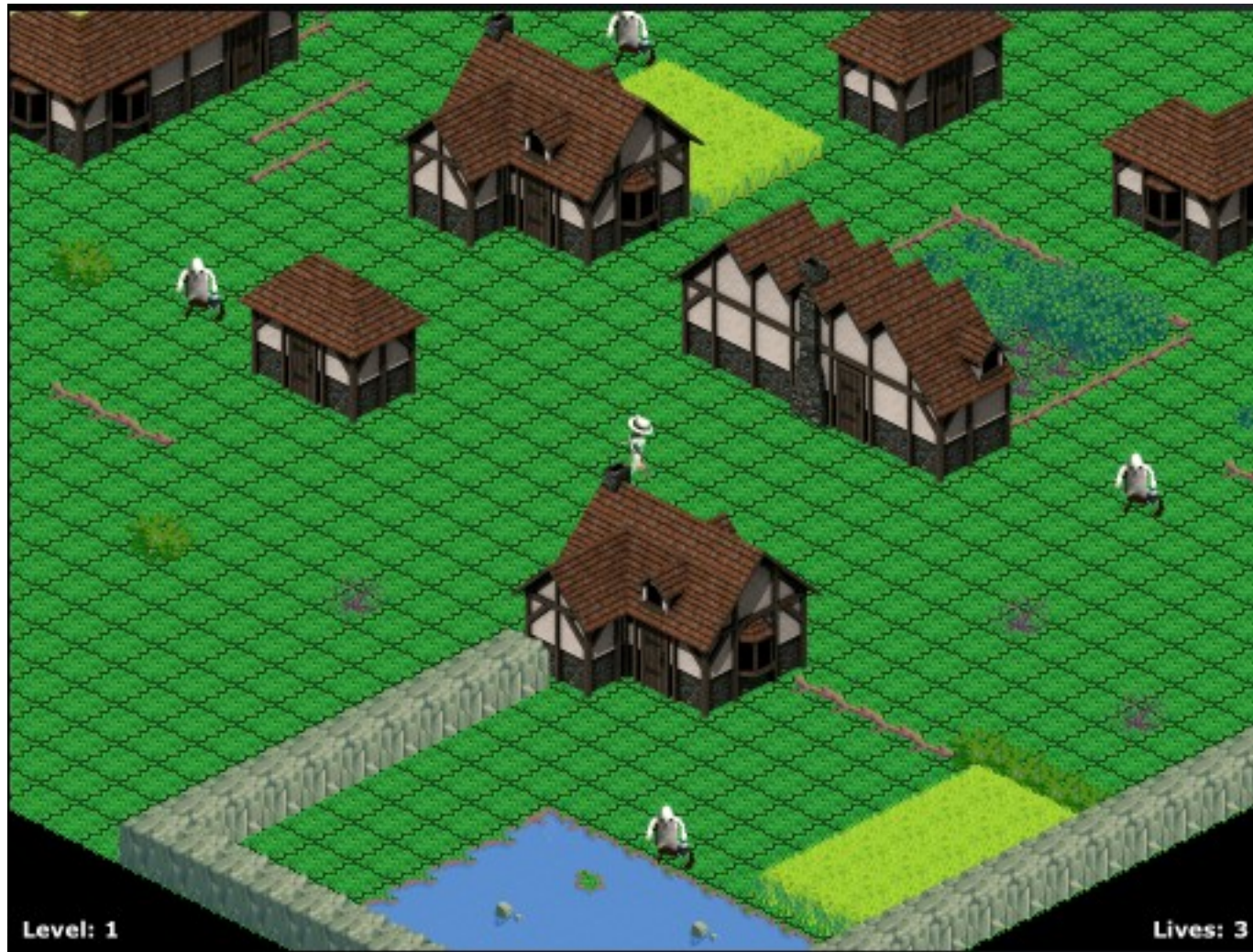
Quien?



Cómo : Aditivas

- Tiles
- Grammars
- Distribution
- Parametric
- Interpretive
- Simulations

Cómo : Tiles



Cómo : Grammars

My Grammar

login scifi ▼ json ▼

```
{
  "firstSyl":["B","C","D","F","G","Z","St","Fl","Bl",
  "Pr","Kr","Li","Chr","Sk","Br","Sth","","H","J","K",
  "L","M","N","P","Qu","R","S","T","V","W","X","Y",
  "Z","","Ch","Dhr","Dr","Sl","Sc","Sh","Thl","Thr",
  "Pl","Fr","Phr","Phl","Wh"],
  "middleSyl":["an","all","ar","art","air","aeon","af",
  "av","ant","app","ab","er","en","eor","eon","ent",
  "enth","irt","ian","ion","iont","ill","il","ipp","in","is",
  "it","ik","ob","ov","orb","oon","ion","uk","uf","un",
  "ull","urk","estr","antr","okl","ackl"],
  "lastSyl":["a","ia","ea","u","y","en","am","is","on",
  "an","o","io","i","el","ios","ax","ox","ix","ex","izz",
  "ius","ian","ean","ekang","anth"],
  "butchName":["Chesty","Manley","Brock","Stone",
  "Brick","Butch","Bruce","Steel","Saber","Tex",
  "Rock","Drake","Ace","Knute","Wolf","Thorax",
  "Brad","Abs","Burt","Slate","Bret","Duke"],
  "alienName":["#firstSyl##middleSyl##lastSyl#",
  "#firstSyl##lastSyl#", "#firstSyl##lastSyl#-
  #firstSyl##lastSyl#"],
  "physicsParticle":["quark","photon","lepton",
  "muon"],
  "scienceVerb":["evaporate","decay","phase-shift",
  "teleport","destabilize","sublimate"],
  "scienceBlargleStart":["holo","hyper","transmuto",
  "digi","nano","electro","transma","magna"],
  "communicationDevice":["#physicsParticle#-
  transmitter","#scienceBlargleStart#phone",
  "#scienceBlargleStart#pager"]
}
```

expansion ▼ the blink of an eye' yelled Senator Shon. 'Yeah, I'll have it done in the time it takes a single unstable muon to evaporate, Steel promised, hanging up the holophone. Steel shrugged with relief. The only thing that could really, I mean really, inspire a prismapoem would be the famously boyish spies of Planet Skionam and for that, he'd have to go to the Jantru system. Steel punched 'Jantru' into the holopager. There was still one ticket left on the hypertram, but he'd have to take a nonstop digi jet the rest of the way to Planet Skionam.'

origin

plot

artPlot

artDemand

shortTime

communicationDevice

artQuest

mcArt

shortTime

mcBoss

prismapoem

by

the blink of an eye

yelled

Senator Shon

.

Yeah, I'll have it done in

physicsParticle

scienceVerb

mc

the time it takes a single unstable

muon

to

evaporate

:

Steel

promised, hanging up the

scienceBlargleStart

mcResponded

conversationally

holo

phone

.

Steel

shrugged

with relief

.

mcArt

sexy

occupation

origin ▼ 1 ▼ reroll step 69864905

'I need that prismapoem by the blink of an eye' yelled Senator Shon. 'Yeah, I'll have it done in the time it takes a single unstable muon to evaporate', Steel promised, hanging up the holophone. Steel shrugged with relief. The only thing that could really, I mean really, inspire a prismapoem would be the famously boyish spies of Planet Skionam and for that, he'd have to go to the Jantru system. Steel punched 'Jantru' into the holopager. There was still one ticket left on the hypertram, but he'd have to take a nonstop digi jet the rest of the way to Planet Skionam.'

Cómo : Grammars / L-Systems

L-system - Wikipedia

https://en.wikipedia.org/wiki/L-system 1.41s

Practical Procedural Gene Procedural generation - V Procedural programming So you want to build a g L-system - Wikipedia

This example yields the same result (in terms of the length of each string, not the sequence of As and Bs) if the rule ($A \rightarrow AB$) is replaced with ($A \rightarrow BA$), except that the strings are mirrored.

This sequence is a [locally catenative sequence](#) because $G(n) = G(n-1)G(n-2)$, where $G(n)$ is the n -th generation.

Example 2: Fractal (binary) tree [\[edit\]](#)

- **variables** : 0, 1
- **constants** : [,]
- **axiom** : 0
- **rules** : ($1 \rightarrow 11$), ($0 \rightarrow 1[0]0$)

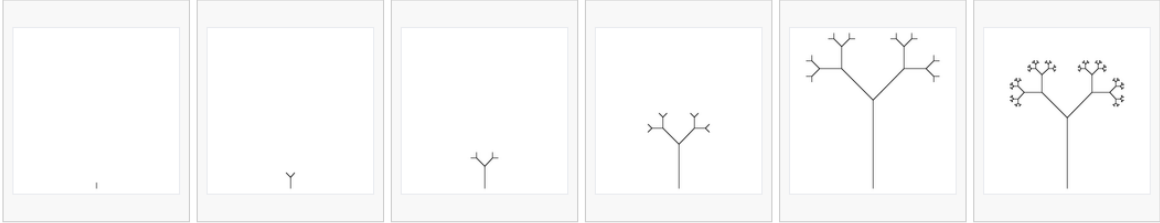
The shape is built by [recursively](#) feeding the axiom through the production rules. Each character of the input string is checked against the rule list to determine which character or string to replace it with in the output string. In this example, a '1' in the input string becomes '11' in the output string, while '[' remains the same. Applying this to the axiom of '0', we get:

axiom: 0
1st recursion: 1[0]0
2nd recursion: 11[1[0]0]1[0]0
3rd recursion: 1111[11[1[0]0]1[0]0]11[1[0]0]1[0]0
...

We can see that this string quickly grows in size and complexity. This string can be drawn as an image by using [turtle graphics](#), where each symbol is assigned a graphical operation for the turtle to perform. For example, in the sample above, the turtle may be given the following instructions:

- 0: draw a [line segment](#) ending in a leaf
- 1: draw a line segment
- [: push position and angle, turn left 45 degrees
-]: pop position and angle, turn right 45 degrees

The push and pop refer to a [LIFO stack](#) (more technical grammar would have separate symbols for "push position" and "turn left"). When the turtle interpretation encounters a '[', the current position and angle are saved, and are then restored when the interpretation encounters a ']'. If multiple values have been "pushed," then a "pop" restores the most recently saved values. Applying the graphical rules listed above to the earlier recursion, we get:

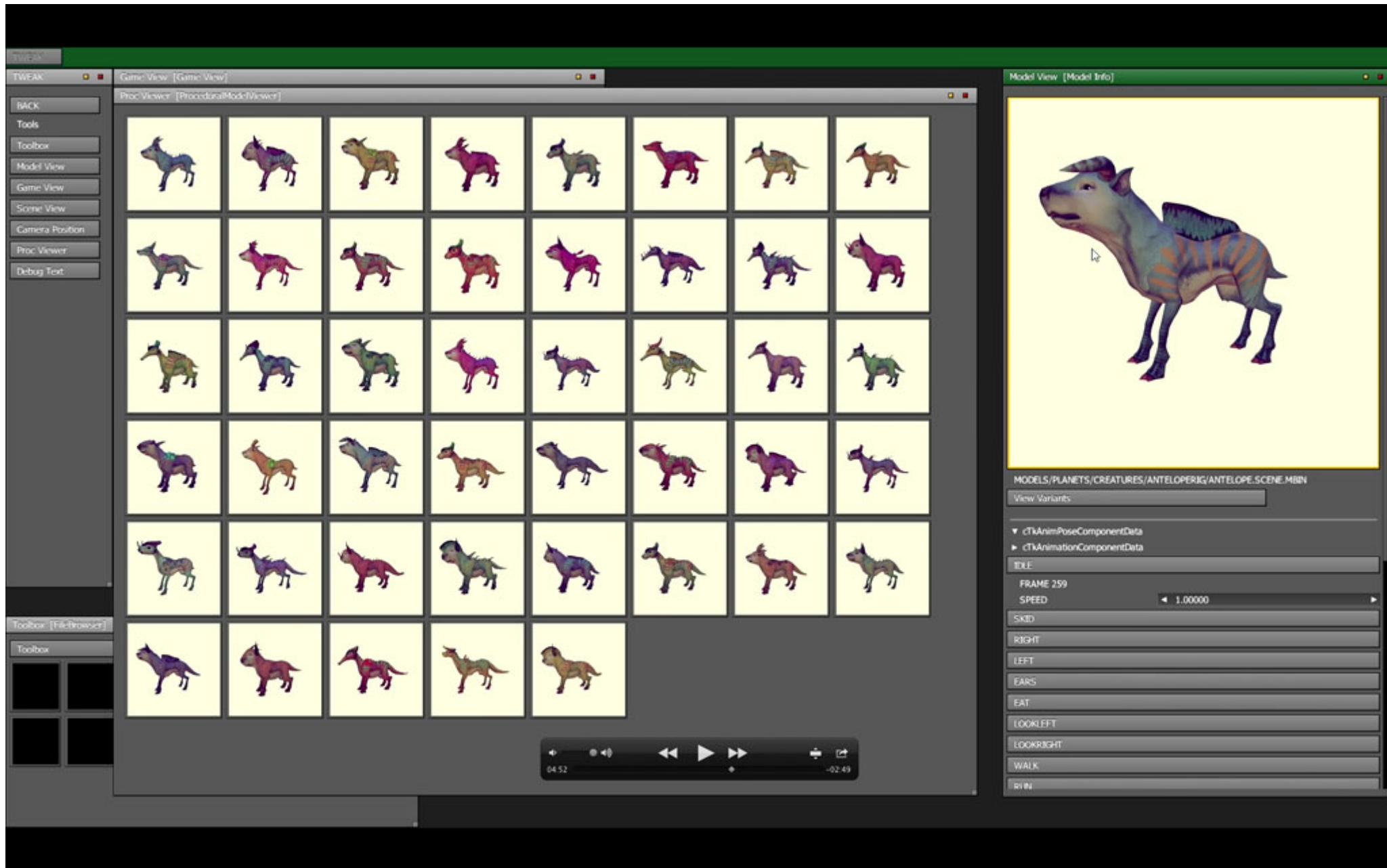


Axiom First recursion Second recursion Third recursion Fourth recursion Seventh recursion, scaled down ten times

Example 3: Cantor set [\[edit\]](#)

variables : A B

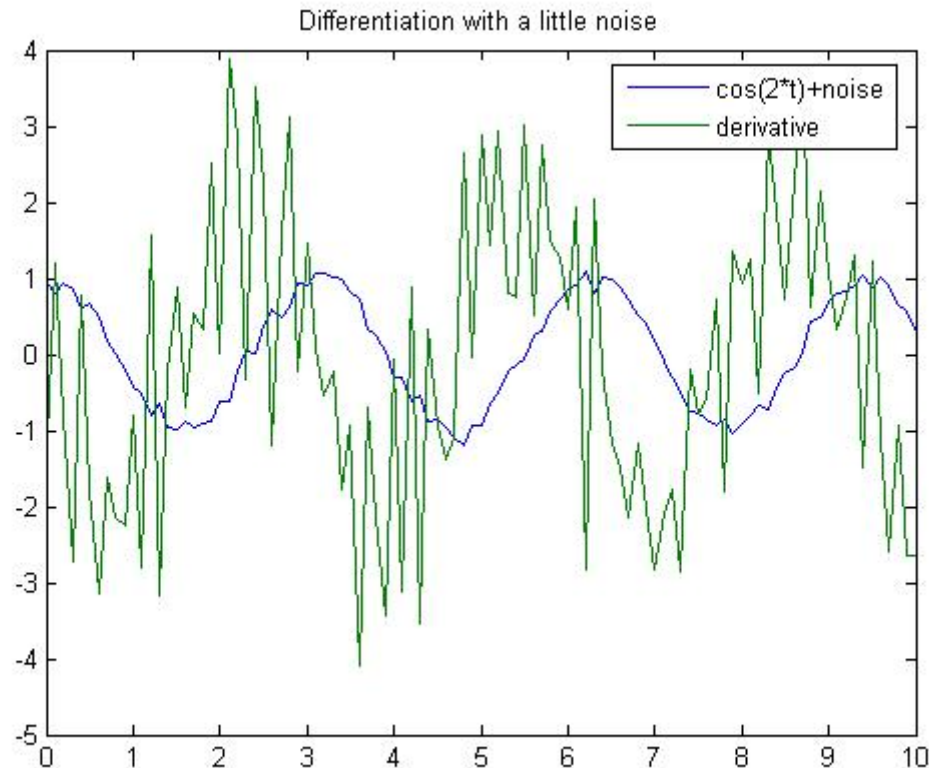
Cómo : Parametricos



Cómo : Interpretive

Agregar complejidad a algo existente.

AR, Kinect, Noise (Perlin), Voronoi,



Cómo : Simulaciones

- Trails: simular trayectorias que reponden a fuerzas.
- Dibujo
- Extrusiones, morphs.
- Cellular automata

Cómo : Ramdoms & Misc

- Aleatorios
- 10.000 bowls de avena
- Guardar aleatorios (No Man's Sky)
- Ownership (No man's Sky) – apropiar contenido y compartir, curar.



Donde : Recursos

- pcgbook.com
- The Nature of Code / Coding Train
- Tracery.io
- Mike Cook
- Ken Perlin
- Kate Compton
- <http://galaxykate0.tumblr.com/post/139774965871/so-you-want-to-build-a-generator>
- P5js.org
- Processing.org
- Unity,

Gracias,

PGC