# Analisi di Substrate

Roberto Ranon
http://users.dimi.uniud.it/~roberto.ranon/
roberto.ranon@uniud.it

- esamineremo un’opera famosa di arte generativa
- impareremo, da un esempio concreto:
  - come da semplici regole è possibile creare un’opera complessa
  - il concetto di casualità negli algoritmi e alcuni possibili utilizzi nell’arte generativa
  - qualcosa in più sulla programmazione in Processing

# Substrate

## Jared Tarbell, 2003

http://www.complexification.net/gallery/machines/substrate/

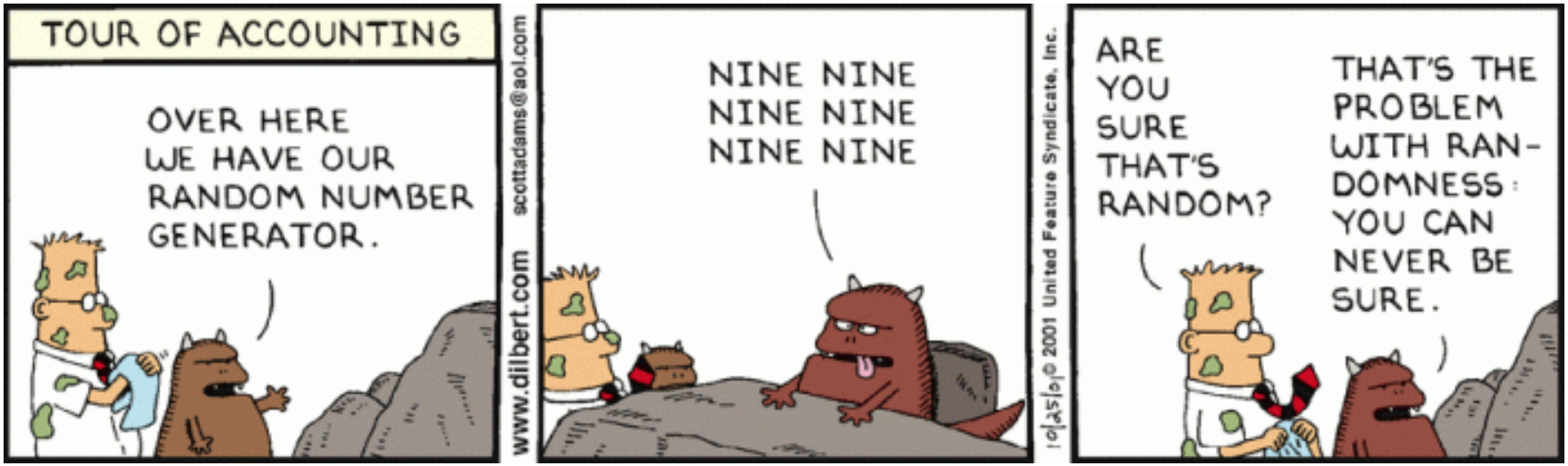
# Jared Tarbell

- artista generativo molto noto
  - infinite.center
  - complexification.net
  - levitated.net
  - etsy.com (cofondatore)

“A single line (known internally as a “crack”) begins drawing itself from some *random* point in some *random* direction. The line continues to draw itself until it either (a) hits the edge of the screen or (b) hits another line, at which point it stops and two more (*random*) lines begin. The one simple rule used in the creation of new lines is that they begin at tangents to existing lines. This process is repeated until there are too many lines to keep track of or the program is stopped.”

*–Jared Tarbell*

# Computers & Casualità



# Computers & Casualità

- un computer non si comporta mai casualmente
- in alcune situazioni, però, la possibilità di generare numeri casuali è fondamentale
  - generare chiavi per la crittografia, simulazione, …
- può un computer generare un numero in modo veramente casuale?
- *Pseudo-Random Number Generators* e *True Random Number Generators*

*Lavarand was a hardware random number generator designed by Silicon Graphics that worked by taking pictures of the patterns made by the floating material in lava lamps, extracting random data from the pictures, and using the result to seed a pseudorandom number generator*

**https://en.wikipedia.org/wiki/Lavarand**

# Random in Processing

- il comando **random** ritorna un numero reale pseudo-casuale
  - **random()**
  - **random(high)**, es. **random(5)**
  - **random(low,high)**, es. **random(1,5)**
  - **randomSeed**, **noise**, **noiseSeed**

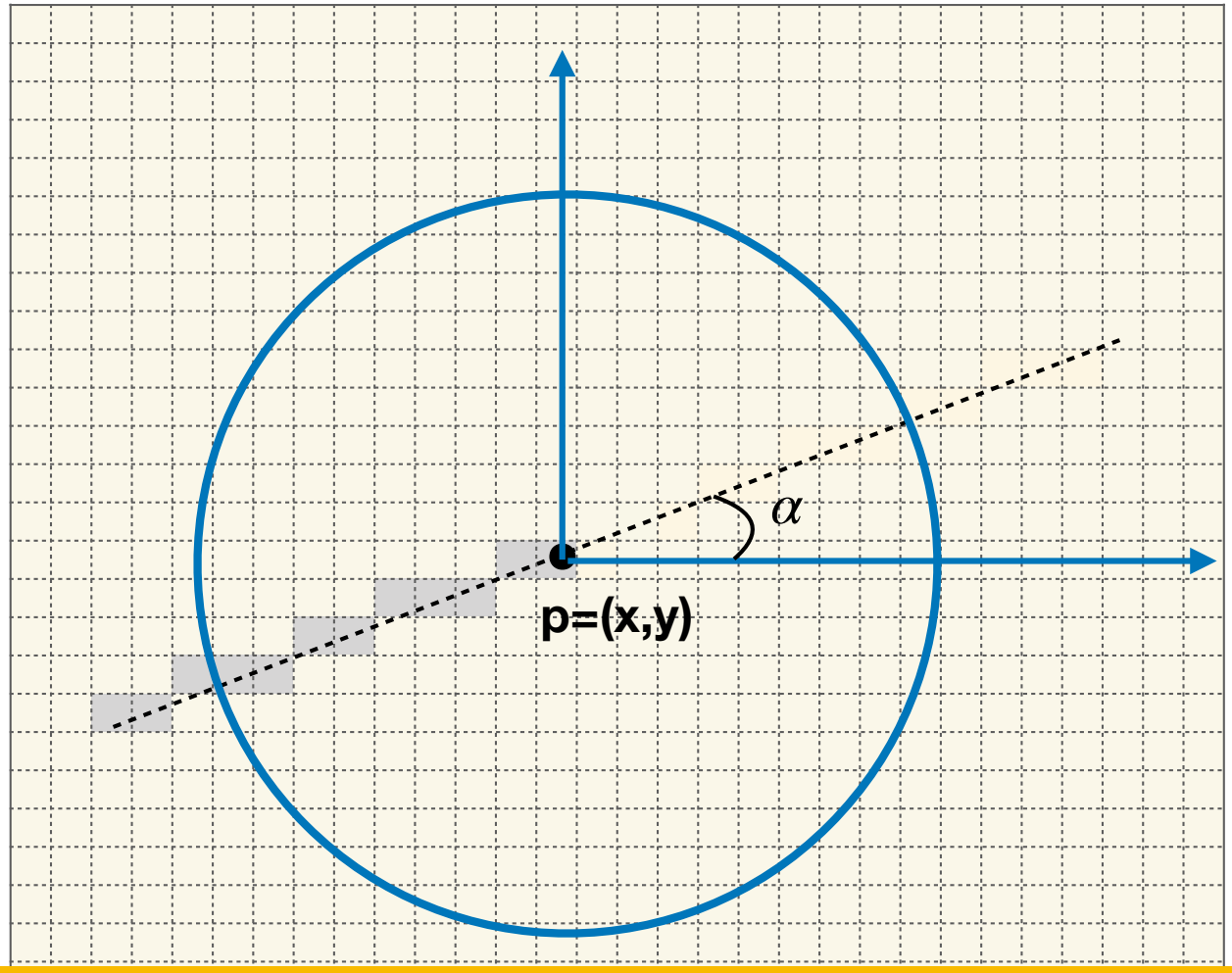
# Disegno di una linea (requisiti)

- ogni linea viene disegnata un punto per frame
- devo poter controllare se una linea incontra il bordo o altre linee
    - una linea continua a crescere solo se non incontra il bordo o altre linee
- le tre linee iniziali vengono generate a caso
- le altre devono originare da un punto a caso su una delle linee esistenti, e crescere perpendicolarmente rispetto alla linea da cui partono

# Disegno di una linea

- ogni linea viene disegnata un punto per frame
- devo poter controllare se una linea incontra il bordo o altre linee
  - una linea continua a crescere solo se non incontra il bordo o altre linee
- le tre linee iniziali vengono generate a caso
- le a delle line esta

**Per rappresentare una linea mi bastano:**
**- l'ultimo punto disegnato, in 2D**
**- la direzione, ad esempio come un angolo**




Qual è il punto successivo a p, noto l'angolo α?
Se s è la distanza che devono avere i punti tra loro, allora sarà ( x + s*cos(α), y + s*sin(α) )

# Disegno di una linea (requisiti)

- ogni linea viene disegnata un punto per frame
- devo poter controllare se una linea incontra il bordo o altre linee
  - una linea continua a crescere solo se non incontra il
- le tre linee iniziali vengono generate a caso
- le altre devono nascere da un punto a caso su una delle linee esistenti, e crescere perpendicolarmente a questa

**Potrei fare questi ragionamenti sull'immagine creata finora, ma sarebbero difficili da fare**

**in questi casi, meglio creare una *struttura dati* adatta ai nostri ragionamenti, nel nostro caso una tabella con le dimensioni dello schermo**

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 30 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 30 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 30 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 30 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |
| 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 | 1001 |

- il valore 30 in una cella indica che:
  - la cella è occupata da una linea
  - la linea in quel punto ha un angolo di 30 gradi
- 10001 è un valore speciale che indica una cella vuota

# in questi casi, meglio creare una *struttura dati* adatta ai nostri ragionamenti, nel nostro caso una tabella con le dimensioni dello schermo


- trovo il valore 60 nel punto in cui vorrei prolungare la linea rossa -> ho trovato un'altra linea
- devo ora creare due nuove linee, ognuna con partenza un punto a caso su un'altra linea, e angolo perpendicolare alla linea
- per ogni nuova linea, basta scegliere una cella a caso con valore < 1001, e sommare al valore della cella l'angolo + 90 (o - 90)

# Esame del codice

substrate_nosandpainting_simpler.pde

substrate_nosandpainting_simpler

```
// Substrate Watercolor
// j.tarbell   June, 2004
// Albuquerque, New Mexico
// complexification.net

// Processing 0085 Beta syntax update
// j.tarbell   April, 2005

int dimx = 900;
int dimy = 900;
int num = 0;        // current number of cracks
int maxnum = 200;   // max number of cracks that

// grid of cracks
int[][] cgrid;
Crack[] cracks;

boolean looping = true;

// MAIN METHODS ----------------------------------

void setup() {
  size(900,900,P3D);
  background(255);

  cgrid = new int[dimx][dimy];
  cracks = new Crack[maxnum];

  begin();
}

void draw() {
  // crack all cracks
  for (int n=0;n<num;n++) {
    cracks[n].move();
  }
}
```

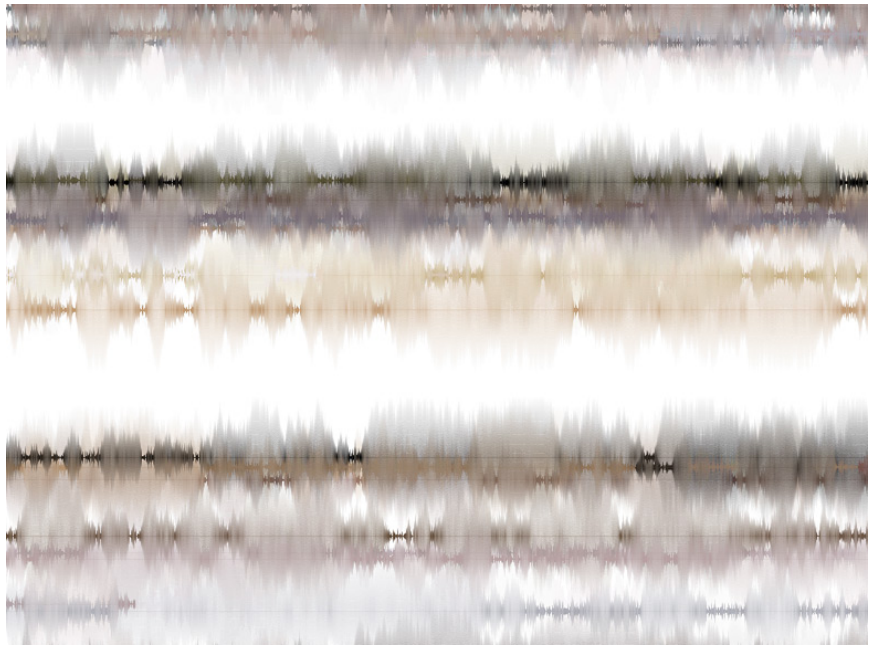
# Casualità come imprecisione

**substrate_nosandpainting.pde**

- lo sketch originale usa il comando **random** per introdurre un po' di imprecisione nel disegno al fine di imitare lo stile di un "disegno a mano"
  - l'angolo di +/- 90 gradi viene alterato di una piccola quantità casuale
  - ogni nuovo punto viene disegnato con un piccolo spostamento casuale
  - il controllo di "collisione" con altre linee viene fatto usando un punto con un piccolo spostamento casuale

# Colori

- lo sketch originale colora lo spazio da una parte di ogni linea mentre la disegna, utilizzando una tecnica nota come *sand painting*
  - scegliamo un colore a caso (da una palette)
  - depositiamo dei granelli di sabbia (disegniamo dei pixel trasparenti) lungo lo spazio vuoto da un lato della linea
  - la posizione dei pixel viene alterata tramite un piccolo valore casuale



**Sand Stroke**
Jared Tarbell, 2004

# Variazioni

- nello sketch originale la palette di colori è presa da un'immagine: cambiate l'immagine con una a vostra scelta … (nello sketch originale)
- lo sketch semplificato si presta meglio a sperimentare variazioni del comportamento dell'algoritmo
  - angolo delle nuove linee
  - dimensione dei punti
  - …

# Alcune risorse

- cercate su twitter / instagram artisti generativi, ad es:
  - https://twitter.com/mattdesl
  - https://twitter.com/inconvergent
  - https://twitter.com/manoloidee
  - https://twitter.com/JoshuaDavis
- alcuni raccontano in dettaglio le idee e tecniche alla base delle loro opere, ad es. https://inconvergent.net/#writing