Baptiste Ory (IMAC 1)

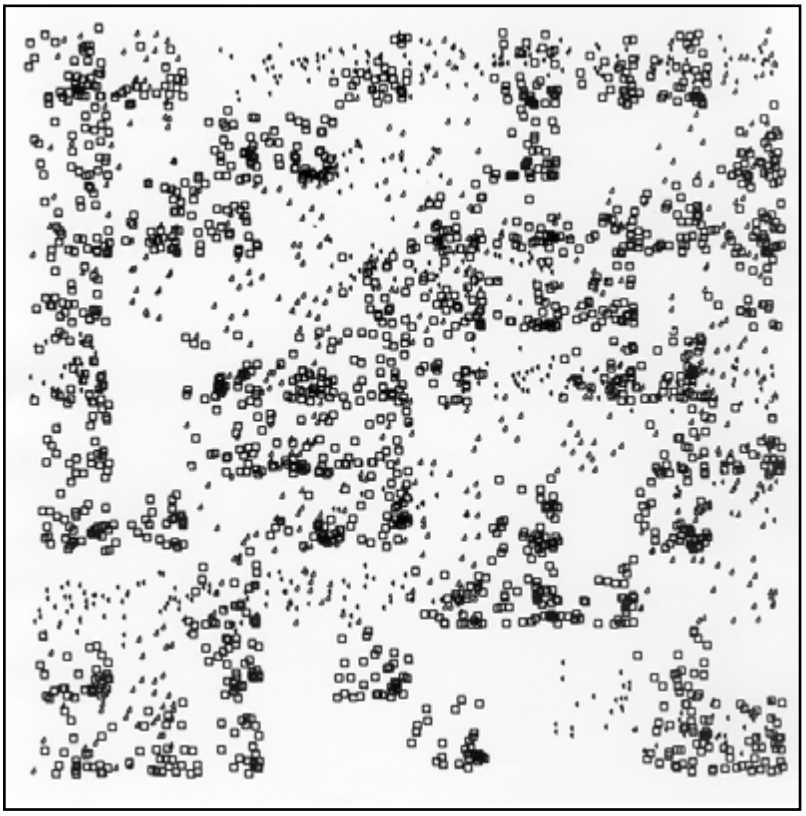
Varying degrees of distribution of elementary signs

/based on the work of **Frieder Nake**

Esthétique algorithmique
/Gaëtan Robillard

/Summary

/The artwork of Nake



Artist	Frieder Nake
Title	“Distribution of elementary signs”, Nr.5
Date	13/09/1965
Type	Bicolored drawing
Material	China ink on paper
Technique	Drawing computer-generated Hardware: ZUSE Graphomat Z64 Program: COMPART ER 56
Dimensions	50×50 cm

First, this work of Frieder Nake is in black and white : **black geometric shapes are drawn on a square white background**. And it seems that these shapes are not distributed according to all the square of the work, but rather **according smaller squares which crisscross the work** : 10 squares of width and 10 squares of height, for a total of 100 squares.

Then, I distinguish **3 sorts of geometrical forms** : empty squares, empty right-angled triangles and small full isosceles triangles. These “signs” are **randomly distributed in the 100 squares**. There may be several types of signs in a square, but also none. A sign that exists in a square has almost always the same number (about thirty). I can therefore assume that the presence of a form in a square is also chosen randomly : **if the type of sign must be present in a square, about thirty signs of this type are distributed randomly**.

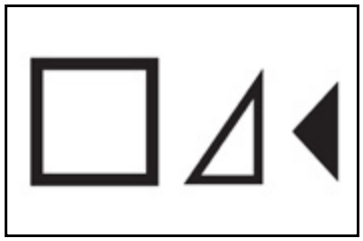
Finally, **the squares are the most common forms**, followed by the right-angled triangles, and finally the isosceles triangles. Even though forms are randomly distributed in the squares that crisscross the work, **empty square forms tend to be more numerous in the lower right corner** of these bigger squares. In addition, it seems that there is a **white border** surrounding the squares that crisscross the work.

/The algorithm

/Explanations

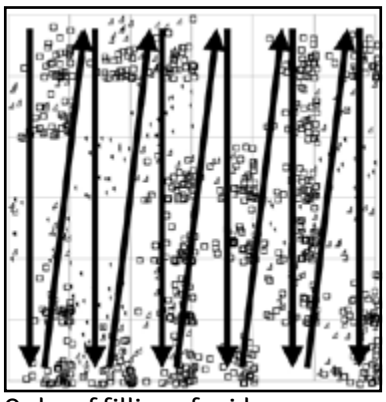
My **algorithm** that I named "Varying degrees of distribution of elementary forms (vddes)" works like this:

There are **three types of forms**: empty squares (form 1), empty right-angled triangles (form 2) and small full isosceles triangles (form 3).



The three forms

For **each square of a grid** (going from top to bottom of the drawing), **the presence of a sort of form is random**: "present" or "not present". But **the form 1 is more likely present in grid squares** than the form 2, than the form 3. Several types of forms can be in a grid square.



Order of filling of grid squares

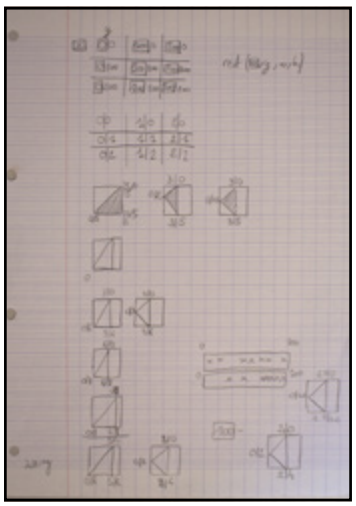
If a type of form can be present in a grid square, **the form will be drawn several times and every time randomly in this grid square**. But there will be more form 1 than form 2, than form 3. In addition, **the form 1 will be more randomly distributed in the lower right corner** of the grid square.

Example for drawing the form 1 in a grid square below.

Knowing that "rect(x-coordinate of the rectangle, y-coordinate of the rectangle, width of the rectangle, height of the rectangle)" :

*"rect((grid square width*number of this grid square on the horizontal axis)+random value smaller than the width of a grid square, (grid square height*number of this grid square on the vertical axis)+random value smaller than the height of a grid square, grid square width divided by 10, grid square width divided by 10)"*

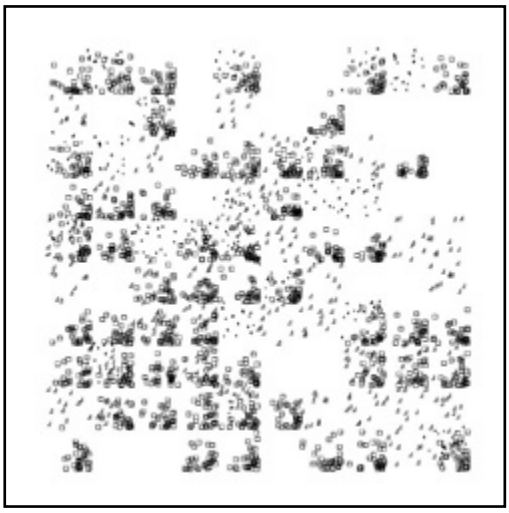
All the stroke weights of the forms and the sizes of the forms and grid squares are **adapted to the size of the window**. In addition, for interactivity, we can **display the grid or not** ('G' key), put the window in **full screen** ('F' key) and **change the number of squares in the grid** (left click and right click).



Draft searches

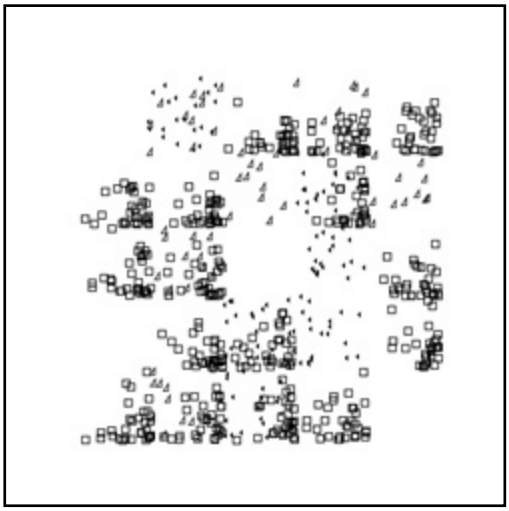
/Pseudo-algorithm

```
/* Number of grid squares (horizontal and vertical) | by default */
squaresGrid <= 12
/* Visible grid | by default */
grid <= false
/* Fullscreen | by default */
fullscreen <= false


Function setup //starts at the beginning
 canva size : 600 by 600 pixels
 resize the window : allow
 draw once
End Function


Function draw //starts after function setup
 /* Size of the grid squares */
 widthBox <= width of the canva/squaresGrid;
 heightBox <= height of the canva/squaresGrid;

 canva background : white

 /* Informations for the user */
 draw rectangle : x-axis 0 | y-axis 0 | width 340 | height 25 | color black
 draw text : size 12 | text "Left click : + / Right click : - / G : grid / F : fullscreen" | x-axis 10 | y-axis 4 | width text box 340 | height text box
25

 /* Grid squares on the horizontal axis */
 For i = 1 to squaresGrid-2  //Leave a white frame around
  /*Grid squares on the vertical axis */
   For j = 1 to squaresGrid-2 //Leave a white frame around

    /* Drawing grid */
    If grid = true
      draw rectangle : x-axis i*widthBox | y-axis j*heightBox | width widthBox | height heightBox | border color grey | border size
widthBox/100
    End If

    /* Drawing empty squares */
    r0 <= random value from 0 to 2
    If r0 = 1 //Random presence of the shape in the grid square (yes or no)
     For k = 0 to k = 30 //Number of shapes in the grid square
      r1 <= widthBox-(random value from 0 to widthbox*random value from 0 to 1) //Random position on the horizontal axis of the grid
square | more right
      r2 <= heightBox-(random value from 0 to widthbox*random value from 0 to 1) //Random position on the vertical axis of the grid
square | more down
      border color of the following form : black
      If squaresGrid = 3 or squaresGrid = 32
       border color of the following form : random  //Random color if minimum or maximum squares of the grid reached
      End If
      draw rect : x-axis (widthBox*i)+r1 | y-axis (heightBox*j)+r2 | width widthBox/10 | height widthBox/10 | border size widthBox/60 //
Drawing the shape according to the position in the grid square and the desired dimensions
     End For
    End If

    /* Drawing empty right-angled triangles */
    r0 <= random value from 0 to 3
    If r0=1 //Random presence of the shape in the grid square (yes or no)
     For k = 0 to k = 20 //Number of shapes in the grid square
      r1 <= random number from 0 to widthbox //Random position on the horizontal axis of the grid square
      r2 <= random number from 0 to heighbox //Random position on the vertical axis of the grid square
      border color of the following form : black
      If squaresGrid = 3 or squaresGrid = 32
       border color of the following form : random  //Random color if minimum or maximum squares of the grid reached
      End If
      draw triangle : x coordinate of the first point ((widthBox*i)+r1)+0 | y coordinate of the first point ((heightBox*j)+r2)+widthBox/10
| x coordinate of the second point ((widthBox*i)+r1)+widthBox/15 | y coordinate of the second point ((heightBox*j)+r2)+0 | x coordinate
of the third point ((widthBox*i)+r1)+widthBox/15 | x coordinate of the third point ((heightBox*j)+r2)+widthBox/10 | border size
widthBox/70 //Drawing the shape according to the position in the grid square and the desired dimensions
```

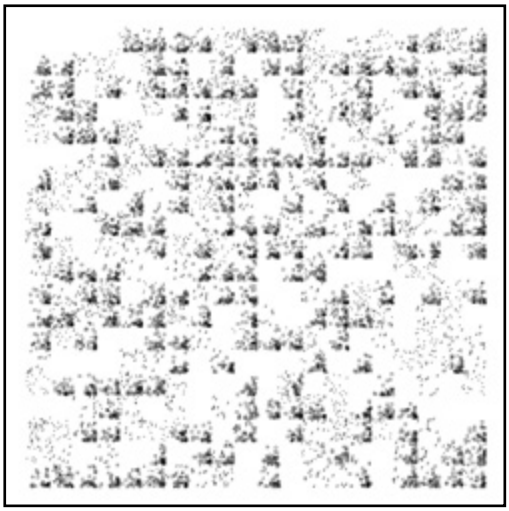


Visual of a version of the algorithm in vector

```
      End For
    End If

    /* Drawing full triangles */
    r0 <= random value from 0 to 4
    If r0 = 1 //Random presence of the shape in the grid square (yes or no)
     For k = 0 to k = 20 //Number of shapes in the grid square
      r1 <= random number from 0 to widthbox //Random position on the horizontal axis of the grid square
      r2 <= random number from 0 to heightbox //Random position on the vertical axis of the grid square
      border color of the following form : black
      If squaresGrid = 3 or squaresGrid = 32
       border color of the following form : random  //Random color if minimum or maximum squares of the grid reached
      End If
      draw triangle : x coordinate of the first point ((widthBox*i)+r1)+0 | y coordinate of the first point ((heightBox*j)+r2)+widthBox/50 |
x coordinate of the second point ((widthBox*i)+r1)+widthBox/50 | y coordinate of the second point ((heightBox*j)+r2)+0 | x coordinate of
the third point ((widthBox*i)+r1)+widthBox/50 | x coordinate of the third point (heightBox*j)+r2)+widthBox/25 | border size widthBox/50
//Drawing the shape according to the position in the grid square and the desired dimensions
     End For
    End If

  End For
 End For
End Function


Function keyPressed //starts when the user presses a key
 /* Display or hide the grid if the user presses the «G» key */
 If key = G
  grid <= inverse of grid
  relaunch function draw
 End If

 /* Fullscreen or not if the user presses the «F» key */
 If key = F
  If fullscreen = true
   location of the canva : x-axis 10 | y-axis 10
   canva always on top : false
   canva size : 600 by 600 pixels
   cursor : hand
  Else If
   location of the canva : x-axis -8 | y-axis -31
   canva always on top : true
   canva size : screen width by screen height pixels
   cursor : hand
  End If
  fullscreen <= inverse of fullscreen
  relaunch function draw
 End If
End Function


Function mousePressed //starts when the user presses the mouse
 /* More grid squares if the user makes a right click */
 If mouse button = right click
  If squaresGrid < 32
   squaresGrid <= squaresGrid+1;
  End If
  relaunch function draw
 End If

 /* Less grid squares if the user makes a left click */
 If mouse button = left click
  If squaresGrid > 3
   squaresGrid <= squaresGrid-1;
  End If
  relaunch function draw
 End If
End Function
```



Visual of a version of the algorithm in vector



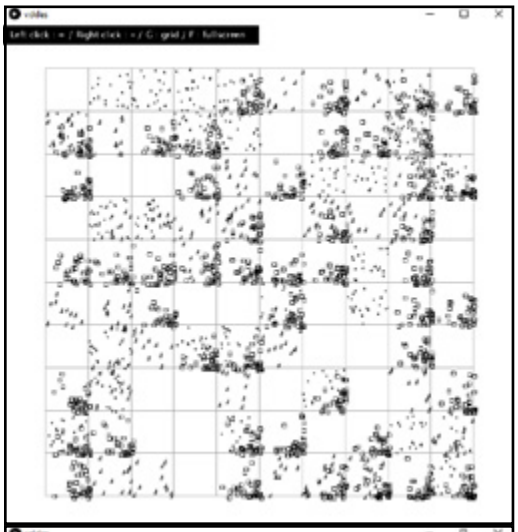
Visual of a version of the algorithm in vector

The algorithm was coded with **Processing 3**.

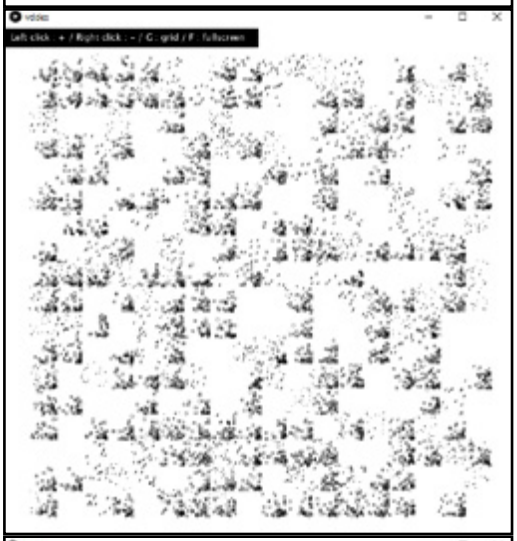
/Visuals

/Interface

10 grid squares



10 grid squares and grid displayed /user clicked ‘G’ key

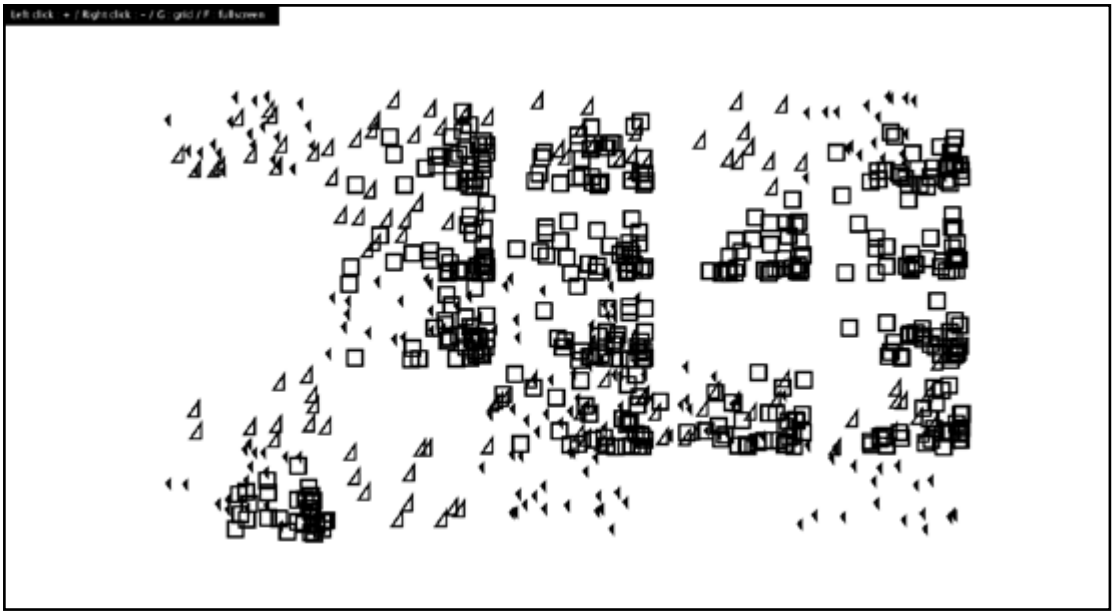


18 grid squares /multiple right clicks of the user

3 grid squares /multiple left clicks of the user

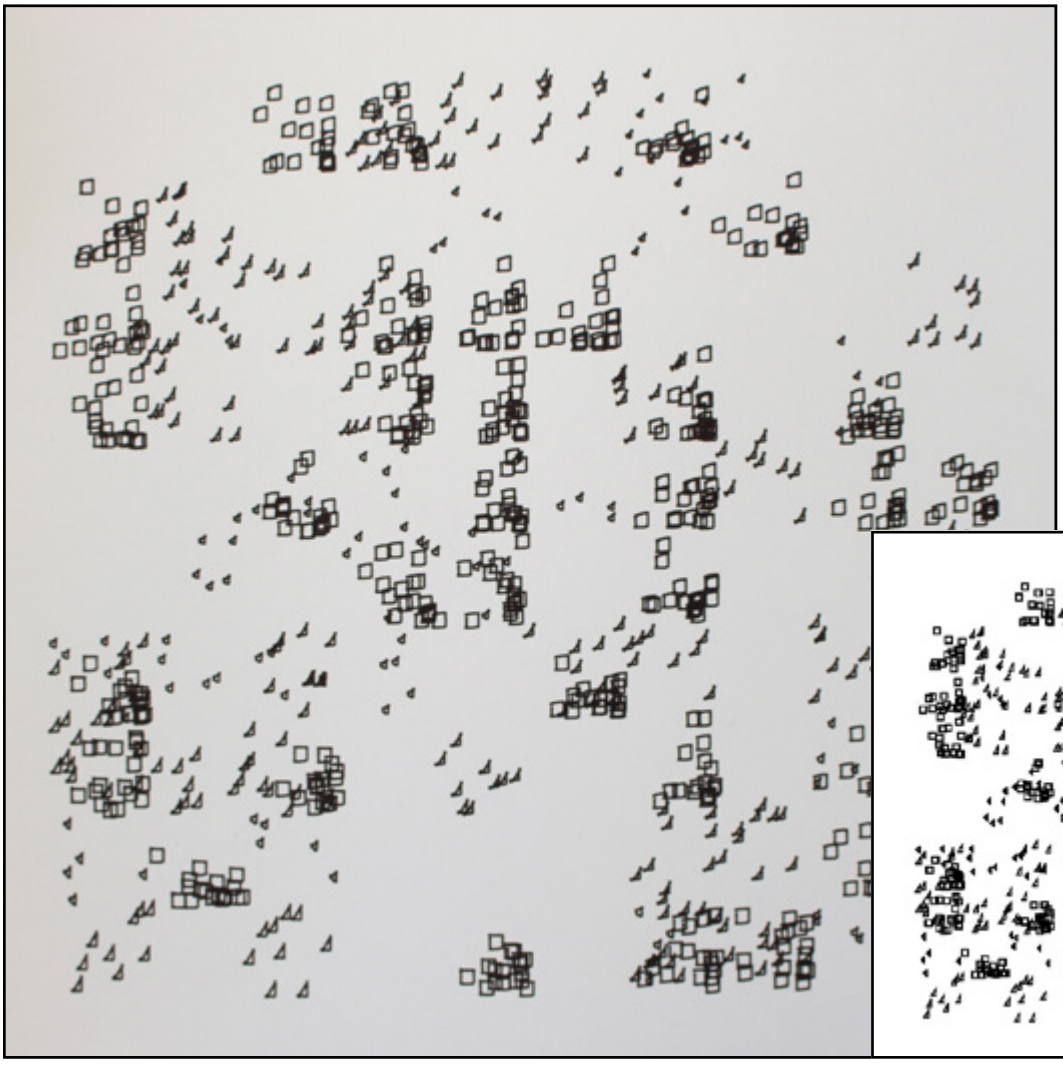
30 grid squares : maximum (all shapes then have a random color)

1 grid square : minimum (all shapes then have a random color)



Fullscreen: the window and the drawing have been resized /user clicked ‘F’ key

/Drawing



Drawing of a version of the algorithm with the machine “XY Plotter MakeBlock” / pen plotter experiment (see opposite) and visual of the same version in vector (see below)

/Sources

NAKE, Frieder. Felder mit Quadratverteilungen, überlagert von zwei Dreiecksverteilungen Nr. 5 (13/09/1965) : compArt [online]. Available on: http://dada.compart-bremen.de/item/artwork/1310

PROCESSING TEAM. Processing : Processing.org [online]. Available on: https://processing.org/

SPALTER, Anne ; MICHAEL, Spalter. 13/9/65 Nr. 5 : Spalter digital [online]. Available on: http://spalterdigital.com/artworks/13965-nr-5/

WEBSTER, Mark. Lecture Frieder Nake : Free Art Bureau [online]. 06/03/2013. Available on: http://freeartbureau.org/fab_activity/lecture-frieder-nake/