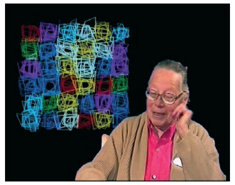
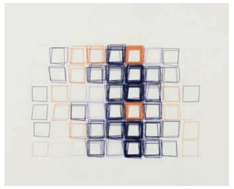
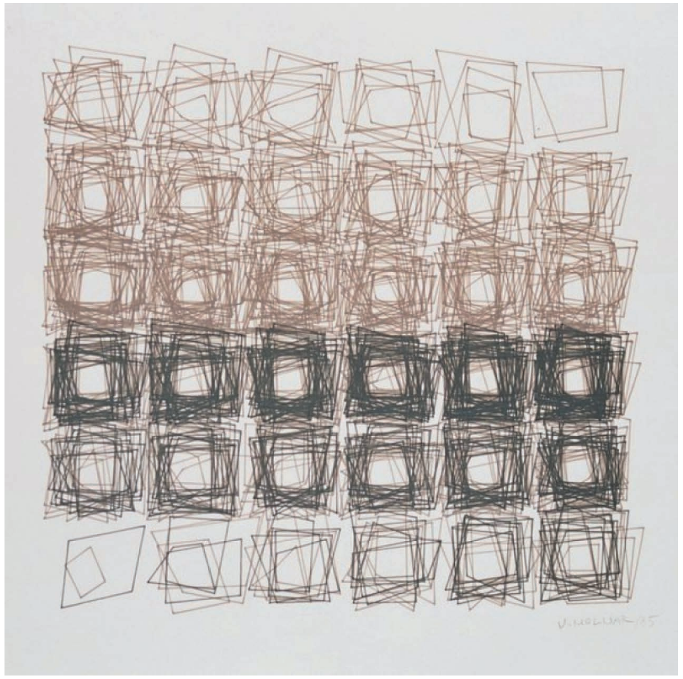
Recoding project algorithmic aesthetics

Tristan Debeaune & Wendy Gervais

Chosen artwork







Véra Molnar, *Structure de quadrilatère*, 1985

Ink on paper. 30 cm x 30 cm
Used technique : algorithm on computer + plotter

Vera Molnar is a pioneer in computer art. Her work is experimental, she tries and makes programs in order to experiment randomness with those shapes, oftenly squares she adores.

Analysis

The version we chose is made by a **squared light-grey canvas**, in which a **black and brown squared lined pattern**.

We see it as a **6 per 6 grid**. Each cell of the grid is scribbled with **quadrangles**. All cells are different, **randomly generated**, and they can overlap. The number of quadrangles drawn over and their colour are **variable according to the position in the grid, as following :**

- The 3 first rows of cells are drawn brown and the density of lines grows row by row, right to left, from top right (2 quadrangles) to bottom left (about thirty).

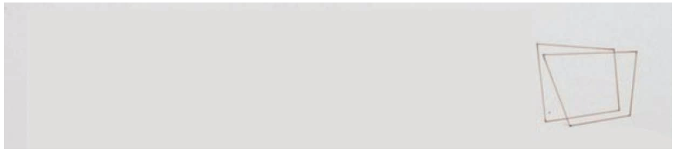
- The 3 last rows of cells are a superposition of half brown and half black lines. The repartition of density is the symetrical of the previous, it is mirrored (the density increases row by row from bottom left to top right).

All-in-all, we were very interested in the chaos created by this grid and tis imbrication of alternative squares, we are now trying to recreate it with an algorithm.

Pseudo-code

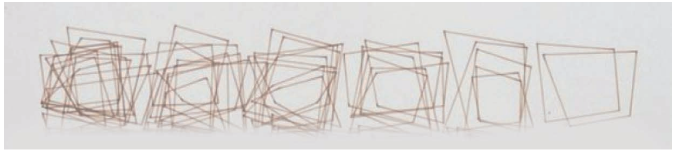
1. Take a brown pen.

2. Start on top-right corner. Draw two deformed and superposed squares, their shapes can be random.

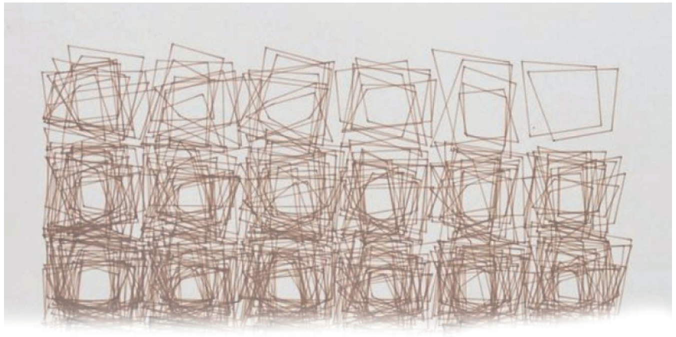


3. On the left of this first shapes, start again drawing 3 more random quadrangles over the same position.

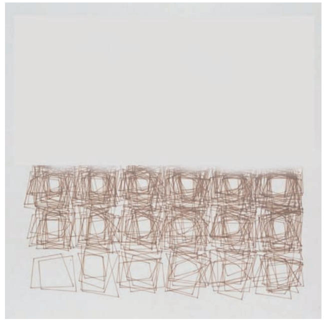
4. Repeat on the row until having 6 squares, denser and denser, increasing the amount of lines each time.



5. Repeat this 2 more times, starting from the right each time.



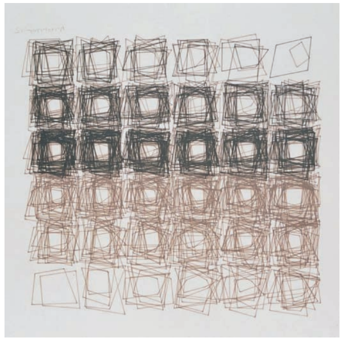
6. Rotate the canvas upside down.

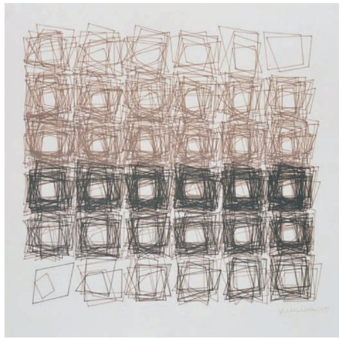


7. Take also a black pen.

8. Repeat the 5 first steps again. But, for each shape, begin by drawing the first half of the lines brown, then the second half black.

9. Rotate the canvas upside down again.





Pseudo-algorithm

```
function setup()
    create a 700*700 canvas
    DrawMesh(6, 6)

function DrawSquare(quad_amount, column, line)
    push()
        translate by 60*column on x axis, 60*line on y axis, 0 on z axis
        for k from 0 to quad_amount, with a step of 1
            dispersion <- []
            add 8 random relative integers between -10 and 10 to the dispersion array
            draw a quadrangle centered on the square generated by the 4 vertices (40,40),
(80,40), (80,80), (40,80) using the 8 integers of dispersion added to the 8 coordinates
        end for

    pop()
```

step 2

```
function DrawMesh(width, height)
    for j from 0 to height/2, with a step of 1
        for i from width to 0, with a step of 1
          ijquad_amount <- 2*(width-i-1 + width*j + 1)
          draw in brown
            DrawSquare(ijquad_amount, i, j)
        end for
    end for
```

steps 1 to 5

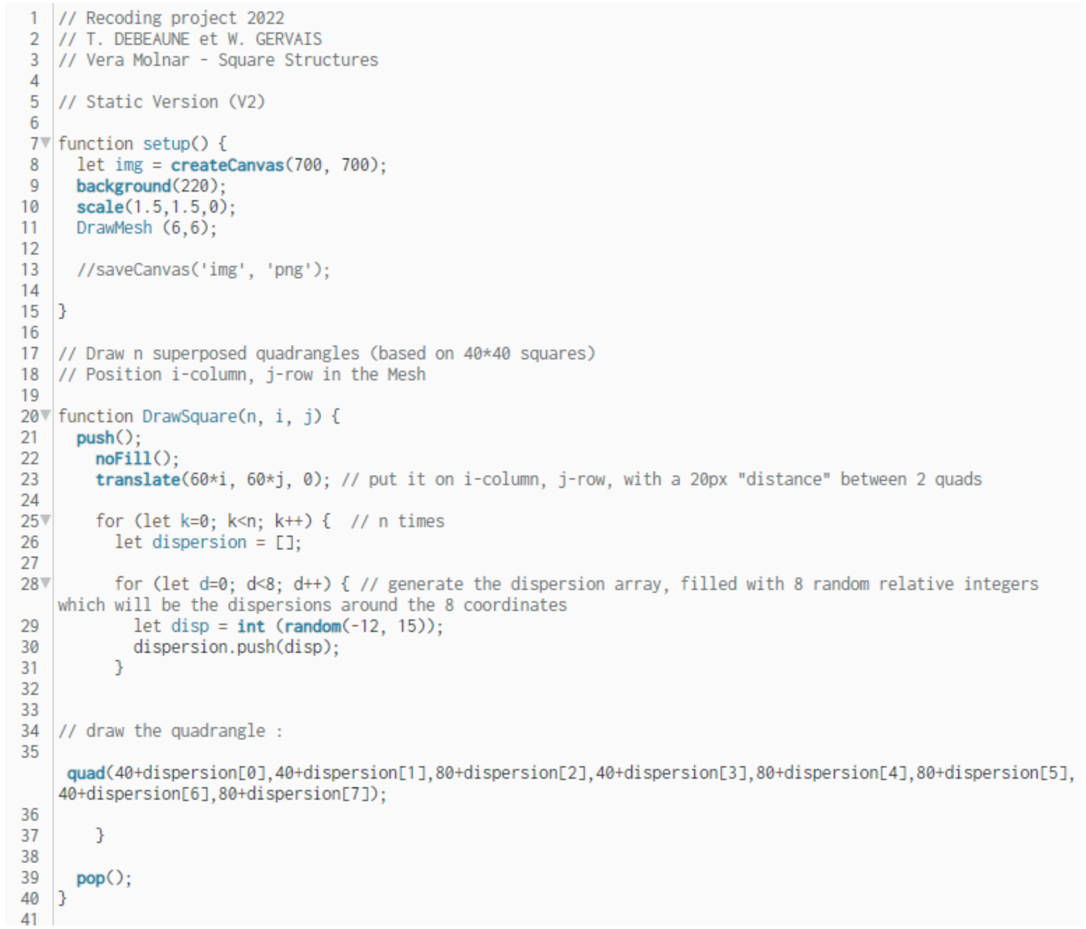
```
    for j from 0 to height/2, with a step of 1
        for i from width to 0, with a step of 1
          ijquad_amount <- width-i-1 + width*j + 1
          drawn in brown
            DrawSquare(ijquad_amount, width-1-i, height-1-j)
          drawn in black
            DrawSquare(ijquad_amount, width-1-i, height-1-j)
        end for
    end for
```

steps 6 to 8

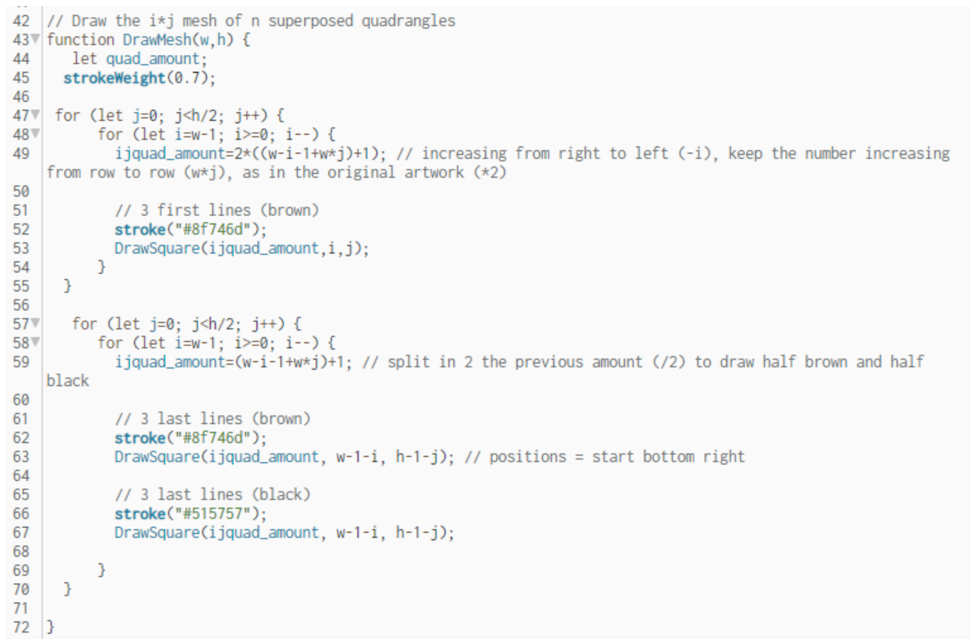
p5 version

p5 - https://editor.p5js.org/kau.grv/sketches/OC2bP--qy
gitlab (STATIC) - https://gitlab.com/debeaunetristan/projet-recoding

Code

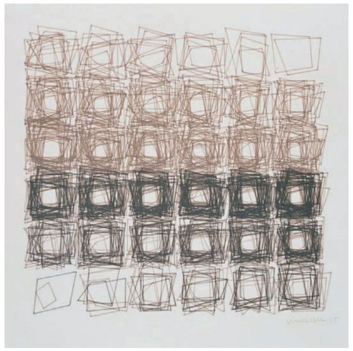


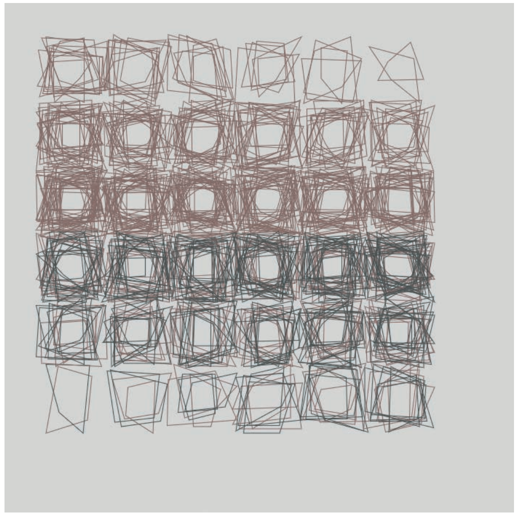
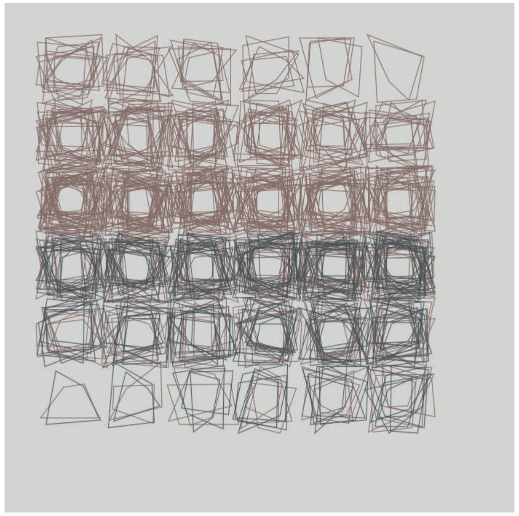
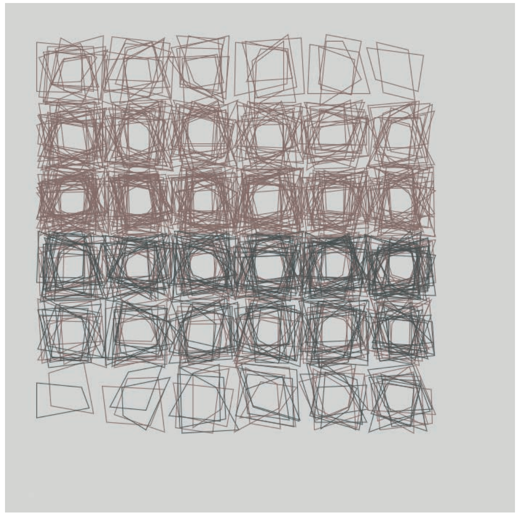
```
// Recoding project 2022
// T. DEBEAUNE et W. GERVAIS
// Vera Molnar - Square Structures

// Static Version (V2)

function setup() {
  let img = createCanvas(700, 700);
  background(220);
  scale(1.5,1.5,0);
  DrawMesh (6,6);

  //saveCanvas('img', 'png');

}

// Draw n superposed quadrangles (based on 40*40 squares)
// Position i-column, j-row in the Mesh

function DrawSquare(n, i, j) {
  push();
    noFill();
    translate(60*i, 60*j, 0); // put it on i-column, j-row, with a 20px "distance" between 2 quads

    for (let k=0; k<n; k++) {  // n times
      let dispersion = [];

      for (let d=0; d<8; d++) { // generate the dispersion array, filled with 8 random relative integers which will be the dispersions around the 8 coordinates
        let disp = int (random(-12, 15));
        dispersion.push(disp);
      }


// draw the quadrangle :

 quad(40+dispersion[0],40+dispersion[1],80+dispersion[2],40+dispersion[3],80+dispersion[4],80+dispersion[5],40+dispersion[6],80+dispersion[7]);

    }

  pop();
}

```

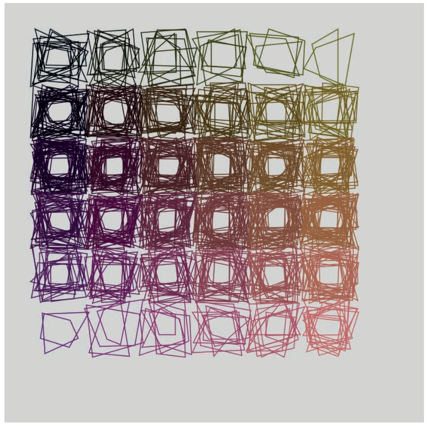


```
// Draw the i*j mesh of n superposed quadrangles
function DrawMesh(w,h) {
   let quad_amount;
  strokeWeight(0.7);

 for (let j=0; j<h/2; j++) {
      for (let i=w-1; i>=0; i--) {
        ijquad_amount=2*((w-i-1+w*j)+1); // increasing from right to left (-i), keep the number increasing from row to row (w*j), as in the original artwork (*2)

        // 3 first lines (brown)
        stroke("#8f746d");
        DrawSquare(ijquad_amount,i,j);
      }
   }

   for (let j=0; j<h/2; j++) {
      for (let i=w-1; i>=0; i--) {
        ijquad_amount=(w-i-1+w*j)+1; // split in 2 the previous amount (/2) to draw half brown and half black

        // 3 last lines (brown)
        stroke("#8f746d");
        DrawSquare(ijquad_amount, w-1-i, h-1-j); // positions = start bottom right

        // 3 last lines (black)
        stroke("#515757");
        DrawSquare(ijquad_amount, w-1-i, h-1-j);

      }
   }

}
```

Outputs


Original

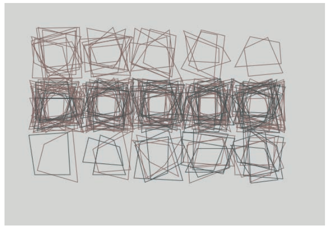



3 consecutive outputs of our code (using savecanvas())


with a color gradient

To look like the original, we used a 0.7 stroke weight, #8f746d and #515757 as brown and black and made a 6*6 grid, but we could change all those parameters in our fonctions to have the liberty of create other grids!

In the first version we used rotate functions to do the two parts, but we eventually managed to respect the pseudo-code better by really drawing each shape in the good order. We also increased the range of the random numbers so as the shapes can overlap.

All-in-all, we discovered the quad() function, we learnt how to implement a grid with «for» loops using their positions in an «aesthecial» way.


5*3