import org.openkinect.*; import org.openkinect.processing.*;

//turns off the Kinect sensing, uses the mouse as input Boolean debugMode = true;

// Showing how we can farm all the kinect stuff out to a separate class KinectTracker tracker; // Kinect Library object Kinect kinect;

//the number of superPixels (things start to slow down quickly if this is increased int xPixels = 100; int yPixels = 76;

float xSize, ySize, x, y; color pixelFill; float isMoving; color backColor; boolean gravity = false;

superPixel[][] pixelArray = new superPixel[xPixels][yPixels];

void setup() { size(1000, 760); noStroke();

//code this //initialize each superPixel, with a nice blueish color for (int i = 0; i < xPixels; i++) { for (int j = 0; j < yPixels; j++) {

xSize = (float)width/xPixels;  
 ySize = (float)height/yPixels;  
 x = (float)xSize \* (i);  
 y = (float)ySize \* (j);  
  
 color pixelFill = color(50);  
  
 pixelArray[i][j] = new superPixel(x, y, pixelFill, xSize, ySize);  
}

} }

void draw() { background(#AAAC9A);

for (int i = 0; i < xPixels; i++) { for (int j = 0; j < yPixels; j++) {

pixelArray[i][j].run();  
  
}

}

if (debugMode){ PVector mouse = new PVector(mouseX, mouseY);

if (mousePressed && (mouseButton == LEFT)) {  
 for (int i = 0; i < xPixels; i++) {  
 for (int j = 0; j < yPixels; j++) {  
 pixelArray[i][j].explode(500, mouse);  
 }  
 }  
}

}

if(!debugMode){ tracker.track(); float force = tracker.getForce(); PVector position = tracker.getPos();

for (int i = 0; i < xPixels; i++) {  
 for (int j = 0; j < yPixels; j++) {  
 pixelArray[i][j].explode(force, position);  
 }  
}

}  
}

void keyPressed() {

if (!debugMode){

//make it easy to adjust our threshold  
int t = tracker.getThreshold();  
if (key == CODED) {  
 if (keyCode == UP) {  
 t+=5;  
 tracker.setThreshold(t);  
 }   
 else if (keyCode == DOWN) {  
 t-=5;  
 tracker.setThreshold(t);  
 }  
}

}

//if we hit space, change the gravity! if (key == ‘’) { gravity = !gravity; println(“gravity:”+gravity); } }