//the superPixels are the actual cubes that fly around.

class superPixel { PVector origin; PVector location; PVector velocity; PVector acceleration; color fillColor; float xSize, ySize; float topspeed; int timer = 0;

//construct the superPixels!! superPixel(float x\_, float y\_, color fill\_, float xSize\_, float ySize\_) {

//remember where we started   
origin = new PVector(x\_, y\_);  
  
//standard location/velocity/acceleration  
location = new PVector(x\_, y\_);  
velocity = new PVector();  
acceleration = new PVector();  
  
//how should we fill the square? how big are they  
fillColor = fill\_;  
xSize = xSize\_;  
ySize = ySize\_;  
topspeed = 15;

}

//our main function, triggering subroutines void run() { timer -= 1; gravity(); checkEdges(); update(); display(); //if the pixel isn’t moving, there is no reason to ask it to move home/bounce off the wall/etc. if (velocity.mag() != 0 && timer < 1 &&! gravity) { returnHome(); } }

// //Dynamic Functions //

//make the superPixels bounce off the edges. increase the velocity for some nice effects!  
 void checkEdges() { if (location.x < 0) { location.x = 0; velocity.x *= -.9; } else if (location.x > width) { location.x = width; velocity.x* = -.9; }

if (location.y < 0) {  
 location.y = 0;  
 velocity.y \*= -.9;  
}   
else if (location.y > height - 2) {  
 if (gravity){  
 float f = randomGaussian();  
 f = -2 \* abs(f) - 1;  
 velocity.y \*= f;  
 velocity.x \*= 10 \* randomGaussian();  
 } else {  
 velocity.y \*= -.9;  
 }  
 location.y = height - 10;  
}

}

//return the super pixels to their starting space void returnHome() { //make a new vector, using the origin of the superPixel as a starting point PVector seek = origin.get();

//the vector now points from the location to the origin of the superPixel  
seek.sub(location);  
  
//how far away are we from the start? useful for the next part  
  
float distance = seek.mag();  
float speed = velocity.mag();  
//the distance test here seems to control the wobble  
  
//if the superPixel is slow and close to it's origin, reset it  
if (distance < .5 && speed < 8) {  
 location = origin.get();  
 velocity.mult(0);  
 acceleration.mult(0);  
}  
//otherwise, move towards home with a random acceleration   
else {  
 seek.normalize();  
 seek.mult(random(.3, .5));  
 acceleration.add(seek);  
}

}

//shoot out away from a location void explode(float force, PVector mouse) {

//make a new vector, starting at the mouse  
PVector gunpowder = mouse.get();  
  
//the vector now goes between the mouse and the superPixel  
gunpowder.sub(location);  
  
//check the distance between the two  
float distance = gunpowder.mag();  
  
 // if (distance < (force \* 7)){  
 timer = 20;  
 gunpowder.normalize();  
  
 gunpowder.mult((-1 \* force) / (distance));  
 applyForce(gunpowder);  
 // }

}

// // Utility Functions //

//basic motion, with some damping to slow everything down void update() { velocity.add(acceleration); velocity.mult(.96); velocity.limit(topspeed); location.add(velocity); acceleration.mult(0); }

//apply some gravity, if it’s turned on void gravity() { if (gravity) { applyForce(new PVector(0,random(.6,.8))); } }

//pass any forces to our object’s acceleration void applyForce(PVector force) { PVector f = force.get(); acceleration.add(f); }

// //Rendering Functions //

//draw the superPixel void display() { fill(fillColor); rect(location.x, location.y, xSize, ySize); }

}