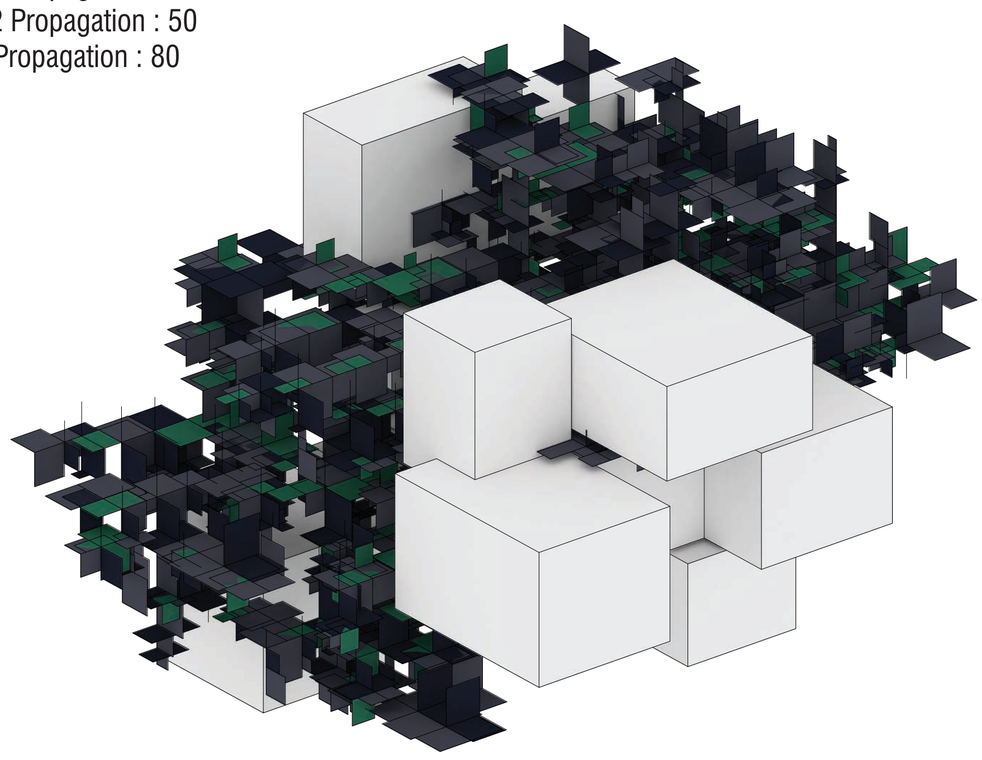
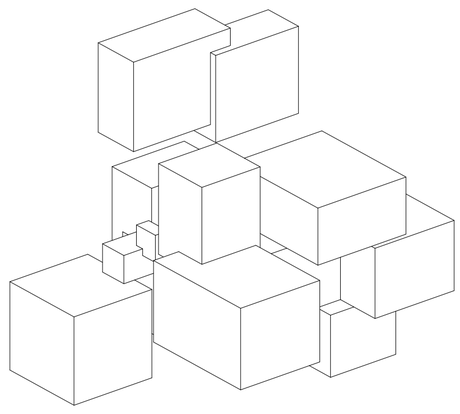
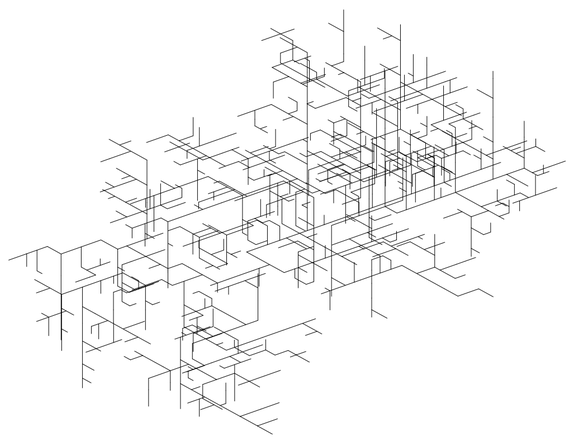
Southern California Institute of Architecture
FALL 2011
Cording Form
Shinji Takagi

Output 1:
IRandom Initial Value: 5
Duration: 25
NextDir1 Propagation : 80
NextDir2 Propagation : 50
Dir.dup Propagation : 80



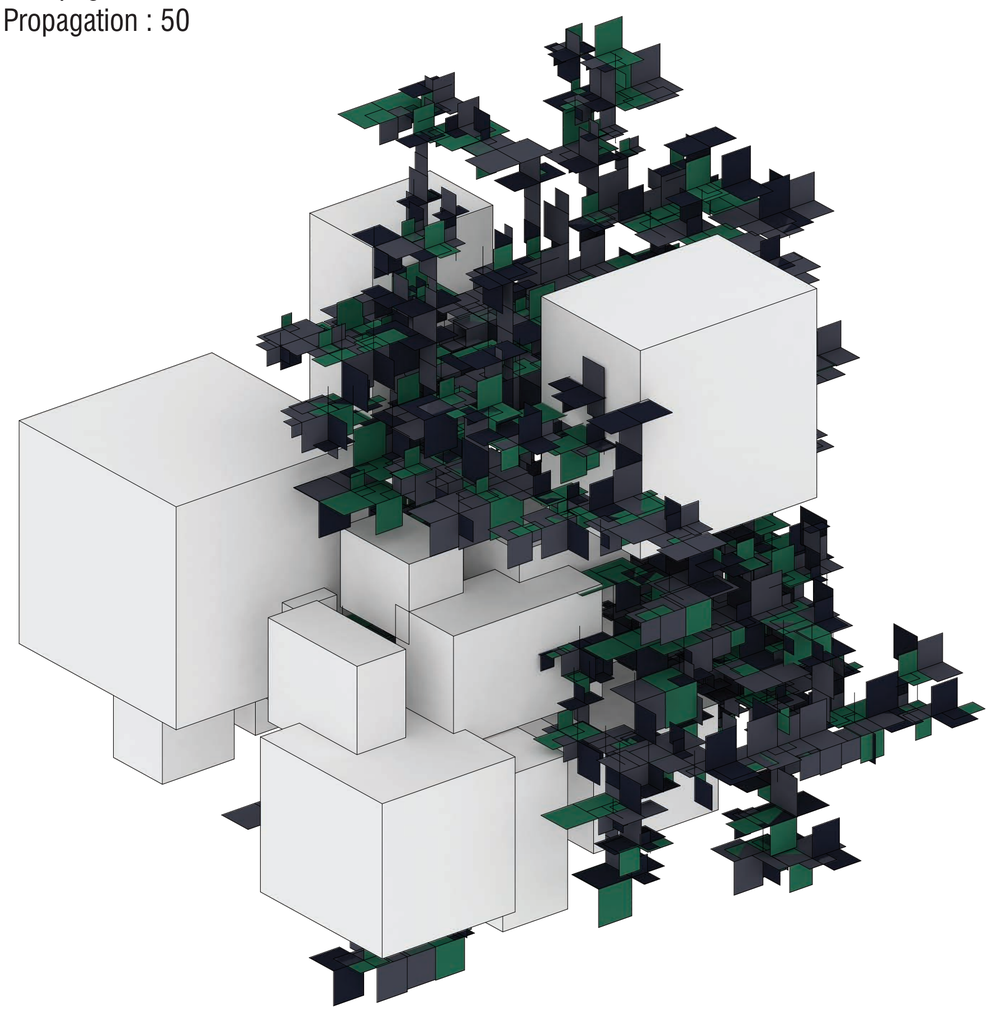


Output 1: Line Block Agents

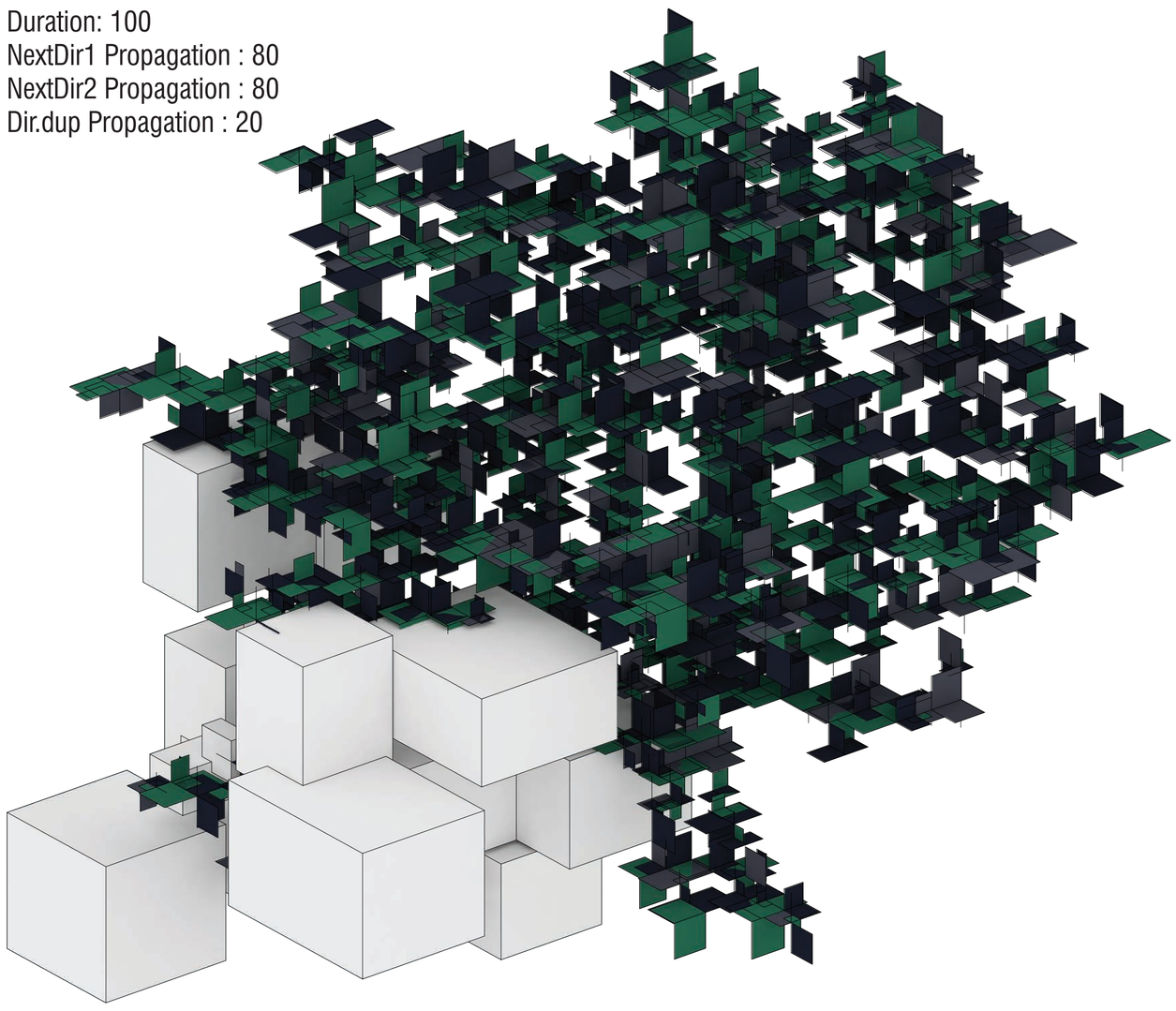


Output 1: Line Agents

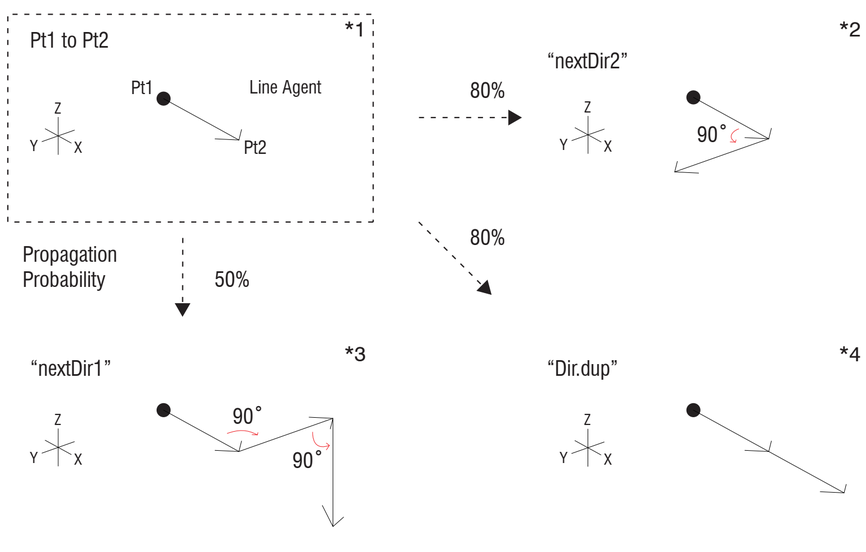
Output 3:
IRandom Initial Value: 1
Duration: 50
NextDir1 Propagation : 60
NextDir2 Propagation : 90
Dir.dup Propagation : 50



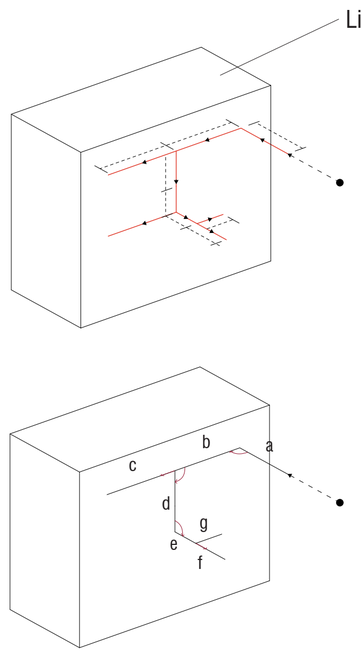
Output 2:
IRandom Initial Value: 5
Duration: 100
NextDir1 Propagation : 80
NextDir2 Propagation : 80
Dir.dup Propagation : 20



Line Agent Diagram



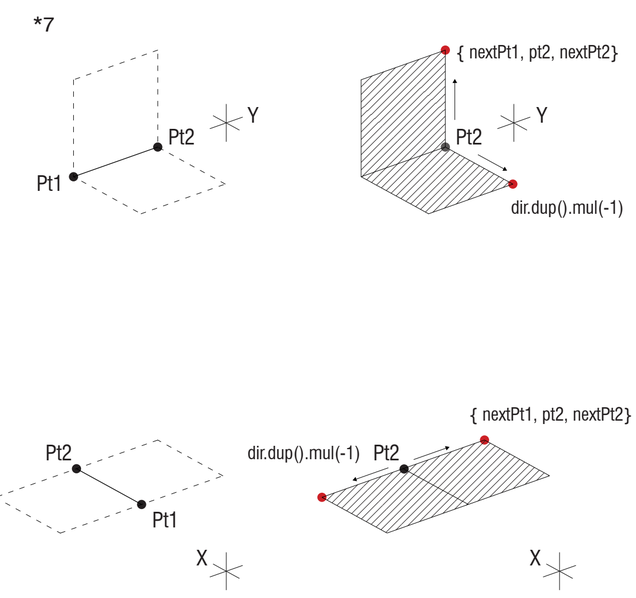
Line Block Agent



*Variation of length defined by "IRandom.get "range between 0.5 to 2 *5

Branch a: Dir.dup
Branch b: nextDir2
Branch c: Dir.dup
Branch d: nextDir1
Branch e: nextDir1
Branch f: Dir.dup
Branch g: nextDir
*6

Agent Geometry Diagram




```
import processing.opengl.*;
import igeo.*;

void setup() {
  size(800, 800, IG.GL);
  IRandom.init(5);

  IG.duration(25);
  new LineAgent(new IVec(14, 14, 14), new IVec(1, 0, 0)).clr(0);

  for (int i=-1; i<2; i++) {
    for (int j=-1; j<2; j++) {
      for (int k=-1; k<2; k++) {
        double size = i*2 *IRandom.get(4, 0.5) * 5;
        if(size<0) size=-size;
        if(size==0) size = 1.3 *IRandom.get(4, 0.5) * 2;
        new LineBlockAgent(new IVec(size*i*1.1, size*j*1.1, size*k*1.1),
                    size*IRandom.get(0.5,1.5), size*IRandom.get(0.5,1.5),
size*IRandom.get(0.5,1.5));
      }
    }
  }

  IG.transparent();
}

static class LineBlockAgent extends IAgent {
  IVec pos;
  double width, height, depth;

  LineBlockAgent(IVec p, double wid, double hei, double dep) {
    pos = p;
    width = wid;
    height = hei;
    depth = dep;
  }

  void interact(ArrayList < IDynamics > agents) {
    super.interact(agents);

    for (int i=0; i < agents.size(); i++) {
      if (agents.get(i) instanceof LineAgent) {
        LineAgent la = (LineAgent)agents.get(i);

        if ( la.pt2.x >= pos.x && la.pt2.x <= pos.x+width &&
          la.pt2.y >= pos.y && la.pt2.y <= pos.y+height &&
          la.pt2.z >= pos.z && la.pt2.z <= pos.z+depth ) {
          la.isColliding=true;
        }
      }
    }
  }

  void update() {
    super.update();
    if (time == 0) {
      new IBox(pos, width, height, depth).clr(255,255,255);
    }
  }
}

static class LineAgent extends IAgent {
  static double length = 5;                                    ------------- *1

  IVec pt1, pt2;
  boolean isColliding=false;

  LineAgent(IVec pt, IVec dir) {
    pt1 = pt;
    pt2 = pt.dup().add(dir.dup().len(length).mul(IRandom.get(0.5, 2)));----- *5
  }

  void interact(ArrayList < IDynamics > agents) {
    super.interact(agents);
    if (time == 0) {
      for (int i=0; i < agents.size() && !isColliding; i++) {
        if (agents.get(i) instanceof LineAgent) {
          LineAgent lineAgent =(LineAgent)agents.get(i);
          if (lineAgent != this) {
            if (lineAgent.pt2.dist(pt2) < pt1.dist(pt2)*0.999) {
              isColliding=true;
            }
          }
        }
      }
    }
  }

  void update() {
    super.update();

    if (isColliding) {
      del();
    }
    else if (time == 0) {
      new ICurve(pt1, pt2).clr(0);

      IVec dir = pt2.diff(pt1);
      IVec nextDir1 = dir.dup().flip().rot(IG.zaxis, -PI/2).rot(IG.yaxis, PI/2);
      IVec nextDir2 = dir.dup().flip().rot(IG.zaxis, PI/2);
      IVec nextPt1 = pt2.cp(nextDir1);
      IVec nextPt2 = pt2.cp(nextDir2);                             ------- *6

      IG.extrude(new IVec[] { nextPt1, pt2, nextPt2}, dir.dup().mul(-1)).clr(clr()); ------ *7

      int gray =
    (clr().getRed()+clr().getGreen()+clr().getBlue())/3 +
    IRandom.getInt(-10,10);
      if (IRandom.percent(80)) {
        new LineAgent(pt2, nextDir1).clr(30,32,46);   ------- *2
      }
      if (IRandom.percent(50)) {
        new LineAgent(pt2, nextDir2).clr(41,105,80); ------- *3
      }
      if (IRandom.percent(80)) {
        new LineAgent(pt2, dir.dup()).clr(68,70,82);   ------- *4
      }
    }
  }

  ISurface createEdgeSurface(IVec pt1, IVec pt2,
  IVec pt3, IVec pt4,
  IVec extrudeDir1,
  IVec extrudeDir2) {
    IVec[][] cpts = new IVec[4][2];
    cpts[0][0] = pt1.dup().add(extrudeDir1);
    cpts[1][0] = pt2.dup().add(extrudeDir1);
    cpts[2][0] = pt3.dup().add(extrudeDir2);
    cpts[3][0] = pt4.dup().add(extrudeDir2);
    cpts[0][1] = pt1.dup().sub(extrudeDir1);
    cpts[1][1] = pt2.dup().sub(extrudeDir1);
    cpts[2][1] = pt3.dup().sub(extrudeDir2);
    cpts[3][1] = pt4.dup().sub(extrudeDir2);
    return new ISurface(cpts, 3, 1).clr(0.3,0.4,0.8);
  }
}
```