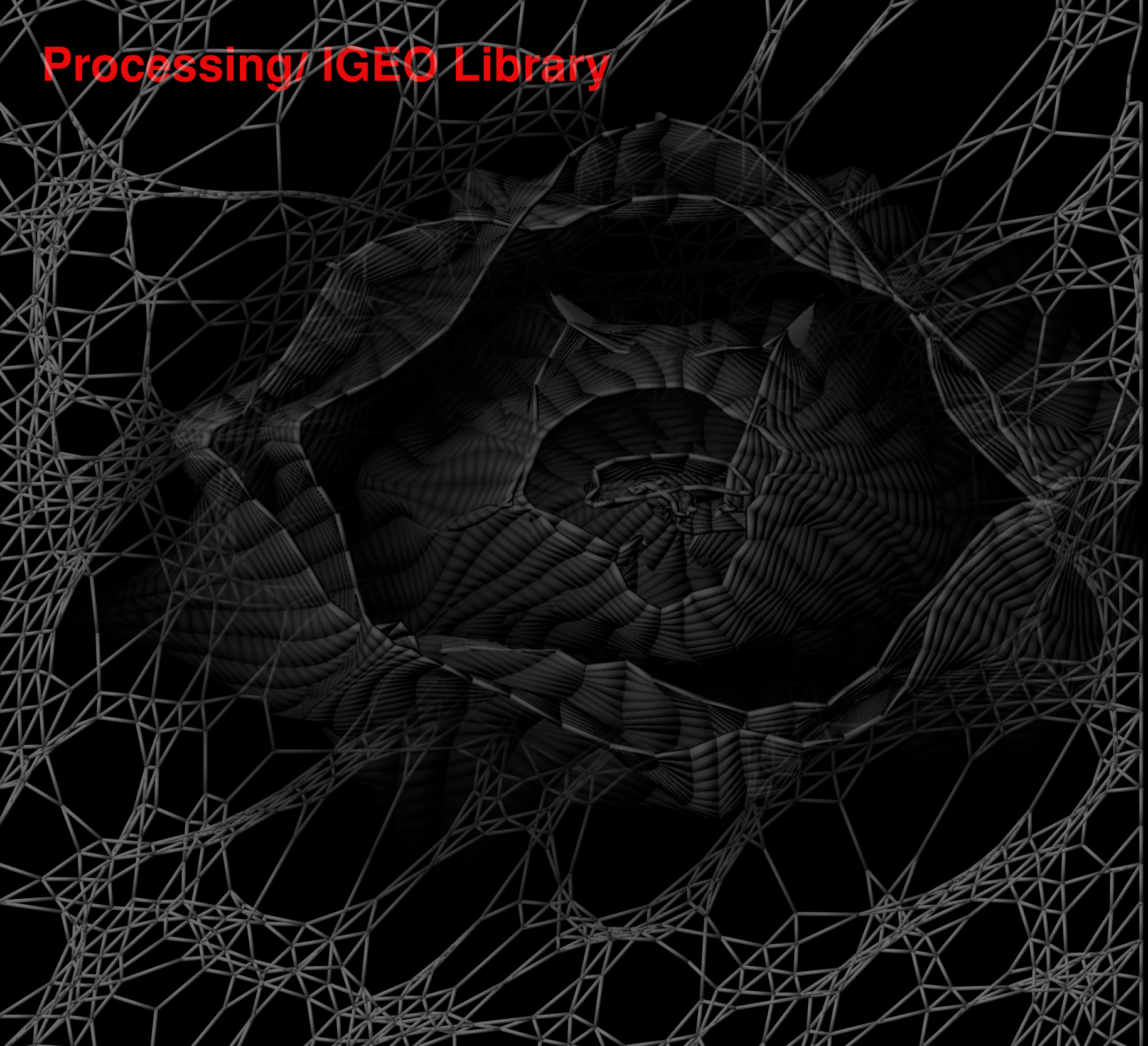
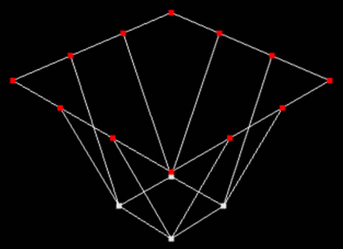
Processing/ IGEO Library

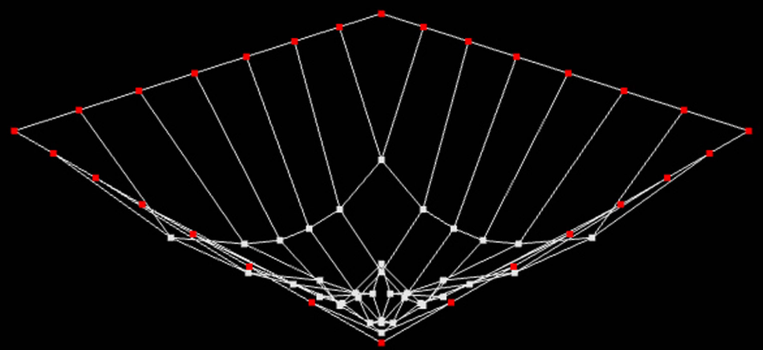


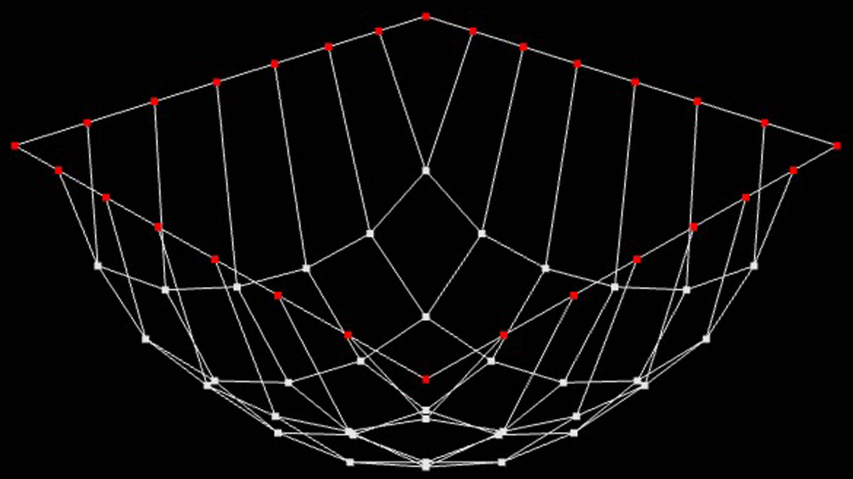
The Imbalancing Act of Entropic Architecture

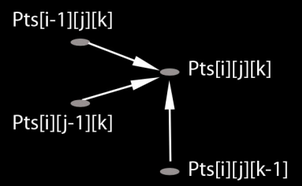
Mehrzad Rafeei
Somayyeh Ramezani

Tension to Particles









number of divitions

Friction =0.01

fix points (in red)

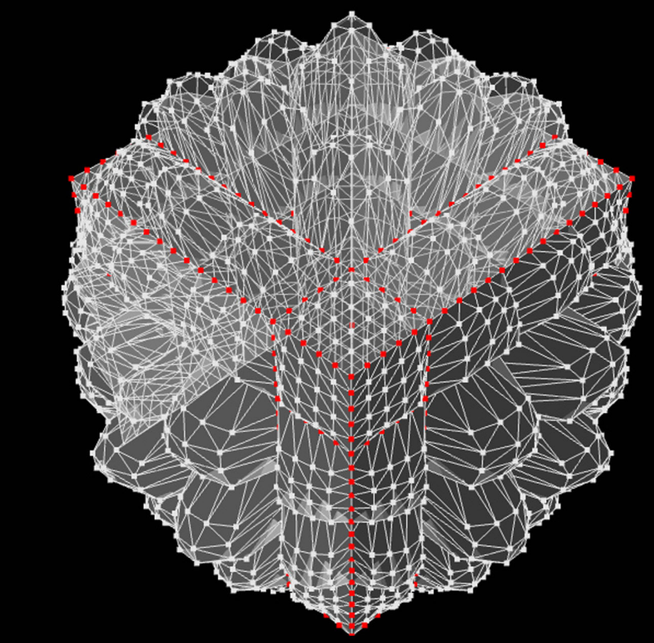
Tention line in X

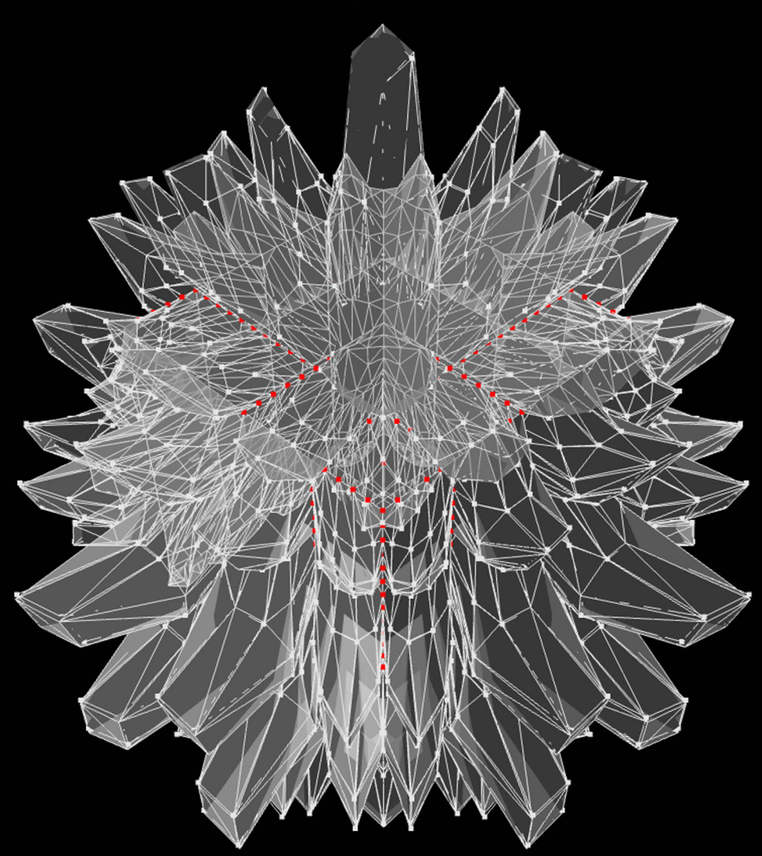
Tention line in Y

Tention line in Z

Amount of Strenght and Threshold distance

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

void setup() {
 size(800, 800, IG.GL);
 IG.duration(800);
 IG.bg(1);
 IG.perspective();
 int num = 7;
 MyParticle[][][] pts = new MyParticle[num+1][num+1][num+1];
 for (int i=0; i <= num; i++) {
  for (int j=0; j <= num; j++) {
   for (int k=0; k <= 0; k++) {
    if ( i==0 || i==num || j==0 || j==num || k==0 || k==num ) {
     pts[i][j][k] = new MyParticle(IG.v(10*i, 10*j, 10*k), IG.v(0, 0, 0));
     pts[i][j][k].fric(0.01).clr(0.9);
          if(i==0 || j==0 || i==num || j==num ){
      pts[i][j][k].fix().clr(1.0,0,0);
     }
          if (i > 0 && (j==0 || j==num || k==0 || k==num) ) {
      new ITensionLine(pts[i-1][j][k], pts[i][j][k], 10).clr(0.9);
     }
     if (j > 0 && (i==0 || i==num || k==0 || k==num) ) {
      new ITensionLine(pts[i][j-1][k], pts[i][j][k], 10).clr(0.9);
     }
     if (k > 0 && (i==0 || i==num || j==0 || j==num) ) {
      new ITensionLine(pts[i][j][k-1], pts[i][j][k], 10).clr(0.9);
      }
    }
   }
  }
 }

 new RepulsionAgent(IG.v(50,50,50));
}
void draw(){
}
class RepulsionAgent extends IPointAgent{
 double strength = 10;
 double thresholdDist = 100;

 RepulsionAgent(IVec p){ super(p); }
 void interact(IDynamics agent){
  if(agent instanceof MyParticle){
   MyParticle p = (MyParticle)agent;
   double dist = p.pos().dist(pos());
   if(dist < thresholdDist){
    IVec force = p.pos().dif(pos());
    force.len(thresholdDist - dist);
    force.mul(strength);
    p.push(force);
   }
  }
 }
}

class MyParticle extends IParticleAgent {
 IVec prevPos;
 MyParticle(IVec pos, IVec vel) {
  super(pos, vel);
 }
}
```





number of divitions
Friction =0.01

Fix points in red (each 5 point)

Tention line in X

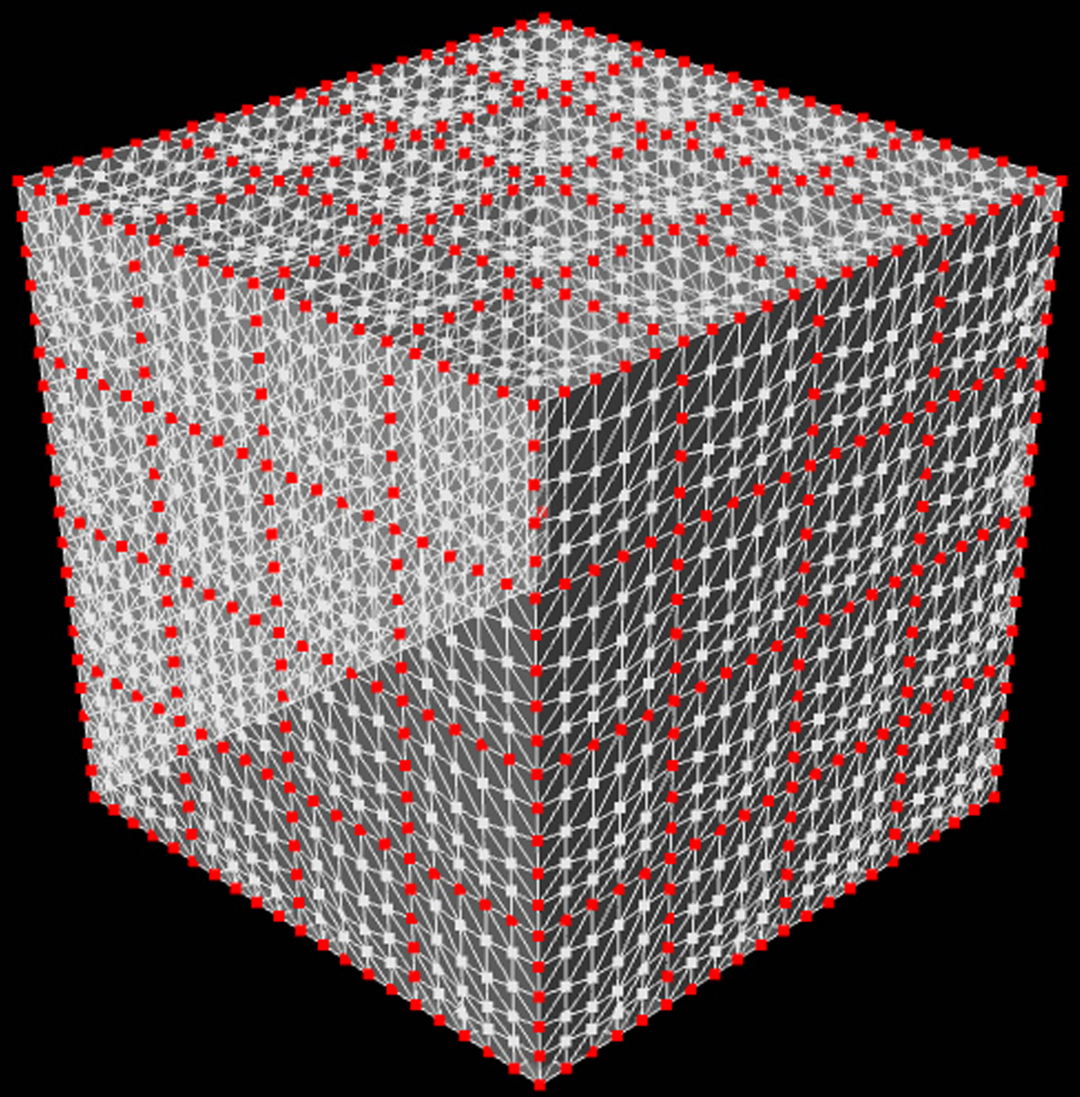
Tention line in Y

Tention line in Z

Draw surface between points

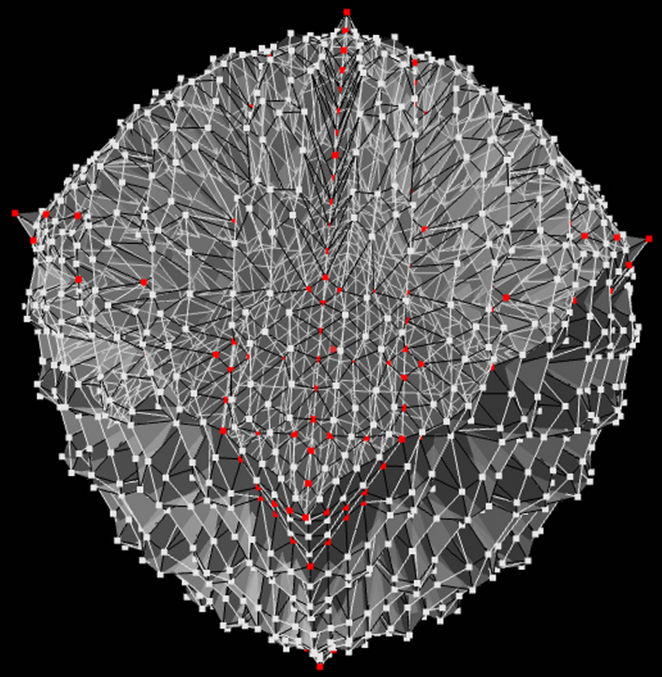
Center point for repulsion
(at the center of the cube)

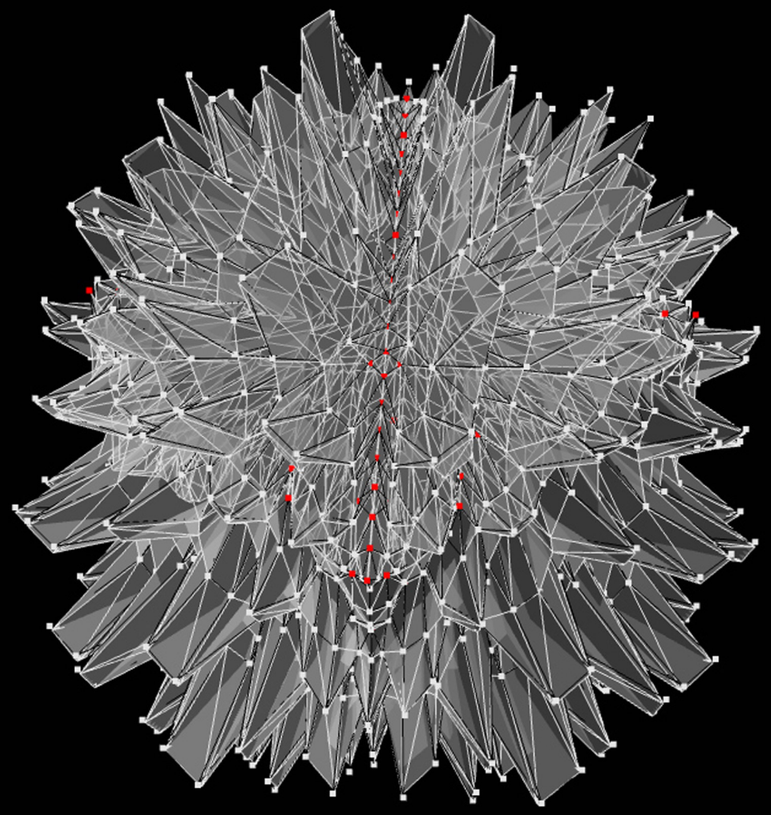
```
iimport processing.opengl.*;
import igeo.*;
void setup() {
 size(1200, 1200, IG.GL);
 IG.duration(500);
  IG.bg(1);
 IG.perspective();
 int num =20;
 MyParticle[][][] pts = new MyParticle[num+1][num+1][num+1];
 for (int i=0; i <= num; i++) {
  for (int j=0; j <= num; j++) {
    for (int k=0; k <= num; k++) {
          if ( i==0 || i==num || j==0 || j==num || k==0 || k==num ) {
       pts[i][j][k] = new MyParticle(IG.v(5*i*.5, 5*j*0.5, 5*k*0.5), IG.v(10,10, 10));
       pts[i][j][k].fric(0.01).clr(0.9);
             if ( (i==0 || i==num) && (j%5==0 || k%5==0)  ||
          (j==0 || j==num) && (i%5==0 || k%5==0)  ||
          (k==0 || k==num) && (i%5==0 || j%5==0)
               ){
         pts[i][j][k].fix().clr(1.0,0,0);
       }
             if (i > 0 && (j==0 || j==num || k==0 || k==num) ) {
         new ITensionLine(pts[i-1][j][k], pts[i][j][k], 5).clr(0.9);
       }
       if (j > 0 && (i==0 || i==num || k==0 || k==num) ) {
         new ITensionLine(pts[i][j-1][k], pts[i][j][k], 5).clr(0.9);
       }
       if (k > 0 && (i==0 || i==num || j==0 || j==num) ) {
         new ITensionLine(pts[i][j][k-1], pts[i][j][k],5).clr(0.9);
       }
     }
        }
  }
 }
  IVec[][] topPts = new IVec[num+1][num+1];
IVec[][] bottomPts = new IVec[num+1][num+1];
IVec[][] leftPts = new IVec[num+1][num+1];
IVec[][] rightPts = new IVec[num+1][num+1];
IVec[][] frontPts = new IVec[num+1][num+1];
IVec[][] backPts = new IVec[num+1][num+1];
  for (int i=0; i <= num; i++) {
  for (int j=0; j <= num; j++) {
    bottomPts[i][j] = pts[i][j][0].pos();
    topPts[i][j] = pts[i][j][num].pos();
     leftPts[i][j] = pts[0][i][j].pos();
    rightPts[i][j] = pts[num][i][j].pos();
    frontPts[i][j] = pts[i][0][j].pos();
    backPts[i][j] = pts[i][num][j].pos();
  }
}
 new IMesh(topPts).clr(0.9);
 new IMesh(bottomPts).clr(0.9);
new IMesh(leftPts).clr(0.9);
new IMesh(rightPts).clr(0.9);
new IMesh(frontPts).clr(0.9);
new IMesh(backPts).clr(0.9);

   new RepulsionAgent(IG.v(25,25,25)).clr(1.0,0,0);
    }
```



Amount of Strenght and Threshold distance

```
class RepulsionAgent extends IPointAgent{
 double strength = 20;
 double thresholdDist =40;
  RepulsionAgent(IVec p){ super(p); }
 void interact(IDynamics agent){
  if(agent instanceof MyParticle){
   MyParticle p = (MyParticle)agent;
   double dist = p.pos().dist(pos());
   if(dist < thresholdDist){
    IVec force = p.pos().dif(pos());
    force.len(thresholdDist - dist);
    force.mul(strength);
    p.push(force);
   }
  }
 }
}
class MyParticle extends IParticleAgent {
 IVec prevPos;
 MyParticle(IVec pos, IVec vel) {
  super(pos, vel);
 }
}
```





number of divitions
Friction =0.01

fix points in red (in cross lines)

Tention line in X

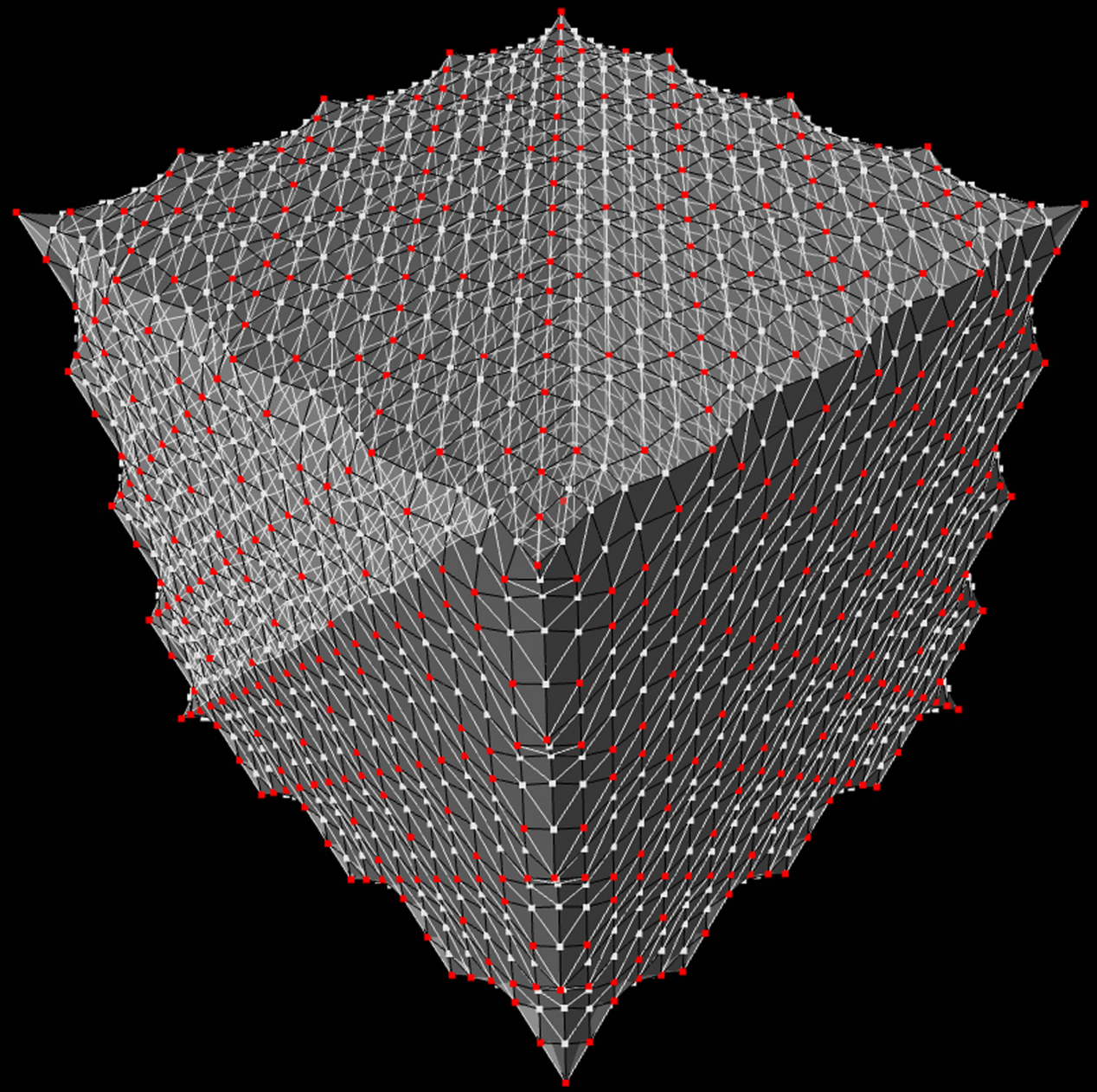
Tention line in Y

Tention line in Z

Draw surface between points

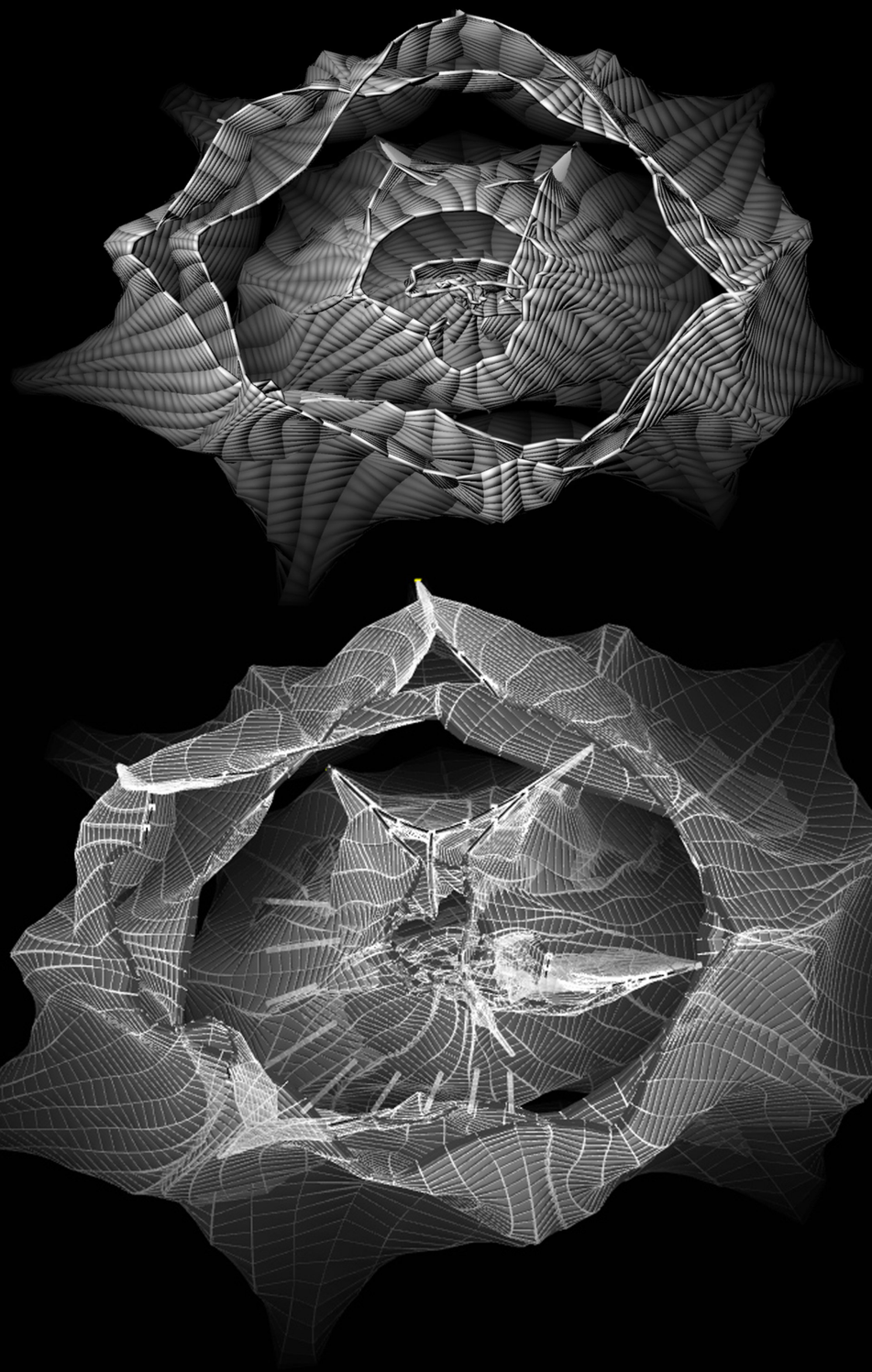
center point for repulsion
(at the center of the cube)

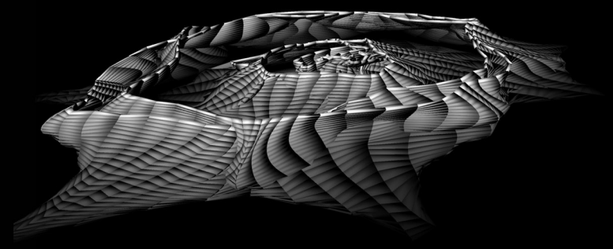
```
import processing.opengl.*;
import igeo.*;
void setup() {
 size(1000, 1000, IG.GL);
 IG.duration(500);
  IG.bg(1);
 IG.perspective();
 int num =20;
 MyParticle[][][] pts = new MyParticle[num+1][num+1][num+1];
 for (int i=0; i <= num; i++) {
   for (int j=0; j <= num; j++) {
     for (int k=0; k <= num; k++) {
           if ( i==0 || i==num || j==0 || j==num || k==0 || k==num ) {
        pts[i][j][k] = new MyParticle(IG.v(5*i*.5, 5*j*0.5, 5*k*0.5), IG.v(10,10, 10));
        pts[i][j][k].fric(0.01).clr(0.9); //friction
             if ( (i==0 || i==num) && ( (j+k)%5==0 || (j-k)%5==0 )  ||
          (j==0 || j==num) && ( (i+k)%5==0 || (i-k)%5==0 )  ||
          (k==0 || k==num) && ( (i+j)%5==0 || (i-j)%5==0 )
        ){
          pts[i][j][k].fix().clr(1.0,0,0);
        }
        if (i > 0 && (j==0 || j==num || k==0 || k==num) ) { //tension line in X
          new ITensionLine(pts[i-1][j][k], pts[i][j][k], 5).clr(0);
        }
        if (j > 0 && (i==0 || i==num || k==0 || k==num) ) { //tension line in Y
          new ITensionLine(pts[i][j-1][k], pts[i][j][k], 5).clr(0);
        }
        if (k > 0 && (i==0 || i==num || j==0 || j==num) ) { //tension line in z
          new ITensionLine(pts[i][j][k-1], pts[i][j][k],5).clr(0);
        }
     }
   }
 }
 IVec[][] topPts = new IVec[num+1][num+1];
 IVec[][] bottomPts = new IVec[num+1][num+1];
 IVec[][] leftPts = new IVec[num+1][num+1];
 IVec[][] rightPts = new IVec[num+1][num+1];
 IVec[][] frontPts = new IVec[num+1][num+1];
 IVec[][] backPts = new IVec[num+1][num+1];
 for (int i=0; i <= num; i++) {
   for (int j=0; j <= num; j++) {
     bottomPts[i][j] = pts[i][j][0].pos();
     topPts[i][j] = pts[i][j][num].pos();
     leftPts[i][j] = pts[0][i][j].pos();
     rightPts[i][j] = pts[num][i][j].pos();
     frontPts[i][j] = pts[i][0][j].pos();
     backPts[i][j] = pts[i][num][j].pos();
   }
 }
 new IMesh(topPts).clr(.9);
 new IMesh(bottomPts).clr(.9);
 new IMesh(leftPts).clr(.9);
 new IMesh(rightPts).clr(.9);
 new IMesh(frontPts).clr(.9);
 new IMesh(backPts).clr(.9);
      new RepulsionAgent(IG.v(25,25,25)).clr(1.0,0,0);
}
class RepulsionAgent extends IPointAgent{
 double strength = 20;
 double thresholdDist =40;
   RepulsionAgent(IVec p){ super(p); }
```

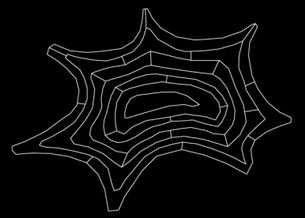


Amount of Strenght and
Threshold distance

```
void interact(IDynamics agent){
  if(agent instanceof MyParticle){
    MyParticle p = (MyParticle)agent;
    double dist = p.pos().dist(pos());
    if(dist < thresholdDist){
      IVec force = p.pos().dif(pos());
      force.len(thresholdDist - dist);
      force.mul(strength);
      p.push(force);
    }
   }
  }
}
class MyParticle extends IParticleAgent {
 IVec prevPos;
 MyParticle(IVec pos, IVec vel) {
   super(pos, vel);
 }
}
```





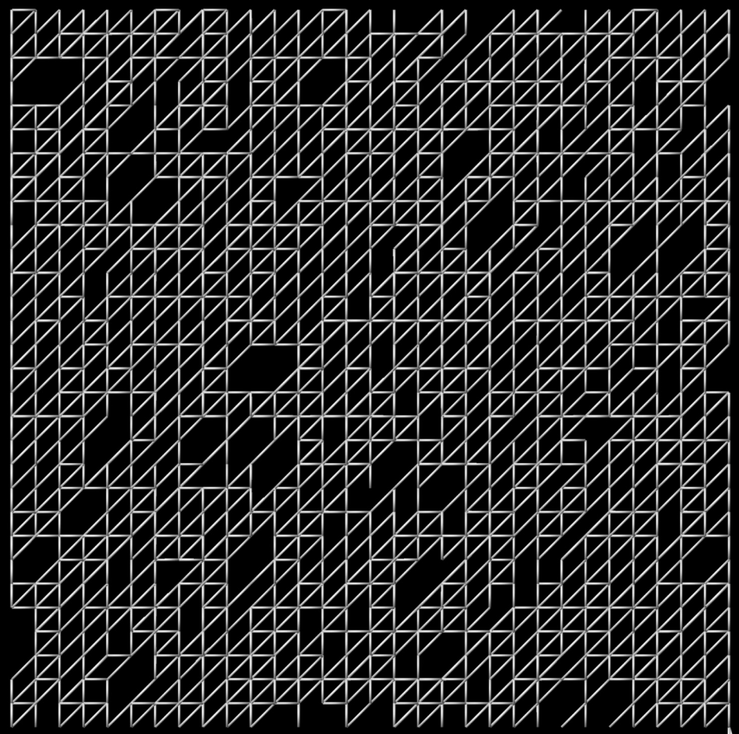


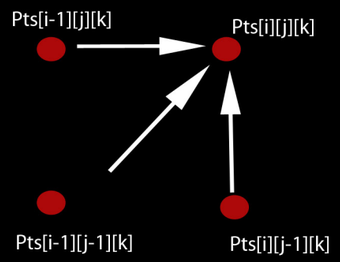
Gravity toward z

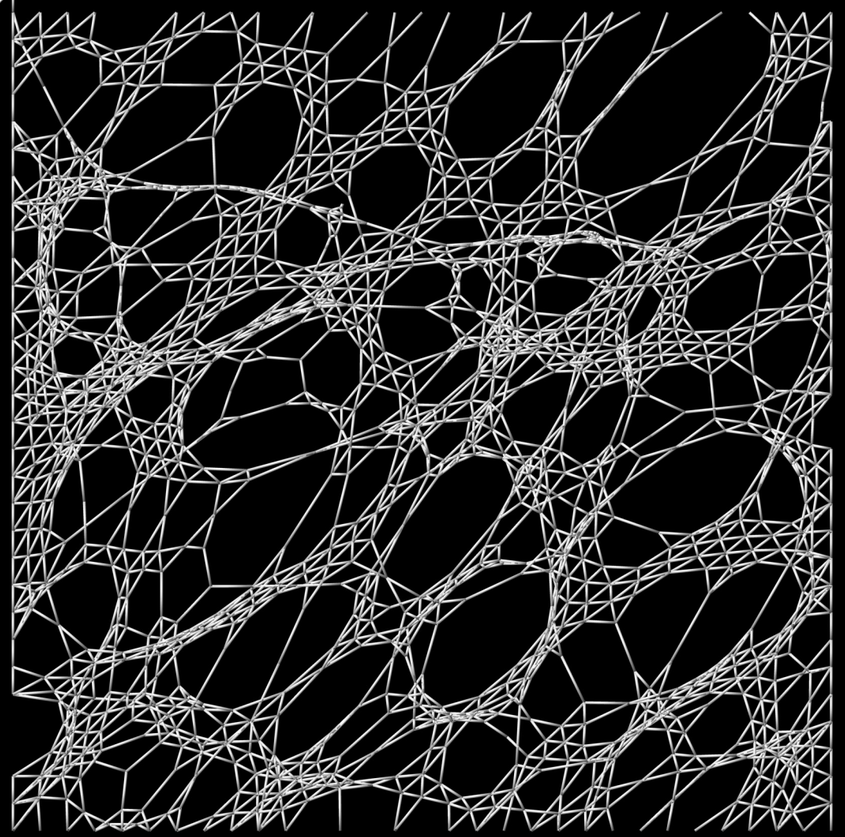
thickness of each box

Draw boxes between points

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

void setup(){
 size(800, 800, IG.GL);
 IG.bg(1);
 IG.perspective();
 IG.open("lines3.3dm");
 IG.duration(500);
 ITensileNet.tensionClass(MyTension.class); //custom tension class
 ITensileNet.create(IG.curves(), IG.points());

 new Gravity(IG.v(0, 0, 3));
}

class MyTension extends ITensionLine{
 IVec prevPos1, prevPos2;

 MyTension(IParticleAgent p1, IParticleAgent p2){
  super(p1,p2);
 }

 void update(){
  double thickness = 5;
  if(IG.time()%10==0){
   IVec p1 = pos1().cp();
   IVec p2 = pos2().cp();
   if(prevPos1!=null && prevPos2!=null ){
    if( !prevPos1.eq(p1) || !prevPos2.eq(p2) ){
     IVec hdir1 = p2.dif(p1);
     IVec hdir2 = prevPos2.dif(prevPos1);
     IVec vdir1 = p1.dif(prevPos1);
     IVec vdir2 = p2.dif(prevPos2);
     if(p1.eq(prevPos1)){ vdir1 = vdir2; } //avoid zero vector
     if(p2.eq(prevPos2)){ vdir2 = vdir1; } //avoid zero vector

     IVec vertex1 = p1.cp().add(hdir1.cross(vdir1).len(thickness/2));
     IVec vertex2 = p2.cp().add(hdir1.cross(vdir2).len(thickness/2));
     IVec vertex3 = prevPos2.cp().add(hdir2.cross(vdir2).len(thickness/2));
     IVec vertex4 = prevPos1.cp().add(hdir2.cross(vdir1).len(thickness/2));
     IVec vertex5 = p1.cp().sub(hdir1.cross(vdir1).len(thickness/2));
     IVec vertex6 = p2.cp().sub(hdir1.cross(vdir2).len(thickness/2));
     IVec vertex7 = prevPos2.cp().sub(hdir2.cross(vdir2).len(thickness/2));
     IVec vertex8 = prevPos1.cp().sub(hdir2.cross(vdir1).len(thickness/2));
     IG.meshBox(vertex1, vertex2, vertex3, vertex4,
           vertex5, vertex6, vertex7, vertex8).clr(IG.time()*0.003);
    }
   }
   prevPos1 = p1;
   prevPos2 = p2;
  }
 }
}

class Gravity extends IAgent{
 IVec gravity;
 Gravity(IVec g){ gravity=g; }
 void interact(IDynamics agent){
  if(agent instanceof IParticleAgent){
   IParticleAgent particle = (IParticleAgent)agent;
   particle.push(gravity);
  }
 }
}
```







number of divitions

Friction =0.01

Tention line in X

Tention line in Y

Tention line in X & Y

fix points (in red)

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

void setup(){

 size(800, 800, IG.GL);
 IG.duration(500);
  IG.bg(.8);
 int num = 30;
 MyParticle[][][] pts = new MyParticle[num+1][num+1][num+1];
 for(int i=0; i <= num; i++){
  for(int j=0; j <= num; j++){
   for (int k=0; k <= 1; k++) {
   if(IRand.pct(80)){
    pts[i][j][k] = new MyParticle(IG.v(10*i,10*j,0), IG.v(0,0,0));
    pts[i][j][k].fric(0.01); //friction

    if(i > 0 && pts[i-1][j][k]!=null){
     new MyTensionLine(pts[i-1][j][k], pts[i][j][k],1).clr(0);
    }
    if(j > 0 && pts[i][j-1][k]!=null){
     new MyTensionLine(pts[i][j-1][k], pts[i][j][k],1).clr(0);
    }
    if(i > 0 && j > 0 && pts[i-1][j-1][k]!=null){
     new MyTensionLine(pts[i-1][j-1][k], pts[i][j][k],1).clr(0);
    }
    if(i==0 || j==0 || i==num || j==num && k==0){ // edge
     pts[i][j][k].fix().clr(0.5,0,0);
    }
   }
  }
 }
}

}


class MyTensionLine extends ITensionLine{
 ICylinder cylinder;
 IMesh stick;
 IVec pt1, pt2;
 MyTensionLine(MyParticle p1, MyParticle p2, double tension){
  super(p1,p2,tension);
  pt1 = p1.pos();
  pt2 = p2.pos();
 }
 void update(){
  if(stick!=null){
   stick.del();
  }
  stick = IG.meshSquareStick(pt1,pt2,1);
 }
}

class MyParticle extends IParticleAgent{
 IVec prevPos;
 MyParticle(IVec pos, IVec vel){ super(pos,vel); }
}
```