

****

实验报告

内容（名称）：社会力模型的实现

|  |  |
| --- | --- |
| 院（系）名称 | 计算机学院 |
| 专业名称 | 计算机科技与技术 |
| 指导教师 | 宋晓 |
| 学号 | 16061194，16231298，76066001，76066002 |
| 姓名 | 肖宏，韩鑫溢，张金源，周美廷 |

2018年12月

**社会力模型实验报告**

1. **实验目的**

给出一个典型的场景，比如单房间疏散或对流（过人行横道），实现图形化仿真。

1. **数学模型**
2. 随着公共场所大规模人群活动越来越多，人群安全问题明显增加。因此人员安全问题备受关注，围绕群体运动行为的描述、规律和引导等展开了大量研究。其中，社会力模型广受关注，在行人流领域特别是逃生方面有了广泛的运用。
3. 社会力（又称社会场或社会影响力）概念：

mi =mi+





其中vi为行人速度，m为行人质量，fij 为两人之间的斥力，fiw 为人受到墙的阻力

1. 模拟了一个名为大运村的场景，人在前往位置时会选择前往认为最近的路线，行人的运动并非机械性运动，下一步行为并非完全由动力方程学决定，因此引入随机因子，多次模拟后，人数增加到50乃至更多没有出现拥堵现象
2. **编程实现与调试过程**

1.主函数：

public class Main{  
 public static void main(String[] args) {  
 try {  
 BeautyEyeLNFHelper.*launchBeautyEyeLNF*();  
 UIManager.*put*("RootPane.setupButtonVisible",false);  
 } catch (Exception e) {  
 }  
 try {  
 //JFrame frame = new JFrame("SimulationPanelMain");  
 *frame* = new JFrame("Epimetheus");  
 //SimulationPanelMain mainPanel = new SimulationPanelMain();  
 ApplicationMain mainPanel = new ApplicationMain();  
 mainPanel.getLoader().add(new DaYunCun());  
 /\*Application org.example.application = new ApplicationForDoorTest();//应用在这里！  
 org.example.application.setApplicationListener(mainPanel);\*/  
 //mainPanel.setLoader(new ApplicationLoader(mainPanel));  
 *frame*.setContentPane(mainPanel.demoP);  
 *frame*.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 //frame.setResizable(false);  
 *frame*.pack();  
 *frame*.setVisible(true);  
  
 //org.example.application.start();  
 } catch (HeadlessException e) {  
 System.*out*.println("GUI Not Supported on this machine.");  
 }  
 }  
}

1. 社会力模型及场景的构建：

社会力由人与人之间的排斥力，墙与人的排斥力，目标与人的作用力三部分组成，关键代码如下

public class NewForceModel implements Model{  
 String parentPath = System.*getProperty*("user.dir")+"/resource/";  
 String locationToSave = "neuralNet/robust.net";  
  
 double timePerStep = 0.05;  
 double min\_div = 0.5;  
 double p = 0.3;  
 MultiLayerNetwork model = null;  
  
 public NewForceModel(){  
  
 try {  
 model = ModelSerializer.*restoreMultiLayerNetwork*(parentPath+locationToSave);  
 } catch (IOException e) {  
 e.printStackTrace();  
 }  
 }  
 @Override  
 public void fieldForce(Pool targets) {  
 for(Object target:targets) {  
 Agent = (Agent)target;  
 Velocity2D expected = (Velocity2D)this.zeroVelocity(), velocity = (Velocity2D) agent.getVelocity();  
 Point current = agent.getShape().getReferencePoint(), goal = agent.getPath().nextStep(current);  
 expected.sub(current);  
 expected.add(goal);  
 double angle = Vector2D.*getRotateAngle*(new Vector2D(1,0), velocity);  
 velocity.rotate(angle);  
 expected.rotate(angle);  
 boolean rotated = false;  
 if(expected.getY() < 0){  
 expected = new Velocity2D(expected.getX(),-expected.getY());  
 rotated = true;  
 }  
 double a\_angle = Vector2D.*getRotateAngle*(expected, new Vector2D(1,0));  
 /\* env matrix *TODO replace by matrix instead of n^2 iteration*\*/  
 int n= 0;  
 for(Object t\_arget:targets){  
 Agent co\_Agent = (Agent)t\_arget;  
 if(!co\_Agent.equals(agent) && co\_Agent.getShape().distanceTo(agent.getShape()) < min\_div){  
 n++;  
 }  
 }  
 /\* end env matrix\*/  
 INDArray input = Nd4j.*create*(new double[]{a\_angle, velocity.getX(), 0, 0, 0, n\*p/2, 0, n\*p/2, 0, 0, 0});  
 INDArray output = model.output(input);  
 Velocity2D newV = new Velocity2D(output.getDouble(0),output.getDouble(1));  
 if(rotated) newV = new Velocity2D(newV.getX(),-newV.getY());  
 newV.rotate(-angle);  
 ((BaseAgent)agent).setCurrVelocity(newV);  
 }  
  
 }  
  
 @Override  
 public Force interactionForce(InteractiveEntity source, InteractiveEntity target) {  
 return zeroForce();  
 }  
  
 @Override  
 public Moment interactionMoment(InteractiveEntity source, InteractiveEntity target) {  
 return zeroMoment();  
 }  
  
 @Override  
 public Vector zeroVector() {  
 return new Vector2D(0,0);  
 }  
  
 @Override  
 public Velocity zeroVelocity() {  
 return new Velocity2D(0,0);  
 }  
  
 @Override  
 public Force zeroForce() {  
 return new Force2D(0,0);  
 }  
  
 @Override  
 public double getTimePerStep() {  
 return timePerStep;  
 }  
  
 @Override  
 public Moment zeroMoment() {  
 return new Moment2D(0);  
 }  
  
 @Override  
 public Model clone() {  
 return new NewForceModel();  
 }  
}

public class SocialForceModel implements Model {  
 double TIME\_PER\_STEP = 0.008;  
 double EXPECTED\_SPEED = 1.5;  
 double EXPECTED\_PALSTANCE = 0;  
 double REACT\_TIME = 0.5;  
  
 long psyT = 0, bodyT = 0, flT = 0;  
 int psyN = 0, bodyN = 0, flN = 0;  
  
 protected List<ForceRegulation> regulations;  
  
 public SocialForceModel() {  
 regulations = new LinkedList<>();  
 regulations.add(new MyPsychologicalForceRegulation(Blockable.class, Agent.class, this));  
 regulations.add(new MyBodyForce(Blockable.class, Agent.class, this));  
 }  
  
 public SocialForceModel(double timePerStep){  
 this();  
 TIME\_PER\_STEP = timePerStep;  
 }  
  
 */\*\*  
 \* 计算源实体对目标实体的作用力。  
 \*  
 \** ***@param*** *source 产生作用力的实体。  
 \** ***@param*** *target 受力的作用的实体。  
 \** ***@return*** *the force. 返回力的大小，其单位是牛。  
 \*/* @Override  
 public Force interactionForce(InteractiveEntity source, InteractiveEntity target) {  
 Force = this.zeroForce();  
 for (ForceRegulation regulation : regulations) {  
 if (regulation.hasForce(source, target)) {  
 Affection temp = regulation.getForce(source, target);  
 if (temp instanceof Force){  
 force.add((Vector) temp);}  
 }  
 }  
 return force;  
 }  
  
 @Override  
 public Moment interactionMoment(InteractiveEntity source, InteractiveEntity target) {  
 Moment = this.zeroMoment();  
 for (ForceRegulation regulation : regulations) {  
 if (regulation.hasForce(source,target)) {  
 Affection temp = regulation.getForce(source, target);  
 if (temp instanceof Moment) {  
 moment.add((Moment) temp);  
 }  
 }  
 }  
 return moment;  
 }  
  
 @Override  
 public Vector zeroVector() {  
 return new Vector2D(0,0);  
 }  
  
 @Override  
 public Velocity zeroVelocity() {  
 return new Velocity2D(0,0);  
 }  
  
 @Override  
 public Force zeroForce() {  
 return new Force2D(0,0);  
 }  
  
 @Override  
 public Moment zeroMoment(){return new Moment2D(0);}  
  
 @Override  
 public double getTimePerStep() {  
 return TIME\_PER\_STEP;  
 }  
  
 public double getExpectedSpeed() {  
 return EXPECTED\_SPEED;  
 }  
  
 public void setExpectedSpeed(double Expect) {  
 EXPECTED\_SPEED = Expect;  
 }  
  
 */\*\*  
 \* 生成模型的场力--即驱动力。  
 \*  
 \** ***@param*** *sources 获有作用力的实体们。  
 \** ***@return*** *the force. 返回力的大小，其单位是牛。  
 \*/* public void fieldForce(Pool sources) {  
 for(Object source : sources){  
 if (!(source instanceof Agent)) {  
 return;  
 }  
 Agent agent = (Agent) source;  
 Velocity expected = this.zeroVelocity();  
 Force force = this.zeroForce();  
 Point current = agent.getShape().getReferencePoint(), goal = agent.getPath().nextStep(current);  
 expected.sub(current);  
 expected.add(goal);  
 expected.scale(EXPECTED\_SPEED / expected.length());  
 force.add(expected);  
 force.sub(agent.getVelocity());  
 force.scale(agent.getMass() / REACT\_TIME);  
 agent.push(force);  
  
 }  
 }  
  
 private <T extends InteractiveEntity> T reg(T entity) {  
 entity.setModel(this);  
 return entity;  
 }  
  
 public Model clone(){  
 return new SocialForceModel();  
 }  
}

1. 场景构建及可视化

构建一个名为大运村的场景，以及实现可视化，关键代码如下

package org.example.application;  
  
import org.example.model.SocialForceModel;  
import org.socialforce.app.Applications.SimpleApplication;  
import org.socialforce.geom.DistanceShape;  
import org.socialforce.geom.impl.\*;  
import org.socialforce.model.impl.\*;  
import org.socialforce.scene.Scene;  
import org.socialforce.scene.SceneLoader;  
import org.socialforce.scene.impl.\*;  
import org.socialforce.strategy.GoalStrategy;  
import org.socialforce.strategy.PathFinder;  
import org.socialforce.strategy.impl.AStarPathFinder;  
import org.socialforce.strategy.impl.FurthestGoalStrategy;  
  
import java.util.Iterator;  
  
*/\*\*  
 \* Created by sunjh1999 on 2017/2/26.  
 \*/*public class DaYunCun extends SimpleApplication {  
 DistanceShape template;  
 public DaYunCun(){  
 }  
 */\*\*  
 \* start the org.example.application immediately.  
 \*/* @Override  
 public void start() {  
 setUpScenes();  
 for (Iterator<Scene> iterator = scenes.iterator(); iterator.hasNext();){  
 currentScene = iterator.next();  
 PathFinder pathFinder = new AStarPathFinder(currentScene, template);  
 GoalStrategy strategy = new FurthestGoalStrategy(currentScene, pathFinder);  
 strategy.pathDecision();  
 while (!toSkip()) {  
 this.StepNext(currentScene);  
 }  
 if(onStop()) return;  
 }  
 }  
  
 */\*\*  
 \* 需要根据parameter的map来生成一系列scene  
 \*/* @Override  
 public void setUpScenes(){  
 template = new Circle2D(new Point2D(0,0),0.486/2);  
 SceneLoader loader = new StandardSceneLoader(new SimpleScene(new Box2D(-50, -50, 100, 100)),  
 new Wall[]{  
 new Wall(new Box2D(-6,-10,1,41)),  
 new Wall(new Box2D(45,-10,1,41)),  
  
 new Wall(new Box2D(-5,30,50,1)),  
 new Wall(new Box2D(5,20,1,10)),  
 new Wall(new Box2D(15,20,1,10)),  
 new Wall(new Box2D(25,20,1,10)),  
 new Wall(new Box2D(35,20,1,10)),  
 new Wall(new Box2D(-5,20,50,1)),  
 new Wall(new Box2D(-5,15,50,1)),  
  
 new Wall(new Box2D(-5,-10,50,1)),  
 new Wall(new Box2D(5,-10,1,10)),  
 new Wall(new Box2D(15,-10,1,10)),  
 new Wall(new Box2D(25,-10,1,10)),  
 new Wall(new Box2D(35,-10,1,10)),  
 new Wall(new Box2D(-5,0,50,1)),  
 new Wall(new Box2D(-5,5,50,1))  
 }).setModel(new SocialForceModel());  
  
 SimpleParameterPool parameters = new SimpleParameterPool();  
  
 parameters.addValuesAsParameter(new RandomEntityGenerator2D(30,new Box2D(3,1,3,8))  
 .setValue(new BaseAgent(template, new Velocity2D(3,0)))  
 ,new RandomEntityGenerator2D(1,new Box2D(3,1,5,8))  
 .setValue(new BaseAgent(template, new Velocity2D(3,0)))  
 );  
  
 //parameters.addValuesAsParameter(new RandomEntityGenerator2D(1,new Box2D(33,1,3,8))  
 //.setValue(new BaseAgent(template, new Velocity2D(-3,0)))  
 //);  
  
 parameters.addValuesAsParameter(new MultipleEntitiesGenerator()  
 .addValue(new SafetyRegion(new Box2D(50,7.5,1,8)))  
 .addValue(new SafetyRegion(new Box2D(55,7.5,1,8)))  
 .addValue(new Exit(new Box2D(44,10,3,3)))  
 .addValue(new Exit(new Box2D(-6,10,2,5)))  
 );  
  
 parameters.addValuesAsParameter(new MultipleEntitiesGenerator()  
 .addValue(new Door(new Rectangle2D(new Point2D(0,0),3,0.5,0),new Point2D(4,0),new double[]{0,Math.*PI*/2},1))  
 .addValue(new Door(new Rectangle2D(new Point2D(3,0),3,0.5,0),new Point2D(6,0),new double[]{-Math.*PI*/2,0},-1))  
 .setCommonName("Door")  
 );  
  
 loader.readParameterSet(parameters);  
 scenes = loader.readScene();  
 for(Scene scene:scenes){  
 scene.setApplication(this);  
 }  
 }

可视化如下

package org.jb2011.lnf.beautyeye;  
  
import java.awt.Color;  
import javax.swing.BorderFactory;  
import javax.swing.LookAndFeel;  
import javax.swing.UIManager;  
import javax.swing.border.Border;  
import org.jb2011.lnf.beautyeye.ch1\_titlepane.\_\_UI\_\_;  
import org.jb2011.lnf.beautyeye.utils.JVM;  
import org.jb2011.lnf.beautyeye.utils.Platform;  
import org.jb2011.lnf.beautyeye.widget.border.BEShadowBorder;  
import org.jb2011.lnf.beautyeye.widget.border.BEShadowBorder3;  
import org.jb2011.lnf.beautyeye.widget.border.PlainGrayBorder;  
  
public class BeautyEyeLNFHelper {  
 public static boolean debug = false;  
 public static boolean translucencyAtFrameInactive = true;  
 public static BeautyEyeLNFHelper.FrameBorderStyle frameBorderStyle;  
 public static Color activeCaptionTextColor;  
 public static Color commonBackgroundColor;  
 public static Color commonForegroundColor;  
 public static Color commonFocusedBorderColor;  
 public static Color commonDisabledForegroundColor;  
 public static Color commonSelectionBackgroundColor;  
 public static Color commonSelectionForegroundColor;  
 public static boolean setMaximizedBoundForFrame;  
  
 static {  
 frameBorderStyle = isSurportedTranslucency() ? BeautyEyeLNFHelper.FrameBorderStyle.translucencyAppleLike : BeautyEyeLNFHelper.FrameBorderStyle.generalNoTranslucencyShadow;  
 activeCaptionTextColor = new Color(0, 0, 0);  
 commonBackgroundColor = new Color(250, 250, 250);  
 commonForegroundColor = new Color(60, 60, 60);  
 commonFocusedBorderColor = new Color(162, 162, 162);  
 commonDisabledForegroundColor = new Color(172, 168, 153);  
 commonSelectionBackgroundColor = new Color(2, 129, 216);  
 commonSelectionForegroundColor = new Color(255, 255, 255);  
 setMaximizedBoundForFrame = true;  
 }  
  
 public BeautyEyeLNFHelper() {  
 }  
  
 protected static void implLNF() {  
 \_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch2\_tab.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch3\_button.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch\_x.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch4\_scroll.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch5\_table.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch6\_textcoms.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch7\_popup.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch8\_toolbar.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch9\_menu.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch10\_internalframe.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch12\_progress.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch13\_radio.cb\_btn.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch14\_combox.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch15\_slider.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch16\_tree.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch17\_split.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch18\_spinner.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch19\_list.\_\_UI\_\_.uiImpl();  
 org.jb2011.lnf.beautyeye.ch20\_filechooser.\_\_UI\_\_.uiImpl();  
 }  
  
 public static String getBeautyEyeLNFStrCrossPlatform() {  
 return "org.jb2011.lnf.beautyeye.BeautyEyeLookAndFeelCross";  
 }  
  
 public static String getBeautyEyeLNFStrWindowsPlatform() {  
 return "org.jb2011.lnf.beautyeye.BeautyEyeLookAndFeelWin";  
 }  
  
 public static LookAndFeel getBeautyEyeLNFCrossPlatform() {  
 return new BeautyEyeLookAndFeelCross();  
 }  
  
 public static LookAndFeel getBeautyEyeLNFWindowsPlatform() {  
 return new BeautyEyeLookAndFeelWin();  
 }  
  
 public static void launchBeautyEyeLNF() throws Exception {  
 if (Platform.isWindows()) {  
 if (debug) {  
 System.out.println("已智能启用Windows平台专用的BeautyEye外观实现(您也可自行启用跨平台实现).");  
 }  
  
 UIManager.setLookAndFeel(getBeautyEyeLNFStrWindowsPlatform());  
 } else {  
 if (debug) {  
 System.out.println("已智能启用跨平台的通用BeautyEye外观实现.");  
 }  
  
 UIManager.setLookAndFeel(getBeautyEyeLNFStrCrossPlatform());  
 }  
  
 }  
  
 public static boolean isSurportedTranslucency() {  
 return JVM.current().isOneDotSixUpdate12OrAfter();  
 }  
  
 public static boolean \_\_isFrameBorderOpaque() {  
 return frameBorderStyle == BeautyEyeLNFHelper.FrameBorderStyle.osLookAndFeelDecorated || frameBorderStyle == BeautyEyeLNFHelper.FrameBorderStyle.generalNoTranslucencyShadow;  
 }  
  
 public static Border \_\_getFrameBorder() {  
 switch($SWITCH\_TABLE$org$jb2011$lnf$beautyeye$BeautyEyeLNFHelper$FrameBorderStyle()[frameBorderStyle.ordinal()]) {  
 case 1:  
 return BorderFactory.createEmptyBorder();  
 case 2:  
 return new BEShadowBorder3();  
 case 3:  
 return new BEShadowBorder();  
 case 4:  
 default:  
 return new PlainGrayBorder();  
 }  
 }  
  
 public static enum FrameBorderStyle {  
 osLookAndFeelDecorated,  
 translucencyAppleLike,  
 translucencySmallShadow,  
 generalNoTranslucencyShadow;  
  
 private FrameBorderStyle() {  
 }  
 }  
  
 public interface \_\_UseParentPaintSurported {  
 boolean isUseParentPaint();  
 }  
}

1. 路径算法

路径算法的关关键代码分为几个类，关键代码如下

PathFinder.Class

package org.socialforce.strategy;  
  
import org.socialforce.geom.Point;  
import org.socialforce.scene.Scene;  
  
public interface PathFinder {  
 Path plan\_for(Point var1);  
  
 Path constraint\_plan\_for(Point var1, Point... var2);  
  
 Point[] getGoals();  
  
 void addSituation(Scene var1, Point var2);  
  
 void addSituation(Point var1);  
  
 void clearCache();  
}

Point.Class

package org.socialforce.geom;  
  
public interface Point extends Vector {  
 double getX();  
  
 double getY();  
  
 double distanceTo(Point var1);  
  
 Vector directionTo(Point var1);  
  
 Point clone();  
  
 Point moveTo(double var1, double var3);  
  
 Point moveBy(double var1, double var3);  
  
 Point scaleBy(double var1);  
  
 double Manhattan\_Distance(Point var1);  
  
 double getAngle();  
  
 Point coordinateTransfer(Point var1, double var2);  
  
 Point inverseCoordinateTransfer(Point var1, double var2);  
}

StraightPathFinder.class

public class StraightPathFinder implements PathFinder {  
 LinkedList<Point> goals = new LinkedList();  
 Shape agentShape;  
 Scene scene;  
  
 public StraightPathFinder(Scene targetScene, Shape agentShape) {  
 this.agentShape = agentShape.clone();  
 this.scene = targetScene;  
 Iterator iter = this.scene.getStaticEntities().selectClass(SafetyRegion.class).iterator();  
  
 while(iter.hasNext()) {  
 SafetyRegion safetyRegion = (SafetyRegion)iter.next();  
 this.goals.addLast(safetyRegion.getShape().getReferencePoint().clone());  
 }  
  
 }  
  
 public Path plan\_for(Point goal) {  
 return new StraightPath(new Point[]{goal});  
 }  
  
 public Path constraint\_plan\_for(Point goal, Point... toBeContained) {  
 Point[] pathPoints = new Point[toBeContained.length + 1];  
  
 for(int i = 0; i < toBeContained.length; ++i) {  
 pathPoints[i] = toBeContained[i];  
 }  
  
 pathPoints[-1] = goal;  
 return new StraightPath(pathPoints);  
 }  
  
 public Point[] getGoals() {  
 Point[] points = new Point[this.goals.size()];  
  
 for(int i = 0; i < points.length; ++i) {  
 points[i] = (Point)this.goals.get(i);  
 }

AStarPathFinder.Class

public AStarPathFinder(int[][] new\_map, Agent agent, Point... goals) {  
 this.min\_div = 1.0D;  
 this.map = new\_map;  
 Point[] var4 = goals;  
 int var5 = goals.length;  
  
 int var6;  
 Point goal;  
 for(var6 = 0; var6 < var5; ++var6) {  
 goal = var4[var6];  
 this.goals.addLast(goal.clone());  
 }  
  
 this.agentShape = agent.getShape().clone();  
 var4 = goals;  
 var5 = goals.length;  
  
 for(var6 = 0; var6 < var5; ++var6) {  
 goal = var4[var6];  
 this.astar(goal);  
 }  
  
}

public void addSituation(Scene scene, Point goal) {  
 assert scene.getBounds().equals(this.templateScene.getBounds());  
  
 Scene newScene = scene.cloneWithStatics();  
 this.scene\_generate(newScene);  
 this.map\_generate(newScene);  
 this.goals.addLast(this.point\_generate(goal.clone()));  
 this.astar((Point)this.goals.getLast());  
}  
  
public void addSituation(Point goal) {  
 this.goals.addLast(this.point\_generate(goal.clone()));  
 this.astar((Point)this.goals.getLast());  
}

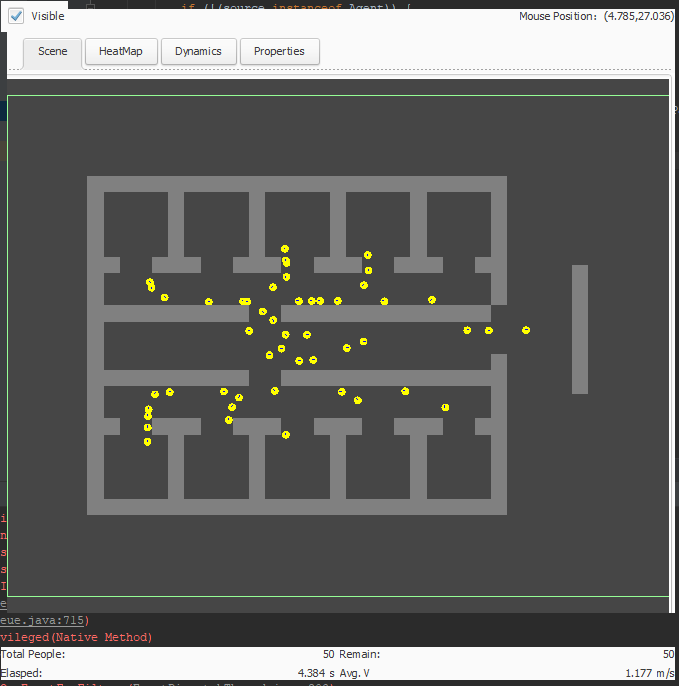
public Point[] getGoals() {  
 Point[] points = new Point[this.goals.size()];  
  
 for(int i = 0; i < points.length; ++i) {  
 points[i] = ((Point)this.goals.get(i)).clone().scaleBy(this.min\_div).moveBy(this.delta\_x, this.delta\_y);  
 }  
  
 return points;  
}

public Point findNext(Point start\_point) {  
 double x = (start\_point.getX() - this.delta\_x) / AStarPathFinder.this.min\_div;  
 double y = (start\_point.getY() - this.delta\_y) / AStarPathFinder.this.min\_div;  
 int xInt = (int)Math.floor(x);  
 int yInt = (int)Math.floor(y);  
 double distance = 1.0D / 0.0;  
 int tempX = 0;  
 int tempY = 0;  
 int available = false;  
  
 for(int offset = 0; !available; ++offset) {  
 for(int i = xInt - offset; i <= xInt + offset + 1; ++i) {  
 for(int j = yInt - offset; j <= yInt + offset + 1; ++j) {  
 if (AStarPathFinder.this.available(i, j) && this.previous[i][j] != null && (new Point2D((double)i, (double)j)).distanceTo(new Point2D(x, y)) < distance) {  
 tempX = i;  
 tempY = j;  
 distance = (new Point2D((double)i, (double)j)).distanceTo(new Point2D(x, y));  
 available = true;  
 }  
 }  
 }  
 }  
  
 Point next = this.previous[tempX][tempY];  
 Point tobeReturn = next.clone().scaleBy(AStarPathFinder.this.min\_div).moveBy(this.delta\_x, this.delta\_y);  
 return tobeReturn;  
}

public double getDistance(Point current) {  
 double x = (current.getX() - this.delta\_x) / AStarPathFinder.this.min\_div;  
 double y = (current.getY() - this.delta\_y) / AStarPathFinder.this.min\_div;  
 double Distance = 1.0D / 0.0;  
 int tempX = 0;  
 int tempY = 0;  
 int available = false;  
  
 for(int offset = 0; !available; ++offset) {  
 for(int i = (int)x - offset; (double)i <= x + (double)offset + 1.0D; ++i) {  
 for(int j = (int)y - offset; (double)j <= y + (double)offset + 1.0D; ++j) {  
 if (AStarPathFinder.this.available(i, j) && this.previous[i][j] != null && (new Point2D((double)i, (double)j)).distanceTo(this.goal) < Distance) {  
 tempX = i;  
 tempY = j;  
 Distance = (new Point2D((double)i, (double)j)).distanceTo(this.goal);  
 available = true;  
 }  
 }  
 }  
 }  
  
 x = (double)tempX;  
 y = (double)tempY;  
 return this.distance[(int)x][(int)y] \* AStarPathFinder.this.min\_div;  
}  
  
public boolean hasPrevious(Point start\_point) {  
 double x = (start\_point.getX() - this.delta\_x) / AStarPathFinder.this.min\_div;  
 double y = (start\_point.getY() - this.delta\_y) / AStarPathFinder.this.min\_div;  
  
 for(double i = x; i <= x + 1.0D; ++i) {  
 for(double j = y; j <= y + 1.0D; ++j) {  
 if (AStarPathFinder.this.available((int)i, (int)j) && this.previous[(int)i][(int)j] != null) {  
 return true;  
 }  
 }  
 }  
  
 return false;  
}

1. **程序运行结果分析**

初次模拟过程中，实现了对人员流动的基本模拟，但是存在多人卡门或者单人卡门的情况，加入随机因子后，方向随机变化，较小的随机力后，再次模拟人数为50的流动，解决了卡门现象



1. **实验感想与反思**

基本完成单出口人群流动现象，但是可能存在人群直接冲出门的情况，在一定程度上就使得“门”失效，以及人的大小视为均一而定，而且在修改和改进模型时遇到一些困难，没有队人群进行更真实的模拟，模型也需要进一步改善，但是也从该次社会力模型实验中受益颇多