Pakman

This java project represents a general packman game.

Game goals:

The goal of the game is for the user to eat as many fruits and packmans as he can within the given game time.

In addition, the user needs to avoid the ghosts that chasing him, each contact with a ghost cause the user to lose points. Also, the player can't pass throw "wall" (the black boxes on the screen) and each contact with the walls cause the user to lose points.

The game board is actually an image of a Ariel university map.

The game options:

- Upload game:
  The user can upload a csv game file to the game board.
  The file contains the game objects data, each line represent a single object in the game. (G for the gohst, B for the black box and F for the fruits)
  For example-

17	3	Radius	Speed/We	Alt	Lon	Lat	ID	Type
		1	10	0	35.20534	32.10333	10	P
		1	10	0	35.20965	32.10294	11	P
		1	10	0	35.20935	32.10469	12	P
			1	0	35.20821	32.10538	45	F
			1	0	35.20973	32.10536	46	F
			1	0	35.21127	32.10535	47	F
			1	0	35.21133	32.10331	48	F
			1	0	35.21132	32.10283	49	F
			1	0	35.21029	32.10282	50	F
			1	0	35.21092	32.10308	51	F
			1	0	35.2104	32.10339	52	F
			1	0	35.21018	32.10307	53	F
			1	0	35.21093	32.10358	54	F
			1	0	35.21047	32.10458	55	F
			1	0	35.20977	32.10461	56	F
			1	0	35.20387	32.10457	57	F
			1	0	35.20384	32.10421	58	F
			1	0	35.20382	32.1039	59	F
			1	0	35.2041	32.10416	60	F
			1	0	35.20408	32.10438	61	F
1	0	35.2033	32.10541	0	35.20282	32.10215	5	B
1	0	35.21119	32.10522	0	35.20263	32.10489	6	B
1	0	35.2116	32.10547	0	35.21154	32.10225	7	B
1	0	35.21186	32.10251	0	35.20244	32.10222	8	B
1	0	35.21191	32.10435	0	35.20627	32.10418	9	B
1	0	35.20556	32.10542	0	35.20495	32.10364	10	B
1	0	35.20461	32.10393	0	35.20419	32.10217	11	B

  Each location of an object given as a GPS coordinate which represent the actual location of the object in google earth, to display the object on the screen we convert the location to a pixel in the image in the correct spot.

An example of a game display after uploading a file:

- Manual game:
  After uploading the game, the user will be able to place the player on the screen.
  By click on the run button the game will start, now the player will move according to the users clicks on the screen. The movement will be in the direction of the pressed point angel.

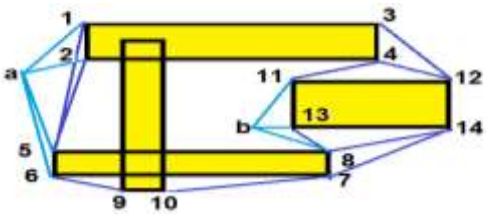
- Automatic game:
  In the Automatic game the player will move without the help of the user, using an algorithm that calculate the path for the player to eat the fruits without passing throw any "walls".

  The algorithm:

  The algorithm based on the Dijkstra algorithm, which calculate the shortest path from given route options.

  The algorithm will calculate all the *possible ways for a player to reach a single fruit.

  *A possible way- a road that doesn't pass through walls. Such as the way from 'a' to '1' in the next image.



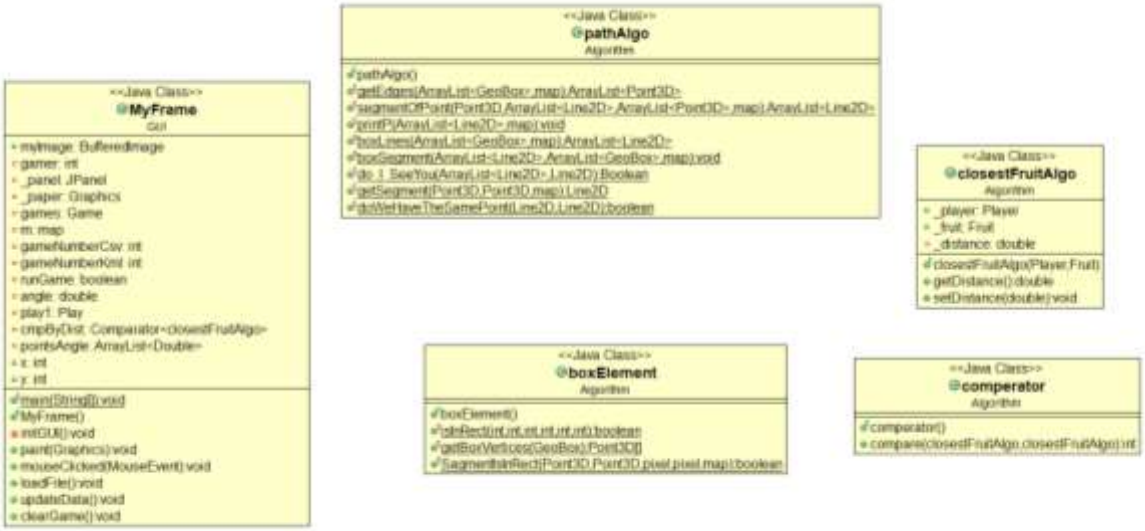
  For calculate all the possible routes the algorithm will use several function as:

  1. Function that returns all the relevant points of the black boxes- the points that the player can reach (and not in a box).

  2. Function that by given two points check if the player can reach it without passing any box in the way.

  3. Function that by a given point, check all the segment to all the reachable points.- this segments will represent all the possible routes from the player to a fruit.

     After we got all the segments we will play the Dijkstra on it and will get the shortest path to a fruit and will move the player by this path using threads.


«Java Class»
MyFrame
Gui
- myImage: BufferedImage
- gamer: int
- _panel: JPanel
- _paper: Graphics
- games: Game
- m: map
- gameNumberCsv: int
- gameNumberKml: int
- runGame: boolean
- angle: double
- play1: Play
- cmpByDist: Comparator<closestFruitAlgo>
- pointsAngle: ArrayList<Double>
- x: int
- y: int
main(String[]):void
MyFrame()
initGUI():void
paint(Graphics):void
mouseClicked(MouseEvent):void
loadFile():void
updateData():void
clearGame():void
«Java Class»
pathAlgo
Algorithm
pathAlgo()
getEdges(ArrayList<GeoBox> map):ArrayList<Point3D>
segmentOfPoint(Point3D,ArrayList<Line2D>,ArrayList<Point3D>,map):ArrayList<Line2D>
printP(ArrayList<Line2D> map):void
boxLines(ArrayList<GeoBox> map):ArrayList<Line2D>
boxSegment(ArrayList<Line2D>,ArrayList<GeoBox>,map):void
do_I_SeeYou(ArrayList<Line2D>,Line2D):Boolean
getSegment(Point3D,Point3D,map):Line2D
doIVeHaveTheSamePoint(Line2D,Line2D):boolean
«Java Class»
closestFruitAlgo
Algorithm
- _player: Player
- _fruit: Fruit
- _distance: double
closestFruitAlgo(Player,Fruit)
getDistance():double
setDistance(double):void
«Java Class»
boxElement
Algorithm
boxElement()
isInRect(int,int,int,int,int,int):boolean
getBoxVertices(GeoBox):Point3D[]
isSegmentInRect(Point3D,Point3D,pixel,pixel,map):boolean
«Java Class»
comperator
Algorithm
comperator()
compare(closestFruitAlgo,closestFruitAlgo):int