Programming in Computer Graphics – Fall 2023

Homework 3 - Crowd Simulation

Pair Submissions Mandatory

Deadline: see canvas

Carefully Follow Instructions Below

In this assignment, we will implement several crowd simulation algorithms, including related data structures such as spatial hashes that we saw in class.

Note that:

- You are strongly encouraged to build on code you submitted in the previous homework.
- External libraries: are not allowed, except the ones mentioned below.

1. Avoidance Force-based Crowd Simulation (distance focused)
   Implement the avoidance force-based crowd simulation method we saw in class. The method is based on the distance between the agents in the current time step. An easy path forward would be to convert the linked Javascript code. Similar to homework 3, the agents should roam around in space, and if they reach one of the edges, they would be transported to the other side as in a “round earth” scenario. Note that the linked Javascript code is not perfect, and might contain bugs. *Can you think of any way to improve it?* Test your code with at least 300 agents.
   For more details, see this book chapter and this paper on “a predictive collision avoidance method for pedestrian simulation”. The paper describes the method in detail, refer to it if you need help with debugging your code.

   Your solution should run with:

   ```
   >python favoid.py
   ```

2. Calculating Frames Per Second

   * With the crowd simulation you implemented in step 1, calculate the number of frames per second. Try to display the frames per second either in the console or on the screen and not flicker, like here:
   * https://jsfiddle.net/tomerwei/7y2ar3nk/81/
   * http://jsfiddle.net/krnlde/u1fL1cs5

   * You should also calculate the overall average number of frames per second till the current point in your simulation.

   Your solution should run with:
   ```
   >python fps.py
   ```

3. Spatial Hash for Collision Detection

   Implement spatial hash for making interagent collision detection more efficient. Follow our class notes. Test your implementation by checking the number of frames per second, which should be a higher number than the non-spatial hash version of your code. This will be true if you have many agents roaming around. The following links would be helpful:
   * Spatial Partitioning from Real Time Collision Detection book (1)
   * Spatial Partition from Game Programming Patterns book (2)
   * Spatial Hashing for fast 2d collision detection blog (3)

   Your solution should run with:
   ```
   >python shash.py
   ```

Source code and Implementation Instructions

- **It is recommended to use code from previous homework.**
- Start early! It might take some time to setup your OpenGL pipeline to compile properly.
- Your solution must include source code in Python..
- OpenGL packages not listed below are not allowed. Contact the instructor if you are not sure.
- External packages not listed below are not allowed. Contact instructor for clarification.
- Python Instructions:
    - **Provide source code, and a requirments.txt file (e.g. here)** you used.
    - **Follow python instructions from the previous homework in regards to setup on compilation.**
    - You may use libraries from previous OpenGL homework, and any other internal python libraries
    - As usual, remember to set up a python virtual environment and then pip install the packages above.

General Instructions

- You are allowed to perform this in **pairs**.
  The file should be named:
  "<your full name1>_<your full name2 (if working with in a pair)>_ex3.zip"
- You can share ideas and discuss general principles with others, but all the code that you submit must be your own work; do not share your homework code with others, and do not look for previous solutions by using a search engine or visiting sites like GitHub. Please consult a TA or the instructor if you have any questions about this policy.
- Work incrementally. Start with a small and simple example and work your way to full functionality.
- Your program should not crash in normal operation as defined by spec above.
- Submit your solution as a zipped file, which is a file that ends with ".zip", to the class website on the class website at canvas