Izel Maras

Last updated 30 June 2014

Force Directed Layout (Java)

The Task

Use Processing to visualize a node-link diagram using forcedirected layout

Design

SPECS

We implementaed mass-spring network where the edges are springs and the nodes have weights (mass).

The animations are not pre-prescribed, but will be dictated by Newtonian physics equations where you need to compute the forces on the springs and the positions, velocities, and accelerations of the nodes.

HOOKE'S AND COULOMB'S LAW

There are two types of forces that we've worked with in this assignment, Hooke's Law and Coulomb's Law.

Hooke's Law describes the forces exerted from springs (f = k1*delta_length),

Coulomb's Law (f = k2/distance) repulses particles away from each other.

Given these forces and the masses of the nodes, we have computed the nodes' accelerations (f = ma, a = f/a).

Given the acceleration, we have computed the velocities (v1 = v0 + at), and the positions (s1 = s0 + vt + ½ at2).

In order for the system to decrease in energy, we also applied "damping" to the nodes

Implementation

PSEUDO CODE (ROUGHLY)

```
a. Parse the input file in the above format and read in the data.

b. Draw each node as a circle (where the area of the circle corresponds to the mass), and a line to represent an edge
between two nodes.

c. Follow the general algorithm below:
    i. Randomly set the positions of the nodes on the screen
    ii. While Total Energy (see additional note section) is greater than some threshold
        1. Loop 1, go through all the nodes in the graph (call one of these nodes A)
            a. Sum up the forces exerted on A from its neighbors
                i. Force 1: from the springs (Hooke's Law)
                ii. Force 2: from repulsion (Coulomb's Law)

        2. Loop 2, go through all the nodes in the graph
            a. Compute the accelerations and velocities of the nodes, remember to apply damping
            b. Update the nodes' positions
            d. Interactions: implement click-and-drag on the nodes such that the user can drag around a node,
            and have the rest of the graph respond to it dynamically.
                i. Note that a node should be highlighted when a mouse hovers over it

e. Part of the challenge for this assignment is to identify the forces that you should apply.
We checked the algorithm in Wikipedia (http://en.wikipedia.org/wiki/Force-based_algorithms_(graph_drawing).

f. Assumptions: you are free to make the assumption that
    i. the graph is connected – meaning that all the nodes belong to the same graph (as opposed to having two or
    more disjointed graphs in the data or the visualization)
    ii. The mass of each node will be at least 1, and the range of possibe values of the mass will be relatively similar
    (i.e., no greater than 10).
```

Notes

TEAM

Skyler Tom - implemented node dragging.
Izel Maras - implemented force and position calculations.
Karl Cronburg - implemented energy and node dragging.

We used the parser from the previous SHF assignment. We all worked together on determining a good framework for Springs and masses / Nodes.