Physically Based Modeling

CS 15-863 Spring 1997 Assignment 4: Tinkertoys

Due Thursday March 20

In this assignment, you'll re-work your mass-and-spring simulator to employ distance constraints in place of (or, if you like, in addition to) springs. You'll implement the constraints using Lagrange multipliers. Rather than hand-coding the whole system, you'll build the constraint matrix on the fly by allowing each constraint to make its own contribution to the global $\mathbf{J}$ and $\dot{\mathbf{J}}$ matrices.

We talked about sparse matrices in class, but you won't need to use them for the assignment. We'll give you C code to do the basic vector/matrix manipulations you need, including solving the linear system. (We'll post source on the web page.)

What to Implement

Using this approach, you can in principle build arbitrary "tinkertoy" structures interactively. However, as before, we're not requiring you to do any interactive construction. You can read a model in from a file, or just wire it into the code. If the latter, do at least the "triangle with a tail" that you did for masses and springs.

You should have gravity and mouse forces as before, but **forget** about collision and contact.

You should implement distance constraints as little structures, analogous to a structure that represents a spring. The structure should point to the pair of particles it influences, and should also point to the three numerical functions that define its behavior: One that computes the *value* of the constraint function $C = |\vec{x}_1 - \vec{x}_2|^2 - r^2$, one that stuffs the derivatives $\partial C / \partial x_1$ and $\partial C / \partial x_2$ into the global J matrix, and another that stuffs their time derivatives into the global $\dot{J}$ matrix.

In order to build the global matrices and vectors, the constraints need to know where to put their derivatives. This is a simple matter of indexing: number all your constraints, and all your particles. The index of the ith constraint is just i, and the indices of the jth particle are $2j$ and $2j+1$, for the x and y components respectively. So, e.g. the derivative of constaint i with respect to the y component of particle j goes into element $i, 2j+1$ of the matrix.

You'll want to have non-movable points, in addition to regular particles, particularly without boundary collisions to keep things on screen.