CSCI 5611 Instr.: Guy

HW 2: Integration Framework

Homework: Fri., Oct 13

Overview. This assignment aims to prepare you for Project 2 by developing a numerical integration framework that supports constraints in the evolution of a system's motion. Simulating systems with constrained dynamics forms the basis of how modern physics engines simulate rigid bodies, ragdolls, ropes, cloth, and fluids.

This is an individual assignment. Each person most complete their own code and report.

For this homework, you must write a numerical simulation of a chain of links using position-based dynamics and then analyze the stability of the resulting simulation.

You may use any programming language you want (besides Python), but you may not import any libraries, modules, or existing code. This means you will likely want to use your own Vec2 functionality (feel free to reuse code from HW1 or in-class). Exceptions: You may use any libraries you need relating to basic math, file input/output, string parsing, and graphing/visualization.

This HW has two parts:

A. Fixed-length Link Simulation

Using a position-based dynamics approach, write a simulation of a 20-node (19-link) chain of nodes, each with a mass of 1, each connected by a fixed rod of length 0.2m (for a total length of 3.8m). Your simulation must use a fixed Δt of 0.05.

Your simulation needs to support two additional parameters: *num_substeps*, which sets the number of substeps within a simulation step used to compute the physics (giving you an effective Δt of less than 0.05), and *num_relaxation_steps*, which sets the number of times to run the constraint solver per substeps. For example, if *num_substeps* = 10 and *num_relaxation_steps* = 10, then you would call 100 constraint solves per fixed Δt of 0.05.

The simulation should always start in the same initial condition with the chain of links perfectly horizontal (all at the same height from the ground) at 5m off the ground. One node of the chain should be pinned perfectly still so that the system of nodes swings back and forth.

This simulation can be in either 2D or 3D.

Submit your code for this simulation in a single zip file. Unlike the Projects, you do not need to submit any video, just the code (and a report PDF for part B).

B. Analysis

Add the following two additional functions to your above simulation:
- The function, *total_length_error()*, should report the sum of the total deviations from the expected fix length of each rod. In other words, *total_length_error()* should be 0 if and only if all rod length constraints are preserved.
- The function, *total_energy()*, should compute the sum of the Kinetic Energy (0.5 * m * v^2) and Potential Energy (m * g * h) for all the nodes in the system. Under ideal conditions, this total energy should remain constant throughout the length of the simulation. In practice, however, you will see the system gain or lose energy due to inaccuracies in your numerical integration.

Use these functions to create the following graphs:

1. For a fixed Δt of 0.05 and no sub-stepping (num_substeps=1), run a simulation for 30s under the following conditions: 1, 10, and 100 relaxation steps. Graph the energy over time each condition on the same graph.

2. For a fixed Δt of 0.05 and using 10 simulation substeps, run a simulation for 30s under the following conditions: 1, 10, and 100 relaxation steps. Graph the energy over time for each condition on the same graph.

3. For a fixed Δt of 0.05 and using 100 simulation substeps, run a simulation for 30s under the following conditions: 1, 10, and 100 relaxation steps. Graph the energy over time for each condition on the same graph.

4. For a fixed Δt of 0.05 and using 10 relaxation steps, run a simulation for 30s under the following conditions: 1, 10, and 100 simulation sub-steps. Graph the energy over time for each condition on the same graph.

5. For a fixed Δt of 0.05 and no sub-stepping (num_substeps=1), run a simulation for 30s under the following conditions: 1, 10, and 100 relaxation steps. Graph the length error over time for each condition on the same graph.

The visuals for graphs do not need to be generated as part of your code. For example, you may wish to save the information to a file and use a program like Excel to create the graphs.

Finally, write a short, ~1-page report that includes:
- A screenshot of your simulation running
- Images of all five above graphs
- A description of the trends observed across the various experiments
- An analysis of the combination of parameters that yielded the best results.

The report should be a single PDF, approximately 1 page long, and written in full English sentences (no bullet points).

Submission [100 points]

Submit 2 separate files: code.zip, report.pdf

1. The file, *code.zip,* should have both your code for computing the simulation, and your code for visualizing the results. [note: you do not need to use processing, but you must supply all files needed to compile your code]

2. The file *report.pdf* should contain your report as a single PDF containing all the graphs from part B.

Unlike the project submissions, you do not need to include any videos.

Extra Credit [up to 10 points]

Instead of doing the analysis on a 20-node link in a line, run your simulation on a 2D mesh containing at least 20 nodes (e.g., a 5x5 mesh). This will need to be a 2D simulation. Also, instead of pinning a single node in place, pin an entire row of nodes in place. This will let the mesh of nodes swing back and forth freely.