CPSC 526: Computer Animation
Assignment 4

Due Fri Oct 27, 2017 (at any hour)

1. (9 points) In this question you will be developing a 2D dynamics simulation of single-link pendulum and a multi-link pendulum using maximal coordinates, hinge constraints, and explicit Euler integration. Each link of your pendulum should be 1 m long and have a mass of 1 kg. You can model it as a line segment or a narrow rectangular block. Use a default time step of 0.01 s. Email a link to the code to Michiel by the due date. Sign up for a time-slot on a Google doc (URL to be posted to Piazza) to briefly demonstrate / discuss your assignment. A specific set of intermediate steps and results is given below. You are free to code this in your favourite language of choice. I would recommend any of: Python (with numpy, matplotlib, PyOpenGL), C++ (with eigen, OpenGL), Matlab / Octave, or even Javascript (with three.js, math.js). However, for this assignment, you should not use existing dynamics or physics-based simulation libraries. Feel free to use Piazza to solicit help from others (and the instructor). Collaborate as much as you like with regard to setup issues for whatever coding framework you are using. Discussing the problem details with others is also fine, although the code should be your own in the end, and you should take the time to briefly document this in the final email that you send to Michiel. I.e., "I discussed aspects A and B of this problem with person C."
   (a) (2 points) Consider a one-link pendulum with no friction. Write out the full 2D equations of motion using maximal coordinates, i.e., with equations for the unconstrained motion of the pendulum link and additional equations for the hinge constraint. Explicitly write out all terms, but structure it in the block form used in class. Note that in the 2D case, the inertia matrix is simply a scalar, and that some terms for the equations of motion are simply zero. Search for "moments of inertia" to find the moments of inertia for any kind of common geometry. Note that in class we developed the Euler equations for rotation about center-of-mass of each link, so you should look for moments of inertia about the center.
   (b) (2 points) Develop a simulation of the one-link no-friction pendulum, dropped from an angle of 45°. Animate the resulting pendulum. Also track the sum of the kinetic and potential energy, and print or illustrate this during the motion, or plot it afterwards. A compelling dynamic illustration would consist of a two stacked bars, of different colours, that respectively represent the kinetic energy and the potential energy. Because of the explicit Euler integration, you can expect that the total energy will slowly increase over time. First implement your method without constraint stabilization, and then add a form of constraint stabilization. Simple Baumgarte stabilization is fine, i.e., what we covered in class.
   (c) (1 point) Add frictional damping to the above simulation, i.e., $\tau - k_d\omega$.
   (d) (2 points) Repeat part (i) for a four-link pendulum, although this time it is not necessary to write out the full equations of motion (although your code will still need to implement them!).

(e) (1 point) Add frictional damping to the four link pendulum.

(f) (1 point) Add a ground plane below the four link pendulum and use a penalty force method to apply an external force to the end of the bottom link such that it does not penetrate too far into the ground plane and instead slides along the surface. Use a smaller time step as needed. The ground plane can be frictionless or you can also experiment with adding friction. A penalty method computes the ground contact force by computing a virtual spring-and-damper force according to: $F_y = k_p(y_0 - y) - k_d\dot{y}$, where F_y is the normal component of the ground reaction force and should always be constrained to be positive, i.e., it should never "pull" a point down towards the ground; y_0 is the height of the ground, and $y, \dot{y}$ represent the height and vertical speed of the point(s) on the pendulum link where we wish to apply the penalty-method contact force.

(g) (2 points) Optional: improve on the simulation in an interesting way of your own choosing.

(h) (3 points) Optional: implement a 3D pendulum with point-to-point constraints between the links. You may wish to add some damping to the system.