Physikbasierte Modellierung und Simulation

Assignment 5

Present your solutions for this sheet in the exercise on Wednesday, December 4, 2013.

In this exercise, you will simulate a double pendulum in two different ways: using a mass-spring system that tries to keep the length of the pendulum close to the desired length, and using an Euler-Lagrange approach with exact constraints.

5.1 Mass-spring system (30 points)

In main.cpp, create a mass-spring system ms_system for the double pendulum. You will need a fixed particle and two moving particles. The particles should have a mass of 1 kg and be located at:

\[ \vec{p}_1 = (\sin 120^\circ, -\cos 120^\circ)^T \cdot 1\text{ m} \]
\[ \vec{p}_2 = \vec{p}_1 + (\sin 60^\circ, -\cos 60^\circ)^T \cdot 1\text{ m} \]

The springs, correspondingly, should have a length of 1 m. Store the particles and springs in the vectors particles and springs, respectively.

You will notice that Particle is now a template; the template parameter names the type of the coordinates stored in the particle. For the mass-spring system, you will need Particle<Length3D>.

Finally, create a solver ms_solver for the system. Note that the solver now also requires a template parameter, namely the type of particle being used.

5.2 Euler-Lagrange solution (30 points)

Ideally, the springs you used for modeling the pendulum would be infinitely stiff, so that their lengths never change. Unfortunately, this is numerically very unstable. A different approach is to transform the problem into a system of generalized coordinates with fewer degrees of freedom, so that the length constraint is always implicitly fulfilled. This is called Lagrangian mechanics.

In the case of the double pendulum, the only degrees of freedom are the two angles \( \theta_1 \) and \( \theta_2 \). The actual three-dimensional coordinates of the particles can be computed from these two angles and the length of
the pendulum. The difficulty lies in transforming the (so far rather simple) equations of motion of the pendulum into the system of generalized coordinates to get differential equations for the angles $\theta_1$ and $\theta_2$ and the corresponding angular velocities $\omega_1$ and $\omega_2$ (you don’t need to do this yourself, here they are):

$$
\dot{\theta}_1 = \omega_1 \\
\dot{\omega}_1 = \frac{m_2 l_1 \omega_1^2 \sin \Delta \cos \Delta + m_2 g \sin \theta_2 \cos \Delta + m_2 l_2 \omega_2^2 \sin \Delta - (m_1 + m_2) g \sin \theta_1}{(m_1 + m_2) l_1 - m_2 l_1 \cos^2 \Delta} \\
\dot{\theta}_2 = \omega_2 \\
\dot{\omega}_2 = -\frac{m_2 l_2 \omega_2^2 \sin \Delta \cos \Delta + (m_1 + m_2) (g \sin \theta_1 \cos \Delta - l_1 \omega_1^2 \sin \Delta - g \sin \theta_2)}{(m_1 + m_2) l_2 - m_2 l_2 \cos^2 \Delta}
$$

Here, $\Delta = \theta_2 - \theta_1$, $m_1$ and $m_2$ are the masses of the particles, $l_1$ and $l_2$ the lengths of the “springs”, and $g = 9.81 \text{ m/s}^2$ is the gravitational acceleration. For a more in-depth explanation, see [http://scienceworld.wolfram.com/physics/DoublePendulum.html](http://scienceworld.wolfram.com/physics/DoublePendulum.html).

Implement `computeAccelerations()` in `DoublePendulum.cpp` using these equations.

5.3 Experiments (40 points)

- Compare both solutions for a few hundred iterations. What do you notice?
- How do both solutions change when you change the step size?
- How does the mass-spring solution change when you increase or decrease the stiffness? What if you introduce damping?
- By increasing `num_el_systems`, you can create several Euler-Lagrange solutions with slightly different initial conditions. What do you observe?