CSC 2521 Computer Graphics, Fall 2006: Assignment 1

Due Sep 30, in class

Written

Questions from the textbook: 2-29, 3-1, 7-4, 7-6.

Notes: I recommend consulting the text and reading relevant sections in order to do these problems. For 3-1, see the text for definitions of natural frequency and period. For 7-6, the moment of inertia of the hoop is $I = mR^2$. For 9-10, see example 9.4, p. 343.

Programming

In this assignment, you will implement a numerical simulation engine for second order ODEs, and test it on the plane pendulum problem.

Plane Pendulum (page 155 and page 232)

The plane pendulum (pages 155-160, 232) consists of a particle with mass m suspended by a rigid rod with length ℓ from a fixed point, at angle θ. The forces on the rod are gravity, which acts downward, and a *constraint force* that keeps the particle attached to the rod. The motion of the pendulum is derived in the text:

$$\ddot{\theta} + \omega_0^2 \sin\theta = 0 \tag{1}$$

where $\omega_0^2 = g/\ell$, and g is the gravitational constant. The kinetic and potential energies are given by:

$$\begin{aligned} T &= \frac{1}{2} m\ell^2\dot{\theta}^2 && (2)\\ U &= mg\ell(1-\cos(\theta)) && (3) \end{aligned}$$

If we add friction at the joint $\tau = -\rho\dot{\theta}$, then the motion becomes:

$$\ddot{\theta} + c\dot{\theta} + \omega_0^2 \sin\theta = 0 \tag{4}$$

where $c = \rho/(m\ell^2)$.

Ordinary Differential Equations

The general form of a first-order ordinary differential equation (ODE) is:

$$\dot{\mathbf{x}} = f(\mathbf{x}, t) \tag{5}$$

where $\mathbf{x}$ is the state variable and f is a function that determines the system's dynamics. The state is a function of time: $\mathbf{x}(t)$, but we usually leave the dependence implicit. There are several algorithms for solving first-order ODEs that we will explore here.

Because physical problems are normally defined in terms of accelerations, they are typically second-order, e.g.:

$$\ddot{\mathbf{y}} + g(\dot{\mathbf{y}}) + h(\mathbf{y}) = 0 \tag{6}$$

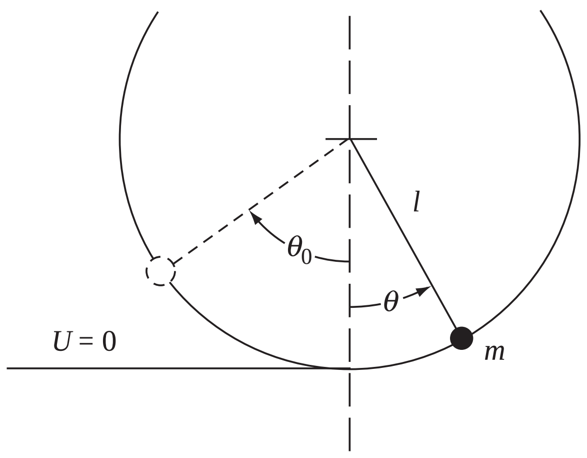

Figure 1: The plane pendulum from Thornton and Marion.

such as the plane pendulum above, where $\mathbf{y} \equiv \theta$. To convert a second-order ODE to first order, we treat the velocity $\mathbf{v} \equiv \dot{\mathbf{y}}$ as an explicit state variable:

$$\mathbf{x} \equiv \begin{bmatrix} \mathbf{y} \\ \mathbf{v} \end{bmatrix} \tag{7}$$

and use the equations:

$$\dot{\mathbf{y}} = \mathbf{v} \tag{8}$$

$$\dot{\mathbf{v}} = -g(\mathbf{v}) - h(\mathbf{y}) \tag{9}$$

This is now a first-order ODE in the form of Equation 5, and can now be solved by a first-order method.

Numerical Solvers

Given a first-order ODE of the form in Equation 5, and initial conditions ($\mathbf{x}(0) = \mathbf{x}_0$), the *initial value problem* is to solve for the evolution of the state over time. We discretize time using a step-size h, and then solve for discrete estimates: $\mathbf{x}_0 = \mathbf{x}(0), \mathbf{x}_1 \approx \mathbf{x}(h), \mathbf{x}_2 \approx \mathbf{x}(2h), ...\mathbf{x}_i \approx \mathbf{x}(ih)$.

Euler's Method. The simplest and easiest first-order ODE solver is Euler integration. We can derive Euler integration by Taylor expansion of Equation 5:

$$\mathbf{x}(t + h) = \mathbf{x}(t) + hf(\mathbf{x}(t)) + O(h^2) \tag{10}$$

(assuming that f doesn't explicitly depend on t). Throwing out higher-order terms and discretizing gives:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + hf(\mathbf{x}_i) \tag{11}$$

It can be shown that the error (i.e., $|x(ht) - x_i|$ in the 1D case) is proportional to h. It turns out that higher-order methods can get much better accuracy without a corresponding performance cost.

The Improved Euler Method. The exact relation between one time-step and the next is:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + \int_t^{t+h} f(\mathbf{x}(\tau))d\tau \tag{12}$$

Hence, Euler's method can be viewed as approximating the integral by:

$$\int_t^{t+h} f(\mathbf{x}(\tau))d\tau \approx \int_t^{t+h} f(\mathbf{x}(t))d\tau = hf(\mathbf{x}_i) \tag{13}$$

The trapezoidal rule gives a better approximation:

$$\int_t^{t+h} f(\mathbf{x}(\tau))d\tau \approx \frac{h}{2}(f(\mathbf{x}_i) + f(\mathbf{x}_{i+1})) \tag{14}$$

However, since we don't yet know $\mathbf{x}_{i+1}$, we also approximate it using Equation 11. Putting it all together gives the following update rule, called *the improved Euler method*:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{h}{2}(f(\mathbf{x}_i) + f(\mathbf{x}_i + hf(\mathbf{x}_i))) \tag{15}$$

It can be shown that the error of this method is proportional to h^2, which is better than the h error of Euler's method (assuming $h < 1$).

Runge-Kutte integration. These same ideas can be extended to higher orders, in the form of Runge-Kutte integration. Higher order methods generally give a better accuracy/time tradeoff. Here we will consider a *Fourth-Order Runge-Kutte method:*

$$\begin{aligned}
\mathbf{k}_1 &= hf(\mathbf{x}_i) && (16)\\
\mathbf{k}_2 &= hf(\mathbf{x}_i + \mathbf{k}_1/2) && (17)\\
\mathbf{k}_3 &= hf(\mathbf{x}_i + \mathbf{k}_2/2) && (18)\\
\mathbf{k}_4 &= hf(\mathbf{x}_i + \mathbf{k}_3) && (19)\\
\mathbf{x}_{i+1} &= \mathbf{x}_i + (\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4)/6 && (20)
\end{aligned}$$

Problems

Your task is to implement the plane pendulum model and the three integration methods above, and compare them. Implement the integration methods as general first-order solvers, and then supply the equations for the plane pendulum. For each solver, and each set of parameter values, generate a phase plot, and a plot of the energies (see example). For each problem, generate a needle plot (e.g., using MATLAB's `quiver` command), showing $\ddot{\mathbf{x}}$ for each point in phase space. I will provide some MATLAB skeleton code to get you started.

The values to use for the trials are:

1. $\theta_0 = \pi/2, \dot{\theta}_0 = 0, g = 9.8m/s^2, \ell = 5m, \rho = 0, m = 1$
2. $\theta_0 = \pi/2, \dot{\theta}_0 = -2, g = 9.8m/s^2, \ell = 5m, \rho = 0, m = 1$
3. $\theta_0 = \pi/2, \dot{\theta}_0 = 0, g = 9.8m/s^2, \ell = 5m, \rho = 0.5, m = 1$
4. $\theta_0 = \pi/2, \dot{\theta}_0 = 0, g = 9.8m/s^2, \ell = 5m, \rho = 2, m = 1$

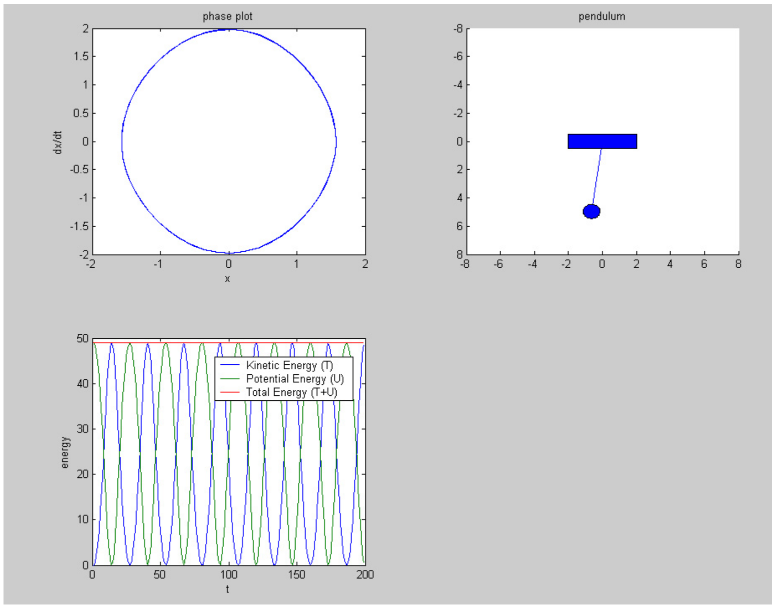

Figure 2: Sample output from an ODE solver.

Describe qualitatively the behavior of the system in each example.

Next, for the first case above, run each solver using a range of step sizes h: $[.001s, .01s, .05s, .1s, .2s, .3s]$, for 20 seconds of simulation time. Note that this means that some simulations will have more steps than others. For each run, compute the absolute value of the difference between the initial total energy (T+U) and the total energy at the final step, and also keep track of computation time (or flops) required to compute the sequence. Why is the difference in a total energy a reasonable measure of error, and what might we do when $\rho \neq 0$? Plot the total error for the three methods, both as a function of step-size, and as a function of computation time.

From these experiments, what empirical conclusions can you draw about the three methods?