Reference List for 15-464 / 15-664 February 15, 2023

These course notes are an excellent introduction to writing a physically based simulator:

Andrew Witkin and David Baraff, "Physically Based Modeling: Principles and Practice," Siggraph '97 Course notes. http://www.cs.cmu.edu/~baraff/sigcourse/

We talked about how to simulate particles and briefly how to extend that simulation to rigid bodies. A photo of my handwritten notes is included at the end of this document.

The integrator you choose can determine the system's behavior (e.g., gaining, losing, or retaining energy). We talked about a variety of integrators to compare, including Euler, Implicit Euler, Symplectic Euler, Verlet, and RK4. (See notes at the bottom of this document for details.) It is possible to develop a fully implicit integrator for cloth, and that is the subject of this paper:

Baraff D, Witkin A. Large steps in cloth simulation. In Proceedings of the 25th annual conference on Computer graphics and interactive techniques 1998 Jul 24 (pp. 43-54). ACM. http://dl.acm.org/citation.cfm?id=280821

We did not really take the time to get into RK4. Personally I find the image on the Wikipedia page to be intuitive for understanding it:
https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods#/media/File:Runge-Kutta_slopes.svg
https://en.wikipedia.org/wiki/Runge%E2%80%93Kutta_methods

This paper has a good practical discussion about different integrators and discusses the design decisions behind Maya nCloth, nParticle, etc.

Stam, Jos. "Nucleus: Towards a unified dynamics solver for computer graphics." In *Computer-Aided Design and Computer Graphics, 2009. CAD/Graphics' 09. 11th IEEE International Conference on*, pp. 1-11. IEEE, 2009. http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5246818

We talked about this paper, which talks about particle simulation with constraints and Verlet integration in the context of simulating rag doll characters for the game Hitman.

Jakobsen, Thomas. "Advanced character physics." In Game Developers Conference, pp. 383-401. 2001. http://www.gotoandplay.it/_articles/2005/08/advCharPhysics.php

I also showed some videos from a constraint based cloth simulation system written by a CMU MS student. The writeup can be found here: http://www.cs.cmu.edu/~ytoh/stickyfingers.pdf

Videos can be seen here: http://www.kentoh.com/publications/

Finally, we took a quick look at the following paper for an overview of point based methods in general for simulation.

Macklin, Miles, Matthias Müller, Nuttapong Chentanez, and Tae-Yong Kim. "Unified particle physics for real-time applications." *ACM Transactions on Graphics (TOG)* 33, no. 4 (2014): 153. https://dl.acm.org/citation.cfm?id=2601152

If you want to think about stability of mass spring simulations and learn more about how to set them up to behave properly, you may want to have a quick look at the following paper, which contains a clear writeup about the spring mass system's behavior and a reminder for how to solve those differential equations. Reading this paper may help you to set damping parameters in a correct proportion to how you set stiffnesses.

Allen, Brian F., and Petros Faloutsos. "Misconceptions of PD control in animation." In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 231-234. Eurographics Association, 2012. http://dl.acm.org/citation.cfm?id=2422389

You may also be interested in this SIGGRAPH course, which seems to be available online in its entirety:

Bargteil, Adam W., Tamar Shinar, and Paul G. Kry. "An introduction to physics-based animation." In *SIGGRAPH Asia 2020 Courses*, pp. 1-57. 2019. https://dl.acm.org/doi/10.1145/3305366.3328050

This 3-hour SIGGRAPH course, which also seems to be available online, has a great discussion of modern techniques for handling contact in Computer Graphics. Page 160 has a tutorial on programming a simulator with frictional contact that would be a fine MiniProject2. If you go this route, keep me posted on how easy or difficult it was!

Andrews, Sheldon, Kenny Erleben, and Zachary Ferguson. "Contact and friction simulation for computer graphics." In *ACM SIGGRAPH 2022 Courses*, pp. 1-172. 2022. https://siggraphcontact.github.io/ https://dl.acm.org/doi/10.1145/3532720.3535640

Some Simulation References

Although we spent a good amount of time looking at the nuts and bolts of simulation systems today, you may not be in the situation of writing your own, except for a class project or for specific research purposes. Fortunately, there are many good simulation engines out there. Here are some references to get you started. There are other references in the MiniProject2 handout, and you can always see what simulation plugins and built-in capabilities are available for your favorite animation environment (e.g., Unity, Houdini…).

NVIDIA FleX: https://developer.nvidia.com/flex

Bullet Physics Library http://bulletphysics.org/wordpress/

Box2D http://box2d.org/

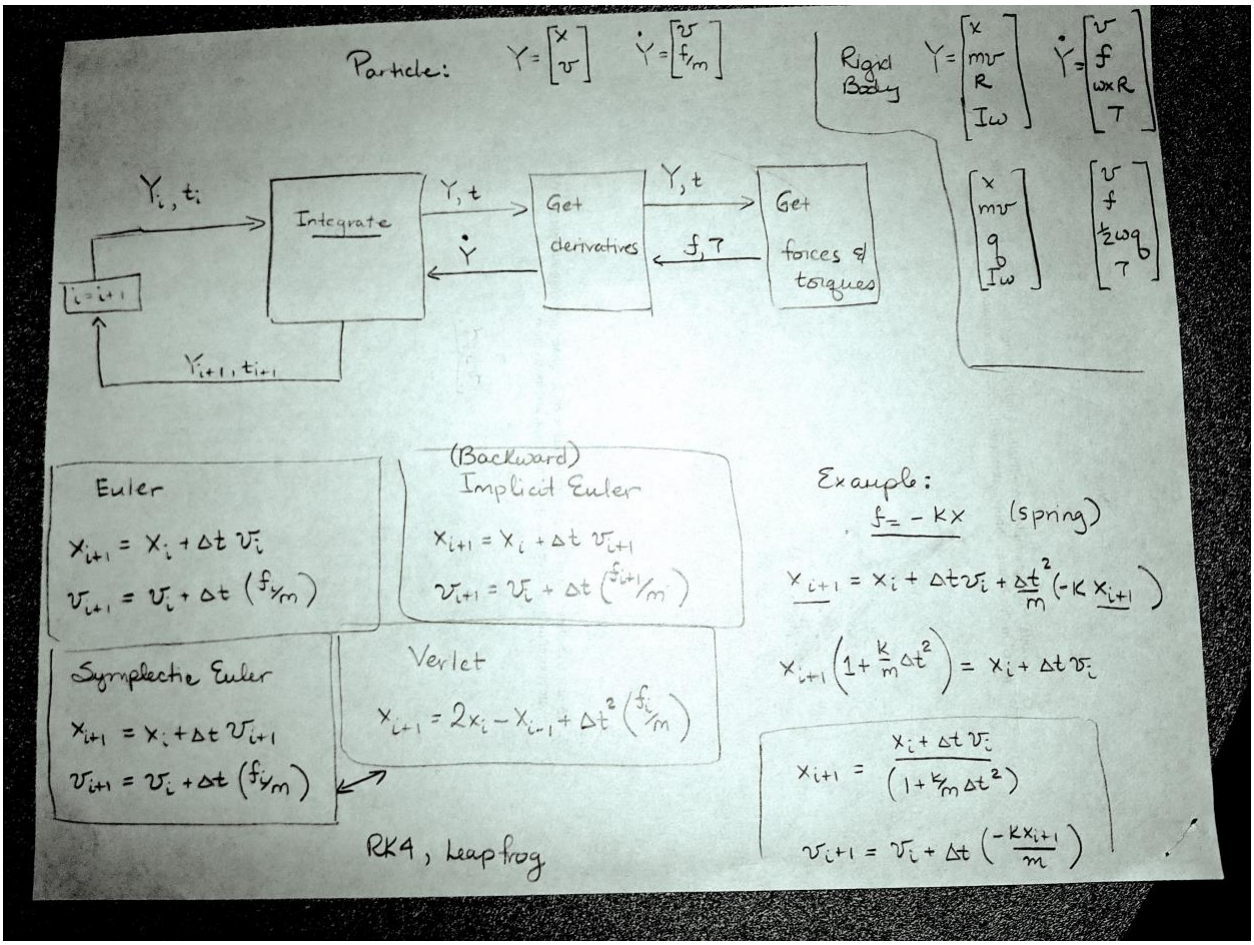
Emanuel Todorov's MuJoCo http://www.mujoco.org/

Gazebo http://gazebosim.org/

Karen Liu's DART http://dartsim.github.io/

Open Dynamics Engine http://www.ode.org/

Also check out this SIGGRAPH 2011 course: http://bulletphysics.org/siggraph2011/


Particle: $Y=\begin{bmatrix} x \\ v \end{bmatrix}$ $\dot{Y}=\begin{bmatrix} v \\ f/m \end{bmatrix}$
Rigid Body $Y=\begin{bmatrix} x \\ mv \\ R \\ I\omega \end{bmatrix}$ $\dot{Y}=\begin{bmatrix} v \\ f \\ \omega \times R \\ \tau \end{bmatrix}$
Y_i, t_i
Integrate
Y, t
$\dot{Y}$
Get derivatives
Y, t
f, τ
Get forces & torques
$i = i+1$
Y_{i+1}, t_{i+1}
$\begin{bmatrix} x \\ mv \\ q \\ I\omega \end{bmatrix}$ $\begin{bmatrix} v \\ f \\ \frac{1}{2}\omega q \\ \tau \end{bmatrix}$
Euler
$x_{i+1} = x_i + \Delta t\, v_i$
$v_{i+1} = v_i + \Delta t\,(f_i/m)$
(Backward) Implicit Euler
$x_{i+1} = x_i + \Delta t\, v_{i+1}$
$v_{i+1} = v_i + \Delta t\,(f_{i+1}/m)$
Example:
$f = -kx$ (spring)
$x_{i+1} = x_i + \Delta t v_i + \frac{\Delta t^2}{m}(-k x_{i+1})$
$x_{i+1}\left(1 + \frac{k}{m}\Delta t^2\right) = x_i + \Delta t\, v_i$
Symplectic Euler
$x_{i+1} = x_i + \Delta t\, v_{i+1}$
$v_{i+1} = v_i + \Delta t\,(f_i/m)$
Verlet
$x_{i+1} = 2x_i - x_{i-1} + \Delta t^2(f_i/m)$
$x_{i+1} = \frac{x_i + \Delta t\, v_i}{(1 + k/m\,\Delta t^2)}$
$v_{i+1} = v_i + \Delta t\left(\frac{-k x_{i+1}}{m}\right)$
RK4, Leapfrog