Visual Interactive Simulation

Review of physics, differential equations and numeric integration

martin.servin@physics.umu.se

November 10, 2011

Presentation

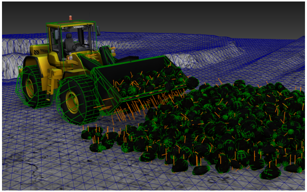
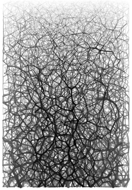
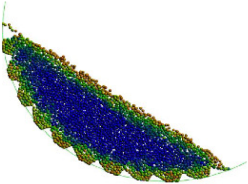
Martin Servin

- Universitetslektor, Dep. Physics, Umeå University
- PhD in theoretical physics, non-linear, relativity & plasma
- Modeling and simulation of complex mechanical systems
- Visual interactive simulations
- Virtual prototyping: control, design, optimization
- UMIT Research Lab
- Algoryx Simulation AB

Example: granular matter

Motivation

- Simulation important tool in R&D involving granular matter
- Major challenge: reduce computing time

Example: Machine and 1K rigid bodies in realtime
Example: 80K spheres in 1:100 of realtime

Todays lecture

- The simulation algorithm
- Basic classical mechanics for particles
- Differential equations and numerical time integration
- Lab 1: particle system

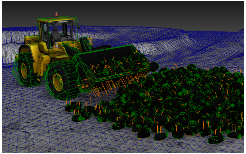
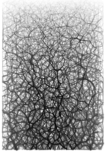
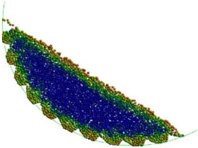
The simulation algorithm

definitions
initialization (t_0)
while running **do**
  collision detection and collision response
  compute forces and constraints
  stepforward:
    solve to update state variables ($t_n \rightarrow t_{n+1}$)
    update derived quantities
  simulation I/O
end while
post-processing

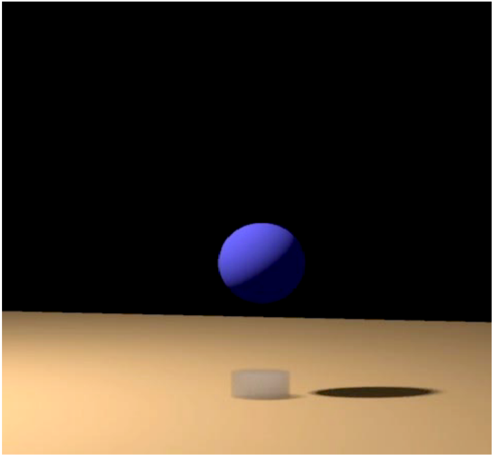
The simulation algorithm

Questions

- How are physical objects represented?
- How do they evolve with time?
- How to perform time stepping?
- What are the forces? Constraints?
- What is the role of energy, momentum etc in simulations?

Particles



Particle representation

Struct representation:

```
particle(i).x        % state variable, vec3
particle(i).v        % state variable, vec3
particle(i).f        % computed quantity, vec3
particle(i).m        % constant, scalar
E_tot                % derived quantity, scalar
```

Particle index $i = 1, 2, \ldots, N_p$.

Particle representation

Vector/matrix representation $\mathbf{x}_{(i)} = \left[x_1^{(i)}, x_2^{(i)}, x_3^{(i)}\right]^T$:

$$x = \begin{bmatrix} \mathbf{x}_{(1)} \\ \mathbf{x}_{(2)} \\ \vdots \end{bmatrix} = \begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \\ x_3^{(1)} \\ x_1^{(2)} \\ x_2^{(2)} \\ x_3^{(2)} \\ \vdots \end{bmatrix}, \quad v = \begin{bmatrix} \mathbf{v}_{(1)} \\ \mathbf{v}_{(2)} \\ \vdots \end{bmatrix}, \quad f = \begin{bmatrix} \mathbf{f}_{(1)} \\ \mathbf{f}_{(2)} \\ \vdots \end{bmatrix}$$

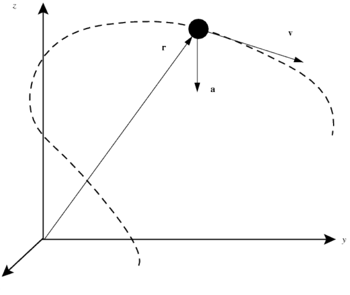
$$M = \begin{pmatrix} m_{(1)} & 0 & 0 & 0 & 0 & 0 & \dots \\ 0 & m_{(1)} & 0 & 0 & 0 & 0 & \dots \\ 0 & 0 & m_{(1)} & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & m_{(2)} & 0 & 0 & \dots \\ 0 & 0 & 0 & 0 & m_{2)} & 0 & \dots \\ 0 & 0 & 0 & 0 & 0 & m_{(2)} & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

$$E_{tot} = \frac{1}{2} v^T M v + \dots$$

Trajectories

Position, velocity, accleration vector are function of time

$$
\begin{aligned}
\mathbf{x}(t) &= [x_1(t), x_2(t), x_3(t)]^T \\
\mathbf{v}(t) &= [v_1(t), v_2(t), v_3(t)]^T = \dot{\mathbf{x}} = \frac{d\mathbf{x}(t)}{dt} \\
\mathbf{a}(t) &= [a_1(t), a_2(t), a_3(t)]^T = \dot{\mathbf{v}} = \frac{d\mathbf{v}(t)}{dt} = \ddot{\mathbf{x}} = \frac{d^2\mathbf{x}(t)}{dt^2}
\end{aligned}
$$



Acceleration

Can be split in tangential and normal components

$$\mathbf{a} = \mathbf{a} = a_t \mathbf{t} + a_n \mathbf{n}$$

where $\mathbf{v} = v\mathbf{t}$ and $\mathbf{t}^T \mathbf{n} = 0$

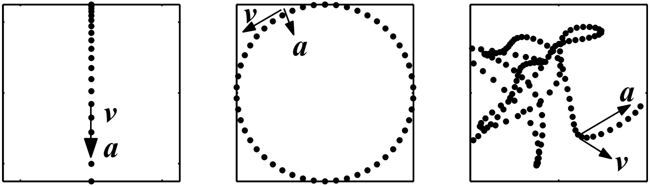


Figure: Linear, circular and irregular motion

But what causes the acceleration?

Newton's laws of motion

1. Every body remains at rest or in uniform motion unless acted on by a force $\mathbf{f}$. The condition $\mathbf{f} = 0$ thus implies a constant velocity $\mathbf{v}$ and constant momentum $\mathbf{p} = m\mathbf{v}$.

2. Application of a force alters the momentum according to the relation
$$\mathbf{f} = \dot{\mathbf{p}}$$

3. To each action, there is an equal and opposite reaction. Thus if $\mathbf{f}$ is a force exerted on body 1 by body 2, then body 1 exerts force $-\mathbf{f}$ on body 2 that is equal in magnitude and opposite in direction.

Time stepping

Newton's second law, $f = m\dot{v}$, is a system of *ordinary differential equations*(ODE)

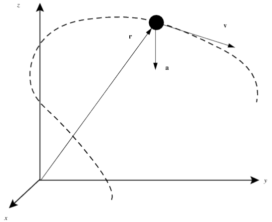
$$\dot{Y} = F(Y)$$

with state vector $Y = (x^T, v^T)$ and $\dot{Y} = (v^T, m^{-1}f^T)$. Time stepping is numerical integration of the ODE

$$Y(t_0) \to Y(t) = \texttt{solveODE}(Y(t_0), F)$$

Because of user interaction and dynamic contacts it is not possible to simply apply standard solvers, e.g., RKF45.

Time discretization



Fix time step h

$$t_{n+1} = t_n + h$$

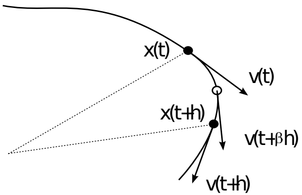
$$\begin{aligned} t &\rightarrow \{t_0, t_1, t_2, \ldots, t_n, t_{n+1}, \ldots\} \\ \mathbf{x}(t) &\rightarrow \{\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n, \mathbf{x}_{n+1}, \ldots\} \\ \mathbf{v}(t) &\rightarrow \{\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n, \mathbf{v}_{n+1}, \ldots\} \end{aligned}$$

Euler discretization

Observe that by the definition of derivative

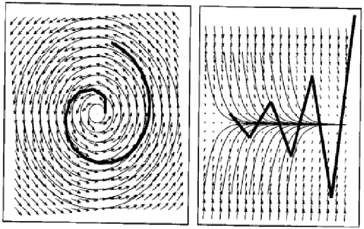
$$\frac{\mathbf{v}(t+h)-\mathbf{v}(t)}{h} \approx m^{-1}\mathbf{f}(t+\alpha h) \quad \rightarrow \quad \frac{d\mathbf{v}(t)}{dt} = m^{-1}\mathbf{f}(t)$$
$$\frac{\mathbf{x}(t+h)-\mathbf{x}(t)}{h} \approx \mathbf{v}(t+\beta h) \quad \rightarrow \quad \frac{d\mathbf{x}(t)}{dt} = \mathbf{v}(t)$$

in the limit of $h \rightarrow 0$ for $0 \leqslant \alpha, \beta \leqslant 1$



Numerical errors and stability

- Taylor expansion $Y(t+h) = Y(t) + hY'(t) + \mathcal{O}(h^2 Y'')$
- Error decrease with time-step or higher-order integrator
- Numerical instability = accumulating numerical errors
- Exponential growth of error = clearly unphysical
- Finer time-step decreases the growth rate
- Stable integrators = globally bounded error



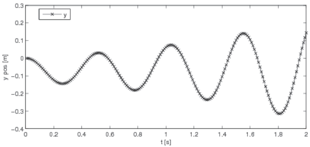
Explicit Euler ($\alpha = \beta = 0$)

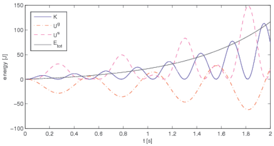
$$\mathbf{v}_{n+1} = \mathbf{v}_n + h m^{-1} \mathbf{f}_n$$
$$\mathbf{x}_{n+1} = \mathbf{x}_n + h \mathbf{v}_n$$

- Also known as Forward Euler
- **Simple to compute!**
- **Unconditionally numerical unstable!**

Explicit Euler ($\alpha = \beta = 0$)

Simulation of particle in spring $f = -k_s(|x| - L) - mg$





$m = 20\text{kg},\ L = 1\text{m},\ k_s = 3\text{kN/m},\ g = 9.81\text{m/s}^2,\ h = 0.01\text{s}$

- **Exponential increase in amplitude!**

Implicit Euler ($\alpha = \beta = 1$)

$$\mathbf{v}_{n+1} = \mathbf{v}_n + h m^{-1} \mathbf{f}_{n+1}$$
$$\mathbf{x}_{n+1} = \mathbf{x}_n + h \mathbf{v}_{n+1}$$

- Also known as Backwards Euler
- Note that $\mathbf{f}_{n+1} = \mathbf{f}(\mathbf{x}_{n+1}, \mathbf{v}_{n+1})$
- **Nonlinear equation - computationally expensive!**
- **Good numerical stability!**

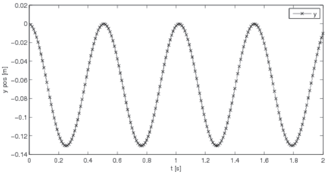
Semi-implicit Euler ($\alpha = 0, \beta = 1$)

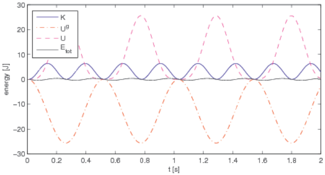
$$\mathbf{v}_{n+1} = \mathbf{v}_n + hm^{-1}\mathbf{f}_n$$
$$\mathbf{x}_{n+1} = \mathbf{x}_n + h\mathbf{v}_{n+1}$$

- Also known as Symplectic Euler
- **Simple to compute!**
- **Stable for** $h < \tau$ - shortest time period of $\mathbf{f}$
- For spring forces $\tau = \omega^{-1} = \sqrt{m/k}$

Semi-implicit Euler ($\alpha = 0, \beta = 1$)

Simulation of particle in spring $f = -k_s(|x| - L) - mg$



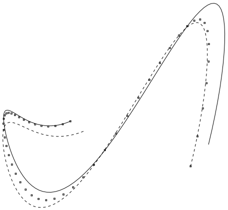


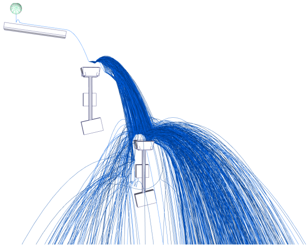
$m = 20\text{kg},\ L = 1\text{m},\ k_s = 3\text{kN/m},\ g = 9.81\text{m/s}^2,\ h = 0.01\text{s}$

- Bounded amplitude
- Bounded total energy
- Conversion between kinetic and potential energy

Accuracy

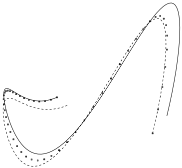
- There time stepping schemes with higher accuracy, e.g., mid-point, Newmark, Adams-Bashforth
- Numerically costly and don't mix well with non-smooth dynamics, e.g., impacts, frictional stick-slip, user interaction, ...
- What type and what level of accuracy are we after?

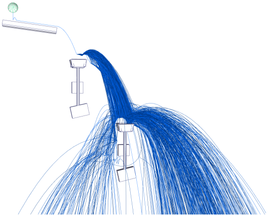




Accuracy

- Numerical trajectory typically deviates increasingly from exact trajectory with time, $|x(t_n) - x_n| \sim e^{\gamma t}$.
- Uncertainties in initial conditions, material properties, geometric shape
- For each exact trajectory there is a number of **shadowing trajectories*** that all are a good approximations for some amount of time





**Computer simulations, exact trajectories, and...*, Hayes, AJP, Vol 72 (2004).

Accuracy

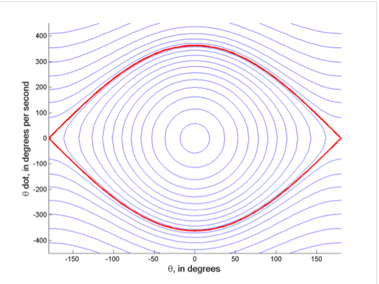
- Many-Worlds Browsing for Control of Multibody Dynamics, D. Twigg and D. James, SIGGRAPH 2007
- http://www.youtube.com/watch?v=dAjYi4ePHmQ

Accuracy

- Alternative measure: preservation of global invariants
- e.g, total energy, linear momentum, angular momentum

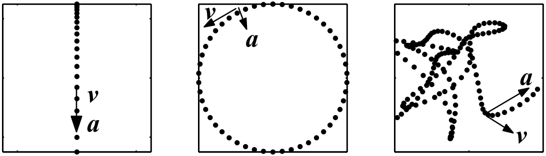
$$\left|\frac{E_n - E_0}{E_0}\right| < \varepsilon$$

- Discrete variational (geometric) integrators preserve momentum exactly and are symplectic
- Semi-implicit Euler = symplectic Euler



Forces

- Recall change in velocity is due to force
- Force may depend on position, velocity and time: $f(x, v, t)$
- Force is either energy conservative or dissipative
- Total force $f = \sum_i f_i$



Conservative forces

Conservative forces preserve total energy and are the gradient of a potential $U(\mathbf{x})$

$$\mathbf{f} = -\nabla_{\mathbf{x}} U(\mathbf{x}) \equiv -\left[\frac{\partial U}{\partial x_1}, \frac{\partial U}{\partial x_2}, \frac{\partial U}{\partial x_2}\right]^T$$

Gravity force with acceleration $\mathbf{g} = [0, 0, -9.81]^T \ m/s^2$

$$U_g(x) = -m\mathbf{x}^T\mathbf{g}$$

$$\mathbf{f}_g = m\mathbf{g}$$

Dissipative forces

Many dissipative forces have a Rayleigh dissipation function $\mathcal{R}(\mathbf{v})$ (rate of energy dissipation)

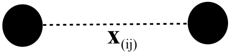
$$f = -\nabla_{\mathbf{v}}\mathcal{R}(\mathbf{v}) \equiv -\left[\frac{\partial \mathcal{R}}{\partial v_1}, \frac{\partial \mathcal{R}}{\partial v_2}, \frac{\partial \mathcal{R}}{\partial v_3}\right]^T$$

Viscous drag

$$\mathcal{R}_d(\mathbf{v}) = \tfrac{1}{2} C \mathbf{v}^T \mathbf{v}$$
$$\mathbf{f}_d(\mathbf{v}) = -k_d \mathbf{v}$$

where the drag coefficient has the form $k_d = \frac{1}{2}\rho A c$, medium mass density ρ, cross-section area A and dimensionless shape/material constant c

Spring force



Scalar form of undamped spring force between bodies i and j

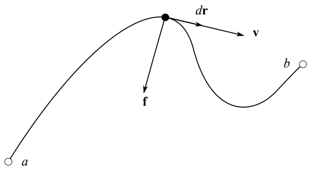
$$\mathbf{f}^{\text{spring}}_{(ij)} = -\left[k_s(|\mathbf{x}_{(ij)}| - L) + k_d \frac{\dot{\mathbf{x}}^T_{(ij)}\mathbf{x}_{(ij)}}{x_{(ij)}} \right] \frac{\mathbf{x}_{(ij)}}{x_{(ij)}}$$

stiffness k_s, damping k_d, length L, separation
$\mathbf{x}_{(ij)} = \mathbf{x}_{(i)} - \mathbf{x}_{(j)}$
Spring potential

$$U^{\text{spring}} = -\tfrac{1}{2}k(|\mathbf{x}_{(ij)}| - L)^2$$

Counter force on j from i: $\mathbf{f}^{\text{spring}}_{(ji)} = -\mathbf{f}^{\text{spring}}_{(ij)}$

Energy



Work - amount of energy exerted by a force $\mathbf{f}$ on a particle between point a and b

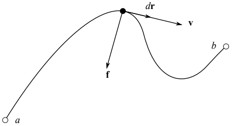
$$W_{ab} \equiv \int_b^a \mathbf{f}^T d\mathbf{x} = \int_{t_b}^{t_a} \mathbf{f}^T \frac{d\mathbf{x}}{dt} dt = m \int_{t_b}^{t_a} \frac{d\mathbf{v}}{dt}^T \mathbf{v} dt$$

$$= \frac{m}{2} \int_{t_b}^{t_a} \frac{d}{dt} |\mathbf{v}|^2 dt = \frac{m}{2} v_b^2 - \frac{m}{2} v_a^2 = K_b - K_a$$

Nonzero work means change in the kinetic energy

$$K_n = \frac{1}{2} m v_n^2$$

Energy



Recall that for a conservative force $\mathbf{f} = -\nabla_{\mathbf{x}} U(\mathbf{x})$ and thus

$$W_{ab} = U(\mathbf{x}_b) - U(\mathbf{x}_a)$$

This implies that the total energy

$$E_{\text{tot}} = K(\mathbf{v}) + U(\mathbf{x})$$

is constant in time, i.e., $\dot{E}_{\text{tot}} = 0$
Important for verifying (debugging) simulation code

Momentum

- For a closed system the total momentum is conserved

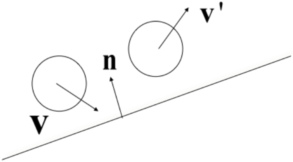
$$\mathbf{p}_{\text{tot}} = \sum_{(i)} m_{(i)} \mathbf{v}_{(i)}$$

- It follows by Netwon's laws that $\dot{\mathbf{p}}_{\text{tot}} = 0$ (check it!)
- Used for verifying (debugging) simulation code
- Used for constructing collision models

Contact with static geometry does not conserve momentum

$$\mathbf{v}' = \mathbf{v} + m^{-1}\mathbf{j}$$

impulse $\mathbf{j} = -(1+e)m(\mathbf{v}^T\mathbf{n})\mathbf{n}$, restitution $0 \leqslant e \leqslant 1$.



Lab 1: Particle system

- System of particles for special effects
- Include also life time, age and color in data representation
- Source/emitter: creates particle at some rate, initial properties, source geometry
- Collisions: particle-surface as reflection against normal
- Force: gravity, air resistance, vortex field, hot air lift force
- Stepforward: semi-implicit Euler
- Destroy/sink: plane, finite volume, expired lifetime

Lab 1: Particle system

definitions
initialization
while running **do**
    collision detection and collision response
    compute forces and constraints
    stepforward:
        solve to update state variables $(Y_n \rightarrow Y_{n+1})$
        update derived quantities
    simulation I/O
end while
post-processing

Lab 1: Particle system

- You are now ready to start Lab 1!!!
- Your simulation will be easy to extend to cloth and further to rigid body systems and fluids
- Good luck and much fun!

Reading instructions

Physics Based Animation, Erleben et al

- Chp 1 - Introduction
- Chp 22.1-22.3 Basic Classical Mechanics
- Chp 23.1 Differential Equations and Numerical Integration
- Chp 8.1-8.5 Particle Systems

SIGGRAPH 97 lecture notes *An Introduction to Physically Based Modeling*

- Differential Equation Basics, A. Witkin and D. Baraff
- Particle System Dynamics, A. Witkin

Notes on Discrete Mechanics, M. Servin, collected lecture notes