Lecture 21:
Particle Dynamics

Reading

Particle Systems Dynamics handout

Optional:

Hocknew and Eastwood. *Computer simulation using particles*. Adam Hilger, New York, 1988.

Gavin Miller. “The motion dynamics of snakes and worms.” *Computer Graphics* 22:169-178, 1988.

Overview

- One lousy particle
- Particle systems
- Forces: gravity, springs
- Implementation

Newtonian particle

- Differential equations: f=ma
- Forces depend on:
- Position, velocity, time

$$\ddot{x} = \frac{f(x, \dot{x}, t)}{m}$$

Second order equations

$$\ddot{x} = \frac{f(x, \dot{x}, t)}{m}$$

Has 2nd derivatives

$$\left[\begin{array}{l} \dot{x} = v \\ \dot{v} = \dfrac{f(x, \dot{x}, t)}{m} \end{array}\right]$$

Add a new variable v to get
a pair of coupled 1st order equations

Phase space

$$\begin{bmatrix} x \\ v \end{bmatrix}$$

Concatenate x and v to make a 6-vector: position in phase space

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix}$$

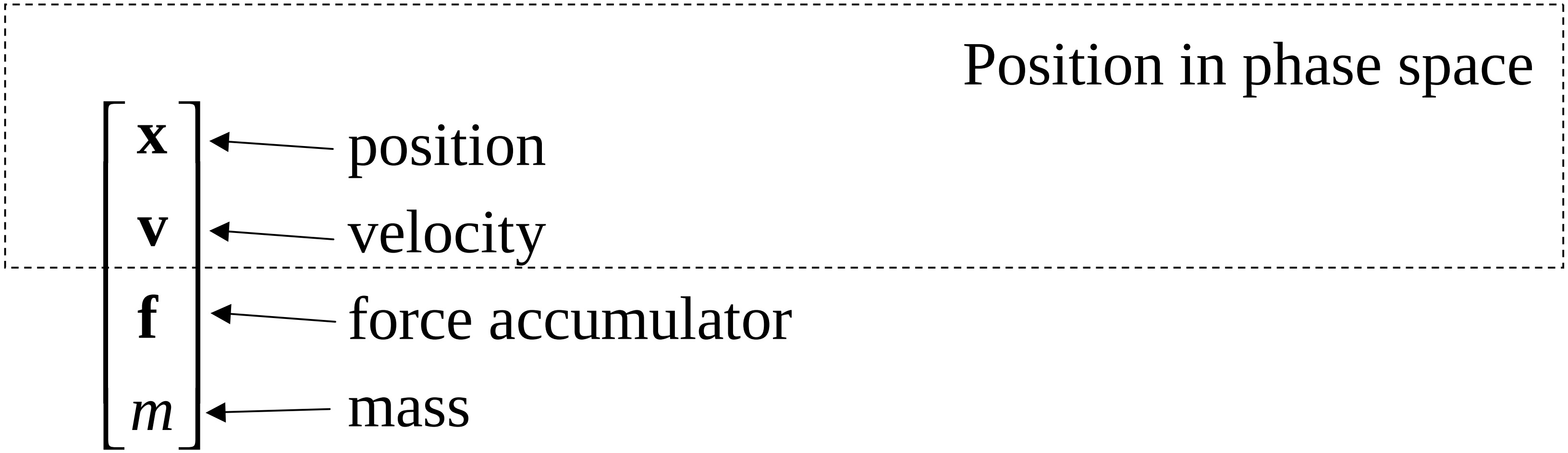
Velocity on Phase space:

Another 6-vector

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ f/m \end{bmatrix}$$

A vanilla 1st-order differential equation

Particle structure



Position in phase space

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{v} \\ \mathbf{f} \\ m \end{bmatrix}$$

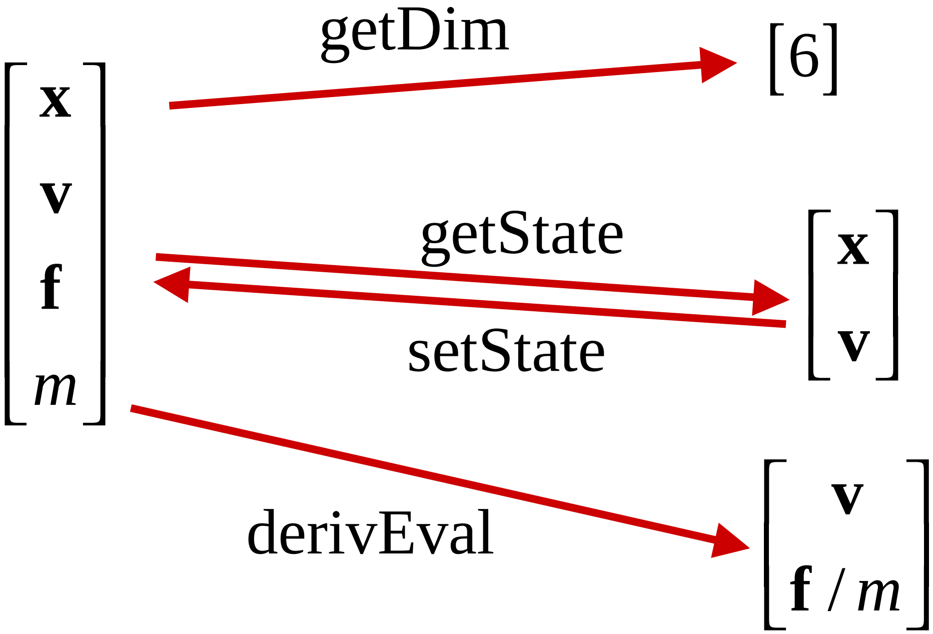
position

velocity

force accumulator

mass

Solver interface



$$\begin{bmatrix} \mathbf{x} \\ \mathbf{v} \\ \mathbf{f} \\ m \end{bmatrix}$$

getDim

$[6]$

getState

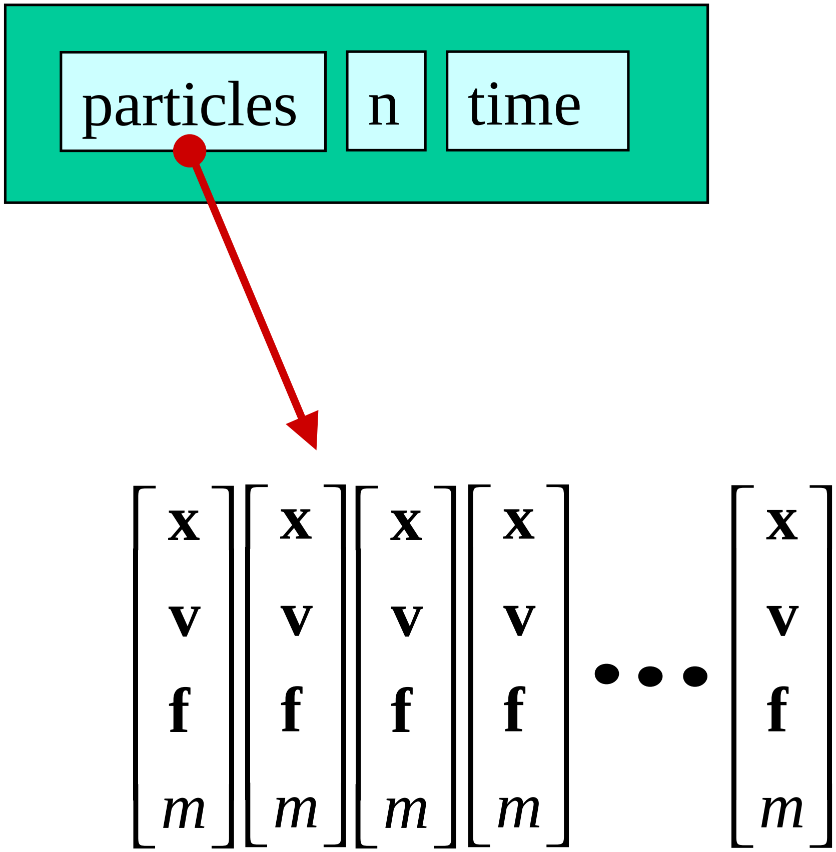
setState

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{v} \end{bmatrix}$$

derivEval

$$\begin{bmatrix} \mathbf{v} \\ \mathbf{f}/m \end{bmatrix}$$

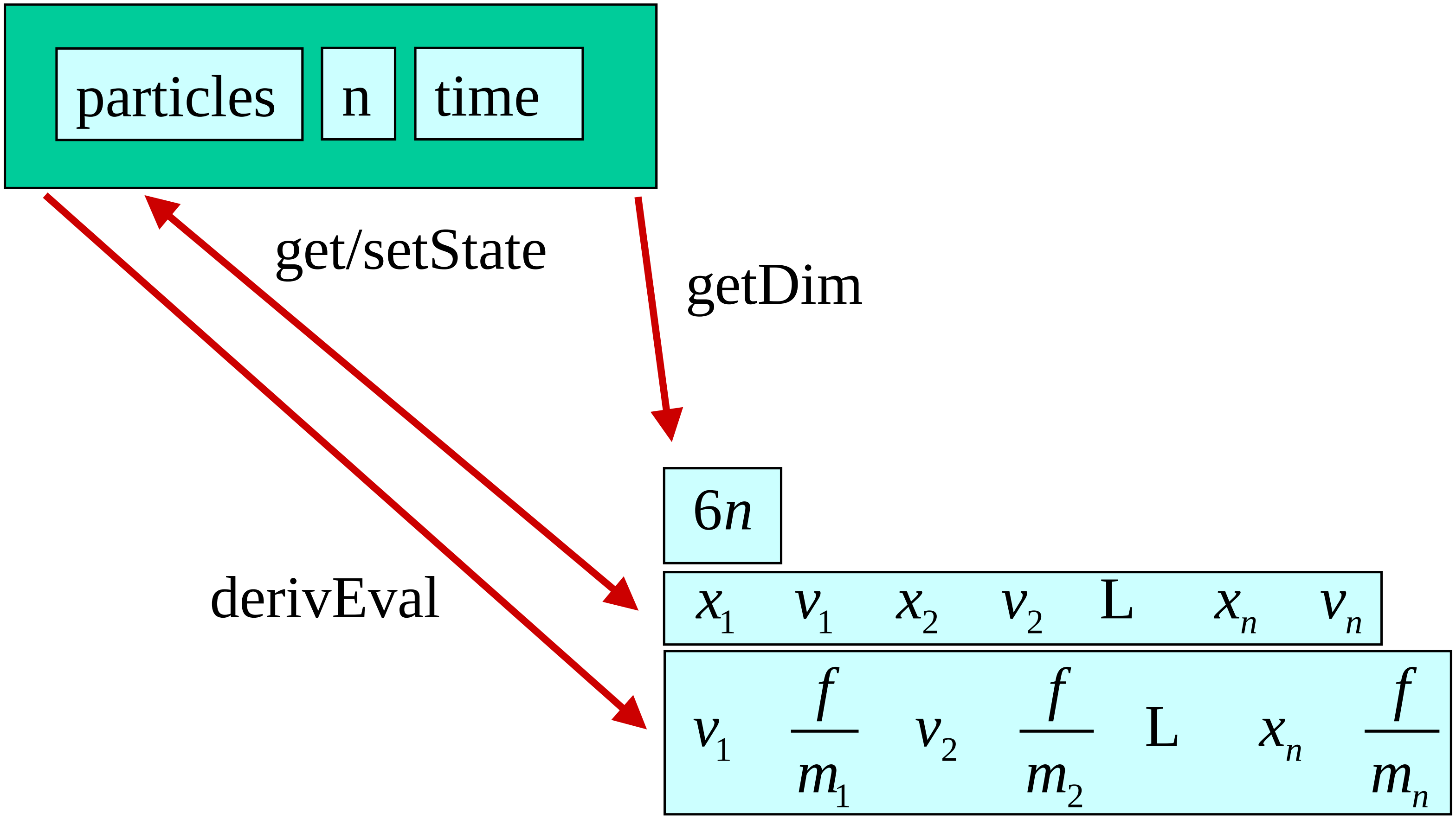
Particle systems



particles n time

$$\begin{bmatrix}\mathbf{x}\\ \mathbf{v}\\ \mathbf{f}\\ m\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \mathbf{v}\\ \mathbf{f}\\ m\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \mathbf{v}\\ \mathbf{f}\\ m\end{bmatrix}\begin{bmatrix}\mathbf{x}\\ \mathbf{v}\\ \mathbf{f}\\ m\end{bmatrix}\cdots\begin{bmatrix}\mathbf{x}\\ \mathbf{v}\\ \mathbf{f}\\ m\end{bmatrix}$$

Solver interface



particles n time

get/setState

getDim

derivEval

$6n$

$x_1 \quad v_1 \quad x_2 \quad v_2 \quad \mathrm{L} \quad x_n \quad v_n$

$v_1 \quad \frac{f}{m_1} \quad v_2 \quad \frac{f}{m_2} \quad \mathrm{L} \quad x_n \quad \frac{f}{m_n}$

Differential equation solver

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ f/m \end{bmatrix}$$

Euler method: $x(t+h) = x(t) + h \cdot \dot{x}(t)$

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \Delta t \cdot \dot{\mathbf{x}}_i$$

$$\mathbf{v}_{i+1} = \mathbf{v}_i + \Delta t \cdot \dot{\mathbf{v}}_i$$

Gets very unstable for large Δt

Higher order solvers perform better: (e.g. Runge-Kutta)

derivEval loop

1. Clear forces
   - Loop over particles, zero force accumulators
2. Calculate forces
   - Sum all forces into accumulators
3. Gather
   - Loop over particles, copying v and f/m into destination array

Forces

- Constant (gravity)
- Position/time dependent (force fields)
- Velocity-dependent (drag)
- N-ary (springs)

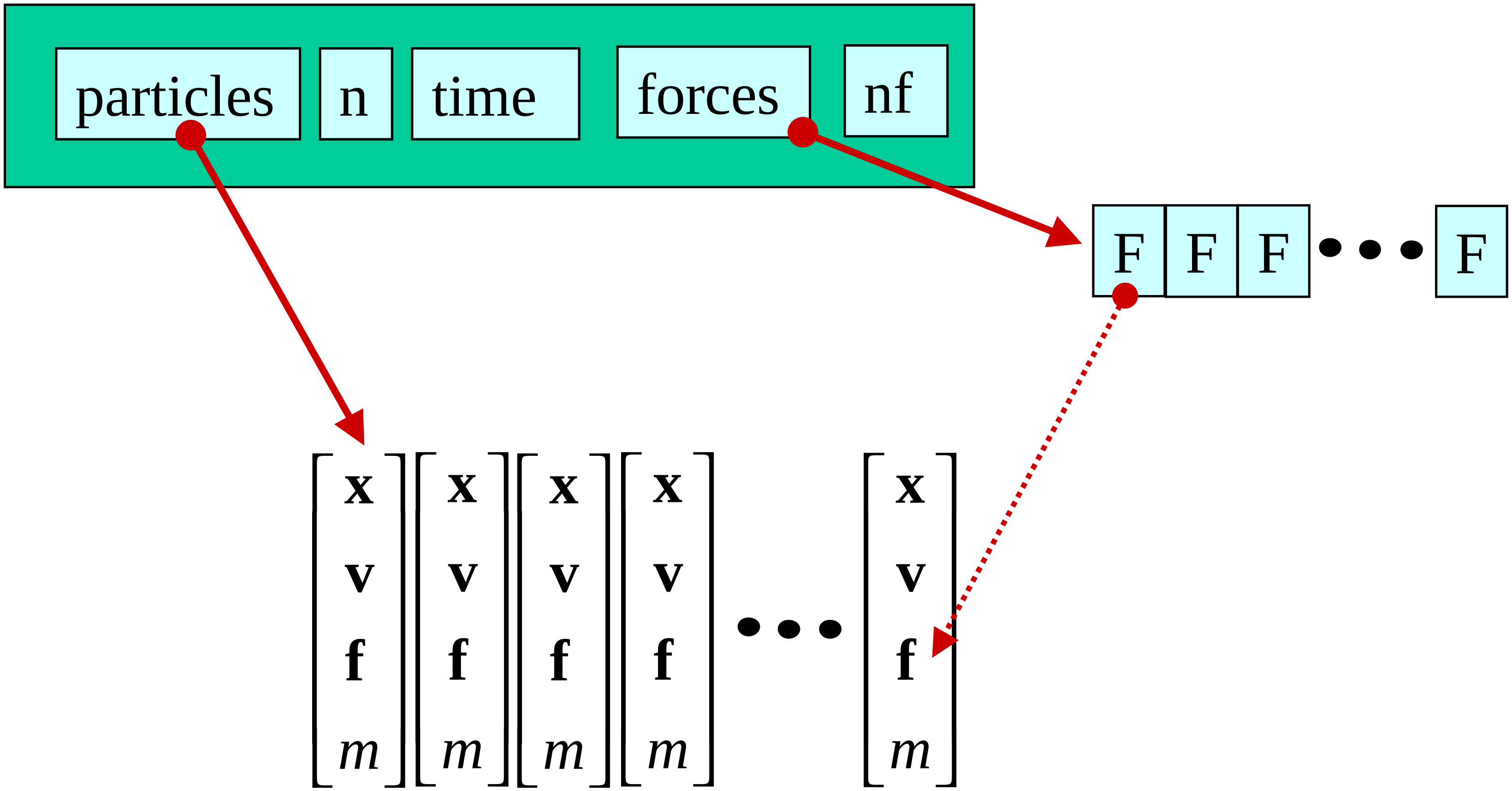
Force structures

Force objects are black boxes that point to the particles they influence, and add in their contribution into the force accumulator.

Global force calculation:

- Loop, invoking force objects

Particle systems with forces



Gravity

Force law:

$$\mathbf{f}_{grav} = m\mathbf{G}$$

```
p->f += p->m * F->G
```

Viscous drag

Force law:

$$\mathbf{f}_{drag} = -k_{drag}\mathbf{v}$$

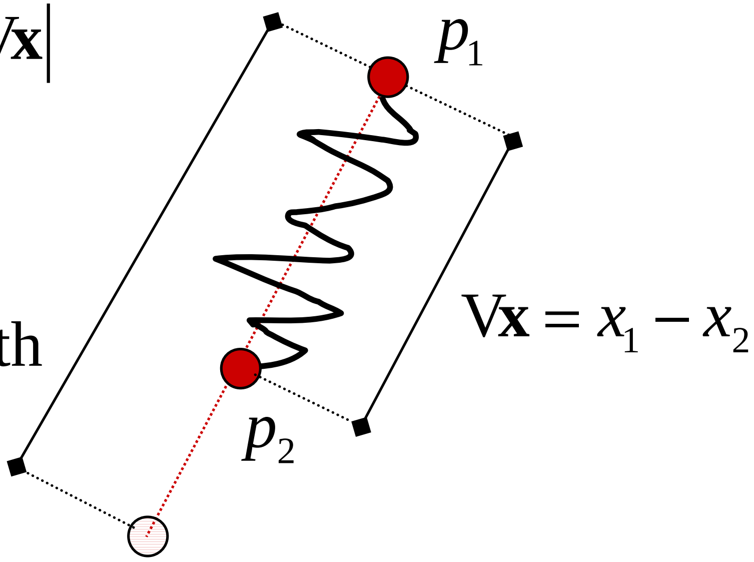
```
p->f -= F->k * p->v
```

Damped spring

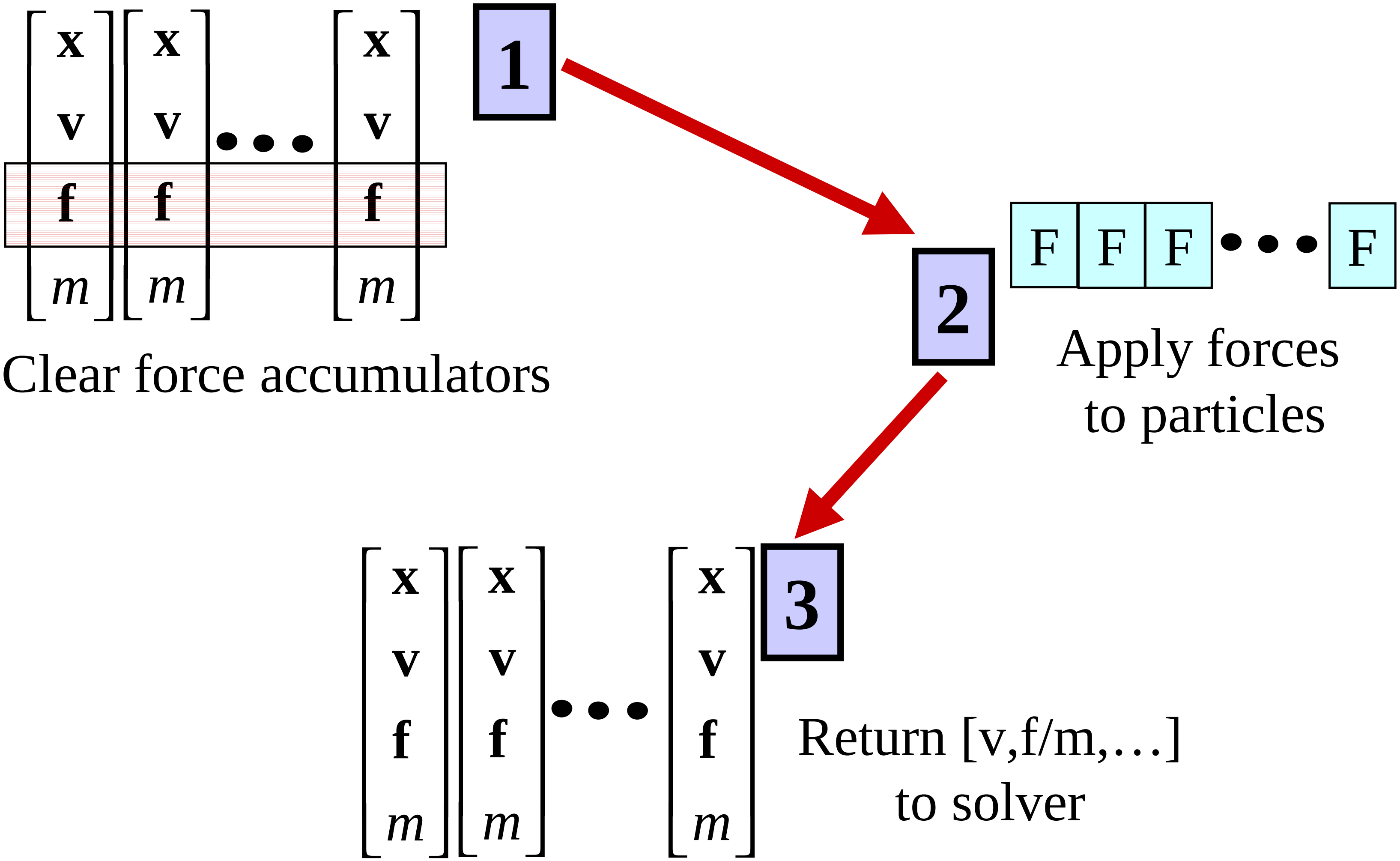
Force law:

$$\mathbf{f}_1 = -\left[k_s\left(\left|\mathrm{V}\mathbf{x}\right| - \mathbf{r}\right) + k_d \left(\frac{\mathrm{V}\mathbf{v}\mathrm{V}\mathbf{x}}{\left|\mathrm{V}\mathbf{x}\right|} \right) \right] \frac{\mathrm{V}\mathbf{x}}{\left|\mathrm{V}\mathbf{x}\right|}$$

$$\mathbf{f}_2 = -\mathbf{f}_1$$


p_1
p_2
$\mathrm{V}\mathbf{x} = x_1 - x_2$
$\mathbf{r}$ = rest length

derivEval Loop

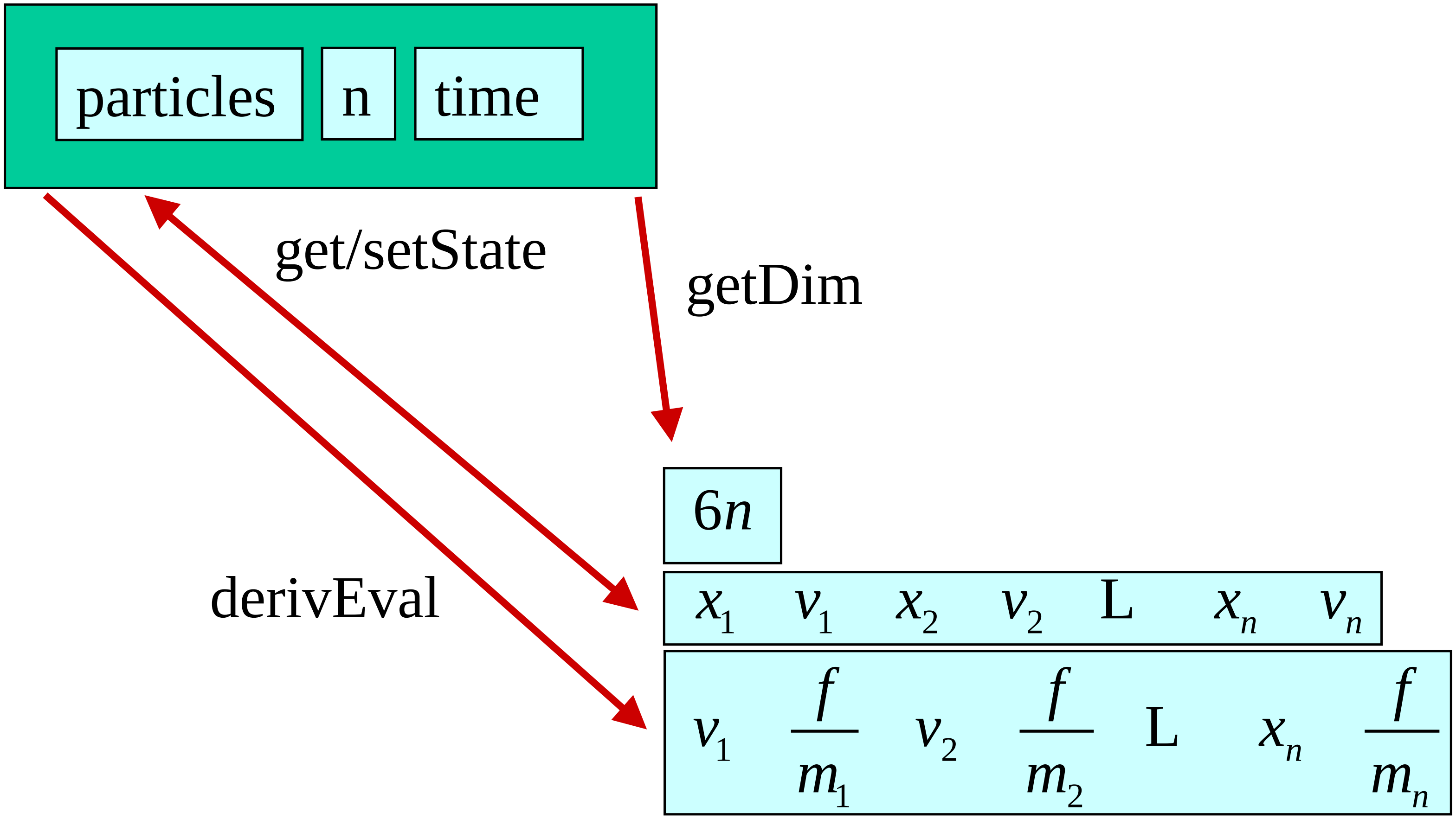


Clear force accumulators

Apply forces
to particles

Return [v,f/m,…]
to solver

Solver interface



$6n$

$x_1 \quad v_1 \quad x_2 \quad v_2 \quad \mathrm{L} \quad x_n \quad v_n$

$v_1 \quad \frac{f}{m_1} \quad v_2 \quad \frac{f}{m_2} \quad \mathrm{L} \quad x_n \quad \frac{f}{m_n}$

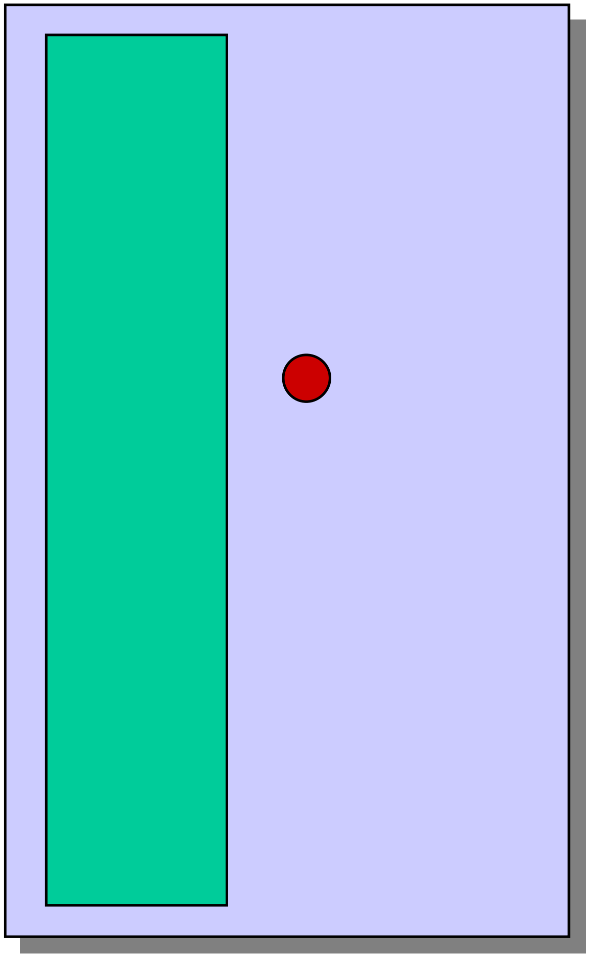
Differential equation solver

$$\begin{bmatrix} \dot{x} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} v \\ f/m \end{bmatrix}$$

Euler method:

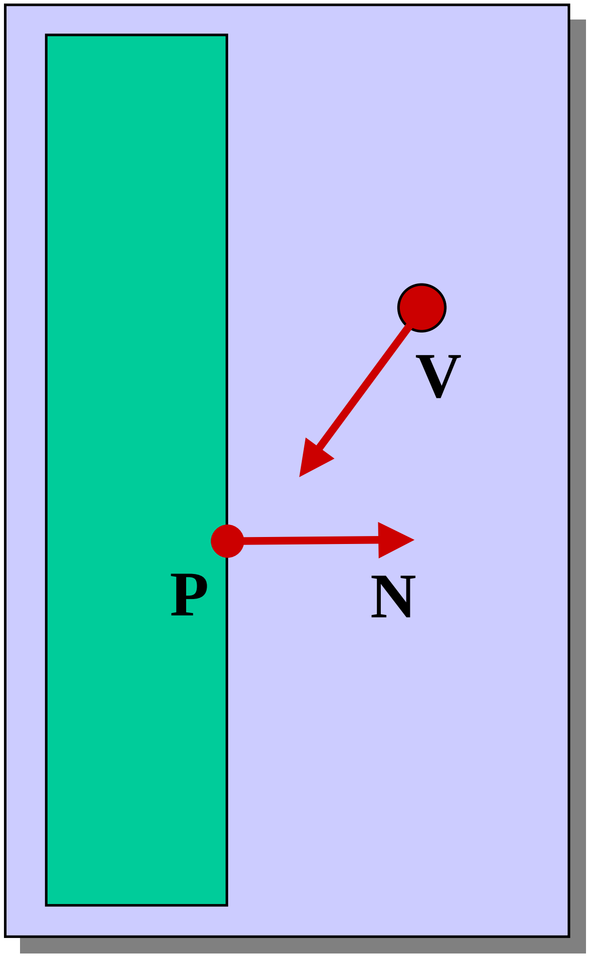
$$\begin{bmatrix} x_1^{i+1} \\ v_1^{i+1} \\ \vdots \\ x_n^{i+1} \\ v_n^{i+1} \end{bmatrix} = \begin{bmatrix} x_1^{i} \\ v_1^{i} \\ \vdots \\ x_n^{i} \\ v_n^{i} \end{bmatrix} + \Delta t \begin{bmatrix} v_1^{i} \\ f_1^{i}/m_1 \\ \vdots \\ v_n^{i} \\ f_n^{i}/m_n \end{bmatrix}$$

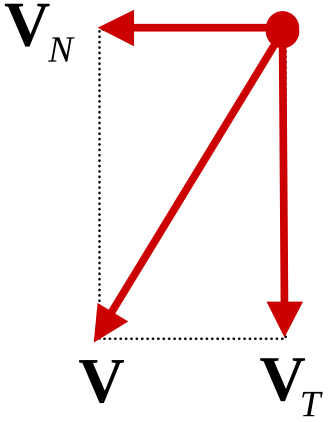
Bouncing off the walls



- Add-on for a particle simulator
- For now, just simple point-plane collisions

Normal and tangential components

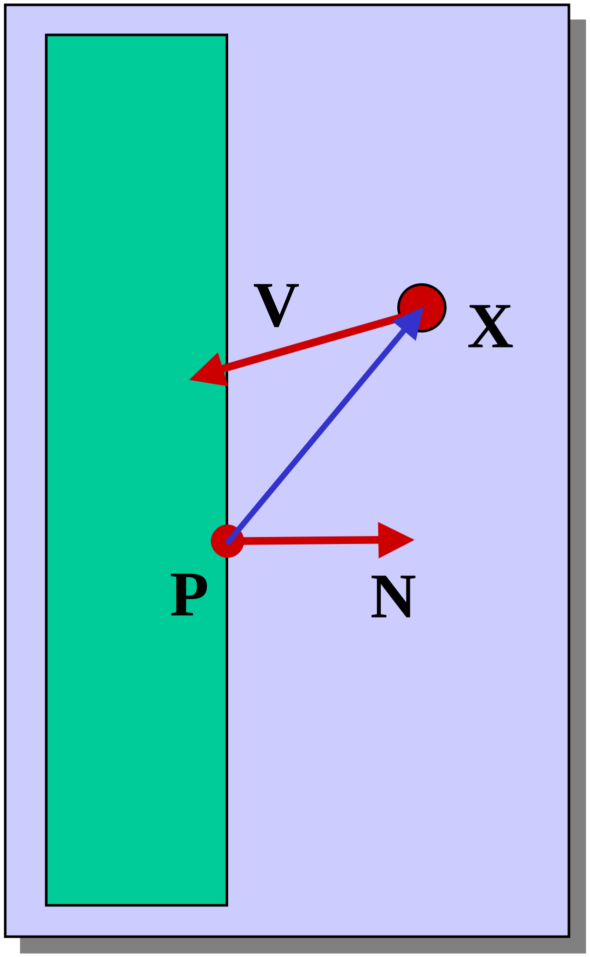




$$V_N = (N \cdot V)N$$

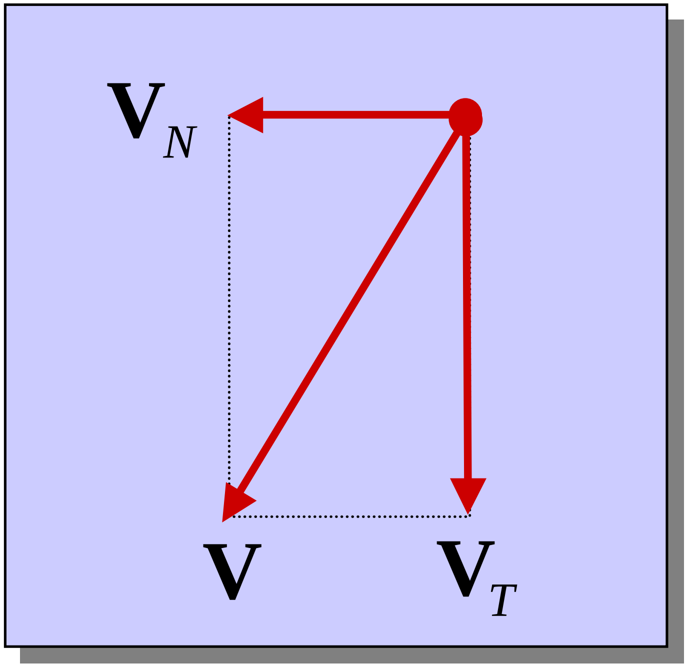
$$V_T = V - V_N$$

Collision Detection

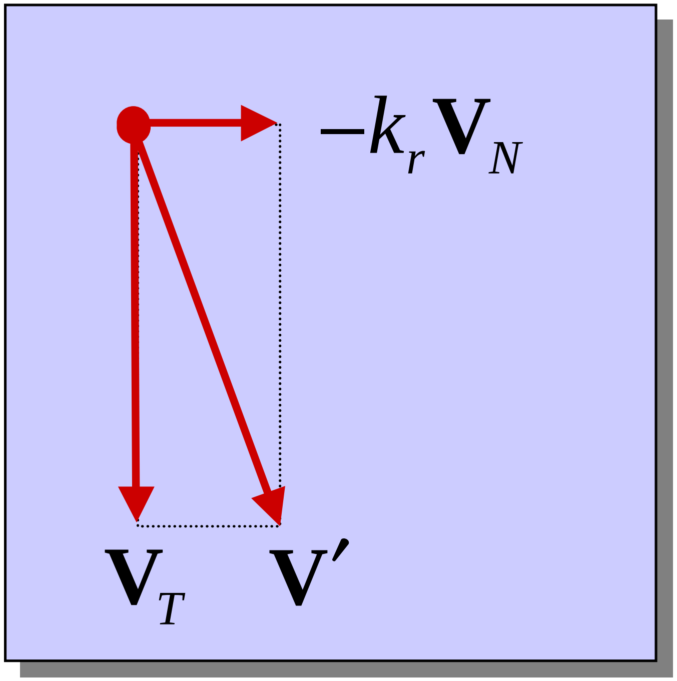


$(\mathbf{X}-\mathbf{P})\cdot\mathbf{N}<\varepsilon$	Within e of the wall
$\mathbf{N}\cdot\mathbf{V}<0$	Heading in

Collision Response



before



after

$$\mathbf{V}' = \mathbf{V}_T - k_r\mathbf{V}_N$$

Summary

- Physics of a particle system
- Various forces acting on a particle
- Combining particles into a particle system
- Euler method for solving differential equations