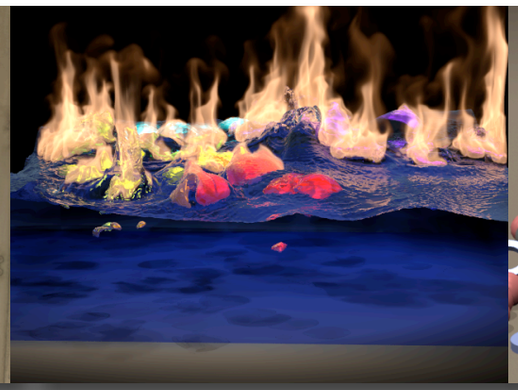
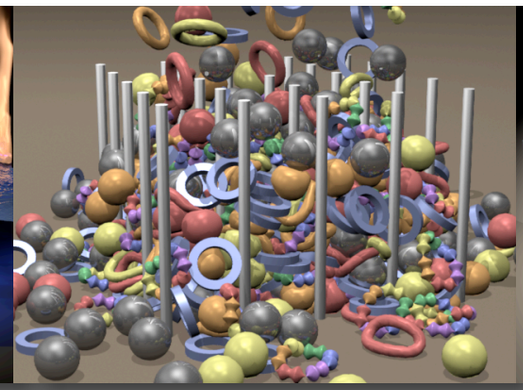
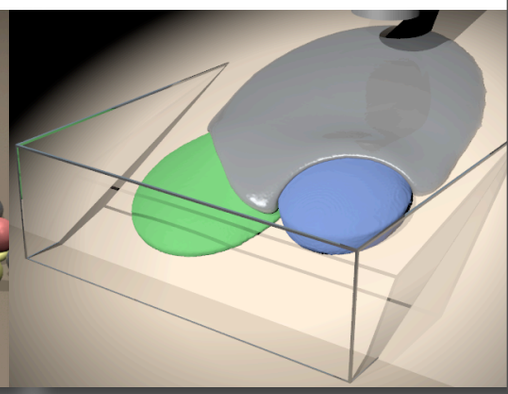
CS260
Lecture 3: Particle Systems



Particle Systems

Andrew Witkin

What is a particle?

What is a particle?

mass m

What is a particle?

mass m

position $\mathbf{x}$

What is a particle?

mass m

position $\mathbf{x}$

velocity $\mathbf{v}$

What is a particle?

mass m

position $\mathbf{x}$

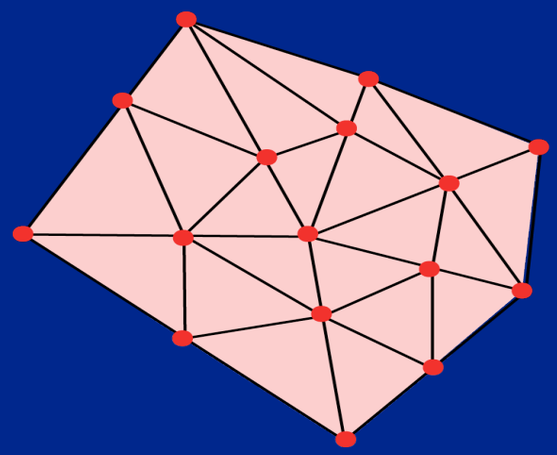
velocity $\mathbf{v}$

forces $\mathbf{f}$

Particle Systems Examples

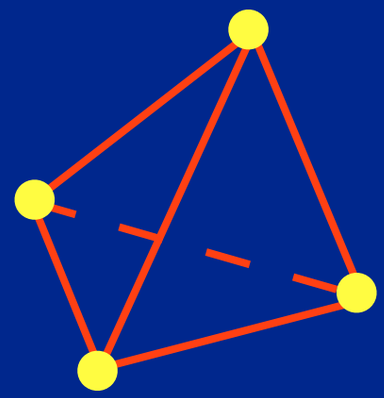
segments

A Mass Spring Model for Hair Simulation, Selle et al., 2008



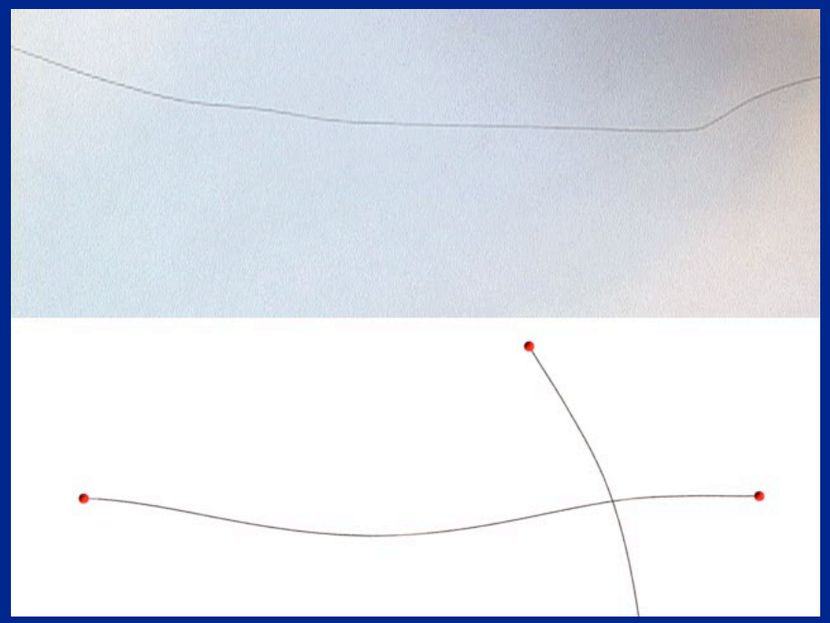
triangles

tetrahedra

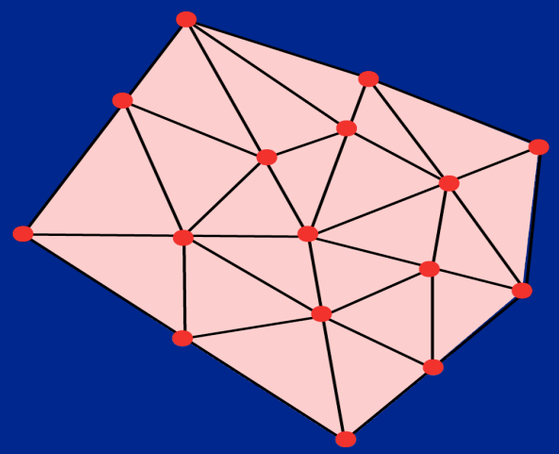


Particle Systems Examples

segments

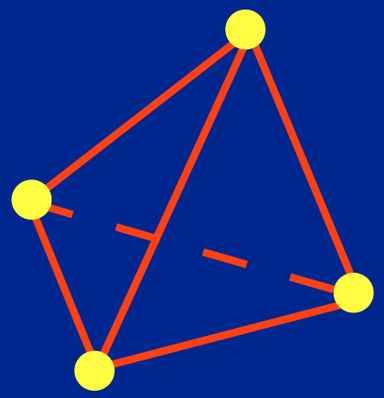


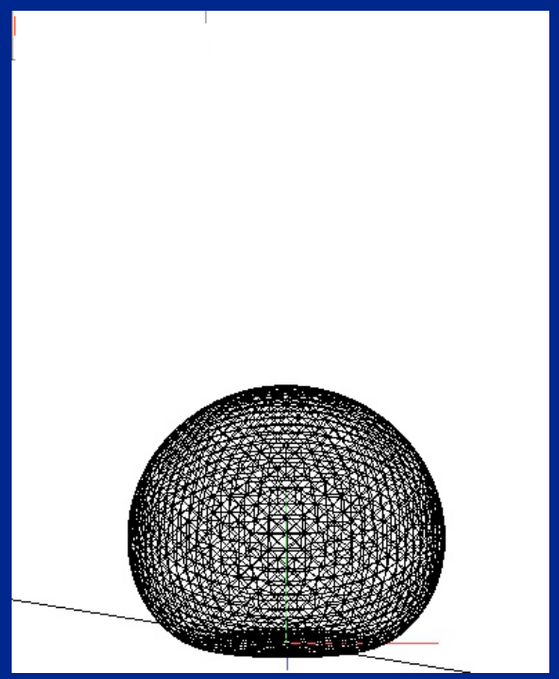
A Mass Spring Model for Hair Simulation, Selle et al., 2008



triangles

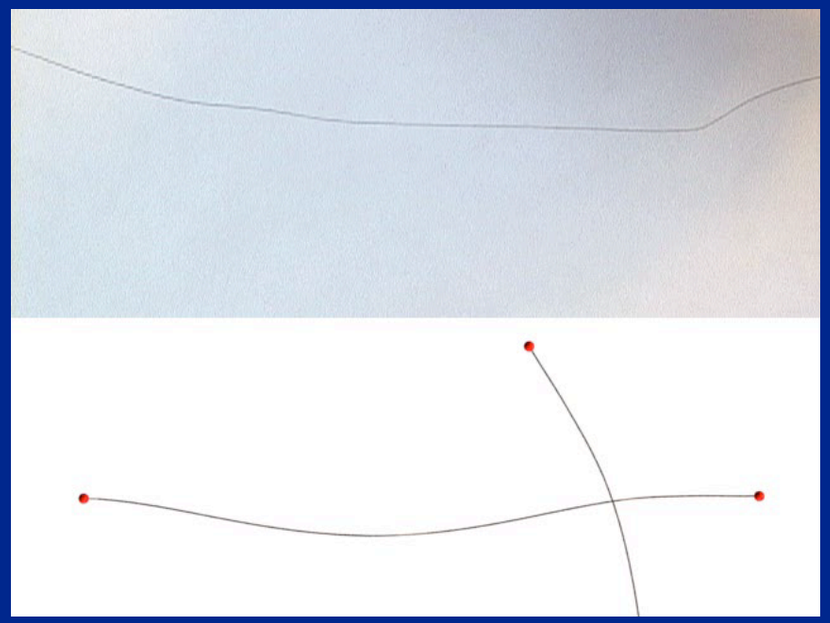
tetrahedra



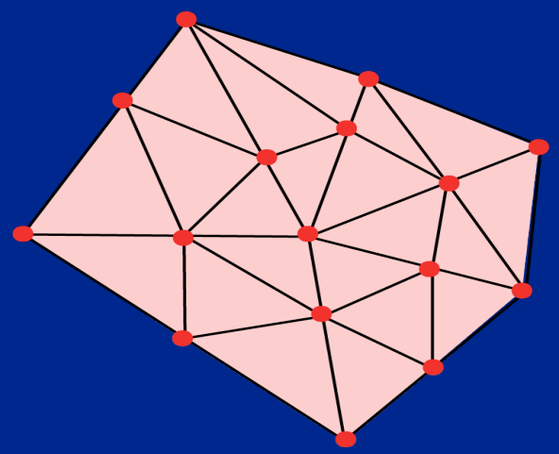


Particle Systems Examples

segments

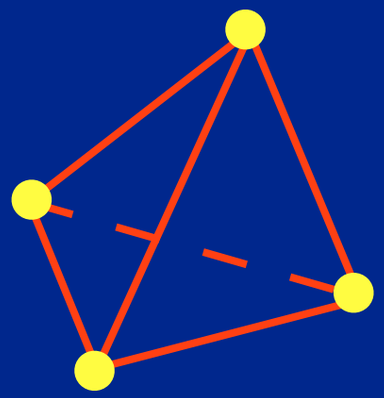


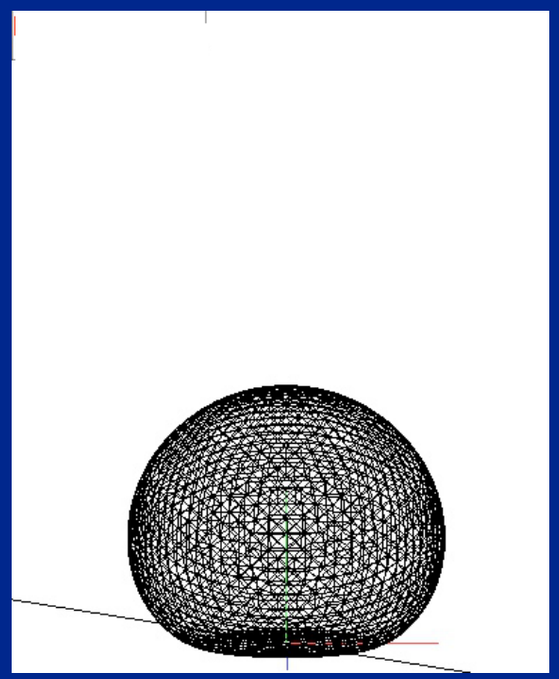
A Mass Spring Model for Hair Simulation, Selle et al., 2008



triangles

tetrahedra





A Newtonian Particle

- Differential equation: f = ma
- Forces can depend on:
  - Position, Velocity, Time

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

Second Order Equations

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

Not in our standard form because it has 2nd derivatives

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

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

Phase Space

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

Concatenate **x** and **v** to make a 6-vector: *Position in Phase Space*.

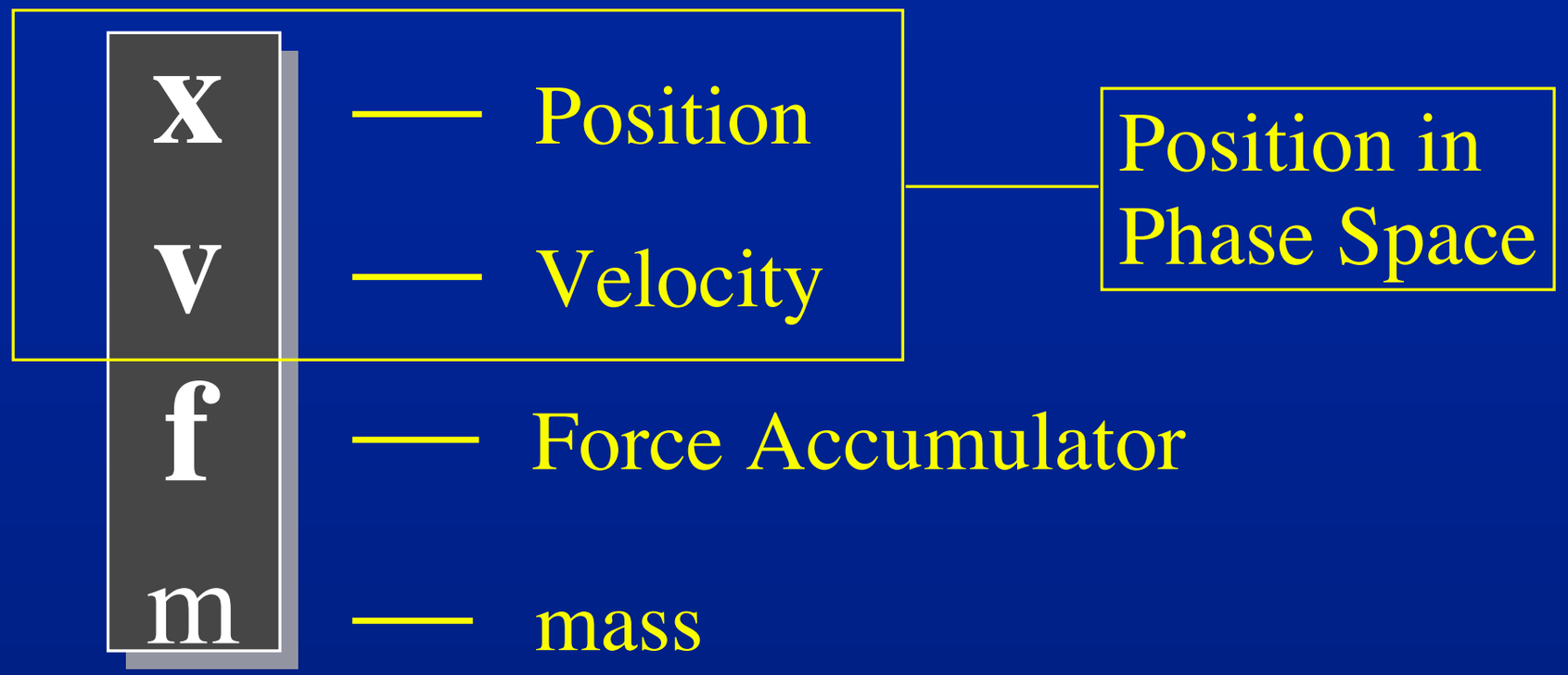
$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{v}} \end{bmatrix}$$

Velocity in Phase Space: another 6-vector.

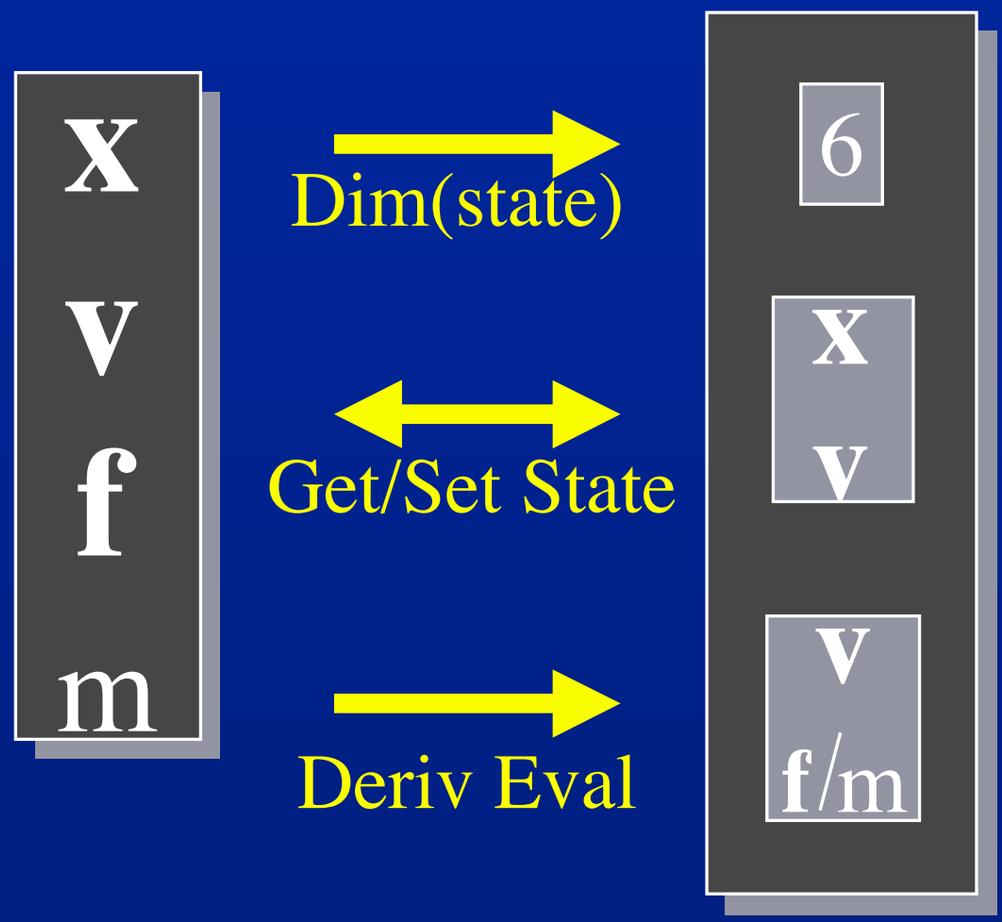
$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{f}/m \end{bmatrix}$$

A vanilla 1st-order differential equation.

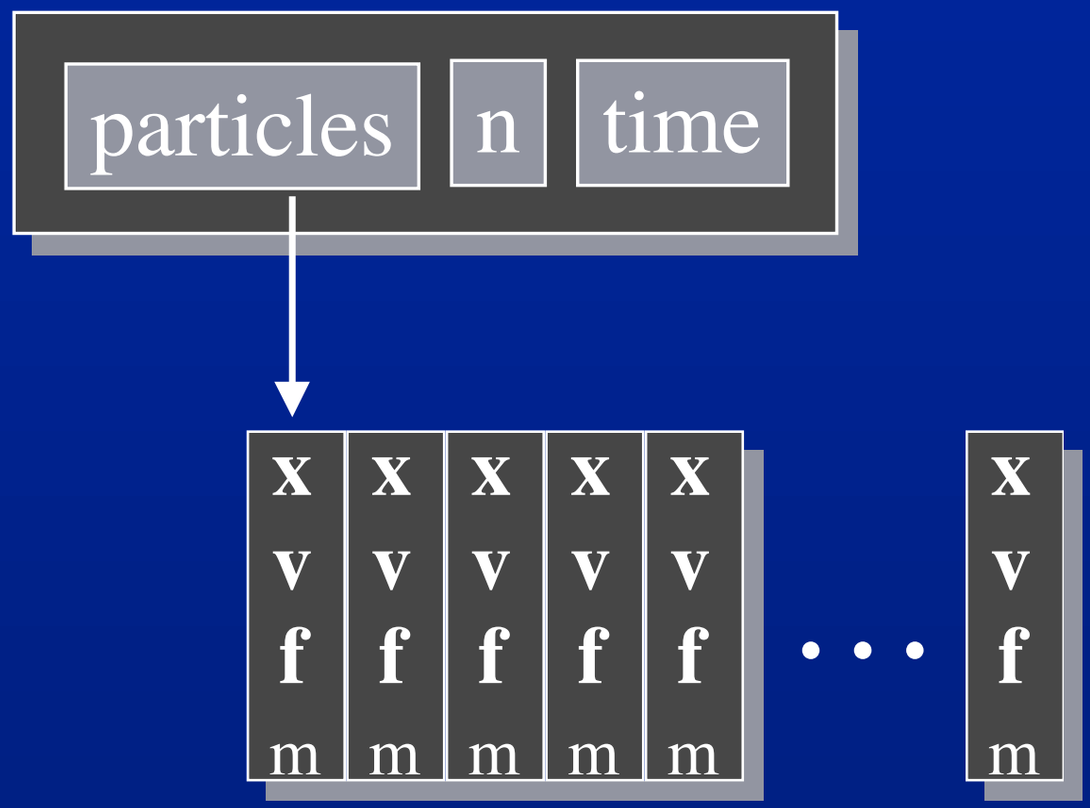
Particle Structure



Solver Interface



Particle Systems



Solver Interface

Particle System

particles	n	time

Dim(State)

Get/Set State

Deriv Eval

Diffeq Solver

6n						
$\mathbf{x}_1$	$\mathbf{v}_1$	$\mathbf{x}_2$	$\mathbf{v}_2$	$\cdots$	$\mathbf{x}_n$	$\mathbf{v}_n$
$\mathbf{v}_1$	$\frac{\mathbf{f}_1}{m_1}$	$\mathbf{v}_2$	$\frac{\mathbf{f}_2}{m_2}$	$\cdots$	$\mathbf{v}_n$	$\frac{\mathbf{f}_n}{m_n}$

A Code Fragment

```
void EulerStep(ParticleSystem* p, float dt)
{
    Vector temp1,temp2;

    temp1 = p->ParticleDerivative();
    temp1 *= dt;
    p->GetState(temp2);
    temp2 += temp1;
    p->SetState(temp2);
    p->time += dt;
}
```

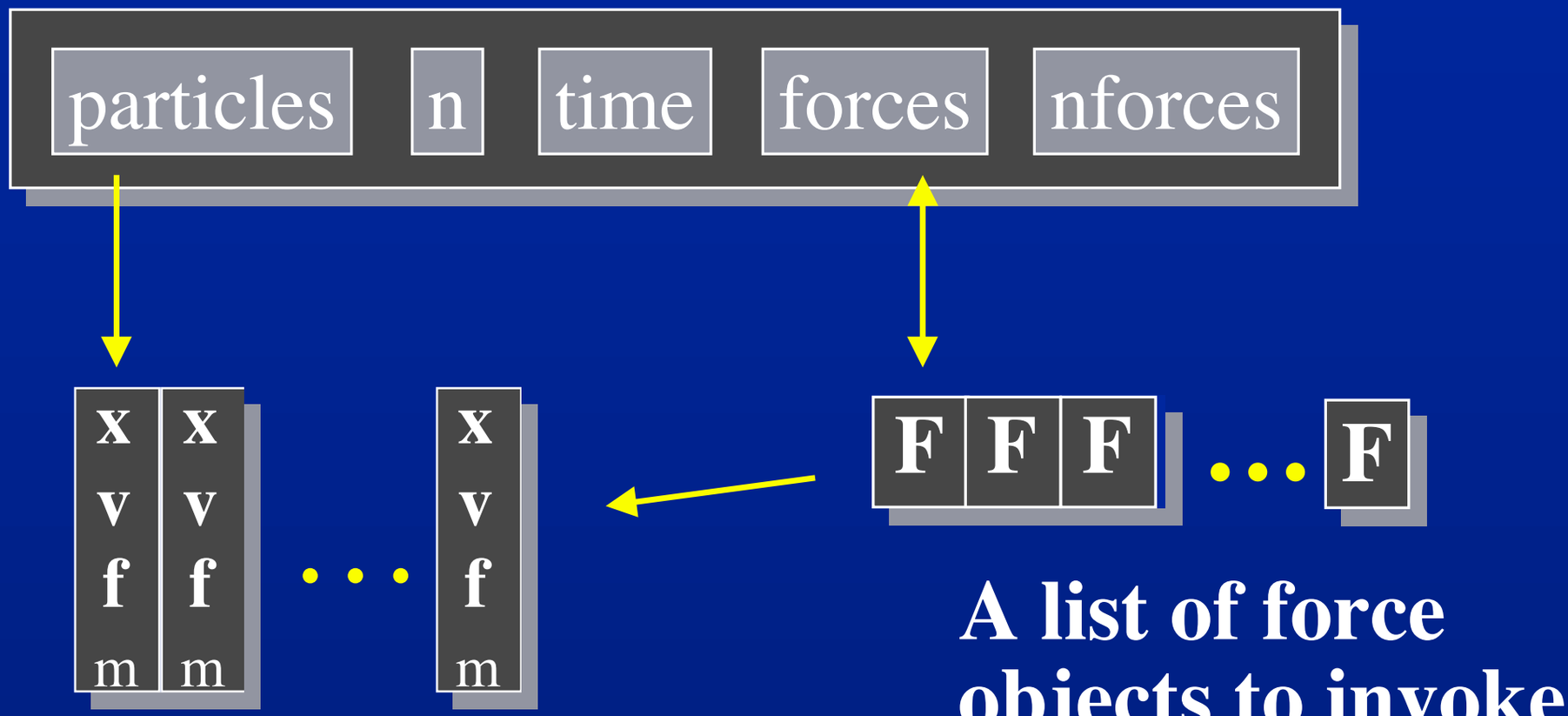
Forces

- Constant — gravity
- Position/time dependent — force fields
- Velocity-Dependent — drag
- n-ary — springs

Force Structures

- **Unlike particles, forces are heterogeneous.**
- **Force Objects:**
  - black boxes
  - point to the particles they influence
  - add in their own forces (type dependent)
- **Global force calculation:**
  - loop, invoking force objects

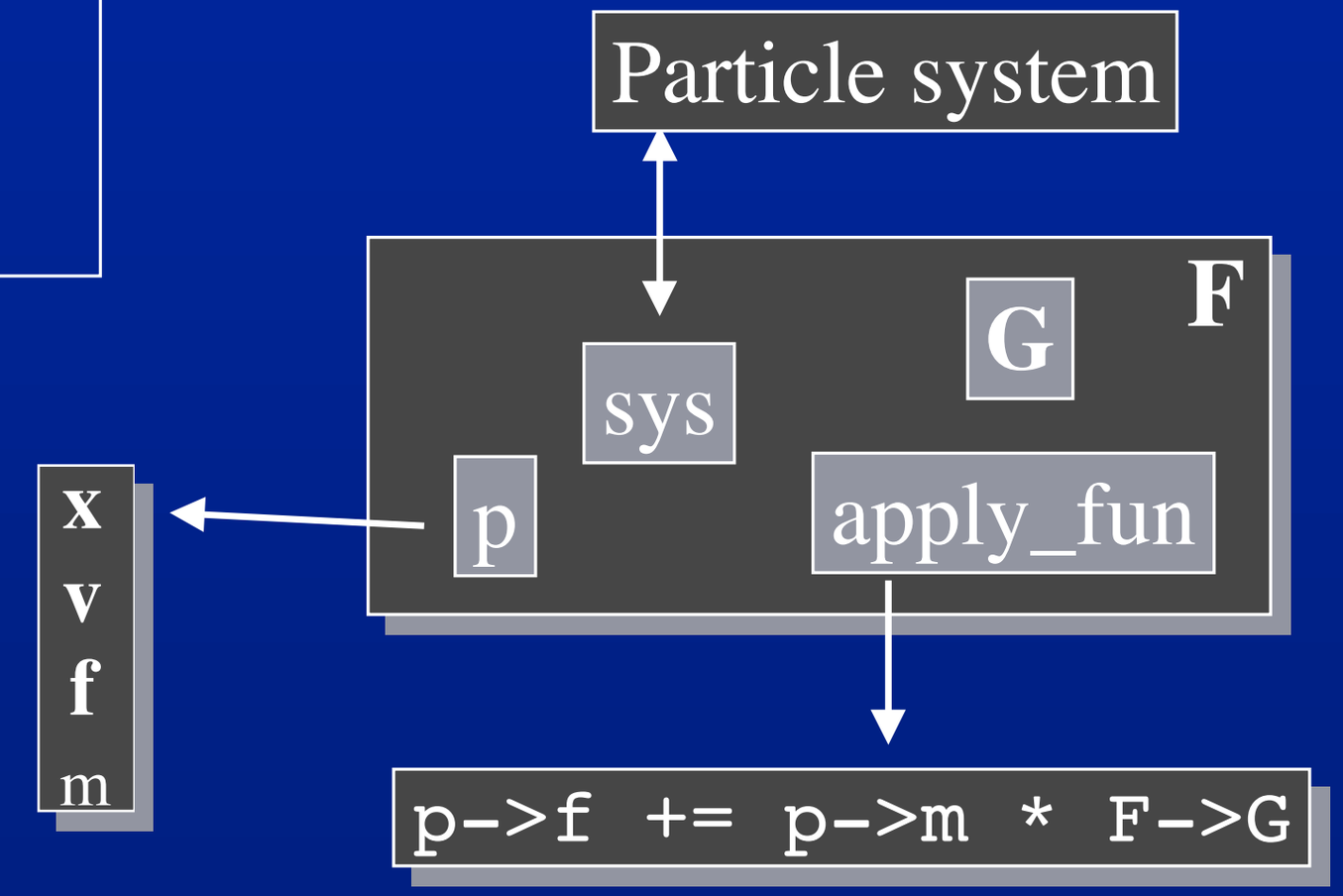
Particle Systems, with forces



Gravity

Force Law:

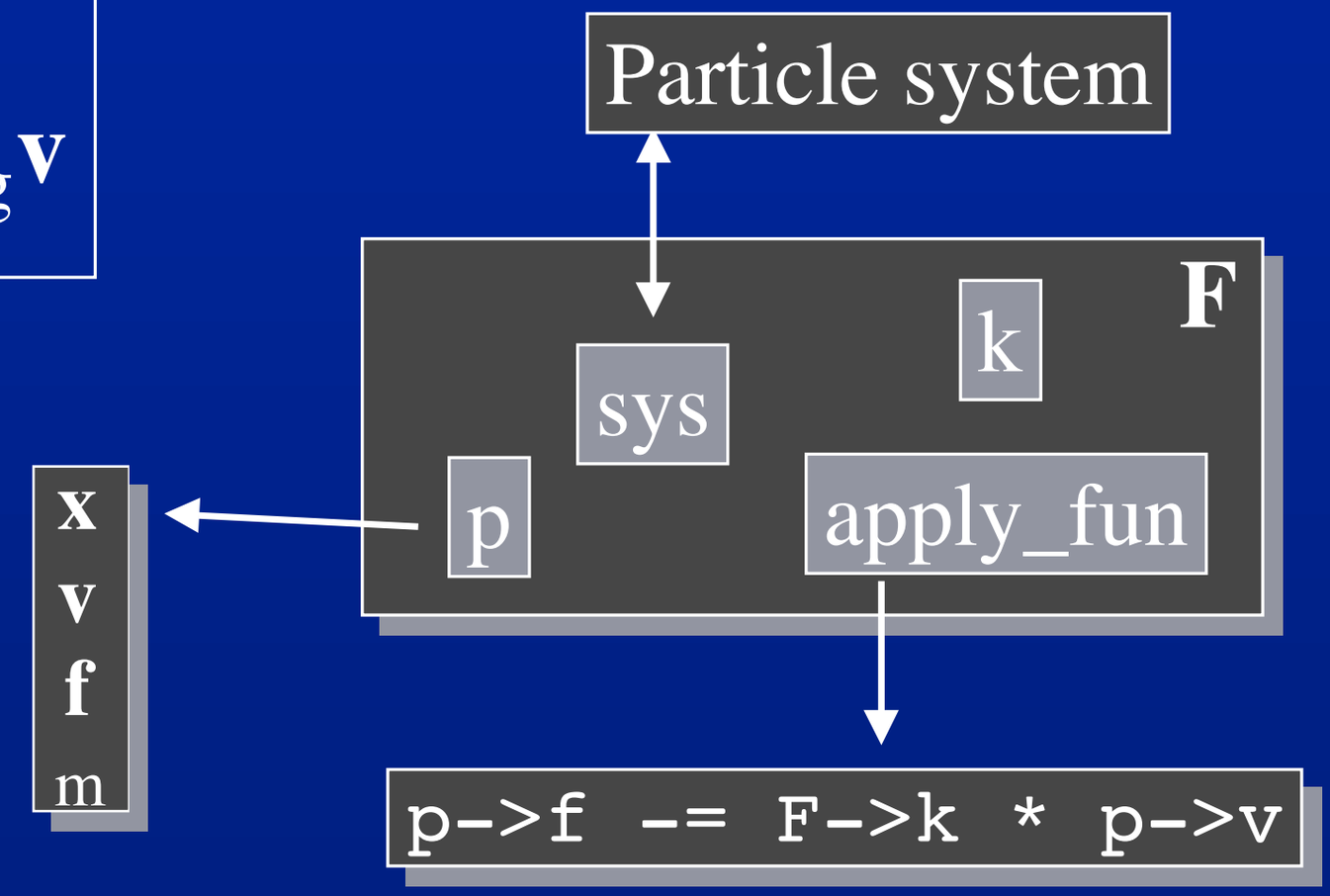
$$\mathbf{f}_{grav} = m\mathbf{G}$$



Viscous Drag

Force Law:

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

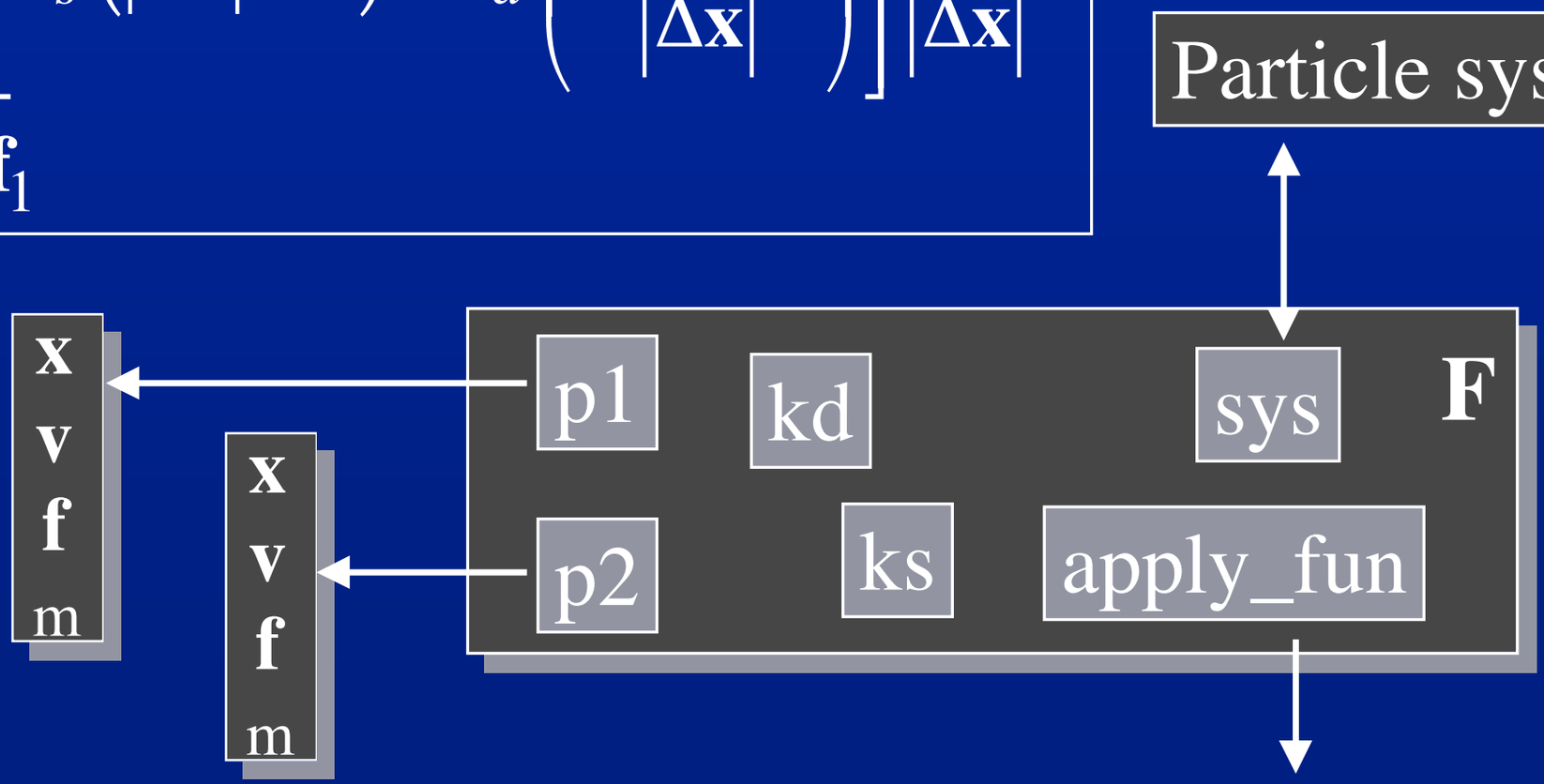


Damped Spring

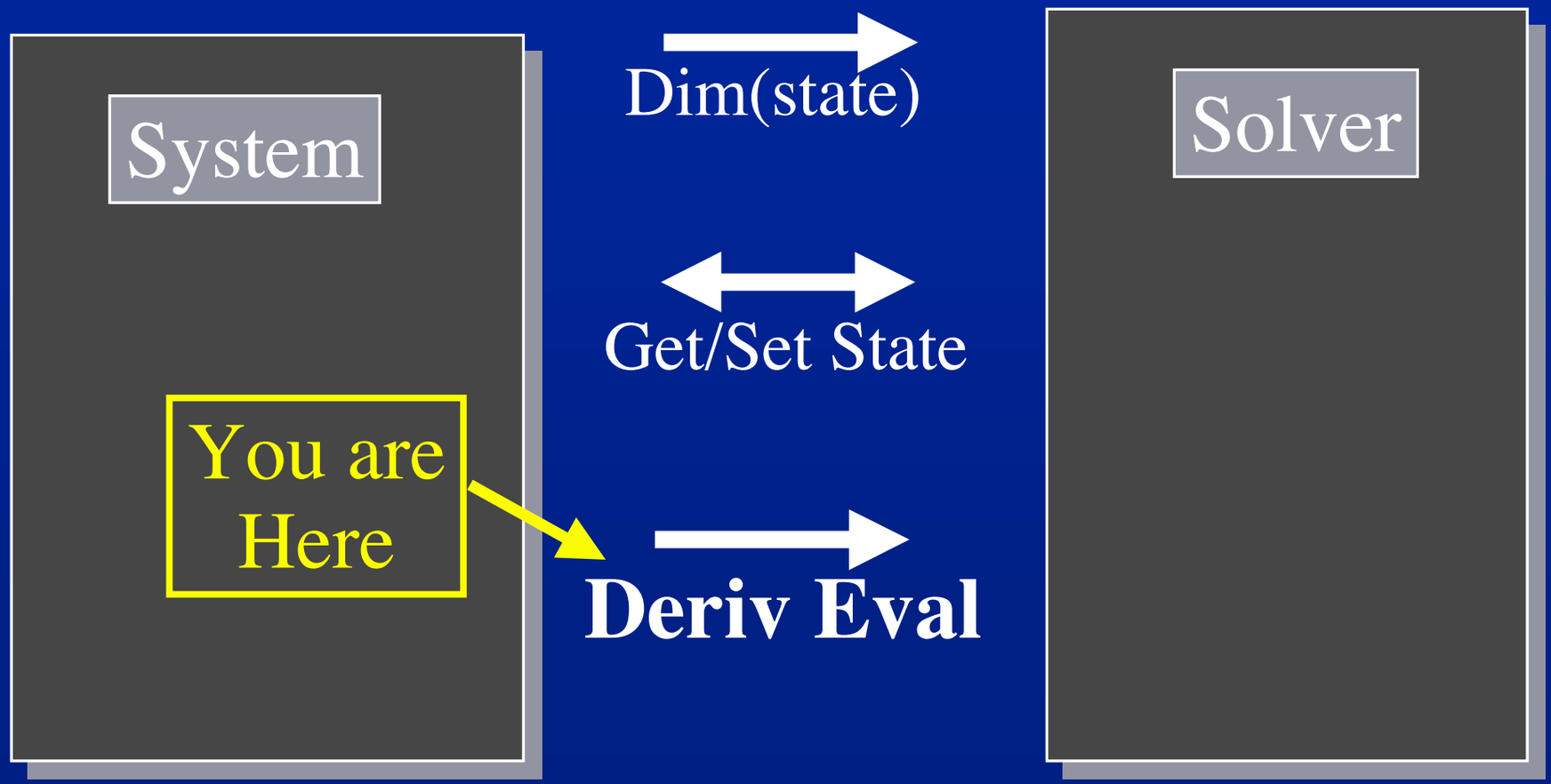
Force Law:

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

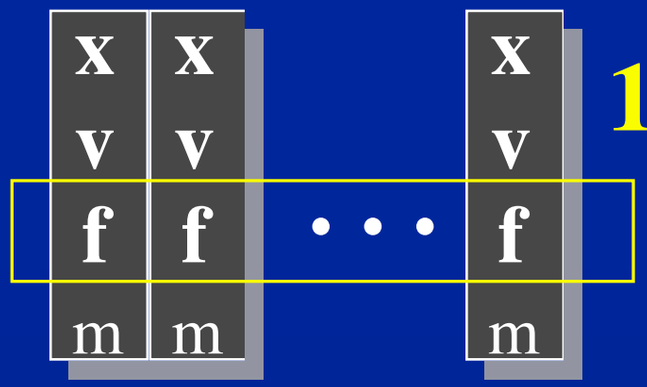
$$\mathbf{f}_2 = -\mathbf{f}_1$$



Solver Interface

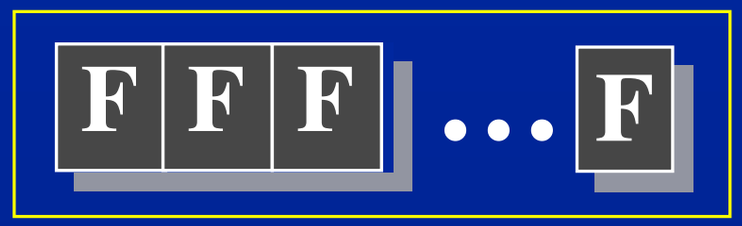


Deriv Eval Loop



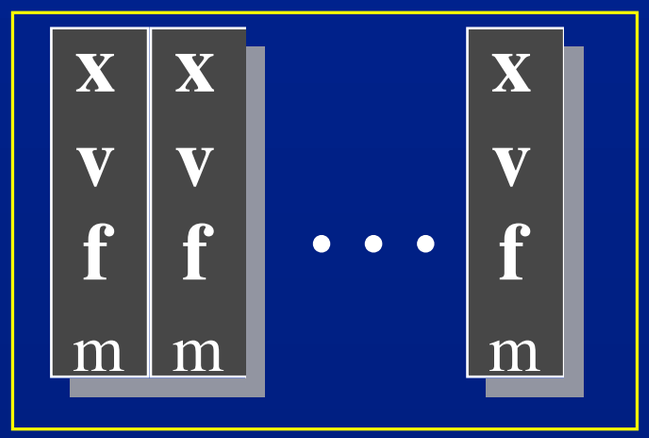
1

Clear Force Accumulators



2

Invoke `apply_force` functions

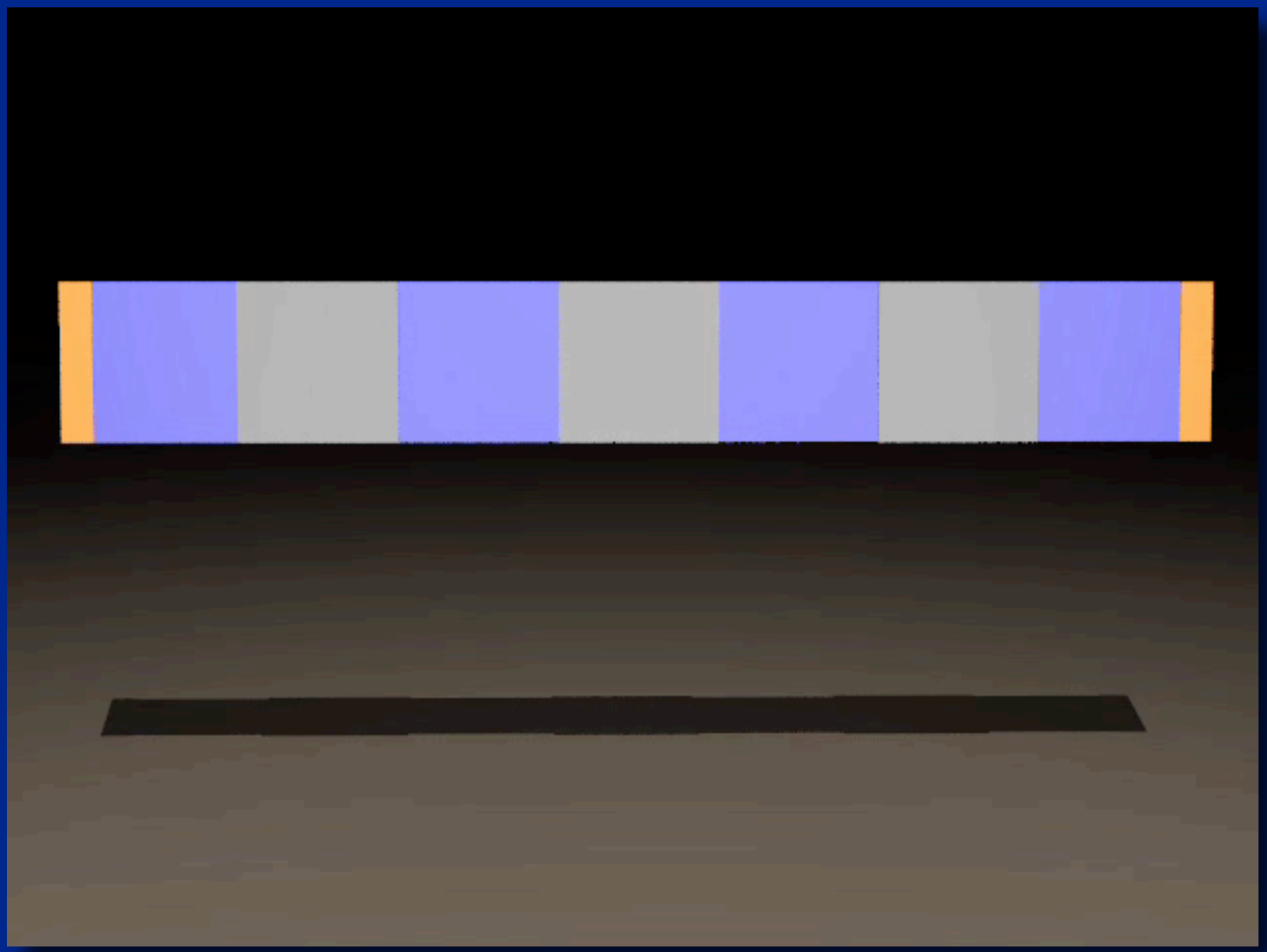


3

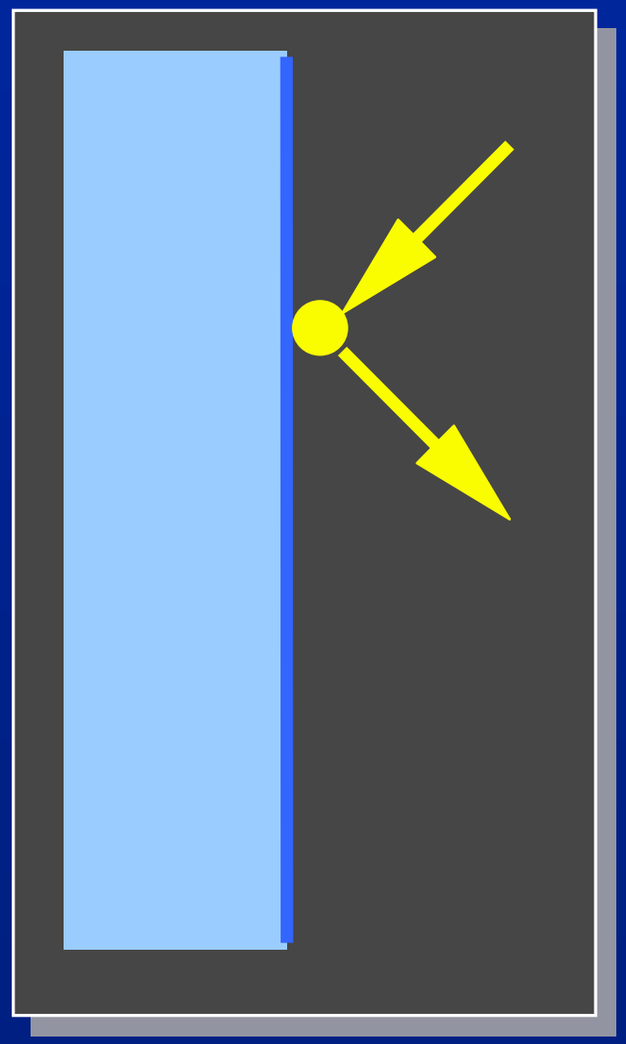
Return [v, f/m,…] to solver.

Controlled Particles

Controlled Particles



Bouncing off the Walls

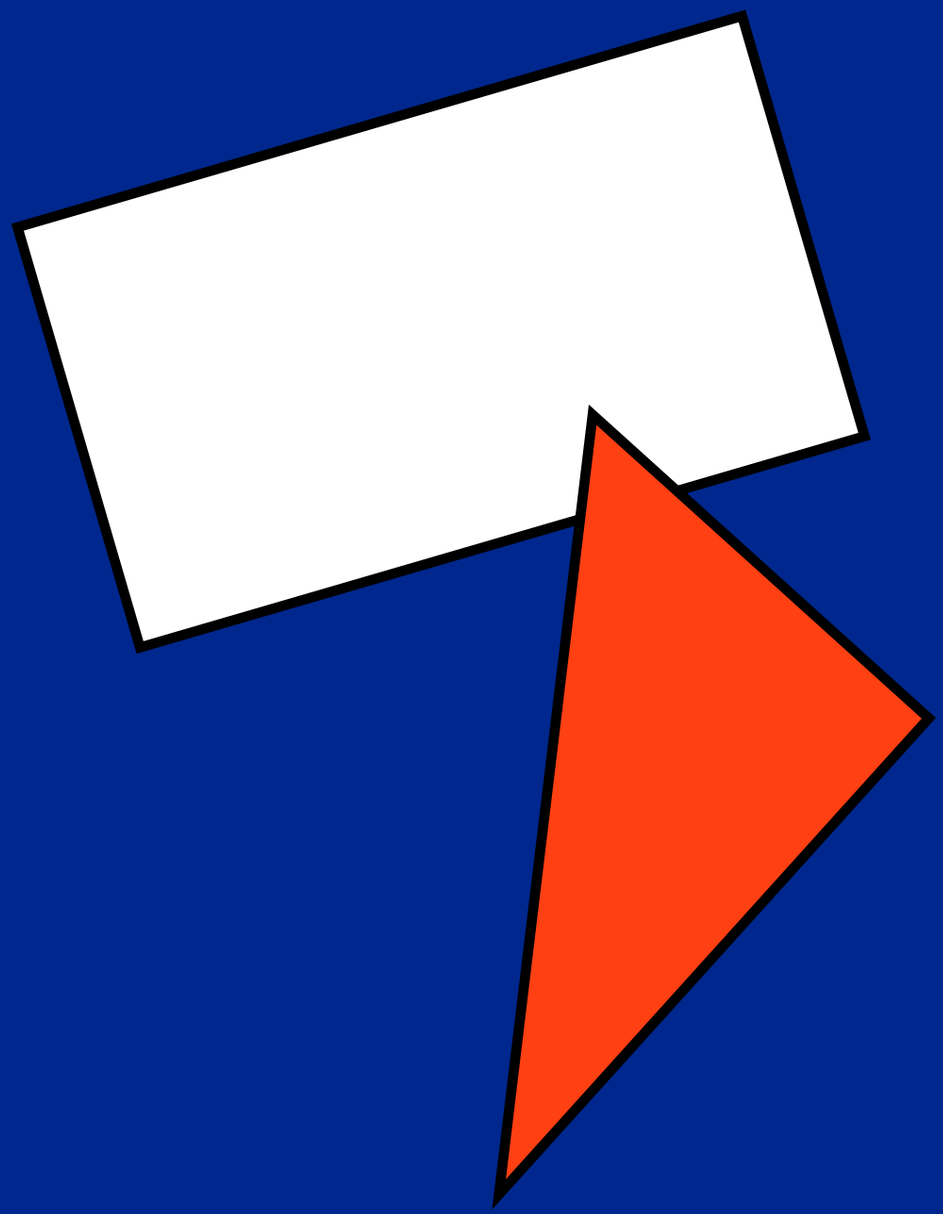


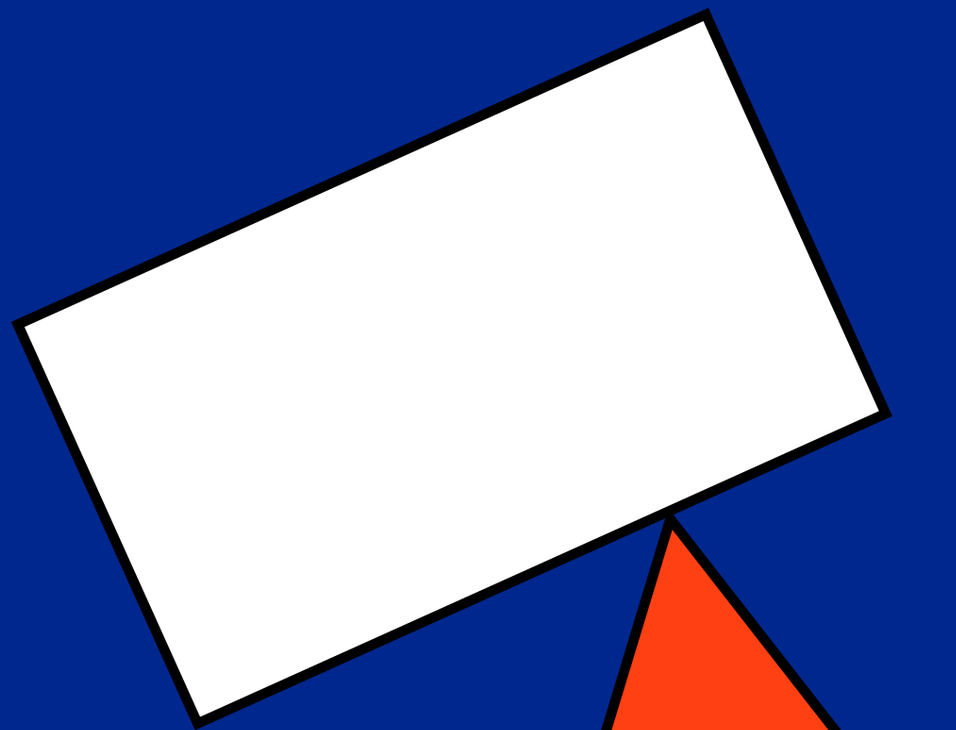
- Later: rigid body collision and contact.
- For now, just simple point-plane collisions.
- Add-ons for a particle simulator.

Collisions: Two Parts

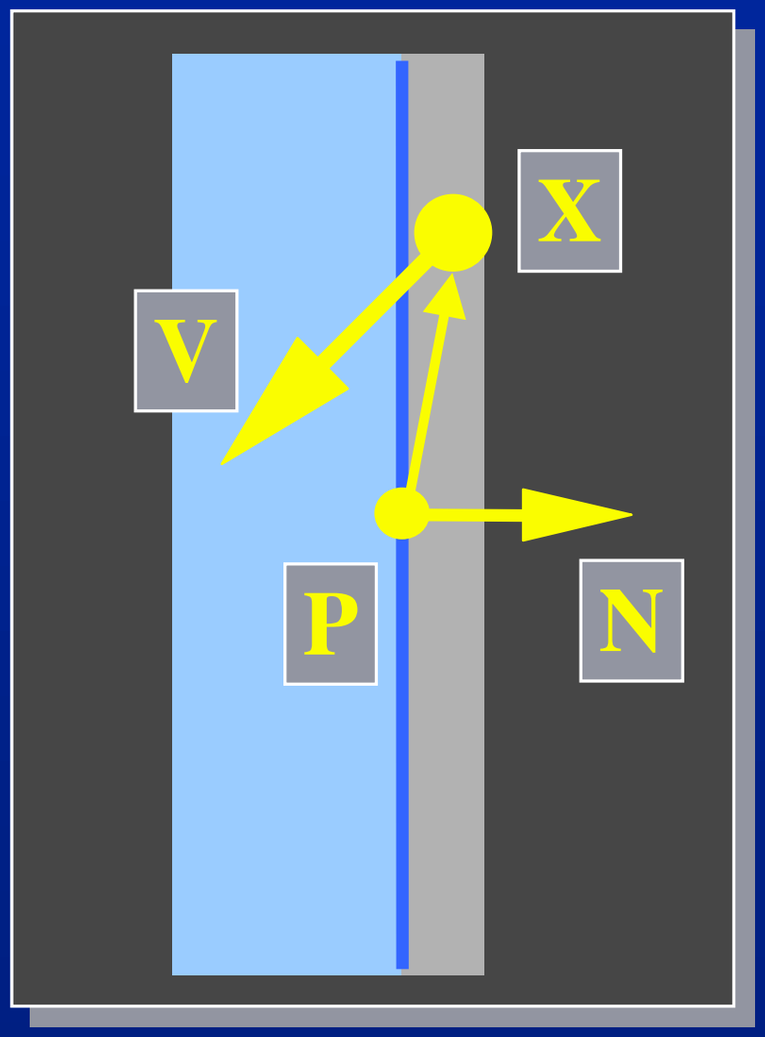
Detection

Response





Collision Detection

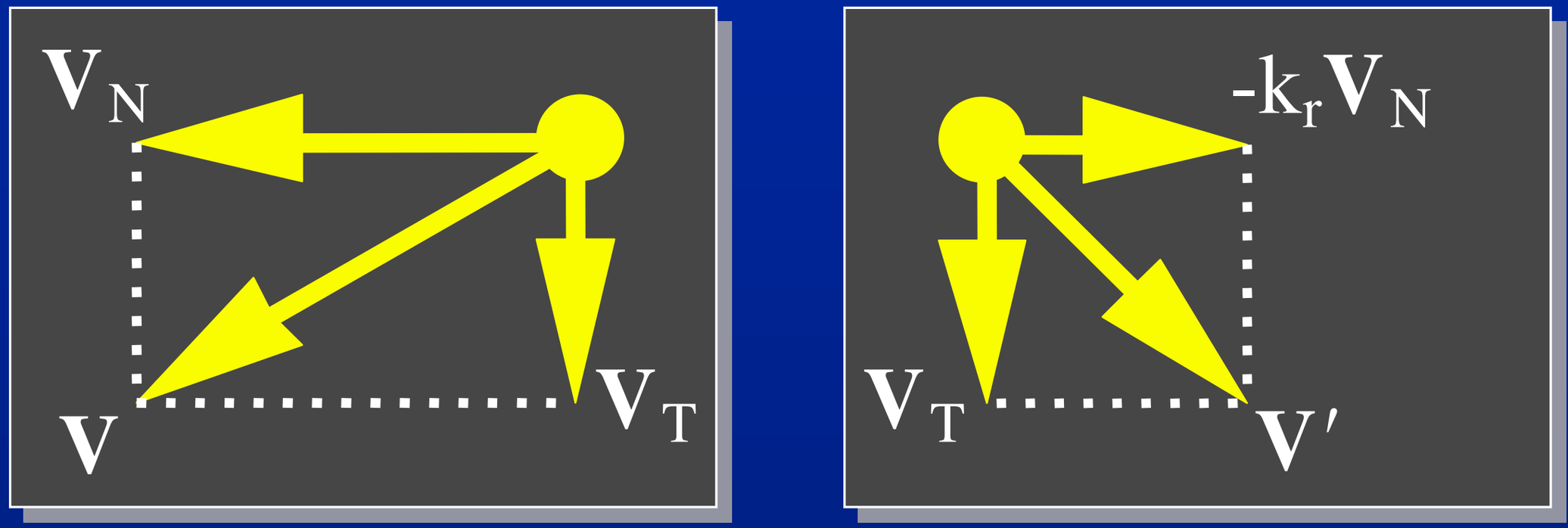


$$(\mathbf{X} - \mathbf{P}) \cdot \mathbf{N} < \varepsilon$$

$$\mathbf{N} \cdot \mathbf{V} < 0$$

- Within ε of the wall.
- Heading in.

Collision Response

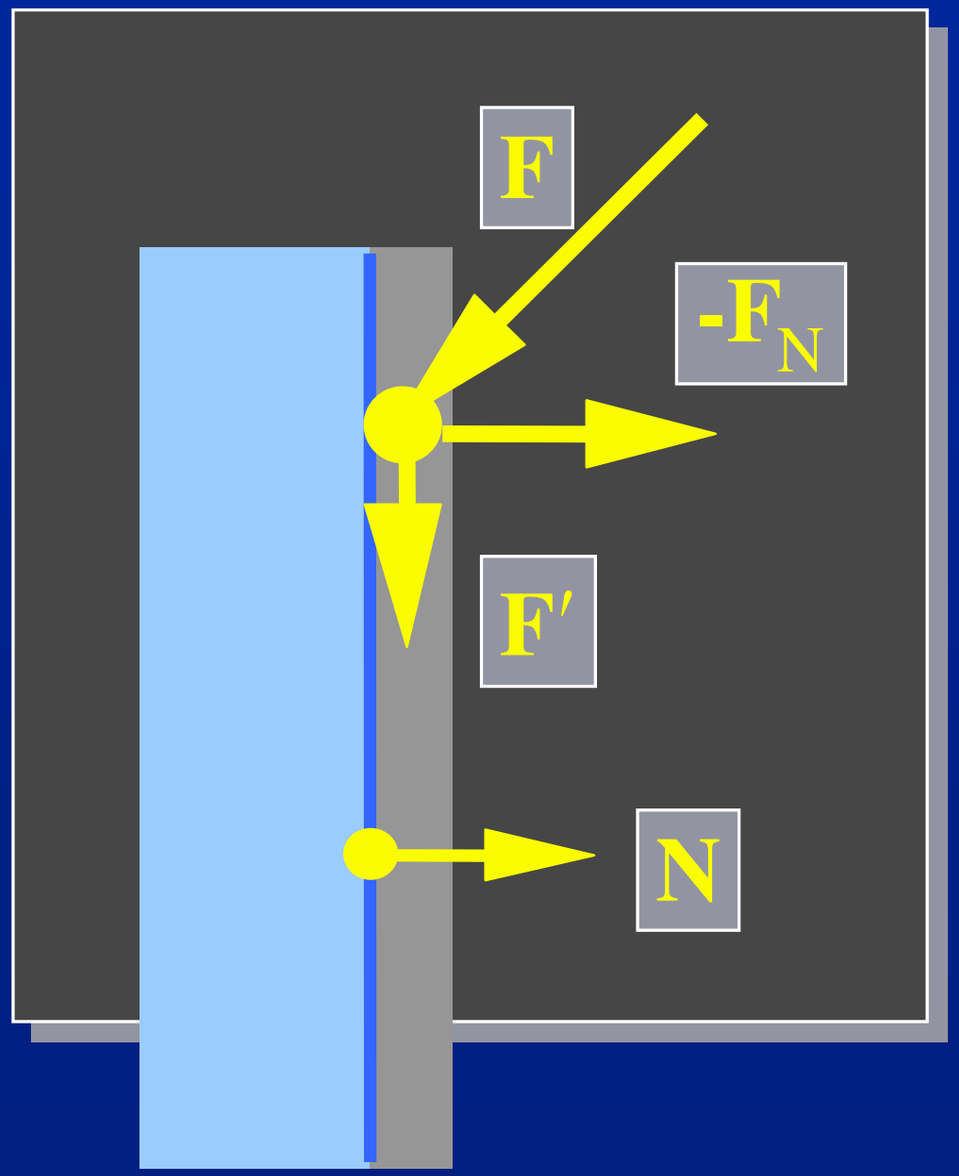


Before

After

$$V' = V_T - k_r V_N$$

Contact Force



$\mathbf{F}' = \mathbf{F}_T$

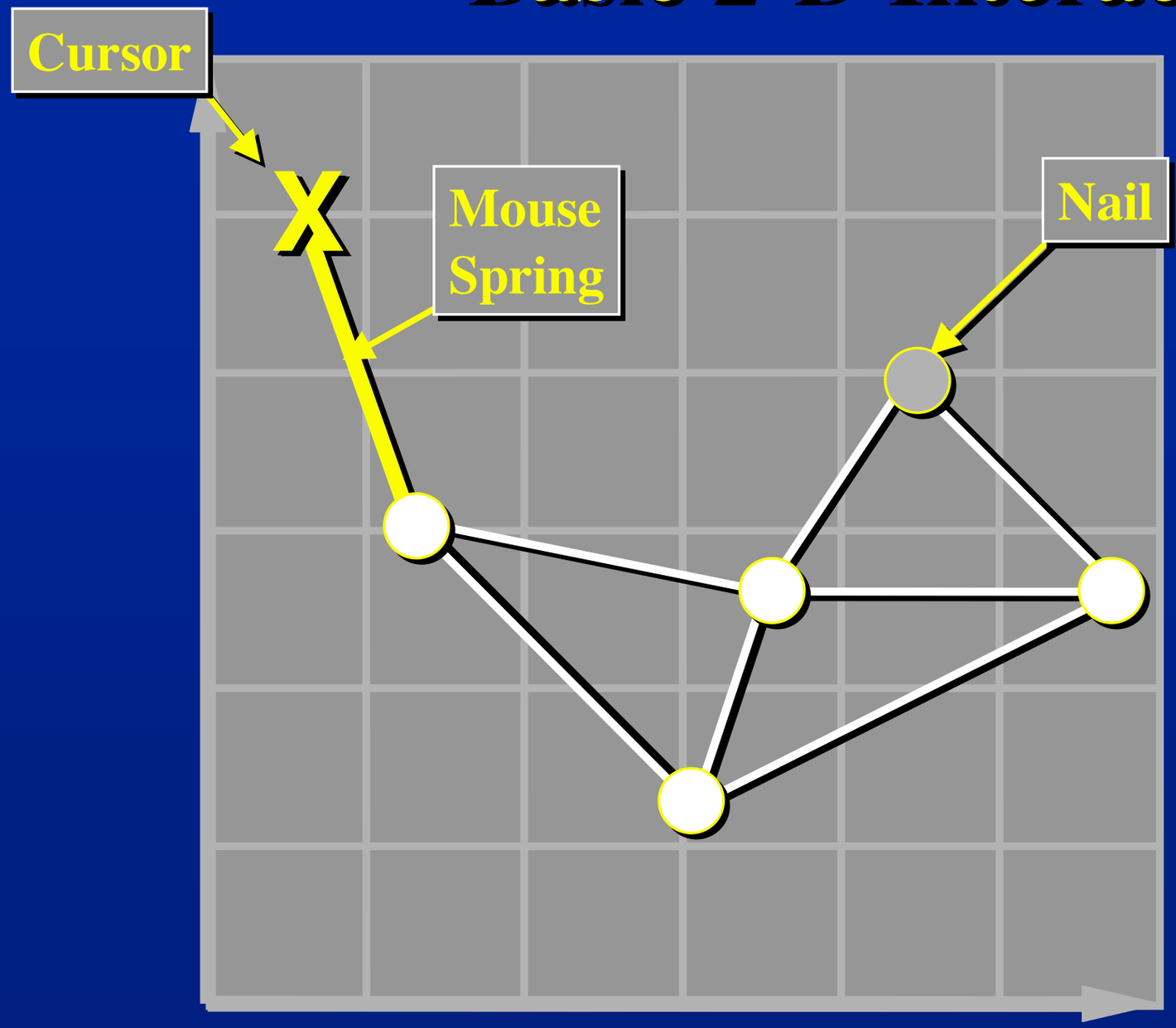
The wall pushes back, cancelling the normal component of F.

(An example of a constraint force.)

Try this at home!

The notes give you everything you need to build a basic interactive mass/spring simulator—try it.

Basic 2-D Interaction



Operations:

- Create
- Attach
- Drag
- Nail