vriphys 06

Madrid, Spain, November 2006

# Position Based Dynamics


Matthias Müller Bruno Heidelberger Marcus Hennix John Ratcliff

ageia

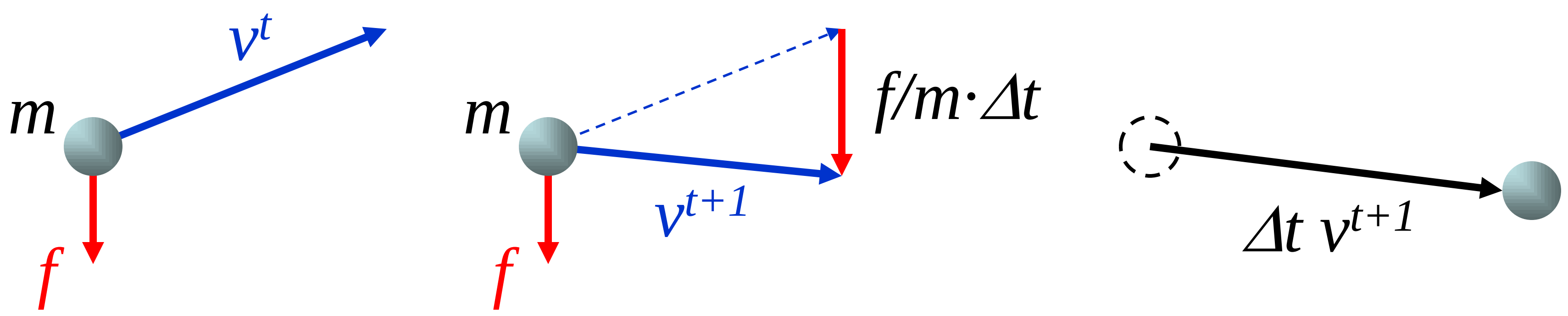

# Dynamics in Games


Cell factor

Bet on soldier

# Simulating a Dynamical System

- **Explicit Euler integration:**




- **Accuracy problem: no issue in a game**
- **Stability problem: big issue in a game**

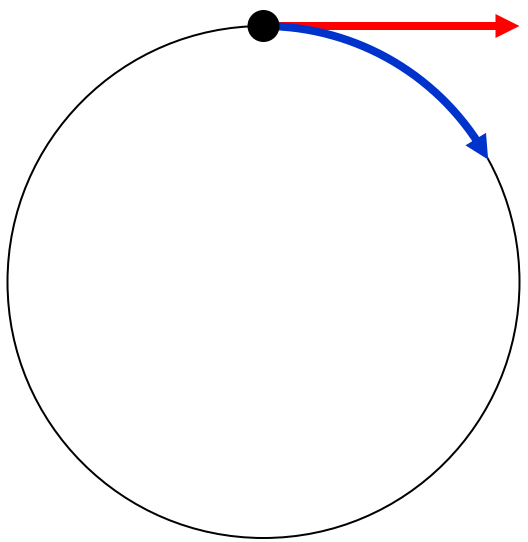
PhysX™ by ageia

# Overshooting Problem

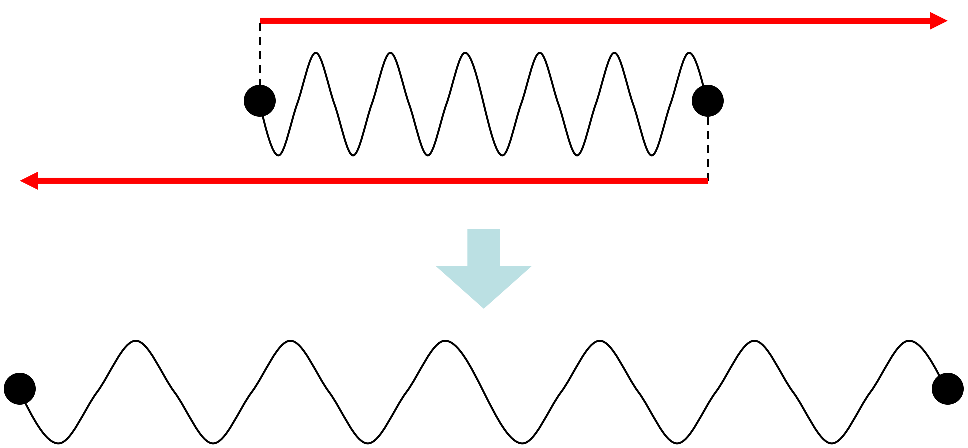
$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \mathbf{v}(t)\cdot\Delta t$$

**vs.**

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \int_{t}^{t+\Delta t} \mathbf{v}(t)\,dt$$

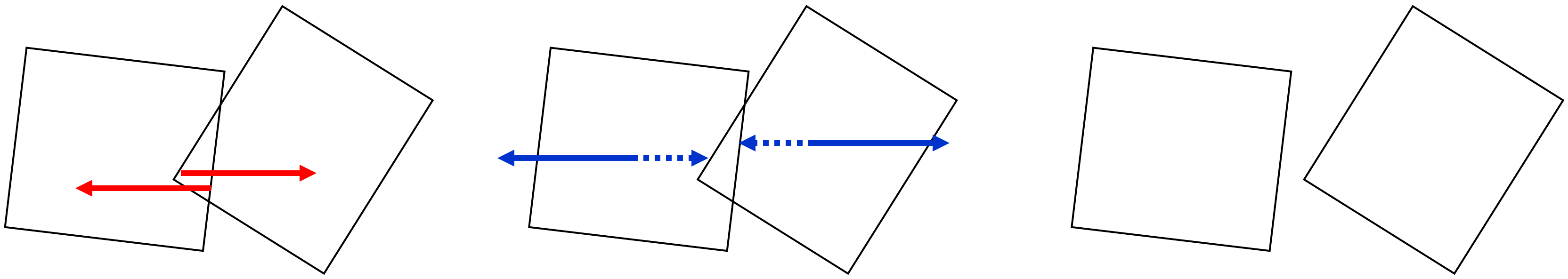


**outwards spin**



**amplitude build up**

# Force Based Update



penetration causes forces

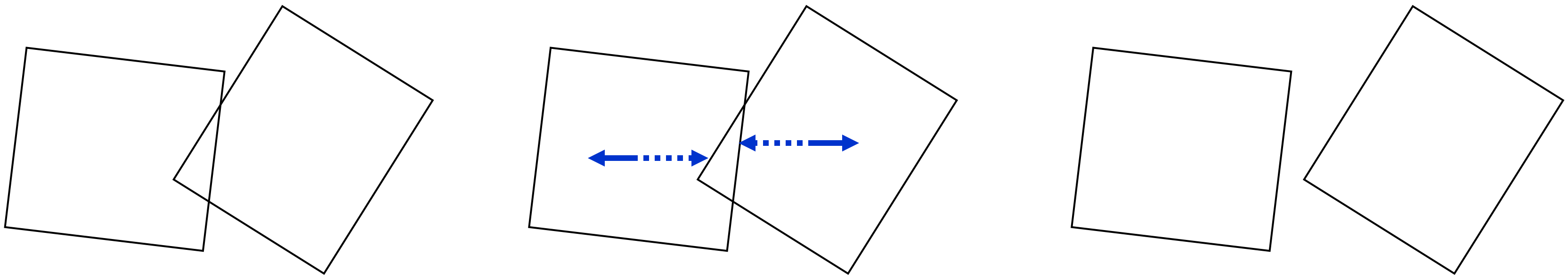
forces change velocities

velocities change positions

- Need overlap
- Reaction lag
- Strong force → stiff system, overshooting
- Weak force → squishy system

PhysX™ by ageia

# Velocity Based Update



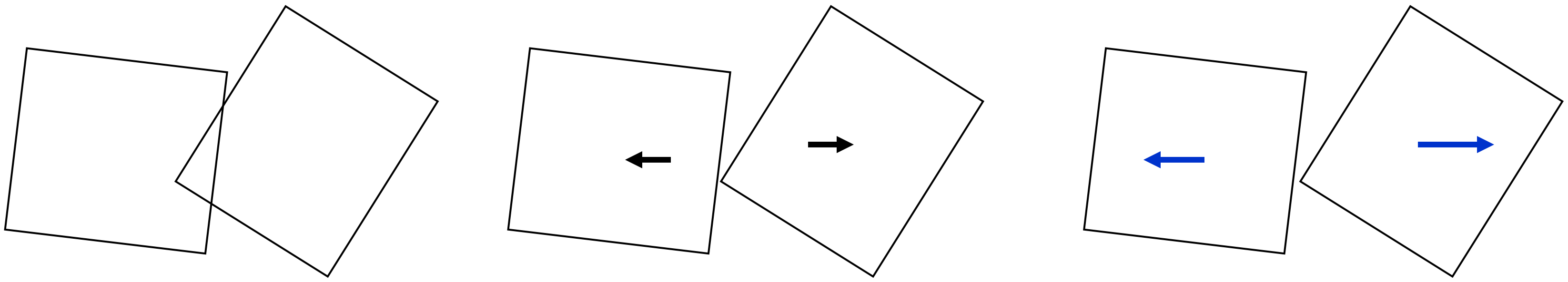
penetration detection only

change velocities so that they separate objects

velocities change positions

- Controlled velocity change (via impulses)
- Only as much as needed → no overshooting
- Drift: Consistent velocities to not guarantee consistent positions!

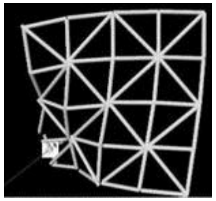
# Position Based Update



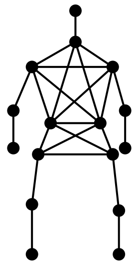

- Controlled position change
- Only as much as needed → no overshooting
- No drift
- Velocity update needed to get 2nd order system!

# Related Work

- Verlet Integration
- Faure, Interactive solid animation using linearized displacement constraints, EGCAS 1998



- Jakobsen, Advanced character physics & the fysx engine, GDC 2001



- Fedor, Fast character animation using particle dynamics, GVIP 2005
- Müller et al., Meshless deformations based on shape matching, Siggraph 2005

# Verlet Integration

- **Derivation**

$$\mathbf{x}(t+\Delta t)=\mathbf{x}(t)+\mathbf{v}(t)\cdot\Delta t+\frac{1}{2}\mathbf{a}(t)\cdot\Delta t^{2}+\frac{1}{2}\dddot{\mathbf{x}}(t)\cdot\Delta t^{3}+O(\Delta t^{4})$$

$$+\quad \mathbf{x}(t-\Delta t)=\mathbf{x}(t)-\mathbf{v}(t)\cdot\Delta t+\frac{1}{2}\mathbf{a}(t)\cdot\Delta t^{2}-\frac{1}{2}\dddot{\mathbf{x}}(t)\cdot\Delta t^{3}+O(\Delta t^{4})$$

---

$$\mathbf{x}(t+\Delta t)+\mathbf{x}(t-\Delta t)=2\mathbf{x}(t)+\Delta t^{2}\mathbf{a}(t)+O(\Delta t^{4})$$

$$\boxed{\mathbf{x}(t+\Delta t)=\mathbf{x}(t)+\underbrace{\left[x(t)-x(t-\Delta t)\right]}_{}+\Delta t^{2}\mathbf{a}(t)+O(\Delta t^{4})}$$

- **Velocity stored implicitly in the previous position**

PhysX™ by ageia

# Position Based Integration

- Verlet: $\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \left[x(t) - x(t-\Delta t)\right] + \Delta t^2 \mathbf{a}(t) + O(\Delta t^4)$

Init $\mathbf{x}(0)$, $\mathbf{v}(0)$

**Loop**

$\mathbf{x}^p(t+\Delta t) = \mathbf{x}(t) + \Delta t \cdot \mathbf{v}(t)$ // prediction

$\mathbf{x}(t+\Delta t)$ = modify $\mathbf{x}^p(t+\Delta t)$ // position correction

$\mathbf{v}^p(t+\Delta t) = [\mathbf{x}(t+\Delta t) - \mathbf{x}(t)] / \Delta t$ // velocity update

$\mathbf{v}(t+\Delta t)$ = modify $\mathbf{v}^p(t+\Delta t)$ // velocity correction

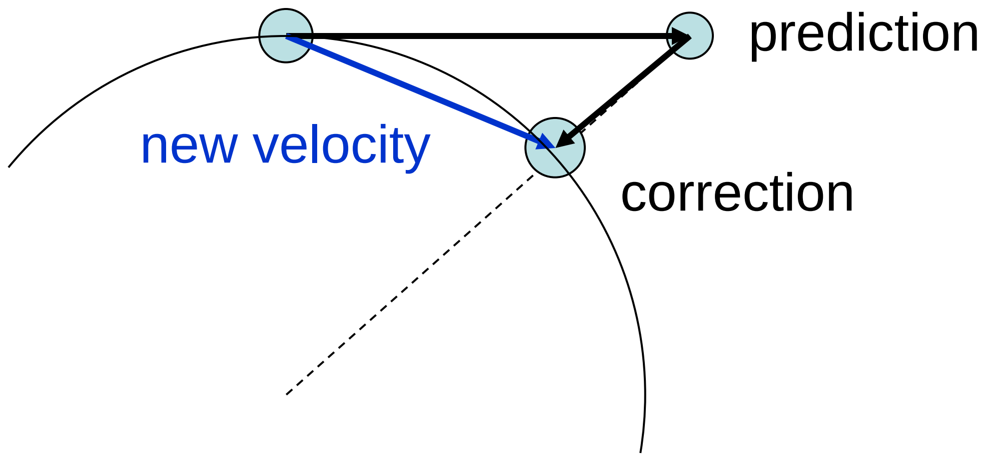
**End loop**

- Verlet plus position / velocity corrections
- Corrections change the dynamic state!

PhysX by ageia

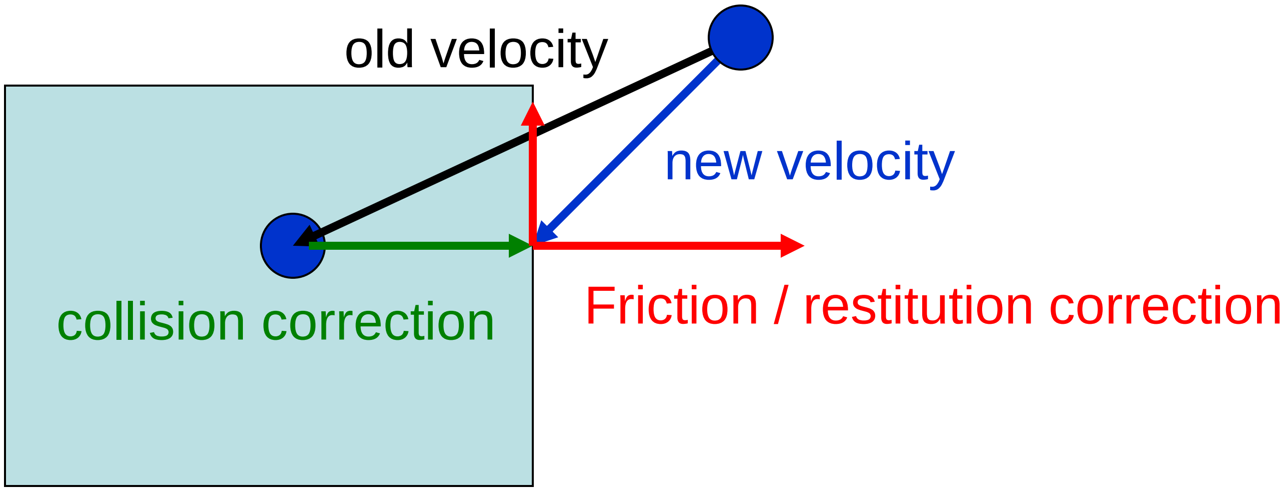
# Position Correction

- Move vertices out of other objects
- Move vertices such that constraints are satisfied
- Example: Particle on circle



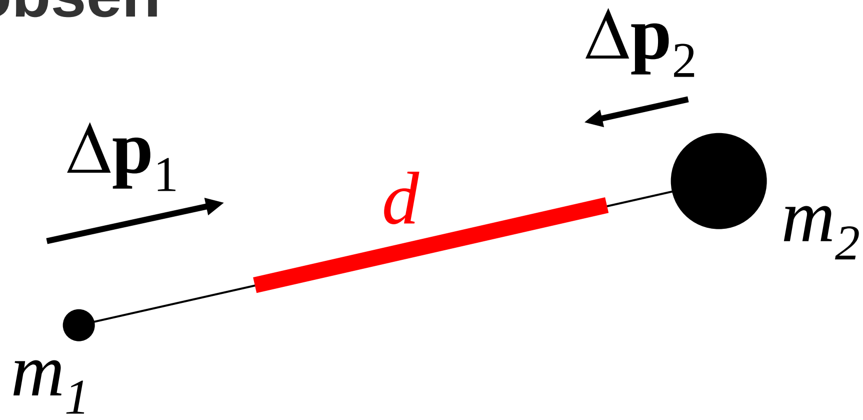
# Velocity Correction

- External forces: $\mathbf{v}(t+\Delta t) = \mathbf{v}^{\mathbf{p}}(t+\Delta t) + \Delta t \cdot \mathbf{f}(t)/m$
- Internal damping
- Friction
- Restitution



# Internal Distance Constraint

- Jakobsen




$$\Delta \mathbf{p}_1 = -\frac{w_1}{w_1 + w_2}\left(|\mathbf{p}_1 - \mathbf{p}_2| - d\right)\frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|}$$

$$w_i = \frac{1}{m_i}$$

$$\Delta \mathbf{p}_2 = +\frac{w_2}{w_1 + w_2}\left(|\mathbf{p}_1 - \mathbf{p}_2| - d\right)\frac{\mathbf{p}_1 - \mathbf{p}_2}{|\mathbf{p}_1 - \mathbf{p}_2|}$$

# General Internal Constraints

- **Scalar constraint function**

$$C(\mathbf{p}) = 0 \quad \Leftrightarrow \text{constraint satisfied}$$

- **Compute $\Delta$p such that**

$$C(\mathbf{p} + \Delta\mathbf{p}) \cong C(\mathbf{p}) + \nabla_{\mathbf{p}} C(\mathbf{p}) \cdot \Delta\mathbf{p} = 0$$

- **Rigid body modes do not change $C(\mathbf{p})$**
- **Do not influence rigid body modes (ghost forces)**
- **Search perpendicular to rigid body modes: $\Delta\mathbf{p} = \lambda \nabla_{\mathbf{p}} C(\mathbf{p})$**

$$\Delta\mathbf{p} = -\frac{C(\mathbf{p})}{\left|\nabla_{\mathbf{p}} C(\mathbf{p})\right|^2} \nabla_{\mathbf{p}} C(\mathbf{p})$$

PhysX™ by ageia

# General Internal Constraints

- General correction for $n$ point constraints
- Including masses

$$\Delta \mathbf{p}_i = -s \frac{n \cdot w_i}{\sum_j w_j} \nabla_{\mathbf{p}_i} C(\mathbf{p}_1, \ldots, \mathbf{p}_n)$$

$$s = \frac{C(\mathbf{p}_1, \ldots, \mathbf{p}_n)}{\sum_j \left| \nabla_{\mathbf{p}} C(\mathbf{p}_1, \ldots, \mathbf{p}_n) \right|^2}$$

- Examples

$$C_{stretch}(\mathbf{p}_1, \mathbf{p}_2) = \left| \mathbf{p}_1 - \mathbf{p}_2 \right| - d \quad \rightarrow \text{Jakobsen}$$

$$C_{bend}(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) = \arccos\left( \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)}{\left|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_3 - \mathbf{p}_1)\right|} \bullet \frac{(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)}{\left|(\mathbf{p}_2 - \mathbf{p}_1) \times (\mathbf{p}_4 - \mathbf{p}_1)\right|} \right) - \varphi_0$$

$$C_{volume}(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4) = \left[ (\mathbf{p}_2 - \mathbf{p}_1 \times (\mathbf{p}_3 - \mathbf{p}_1) \right] \cdot (\mathbf{p}_4 - \mathbf{p}_1) - v_0$$

# Position Solver

- **Non-linear Gauss Seidel**
- **Iterate**
  - Go through all constraints
    - Move (project) points according to the constraint

- **Remarks**
  - Gauss Seidel is order dependent
  - Last constraints are strongest
  - Projection is a non-linear step
  - Takes time for pressure waves to propagate through objects

# Results

# Conclusions / Future Work