Tema 5 - Física y colisiones. Efectos especiales.

5.1 Introducción a los motores físicos.

Germán Arroyo, Juan Carlos Torres

5 de febrero de 2021

Contenido del tema

Tema 5: Física y colisiones. Efectos especiales.

5.1 Introducción a los motores físicos.

5.2 Interacción con dispositivos de entrada y dispositiv

5.3 Técnicas de optimización.

5.4 Personalización de fuerzas

5.5 Efectos especiales y técnicas volumétricas.

5.6 Shaders de vértices y técnicas avanzadas.

5.1 Introducción a los motores físicos.

La simulación física trata de reproducir el comportamiento dinámico y cinemático de los objetos de la escena.

Implica:

- Representar el estado de los objetos: posición, velociad, aceleración y momento angular.
- Representar propiedades físicas de los objetos: densidad, elasticidad, coeficiente de fricción.
- Resolver las ecuaciones de la mecánica del sistema (integración en el tiempo).

Componentes

Detección de colisiones:

- Cinemática.
- Cálculo de velocidades.
- Dinámica.
- Cálculo de fuerzas.
- Fractura.

Objetos

- Puntual.
- Rígido.
- Deformable.
- Fluido.
- Tela.

What is Time Integration ?

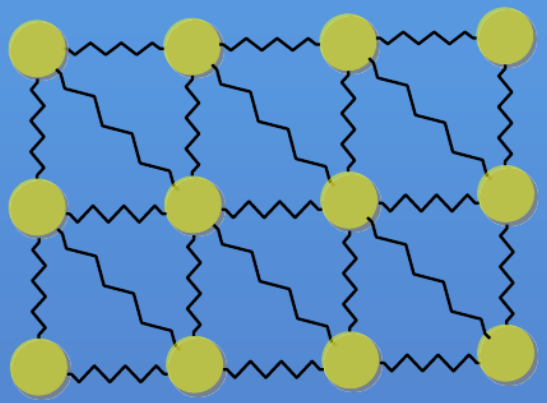
- Computing the simulation state at the next time step *t+h* given :
  - State at the current time *t*
    - positions $\mathbf{x}_t$
    - velocities $\mathbf{v}_t$
  - Equations and constraints expressing the required mechanics
    - $\mathbf{a} = a(\mathbf{x}, \mathbf{v}, t)$
    - $c(x, v, t) \geq 0$
- Main issue : lively but stable simulation

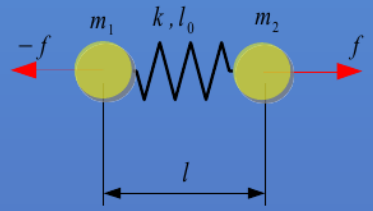
$$\mathbf{v}(t) = \mathbf{v}(0) + \int_0^t \mathbf{a}(\mathbf{x}, \mathbf{v}, t)dt$$

$$\mathbf{x}(t) = \mathbf{x}(0) + \int_0^t \mathbf{v}(t)dt$$

Example : Springs

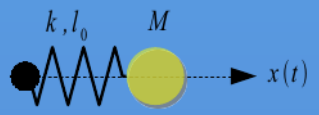
- Particle
  - mass *m*,
  - position, velocity
- Spring
  - rest length l_0
  - stiffness *k*
- Force
  - $l = (x_2 - x_1)$
  - $f = k(l - l_0)$
- *Acceleration*
  - $a = f / m$



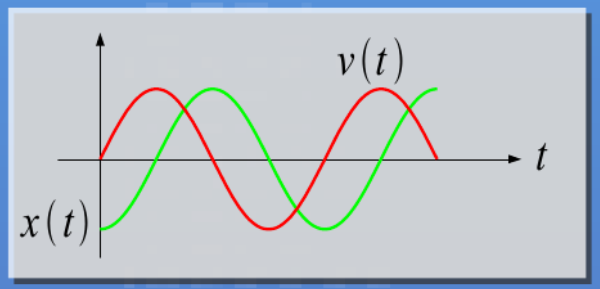


Métodos de simulación física (III)

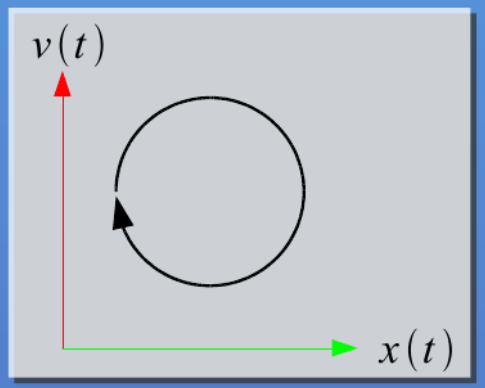
Simplest Case



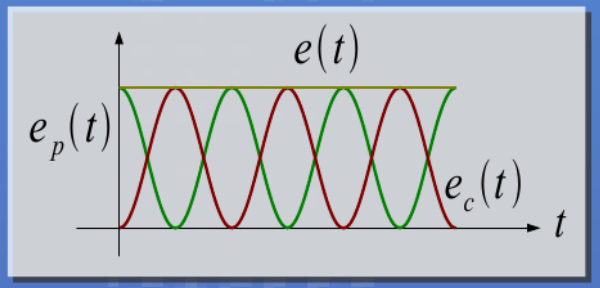
- Single 1D Particle
- Theoretical solution :



in phase space :



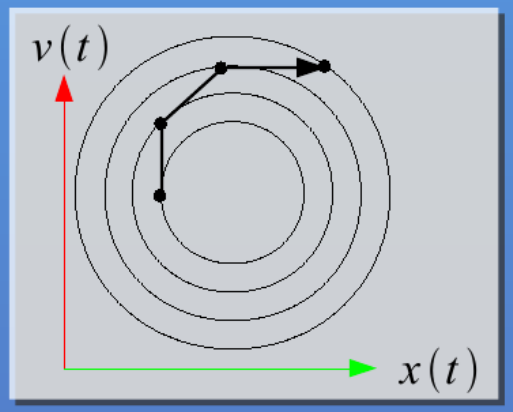
- Conservation of Energy :



Explicit Integration

- Principle : use velocity and acceleration at the **begin** of the time-step
- Forward Euler :

$$\begin{aligned} \mathbf{a}_t &= \mathbf{M}^{-1}\mathbf{f} \\ \mathbf{x}_{t+h} &= \mathbf{x}_t + h\mathbf{v}_t \\ \mathbf{v}_{t+h} &= \mathbf{v}_t + h\mathbf{a}_t \end{aligned}$$



- Simple
- Exagerates motion
- Damping is required
- Stiff systems require very small time steps

Métodos de simulación física (V)

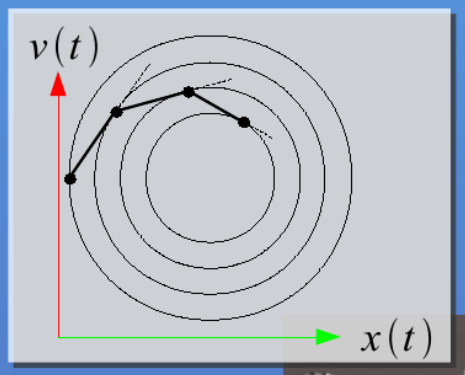
Implicit Integration

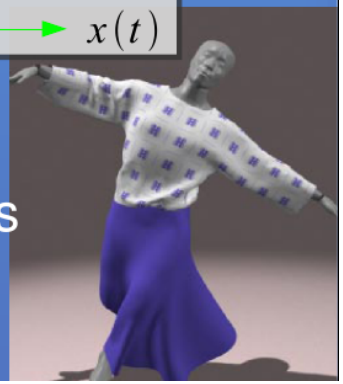
- Principle : use velocity and acceleration at the **end** of the time-step
- Backward Euler :
  - Solve $\left(\mathbf{M} - \frac{1}{h}\frac{\partial \mathbf{f}}{\partial \mathbf{v}} - \frac{1}{h^2}\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right)\mathbf{a} = \left(\mathbf{f} + h\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\mathbf{v}\right)$

$$\mathbf{v}_{t+h} = \mathbf{v}_t + h\mathbf{a}_{t+h}$$

$$\mathbf{x}_{t+h} = \mathbf{x}_t + h\mathbf{v}_{t+h}$$

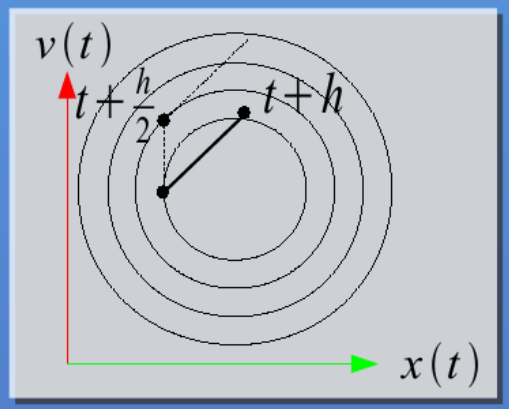
- Complex (requires a solver)
- Under-estimates motion
- Introduces additional damping
- Stable stiff systems even with large timesteps
- Well-suited for soft bodies





Explicit Runge-Kutta Methods

- 2nd-order Runge-Kutta (RK2) :
  - Go to *t + h/2* using forward Euler
  - Compute the derivative
  - Use this derivative in a full forward Euler step
- Simple
- More precise than forward Euler
- Still exagerates motion
- Well-suited for rigids



Colisiones en sistemas dinámicos (I)

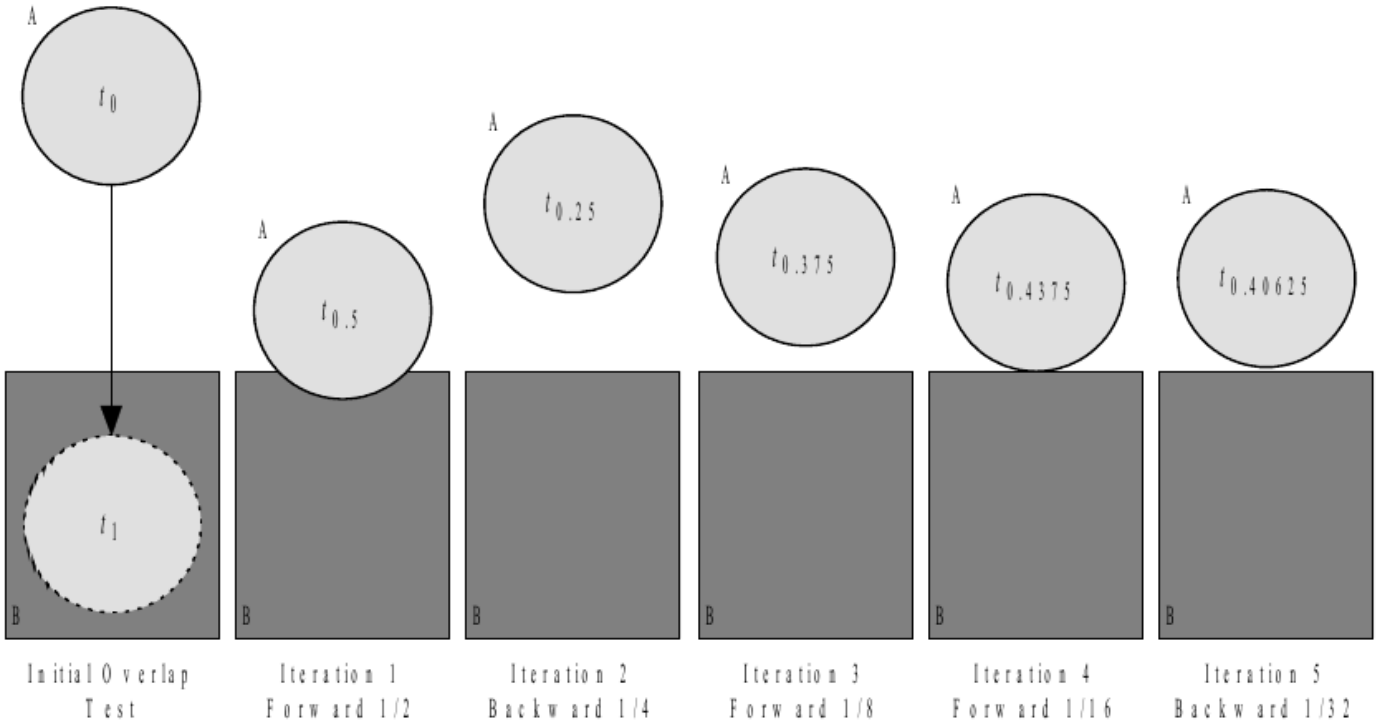
Escenas dinámicas: Si al menos uno de los objetos está en movimiento, a partir de sus posiciones iniciales y finales.

- El objetivo es determinar el punto de contacto.
- En simulación necesitamos además saber el tiempo de la colisión.

Colisiones en sistemas dinámicos (II)

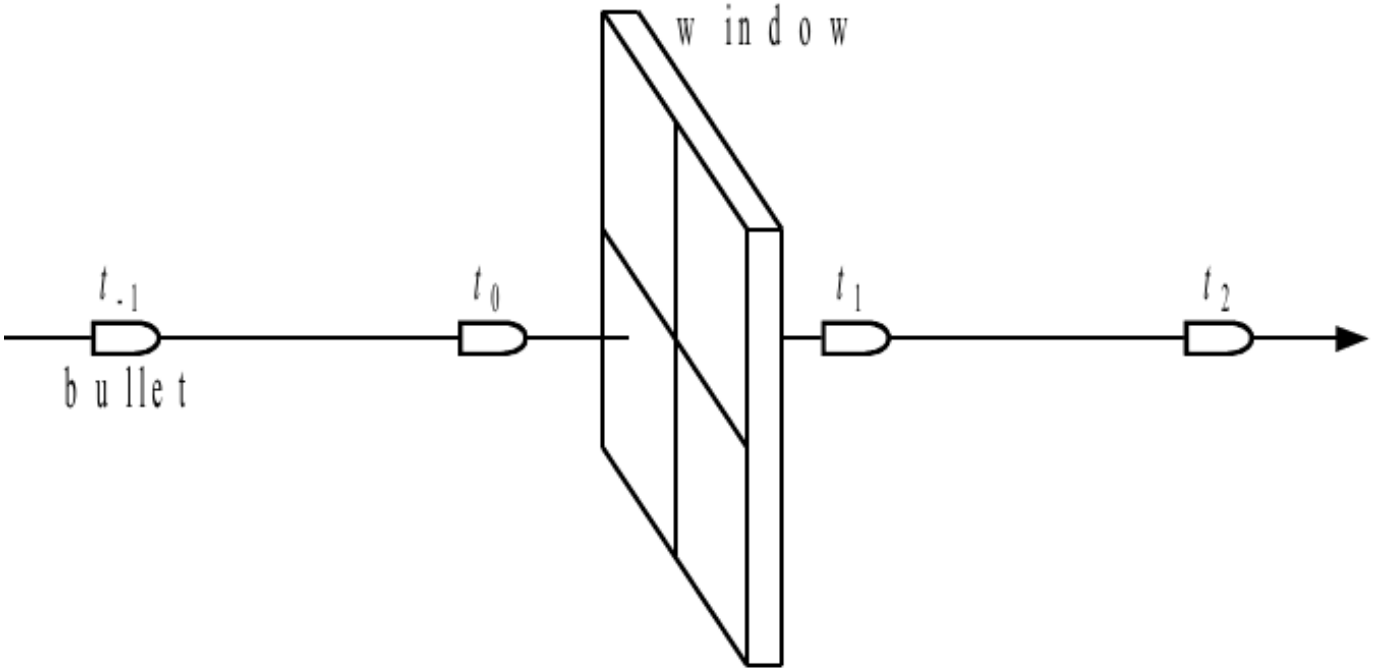
Tiempo de colisión: Se puede calcular retrocediendo en el tiempo hasta el momento de colisión.

- Se pueden usar técnicas de bisección.



Colisiones en sistemas dinámicos (III)

Cálculo de colisiones: Determinar superposición puede hacer que no se detecten colisiones si el tiempo de integración es muy alto.



Simulación física

La simulación es costosa:

- **Tiempo real.** Cálculo aproximado: Juegos, Entornos virtuales, Sistemas interactivos.
- **Precisos (lentos).** Películas, Aplicaciones científicas y técnicas.

Motores físicos

Open source:

- ODE.
- NEWTON.
- Bullet.

Comerciales:

- Havok (Intel).
- Physx (nVidia).
- Vortex (Montreal).
- Realflow (Nextlimit, España).