

Game Programming

Particles Physics

Frederic Koriche

LIRMM, Université Montpellier II

Parcours Intelligence Artificielle
Master Informatique 2ème Année
Septembre 2008

1 Particles

- The Particle Object
- Kinetics Laws
- Laws of Motion
- Motion Algorithms
- Gravitational Forces

2 Mass-Aggregate Systems

- Spring Forces
- Springlike Systems
- Anchors
- Buoyancy

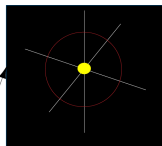
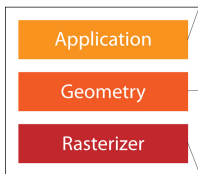
Outline

1 Particles

- The Particle Object
- Kinetics Laws
- Laws of Motion
- Motion Algorithms
- Gravitational Forces

2 Mass-Aggregate Systems

- Spring Forces
- Springlike Systems
- Anchors
- Buoyancy



Particle

A particle is an object which has a position but no orientation: we cannot tell in what direction the object is pointing.

Attributes

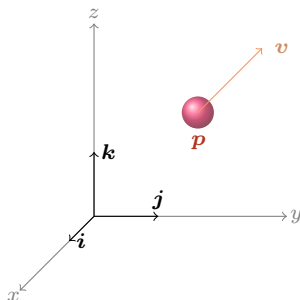
The *attributes* of a particle are:

■ *mass*: $m > 0$

■ *position*: $\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$

■ *velocity*: $\mathbf{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$

■ *acceleration*: $\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$



Normalization

■ *velocity*: $\mathbf{v} = v \hat{\mathbf{v}}$

■ *acceleration*: $\mathbf{a} = a \hat{\mathbf{a}}$

Attributes

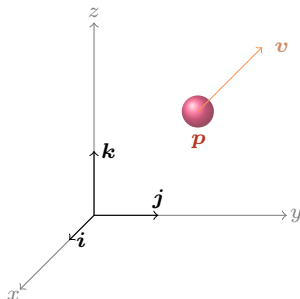
The *attributes* of a particle are:

■ *mass*: $m > 0$

■ *position*: $\mathbf{p} = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$

■ *velocity*: $\mathbf{v} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix}$

■ *acceleration*: $\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$



Normalization

■ *velocity*: $\mathbf{v} = v\hat{\mathbf{v}}$

■ *acceleration*: $\mathbf{a} = a\hat{\mathbf{a}}$

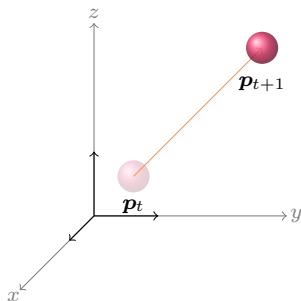
Velocity

The velocity is the gradient of the position

$$\mathbf{v} = \begin{bmatrix} \frac{\partial p_x}{\partial t} \\ \frac{\partial p_y}{\partial t} \\ \frac{\partial p_z}{\partial t} \end{bmatrix}$$

If a particle is moving with constant velocity during an interval of length τ then

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}\tau$$



Acceleration

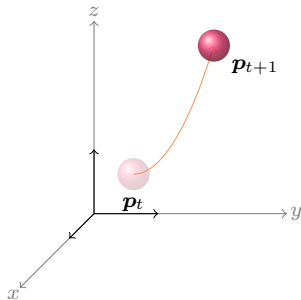
The acceleration is the gradient of the velocity

$$\mathbf{a} = \begin{bmatrix} \frac{\partial v_x}{\partial t} \\ \frac{\partial v_y}{\partial t} \\ \frac{\partial v_z}{\partial t} \end{bmatrix}$$

If a particle is moving with constant acceleration during an interval of length δ then

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \mathbf{a}\tau$$

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}_t\tau + \mathbf{a}\frac{\tau^2}{2}$$



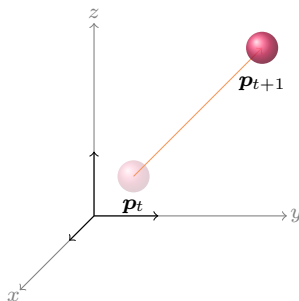
Velocity

The velocity is the gradient of the position

$$\mathbf{v} = \begin{bmatrix} \frac{\partial p_x}{\partial t} \\ \frac{\partial p_y}{\partial t} \\ \frac{\partial p_z}{\partial t} \end{bmatrix}$$

If a particle is moving with constant velocity during an interval of length τ then

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}\tau$$



Acceleration

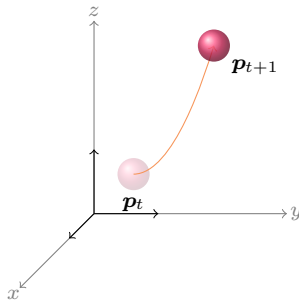
The acceleration is the gradient of the velocity

$$\mathbf{a} = \begin{bmatrix} \frac{\partial v_x}{\partial t} \\ \frac{\partial v_y}{\partial t} \\ \frac{\partial v_z}{\partial t} \end{bmatrix}$$

If a particle is moving with constant acceleration during an interval of length δ then

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \mathbf{a}\tau$$

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \mathbf{v}_t\tau + \mathbf{a}\frac{\tau^2}{2}$$



1st Law

A particle will continue at a constant velocity unless acted upon by an external force.

2nd Law

A force acting on a particle produces an acceleration that is proportional to its mass.

$$a = \frac{1}{m} f$$

3rd Law

Whenever a particle A exerts a force on another particle B , B simultaneously exerts a force on A with the same magnitude in the opposite direction.

$$f_A = -f_B$$

D'Alembert Principle

All forces acting on an object can be replaced with a single force.

$$a_t = \frac{1}{m} \sum_{i=1}^n f_{t,i}$$

1st Law

A particle will continue at a constant velocity unless acted upon by an external force.

2nd Law

A force acting on a particle produces an acceleration that is proportional to its mass.

$$\mathbf{a} = \frac{1}{m} \mathbf{f}$$

3rd Law

Whenever a particle A exerts a force on another particle B , B simultaneously exerts a force on A with the same magnitude in the opposite direction.

$$\mathbf{f}_A = -\mathbf{f}_B$$

D'Alembert Principle

All forces acting on an object can be replaced with a single force.

$$\mathbf{a}_t = \frac{1}{m} \sum_{i=1}^n \mathbf{f}_{t,i}$$

1st Law

A particle will continue at a constant velocity unless acted upon by an external force.

2nd Law

A force acting on a particle produces an acceleration that is proportional to its mass.

$$\mathbf{a} = \frac{1}{m} \mathbf{f}$$

3rd Law

Whenever a particle A exerts a force on another particle B , B simultaneously exerts a force on A with the same magnitude in the opposite direction.

$$\mathbf{f}_A = -\mathbf{f}_B$$

D'Alembert Principle

All forces acting on an object can be replaced with a single force.

$$\mathbf{a}_t = \frac{1}{m} \sum_{i=1}^n \mathbf{f}_{t,i}$$

1st Law

A particle will continue at a constant velocity unless acted upon by an external force.

2nd Law

A force acting on a particle produces an acceleration that is proportional to its mass.

$$\mathbf{a} = \frac{1}{m} \mathbf{f}$$

3rd Law

Whenever a particle A exerts a force on another particle B , B simultaneously exerts a force on A with the same magnitude in the opposite direction.

$$\mathbf{f}_A = -\mathbf{f}_B$$

D'Alembert Principle

All forces acting on an object can be replaced with a single force.

$$\mathbf{a}_t = \frac{1}{m} \sum_{i=1}^n \mathbf{f}_{t,i}$$

Class Particle

Attributes

Particle state

Vector3 \mathbf{p} ;

Vector3 \mathbf{v} ;

Vector3 \mathbf{a} ;

Force accumulator

Vector3 \mathbf{f} ;

Inverse of the mass

float im;

Methods

Clear force accumulator

```
void Clear()
{  $\mathbf{f}$ .zero(); }
```

Collect force

```
void Add(const Vector3D&  $\mathbf{g}$ )
{  $\mathbf{f} += \mathbf{g}$ ; }
```

Update particle

```
void Update(const float  $\tau$ );
```

Integration Method

Let t be the frame interval. Based on Taylor's series:

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \tau \mathbf{p}'_t + \frac{\tau^2}{2} \mathbf{p}''_t + \dots + \frac{\tau^n}{n!} \mathbf{p}^{(n)} + \dots$$

If τ is small and \mathbf{p}'_t is nearly constant during τ , then

$$\mathbf{p}_{t+1} \sim \mathbf{p}_t + \tau \mathbf{p}'_t$$

Update(const float t)

Update acceleration

```
 $\mathbf{a} = \text{im} * \mathbf{f}$ ;
 $\mathbf{a}$ .clean();
```

Update position

```
 $\mathbf{p} += \mathbf{v} * t$ ;
```

Update velocity

```
 $\mathbf{v} += \mathbf{a} * t$ ;
 $\mathbf{a}$ .clean();
```

Class Particle

Attributes

Particle state

Vector3 \mathbf{p} ;
 Vector3 \mathbf{v} ;
 Vector3 \mathbf{a} ;

Force accumulator

Vector3 \mathbf{f} ;

Inverse of the mass

float im;

Methods

Clear force accumulator

```
void Clear()
{  $\mathbf{f}$ .zero(); }
```

Collect force

```
void Add(const Vector3D&  $\mathbf{g}$ )
{  $\mathbf{f} += \mathbf{g}$ ; }
```

Update particle

```
void Update(const float  $\tau$ );
```

Integration Method

Let t be the frame interval. Based on Taylor's series:

$$\mathbf{p}_{t+1} = \mathbf{p}_t + \tau \mathbf{p}'_t + \frac{\tau^2}{2} \mathbf{p}''_t + \dots + \frac{\tau^n}{n!} \mathbf{p}^{(n)}_t + \dots$$

If τ is small and \mathbf{p}'_t is nearly constant during τ , then

$$\mathbf{p}_{t+1} \sim \mathbf{p}_t + \tau \mathbf{p}'_t$$

Update(const float t)

Update acceleration

```
 $\mathbf{a} = \text{im} * \mathbf{f}$ ;  

 $\mathbf{a}$ .clean();
```

Update position

```
 $\mathbf{p} += \mathbf{v} * t$ ;
```

Update velocity

```
 $\mathbf{v} += \mathbf{a} * t$ ;  

 $\mathbf{a}$ .clean();
```

Gravitational Forces

Gravity applies between every pair of objects by attracting them together with a force that depends on their mass and distance apart.

$$f = G \frac{m_A m_B}{r^2}$$

- m_A and m_B are the masses of objects
- r is their distance
- G is the universal gravity constant

$$G = 6.67300 \times 10^{-11} \text{ m}^3 \text{ kg}^{-1} \text{ s}^{-2}$$

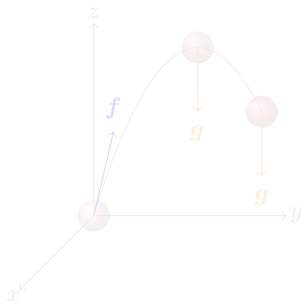
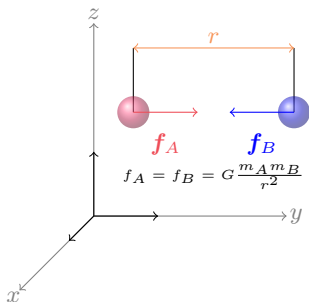
Earth gravity

For each object of mass m on Earth

$$f_{\text{gra}} = m \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}$$

The value of g

Real world	$g = 9.807 \text{ ms}^{-2}$
Shooters	$g \sim 15 \text{ ms}^{-2}$



Gravitational Forces

Gravity applies between every pair of objects by attracting them together with a force that depends on their mass and distance apart.

$$f = G \frac{m_A m_B}{r^2}$$

- m_A and m_B are the masses of objects
- r is their distance
- G is the universal gravity constant

$$G = 6.67300 \times 10^{-11} \text{m}^3 \text{kg}^{-1} \text{s}^{-2}$$

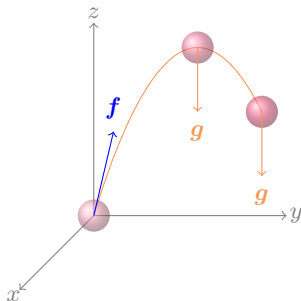
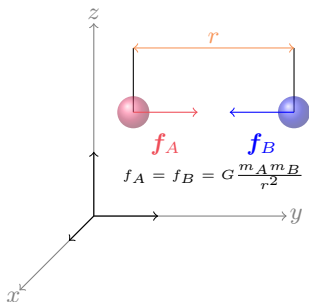
Earth gravity

For each object of mass m on Earth

$$\mathbf{f}_{\text{gra}} = m \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix}$$

The value of g

Real world	$g = 9.807 \text{ms}^{-2}$
Shooters	$g \sim 15 \text{ms}^{-2}$



Outline

1 Particles

- The Particle Object
- Kinetics Laws
- Laws of Motion
- Motion Algorithms
- Gravitational Forces

2 Mass-Aggregate Systems

- Spring Forces
- Springlike Systems
- Anchors
- Buoyancy

Spring Forces

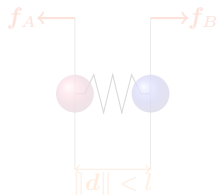
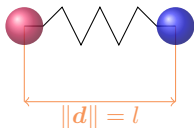
Suppose that two objects A and B are connected by a spring. If the spring is extended or compressed, the objects will be attracted or repulsed by the same force, given by *Hook's law*:

$$\mathbf{f} = -k(\|\mathbf{d}\| - l)\hat{\mathbf{d}}$$

- k is the *spring constant*
- l is the *length at rest*
- \mathbf{d} is the vector from the end of the spring attached to the object we are generating a force.

$$\mathbf{d} = \mathbf{p}_A - \mathbf{p}_B \text{ for object } A$$

$$\mathbf{d} = \mathbf{p}_B - \mathbf{p}_A \text{ for object } B$$



Spring Forces

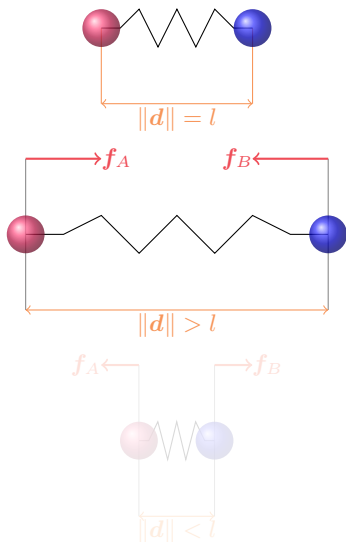
Suppose that two objects A and B are connected by a spring. If the spring is extended or compressed, the objects will be attracted or repulsed by the same force, given by *Hook's law*:

$$\mathbf{f} = -k(\|\mathbf{d}\| - l)\hat{\mathbf{d}}$$

- k is the *spring constant*
- l is the *length at rest*
- \mathbf{d} is the vector from the end of the spring attached to the object we are generating a force.

$$\mathbf{d} = \mathbf{p}_A - \mathbf{p}_B \text{ for object } A$$

$$\mathbf{d} = \mathbf{p}_B - \mathbf{p}_A \text{ for object } B$$



Spring Forces

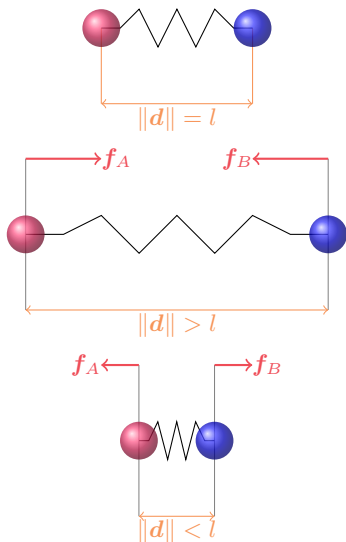
Suppose that two objects A and B are connected by a spring. If the spring is extended or compressed, the objects will be attracted or repulsed by the same force, given by *Hook's law*:

$$\mathbf{f} = -k(\|\mathbf{d}\| - l)\hat{\mathbf{d}}$$

- k is the *spring constant*
- l is the *length at rest*
- \mathbf{d} is the vector from the end of the spring attached to the object we are generating a force.

$$\mathbf{d} = \mathbf{p}_A - \mathbf{p}_B \text{ for object } A$$

$$\mathbf{d} = \mathbf{p}_B - \mathbf{p}_A \text{ for object } B$$



Class SpringedParticle

Attributes

Particle

Vector3 p ;Vector3 v ;Vector3 a ;Vector3 f ;float im ;

Spring components

Particle* other;

float k ;float l ;

Methods

Clear force accumulator

void Clear();

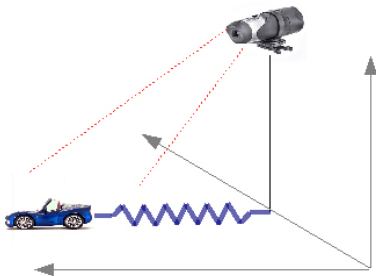
Collect force

void Add(const Vector3D& g);

Update particle

void Update(const float τ);

Set spring force

void SetSpringForce(const float τ);SetSpringForce(const float t)

Set force to difference

Vector3 h ; $h = p - \text{other}.p$;

Calculate magnitude

float magnitude = h .magnitude();magnitude = abs(magnitude - l);magnitude *= k ;

Calculate final force

 h .normalize(); $h *= -\text{magnitude}$;

Class SpringedParticle

Attributes

Particle

Vector3 p ;Vector3 v ;Vector3 a ;Vector3 f ;float im ;

Spring components

Particle* other;

float k ;float l ;

Methods

Clear force accumulator

void Clear();

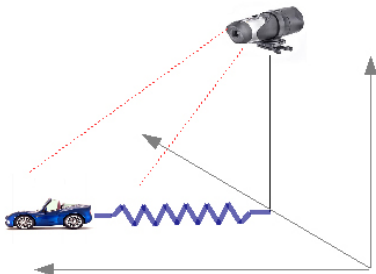
Collect force

void Add(const Vector3D& g);

Update particle

void Update(const float τ);

Set spring force

void SetSpringForce(const float τ);SetSpringForce(const float t)

Set force to difference

Vector3 h ; $h = p - \text{other}.p$;

Calculate magnitude

float magnitude = h .magnitude();magnitude = abs(magnitude - l);magnitude *= k ;

Calculate final force

 h .normalize(); h *= -magnitude;

Class AnchoredParticle

Attributes

Particle

Vector3 p ;

Vector3 v ;

Vector3 a ;

Vector3 f ;

float im ;

Anchor components

Vector3 **anchor**;

float k ;

float l ;

Methods

Clear force accumulator

```
void Clear();
```

Collect force

```
void Add(const Vector3D& g);
```

Update particle

```
void Update(const float  $\tau$ );
```

Set spring force

```
void SetSpringForce(const float  $\tau$ );
```



SetSpringForce(const float t)

Set force to difference

```
Vector3  $h$ ;
```

```
 $h = p - \text{anchor}$ ;
```

Calculate magnitude

```
float magnitude =  $h$ .magnitude();
```

```
magnitude = abs(magnitude -  $l$ );
```

```
magnitude *=  $k$ ;
```

Calculate final force

```
 $h$ .normalize();
```

```
 $h$  *= -magnitude;
```

Class BuoyancyParticle

Attributes

Particle

Vector3 p ;

Vector3 v ;

Vector3 a ;

Vector3 f ;

float im ;

Submersion depth

float maxDepth;

Object volume

float volume;

Liquid height

float liquidHeight;

Liquid density

float liquidDensity;

Methods

Clear force accumulator

void Clear();

Collect force

void Add(const Vector3D& g);

Update particle

void Update(const float τ);

Set buoyancy force

void SetBuoyancyForce(const float τ);



SetBuoyancyForce()

Vector3 **force**(0,0,0);

float depth = $p.z$;

Check if we are out of water

if(depth >= liquidHeight + subDepth)

return **force**;

Check if we are at max depth

else if(depth <= liquidHeight - subDepth)

force.y = liquidDensity * volume;

return **force**;

Otherwise we are partly submerged

else

float d = (depth - maxDepth - waterHeight) / (2 * maxDepth);

force.y = liquidDensity * volume * d;