Modelado y Animación por Computador

Tema 2: Modelado

Dr. Miguel Davia Aracil

Tema 2: Modelado

1.- Introducción
2.- Modelos geométricos de representación
3.- Técnicas de modelado
4.- Transformaciones geométricas
5.- Deformadores
6.- Sistemas de partículas
7.- Fuerzas
8.- Efectos atmosféricos

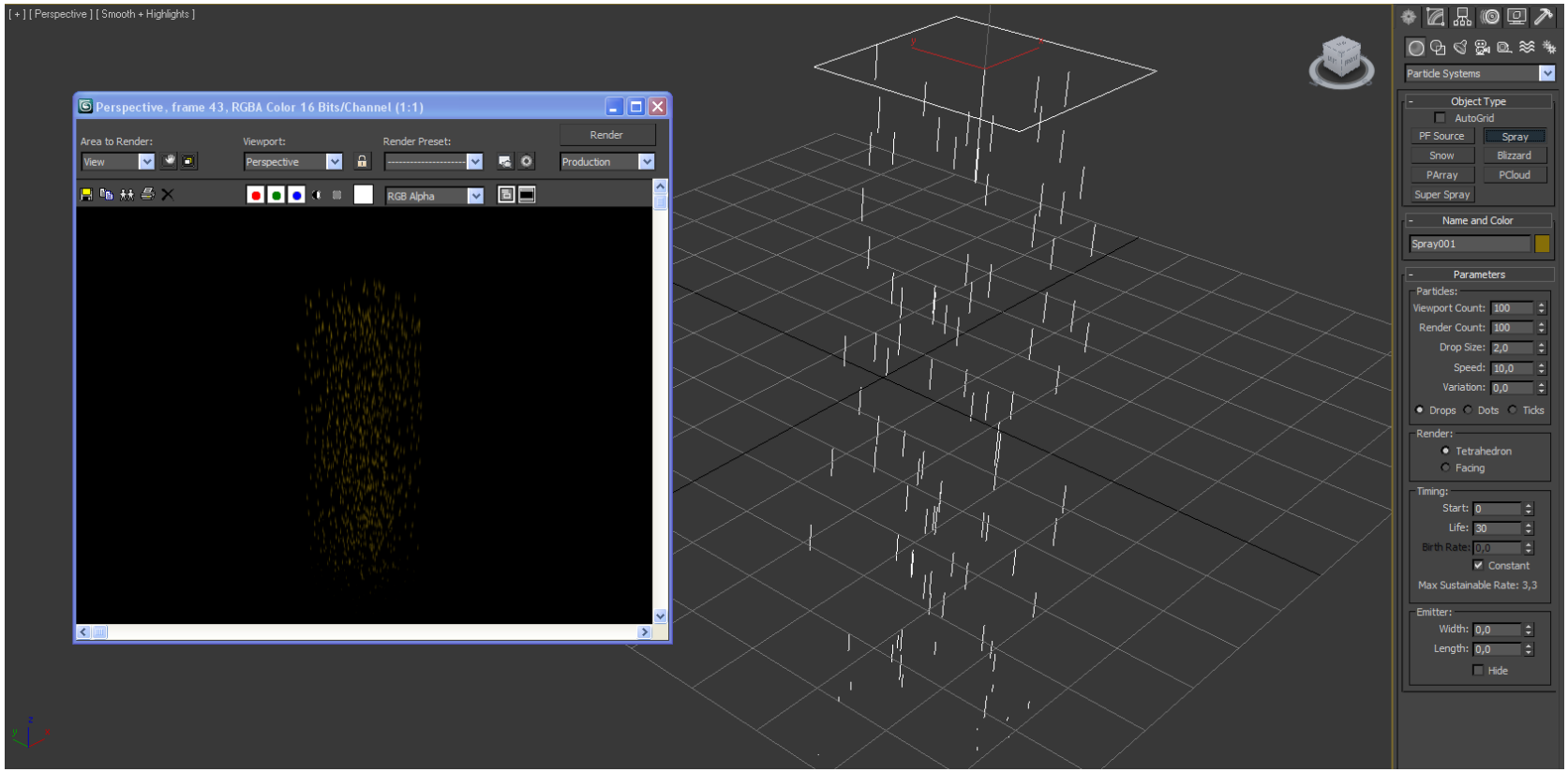
Sistemas de partículas

Introducción

- Los sistemas de partículas han sido ampliamente utilizados en animación por computador y generación de efectos especiales desde su introducción en la industria en los años 80.
- Las reglas que rigen el comportamiento de una partícula son relativamente simples, la complejidad viene determinada por el comportamiento de grandes cantidades de ellas.
- Normalmente, las partículas se regirán por una combinación de leyes físicas dependiendo de su posición exacta en el espacio.

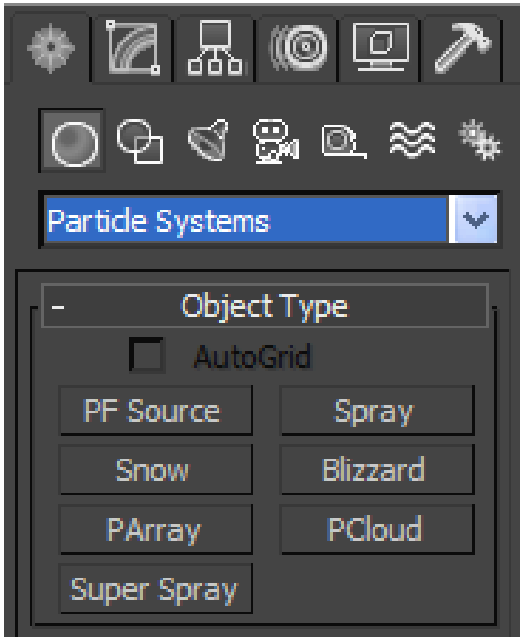
Sistemas de partículas

3D Studio Max. Geometry – Particle Systems



Sistemas de partículas

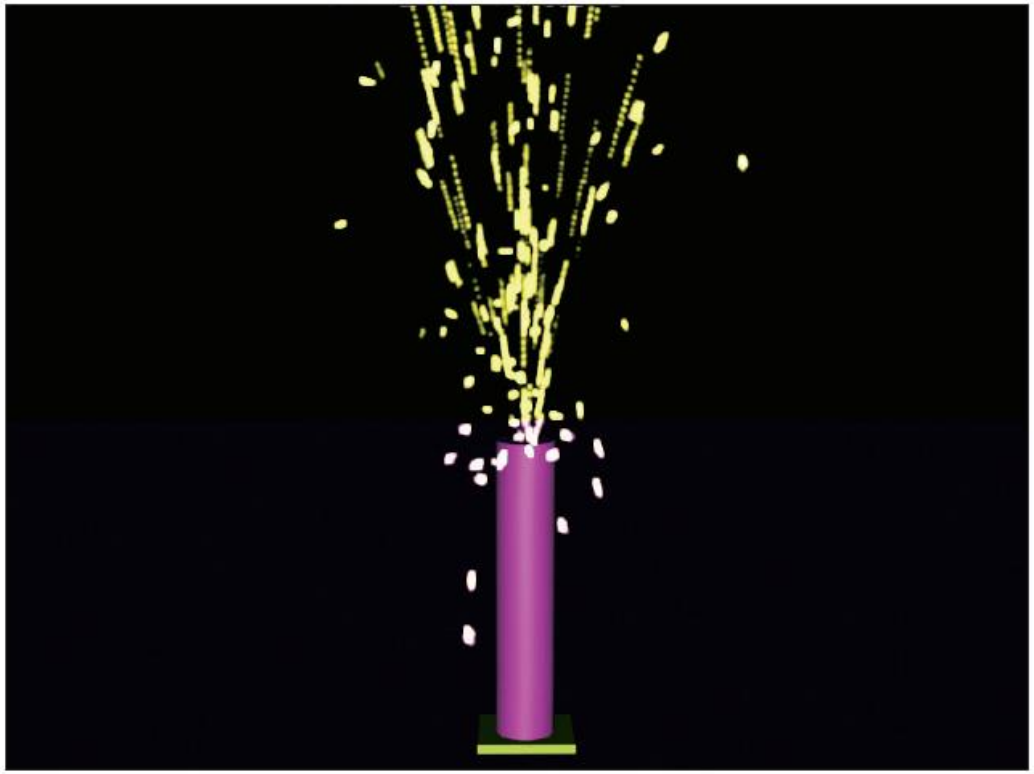
3D Studio Max. Geometry – Particle Systems



Sistemas de partículas

3D Studio Max. Geometry – Particle Systems

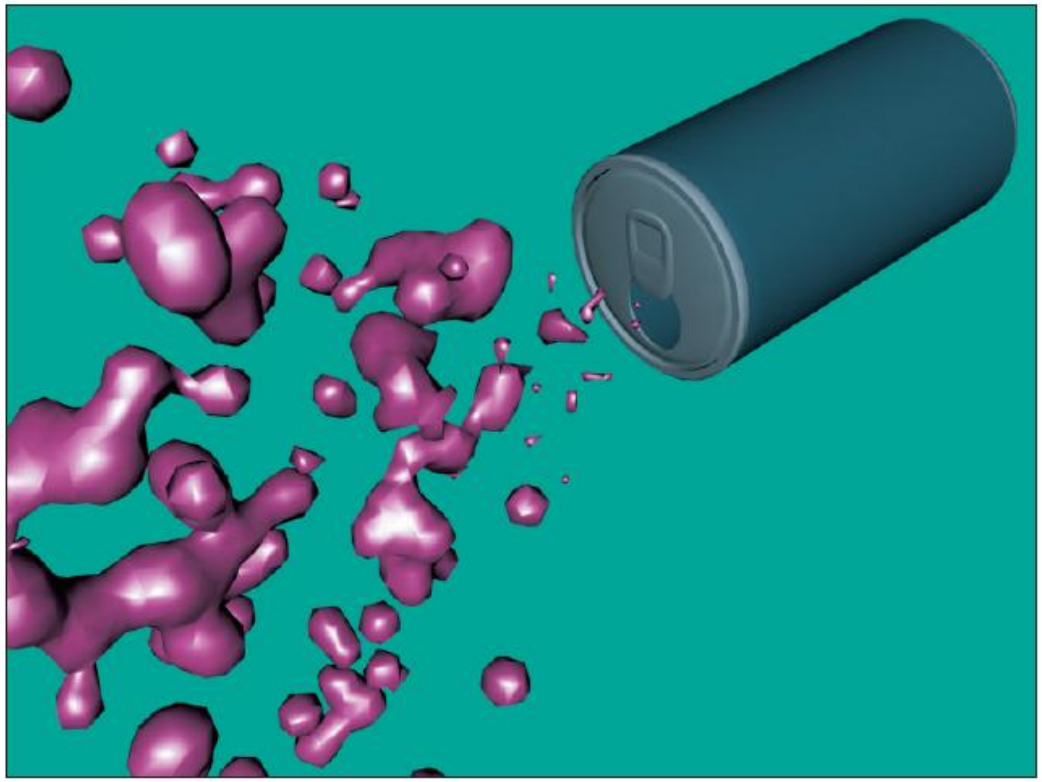
The Super Spray particle system is used to create fireworks sparks.



Sistemas de partículas

3D Studio Max. Geometry – Particle Systems

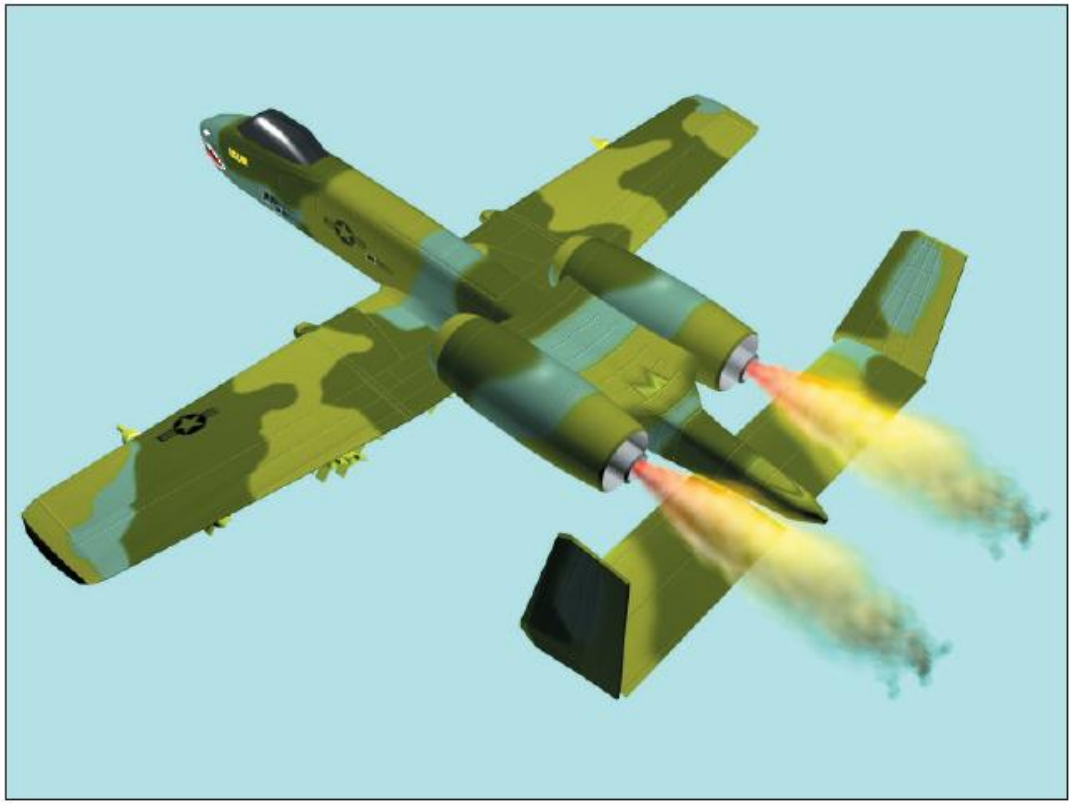
MetaParticles emitting from the opening of a soda can



Sistemas de partículas

3D Studio Max. Geometry – Particle Systems

Realistic jet flames created using the Particle Age and MBlur maps



Sistemas de partículas

3D Studio Max. Geometry – Particle Systems

A Plane object positioned beneath the vent is an emitter for the particle system.

Sistemas de partículas

Cinemática de partículas

- Se define la posición de una partícula 3D en función del tiempo como $\mathbf{r}(t)$
- Por definición, la **velocidad** es la primera derivada de la posición y la **aceleración** es la segunda derivada
- Para **renderizar** una partícula es necesario conocer su posición **r**

$$\mathbf{r} = \mathbf{r}(t)$$

$$\mathbf{v} = \frac{d\mathbf{r}}{dt}$$

$$\mathbf{a} = \frac{d\mathbf{v}}{dt} = \frac{d^2\mathbf{r}}{dt^2}$$

Sistemas de partículas

Aceleración uniforme

- ¿Cómo se mueve una partícula cuando está sujeta a una aceleración constante?

$$\mathbf{a} = \mathbf{a}_0$$

$$\mathbf{v} = \int \mathbf{a}\,dt = \mathbf{v}_0 + \mathbf{a}_0 t$$

$$\mathbf{r} = \int \mathbf{v}\,dt = \mathbf{r}_0 + \mathbf{v}_0 t + \frac{1}{2}\mathbf{a}_0 t^2$$

- Se especifican dos vectores adicionales $\mathbf{r}_0$ y $\mathbf{v}_0$ para completar la ecuación. Estos representan la posición y velocidad inicial en el instante **t=0**

Sistemas de partículas

Masa y momento

- Es necesario asociar una masa **m** a cada partícula. Se asumirá que la masa es constante en todo el intervalo de tiempo

$$m = m_0$$

- Se define una **magnitud** vectorial llamada **momento** (o cantidad de movimiento) **p** de una partícula como

$$\mathbf{p} = m\mathbf{v}$$

Sistemas de partículas

Primera ley de Newton: Ley de inercia

- La primera ley de Newton establece que un cuerpo en **movimiento** con velocidad constante permanecerá en movimiento y que un cuerpo en **reposo** permanecerá en reposo a menos que sobre él actúe alguna **fuerza**
- Esto implica que una partícula libre en movimiento en el espacio viajará definiendo un movimiento **rectilíneo**

$$\mathbf{a} = 0$$

$$\mathbf{v} = \mathbf{v}_0 \qquad \mathbf{p} = \mathbf{p}_0 = m\mathbf{v}_0$$

$$\mathbf{r} = \mathbf{r}_0 + \mathbf{v}_0 t$$

Sistemas de partículas

Fuerza

- Una **fuerza** ejercida sobre una partícula es definida como la variación instantánea de su **momento** lineal

$$\mathbf{f} = \frac{d\mathbf{p}}{dt}$$

- Desarrollando

$$\mathbf{f} = \frac{d(m\mathbf{v})}{dt} = \frac{dm}{dt}\mathbf{v} + m\frac{d\mathbf{v}}{dt} = m\mathbf{a}$$

Sistemas de partículas

Segunda ley de Newton: Ley de fuerza

- La segunda ley de Newton dice

$$\mathbf{f} = \frac{d\mathbf{p}}{dt} = m\mathbf{a}$$

- La **aceleración** que adquiere un cuerpo es proporcional a la fuerza aplicada y la **constante de proporcionalidad** es la masa del cuerpo
- Es decir, la aplicación de una fuerza sobre una partícula implica una variación en su aceleración

Sistemas de partículas

Tercera ley de Newton: Ley de acción y reacción

- La tercera ley de Newton dice que si un cuerpo ejerce una **fuerza** sobre otro, este último ejerce sobre el primero una fuerza **igual** en módulo, pero de sentido **contrario** a la primera

$$\mathbf{f}_{AB} = -\mathbf{f}_{BA}$$

- Dicho de otra manera: cada **acción** tiene una **reacción** igual y opuesta
- Esto es muy importante cuando combinada con la segunda ley, la dos juntas, implican la **conservación del momento**

Sistemas de partículas

Conservación del momento

- Cualquier **ganancia de momento** de una partícula implica una pérdida igual y opuesta de momento de otra partícula. Además, el momento total en un sistema cerrado permanecerá constante (suma total de los momentos de las partículas independientes)
- No siempre se obedece explícitamente esta ley, pero si se obedece implícitamente…
- En otras palabras, ocasionalmente se aplican fuerzas **sin aplicar estrictamente** una fuerza opuesta igual a otra partícula

Sistemas de partículas

Energía

- La cantidad de energía es muy importante en Física, y el movimiento de una partícula también puede ser planteado en términos de **energía**
- La **energía** es otro elemento importante que es conservado en las interacciones en sistemas físicos **reales**
- No obstante, habitualmente se utiliza la formulación de Newton usando el **momento de fuerza**

Sistemas de partículas

Fuerzas sobre la partícula

- Normalmente, una partícula estará sometida a varios **vectores de fuerza simultáneos** provenientes de diferentes fuentes
- Todas esta fuerzas individuales conforman el **vector fuerza final** aplicado a cada partícula del sistema de partículas

$$\mathbf{f}_{total} = \sum \mathbf{f}_i$$

Sistemas de partículas

Simulación

- La **cinemática básica** permite relacionar la aceleración de una partícula y el movimiento resultante
- Las leyes de Newton permiten **relacionar aceleración y fuerza**, lo cual es importante porque la fuerza es conservada en un sistema y sirve para describir interacciones en el mismo
- Este es un posible **esquema** para **simular** sistemas de partículas:

Sistemas de partículas

Simulación

1. Calcular todas las fuerzas que actúan en un sistema (asegurándose que cumplen la tercera ley de Newton)
2. Calcular la aceleración resultante para cada partícula (***a=f/m***) e integrarla para calcular su posición

- Repetir

- Esta es la **aproximación de Newton** para simulación de sistemas de partículas. Puede ser extendida a “rigid bodies”, “deformable bodies”, “fluids” …

Sistemas de partículas

Ejemplo (fuerza gravedad)

```
class Particle
{
      float Mass;      // Constante
      Vector3 Position;  // Evoluciona frame a frame
      Vector3 Velocity;  // Evoluciona frame a frame
      Vector3 Force;  // Recalculado cada frame
    public:
      void Update();
      void Draw();
      void ApplyForce(Vector3 &f){Force.Add(f);}
};
```

Sistemas de partículas

Ejemplo

```
class ParticleSystem
{
      int NumParticles;
      Particle *P; //Vector de partículas
   public:
      void Update();
      void Draw();
};
```

Sistemas de partículas

Ejemplo

```
ParticleSystem::Update(float time)
{
      // Calculo de fuerzas
      Vector3 gravity(0,-9.8,0);
      for(i=0;i<NumParticles;i++) {
       Vector3 force=gravity*Particle[i].Mass;//f=mg
       Particle[i].ApplyForce(force);
      }
      // Integración
      for(i=0;i<NumParticles;i++)
       Particle[i].Update(time);
}
```

Sistemas de partículas

Ejemplo

```
Particle::Update(float time)
{
      // Calcular aceleración (Segunda ley de Newton)
      Vector3 Accel=(1.0/Mass) * Force;
      // Calcular nueva posición y velocidad
      Velocity+=Accel*time;
      Position+=Velocity*time;
      // Resetear vector fuerza
      Force.Zero();
}
```

Sistemas de partículas

Ejemplo

- Este sistema de partículas se mantiene **activo** en función del conjunto de fuerzas que se le aplican
- Estas fuerzas provienen de **varias fuentes**, internas o externas al sistema de partículas
- Un ejemplo clásico es la **fuerza de la gravedad**, no obstante, este esquema puede ser extendido fácilmente a cualquier tipo de fuerzas
- El esquema que integra el *sistema + fuerzas* es denominado ‘forward Euler integration’ y es el método más sencillo posible

Fuerzas

Gravedad uniforme

- Es una fuerza sencilla y muy útil, la **gravedad uniforme**:

$$\mathbf{f}_{gravity} = m\mathbf{g}$$

$$\mathbf{g} = [0 \quad -9.8 \quad 0] \quad \frac{m}{s^2}$$

- Como no se aplica una fuerza igual y opuesta, parece que no se cumpla la tercera ley de Newton, por ello se asume que se intercambia fuerza con una masa infinita (planeta)

Fuerzas

Gravedad

- Si los cuerpos se encuentran los suficientemente alejados, se puede usar la **Ley Gravitacional** de Newton:

$$\mathbf{f}_{gravity} = \frac{G m_1 m_2}{d^2}\mathbf{e}$$

$$G = 6.673 \times 10^{-11} \quad \frac{m^3}{kgs^2}$$

Fuerzas

Gravedad

- Esta ley define una fuerza igual y opuesta intercambiada entre dos cuerpos, donde la fuerza es **directamente** proporcional al producto de sus **masas** e **inversamente** proporcional a su **distancia**. La fuerza actúa en una dirección ***e*** a lo largo de una línea desde el centro de un cuerpo hasta el otro (atracción)

$$\mathbf{f}_{gravity} = \frac{Gm_1m_2}{d^2}\mathbf{e} \qquad \mathbf{e} = \frac{\mathbf{r}_1 - \mathbf{r}_2}{|\mathbf{r}_1 - \mathbf{r}_2|}$$

Fuerzas

Gravedad

- Esta ecuación describe la fuerza gravitacional entre **dos partículas**
- Para calcular las fuerzas en un **sistema completo** de partículas, debe ser considerada la interacción entre **cada par** de partículas
- La **complejidad** del algoritmo es de N^2

Fuerzas

Interacciones aerodinámicas

- Las **interacciones aerodinámicas** son normalmente muy complejas y difíciles de modelar con precisión
- Una **simplificación** razonable para describir la fuerza total aerodinámica sobre un objeto es

$$\mathbf{f}_{aero} = \frac{1}{2}\rho |\mathbf{v}|^2 c_d a \mathbf{e} \qquad \mathbf{e} = -\frac{\mathbf{v}}{|\mathbf{v}|}$$

- Donde **ρ** es la densidad del elemento (aire o agua…), $\mathbf{c_d}$ es el coeficiente de resistencia del objeto, **a** es el área de la sección transversal del objeto y **e** es un vector unitario en dirección opuesta a la velocidad

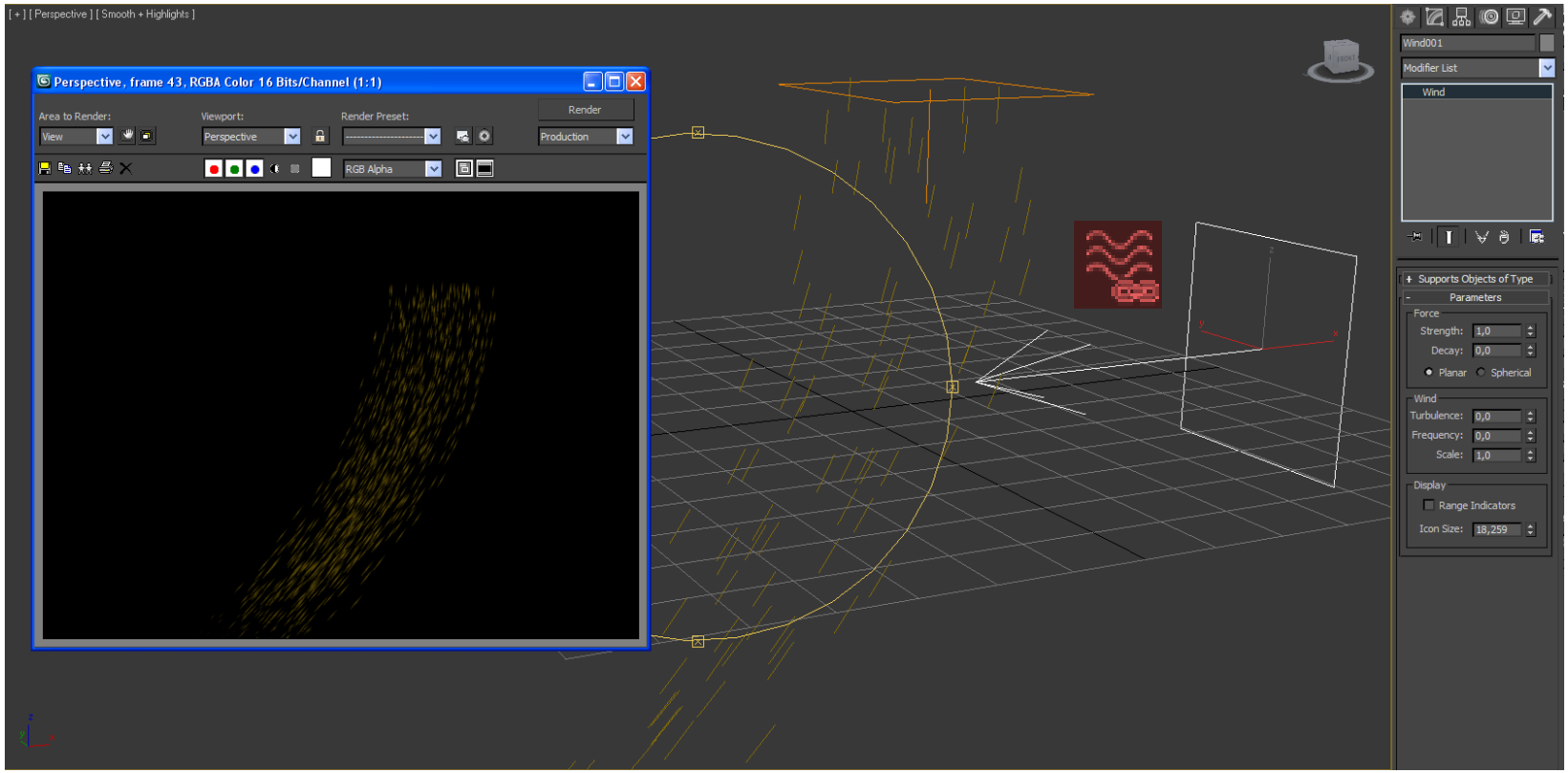
Fuerzas

Interacciones aerodinámicas

- Al igual que la gravedad, las fuerzas aerodinámicas parecen **violar** la tercera ley de Newton, ya que se aplica una fuerza pero no se responde con ninguna otra opuesta
- Se puede **justificar** esto diciendo que la partícula normalmente aplica una fuerza sobre su entorno (aire), y se asumirá que el movimiento resultante es amortiguado por la viscosidad del aire

Fuerzas

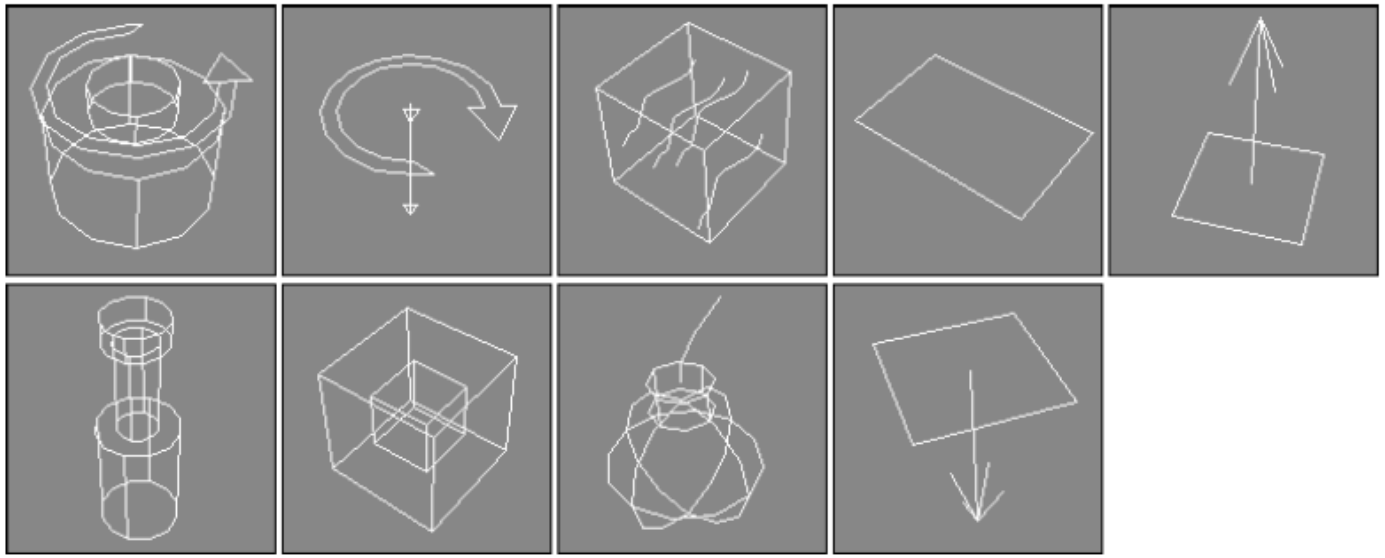
3D Studio Max. Geometry – Particle Systems – Space Warps - Forces



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

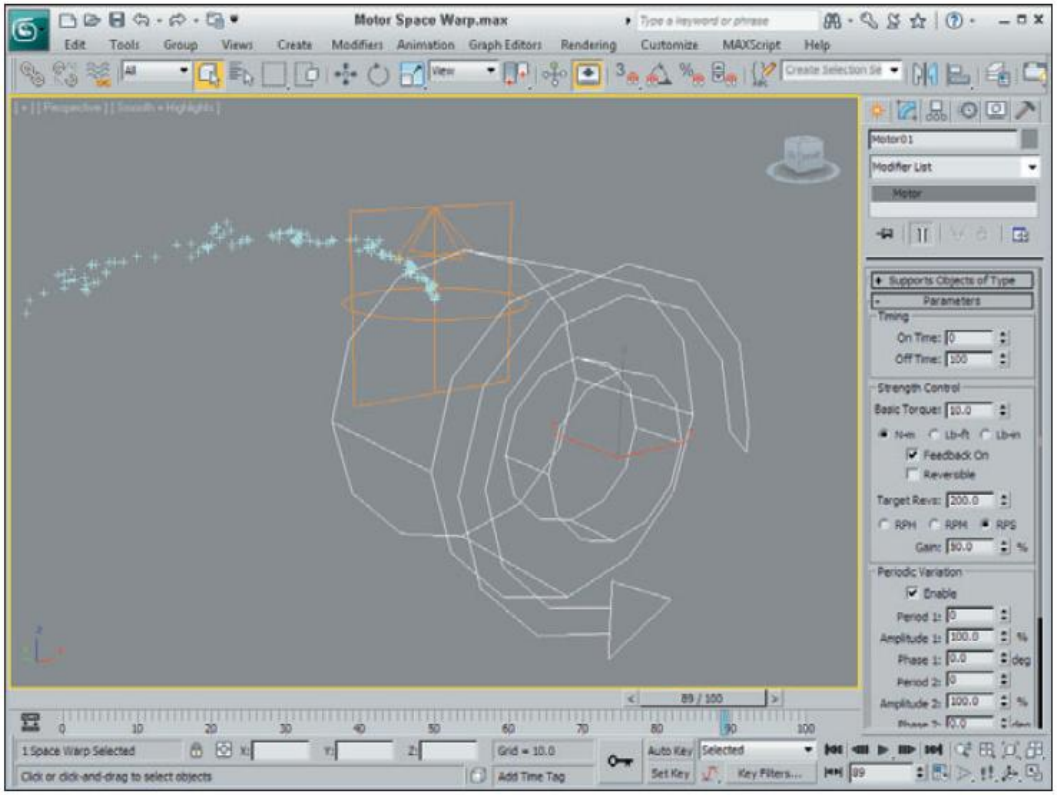
The Force Space Warps: Motor, Vortex, Path Follow, Displace, Wind, Push, Drag, PBomb, and Gravity



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

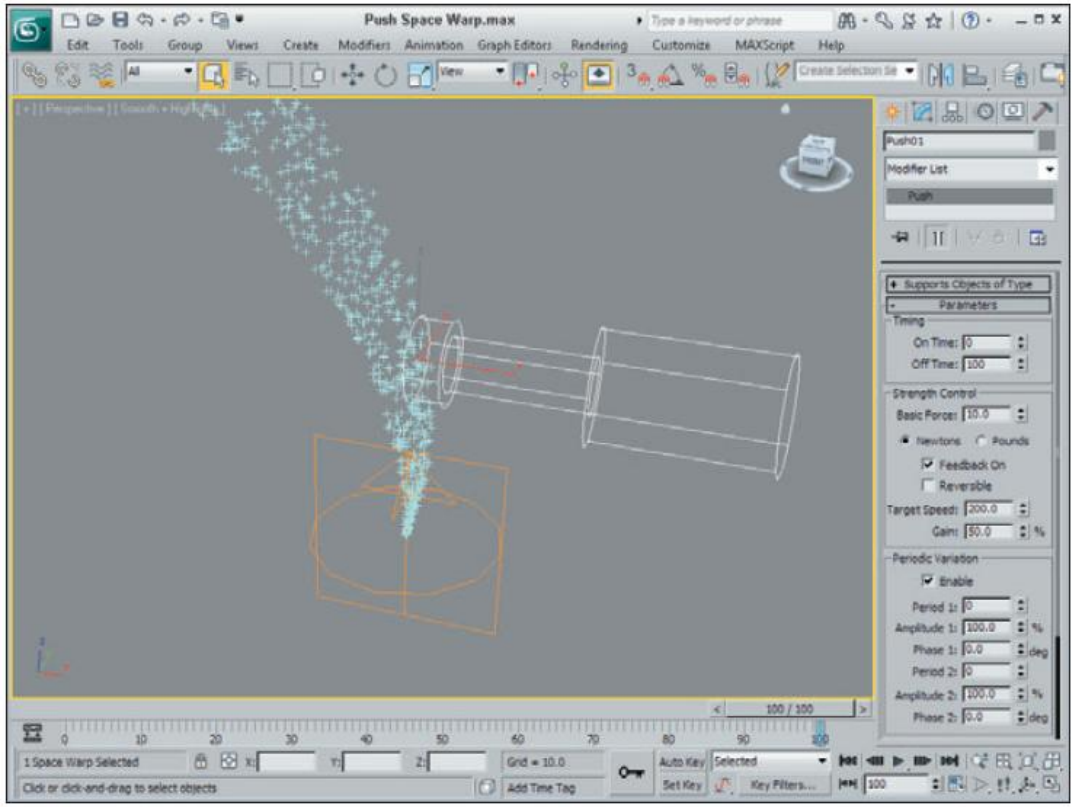
You can use the Motor Space Warp to apply a twisting force to particles and dynamic objects.



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

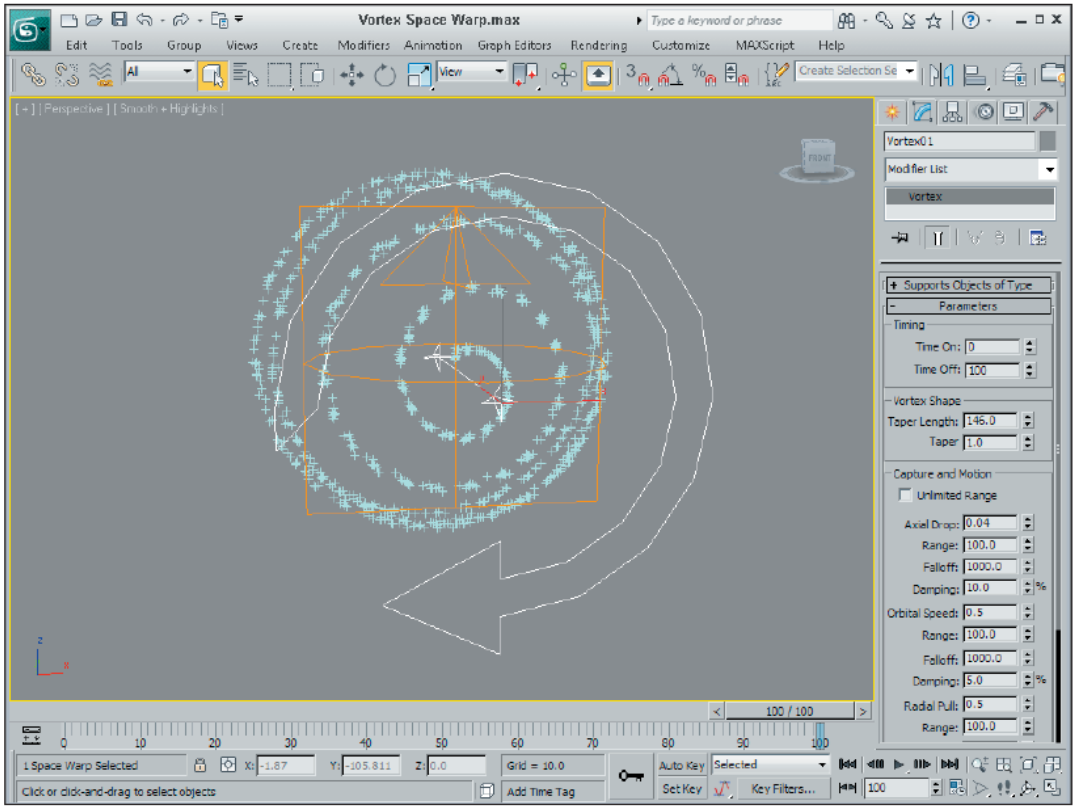
You can use the Push Space Warp to apply a controlled force to particles and dynamic objects.



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

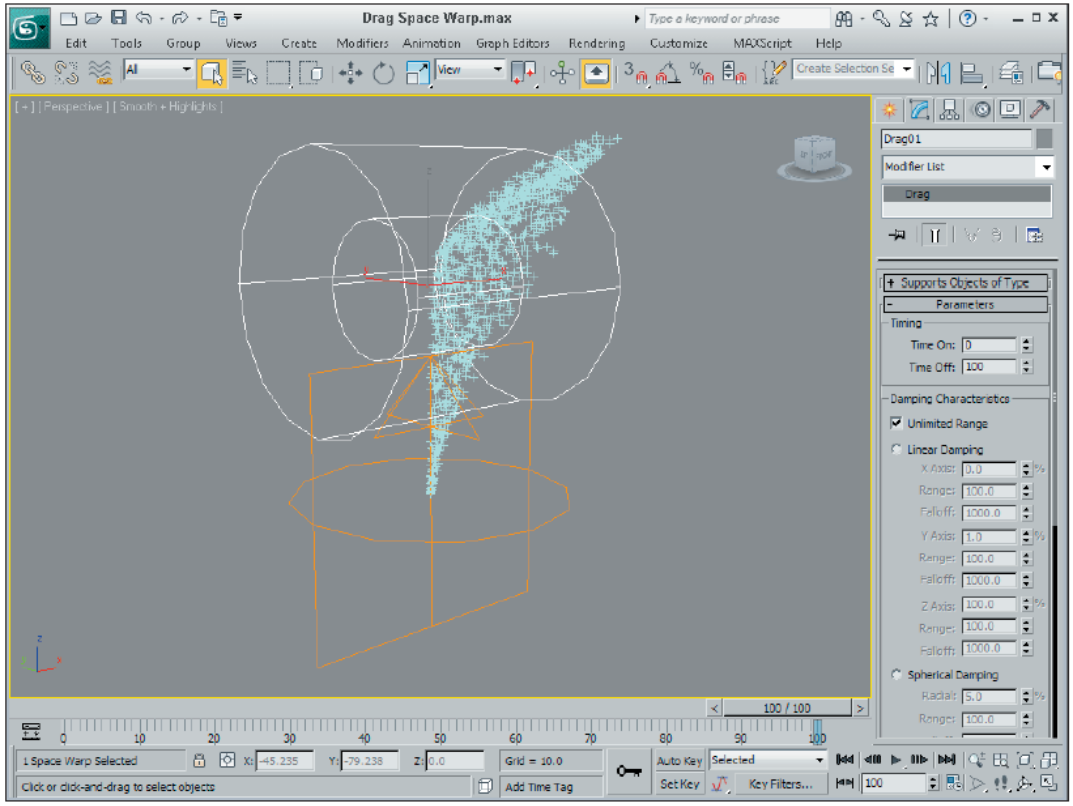
You can use the Vortex Space Warp to force a particle system into a spiral like a whirlpool.



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

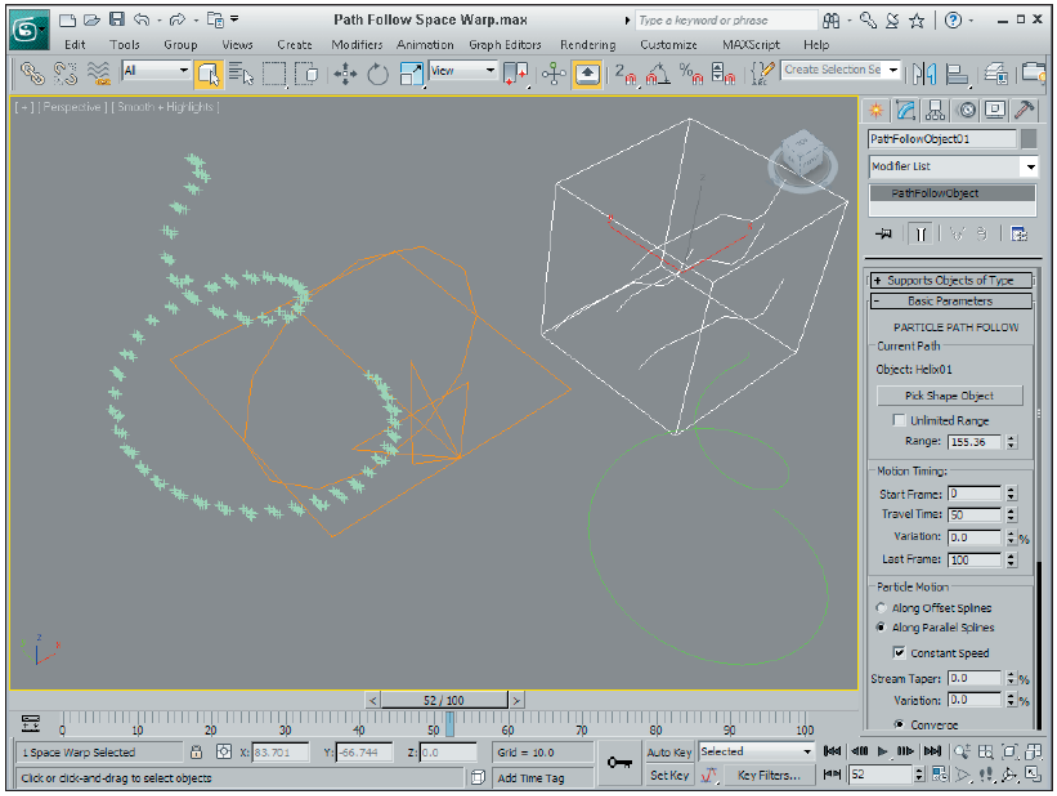
You can use the Drag Space Warp to slow the velocity of particles.



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

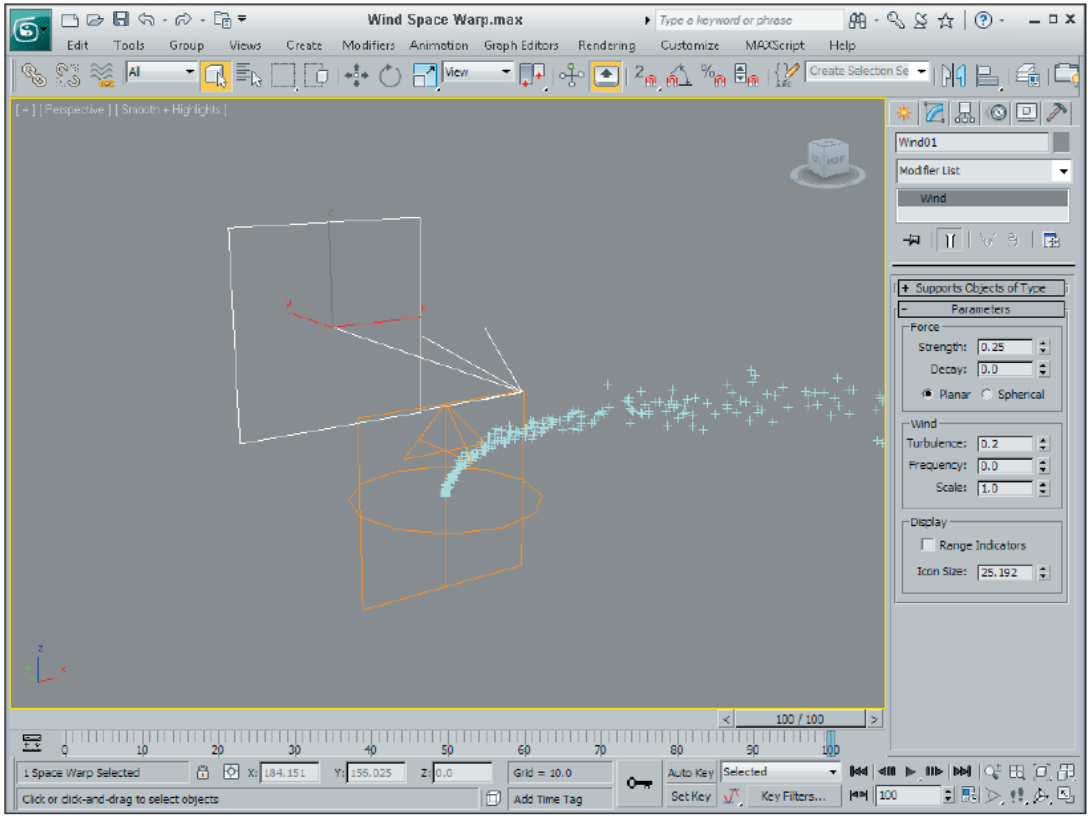
A Path Follow Space Warp bound to an emitter from the Super Spray particle system and following a Helix path



Fuerzas

3D Studio Max. Geometry – Particle Systems – Space Warps - Forces

You can use the Wind Space Warp to blow particles and dynamic objects.



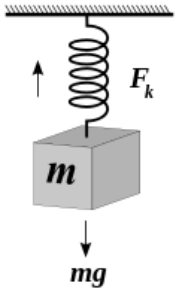
Fuerzas

Springs (fuerza elástica)

- Una fuerza 'spring' puede ser definida como

$$\mathbf{f}_{spring} = -k_s\mathbf{x}$$

- Donde **k** es una constante que describe la rigidez del 'spring' y **x** es un vector que describe el desplazamiento. Es negativa por su oposición a la carga aplicada. Entre partículas la fuerza se basa en la posición de éstas



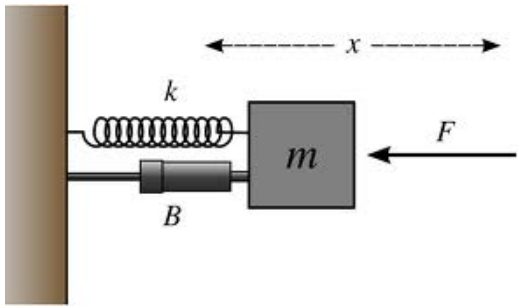
Fuerzas

Dampers (amortiguadores)

- Se puede definir la fuerza de amortiguación entre dos partículas como

$$\mathbf{f}_{damp} = -k_d \mathbf{v}$$

- Los amortiguadores definen la **diferencia de velocidad** entre partículas y la fuerza resultante por ello



Fuerzas

Dampers

- Los **'dampers'** y **'springs'** son combinados habitualmente en como una sola fuerza **'spring-damper'**

$$f_{sd} = -k_s\left(l_0 - l\right) - k_d\left(v_1 - v_2\right)$$

- Un sencillo **'spring-damper'** podría ser:

```
class SpringDamper
{
        float SpringConstant,DampingFactor;
        float RestLength;
        Particle *P1,*P2;
  public:
        void ComputeForce();
};
```

Fuerzas

Dampers

- Para calcular la fuerza de amortiguación entre partículas, se necesita saber la **velocidad** con la que se acercan la una a la otra

$$\mathbf{e} = \frac{\mathbf{r}_1 - \mathbf{r}_2}{|\mathbf{r}_1 - \mathbf{r}_2|}$$

$$v = \mathbf{v}_1 \cdot \mathbf{e} - \mathbf{v}_2 \cdot \mathbf{e}$$

- Esta fórmula describe la **velocidad instantánea** de dos partículas

Fuerzas

Dampers

- Otra forma de calcular esta velocidad es comparando la **distancia** entre las dos partículas en un intervalo de tiempo y la distancia anterior en el último ‘frame’

$$v = \frac{|\mathbf{r}_1 - \mathbf{r}_2| - d_0}{\Delta t}$$

- La diferencia es que ésta es la **solución numérica** que **aproxima** a la propuesta anterior que representa la forma analítica exacta

Fuerzas

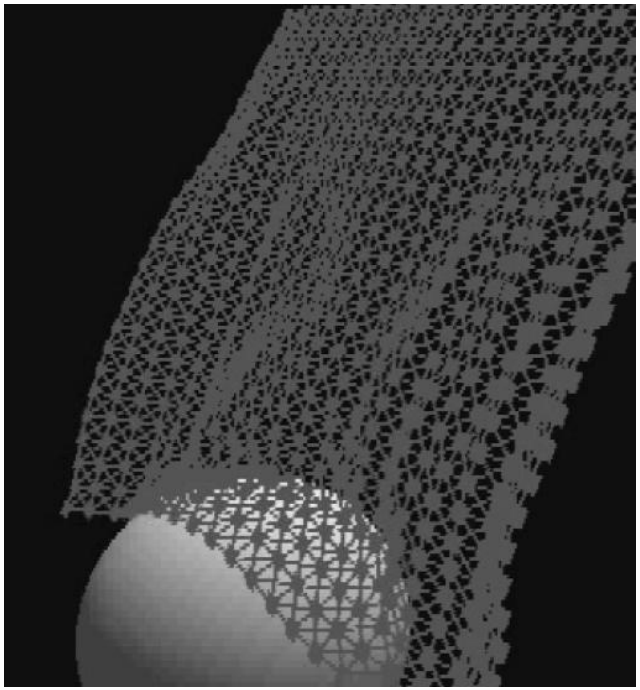
Dampers

- La aproximación **analítica** es mejor por varias razones:
  - No requiere almacenamiento extra
  - Fácil de comenzar la simulación (no necesita ninguna información del último frame)
  - Proporciona el resultado exacto y no una aproximación

Fuerzas

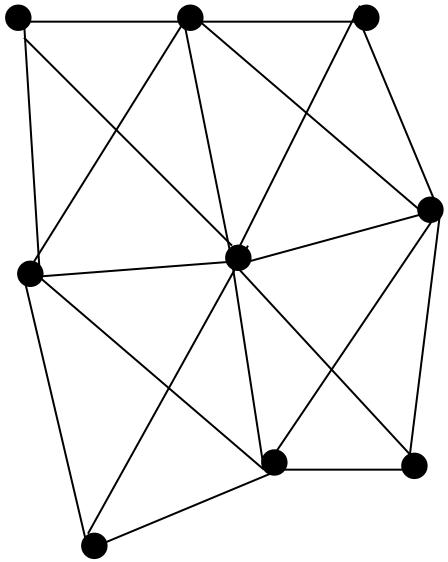
Usos spring-damper

Cloth simulation



- Partícula

Spring-damper



Fuerzas

Campos de fuerzas

- Se pueden definir campos de fuerza arbitrarios. En este caso la fuerza a aplicar a una partícula está relacionada con respecto a la **posición** de la partícula en el interior del campo de fuerzas

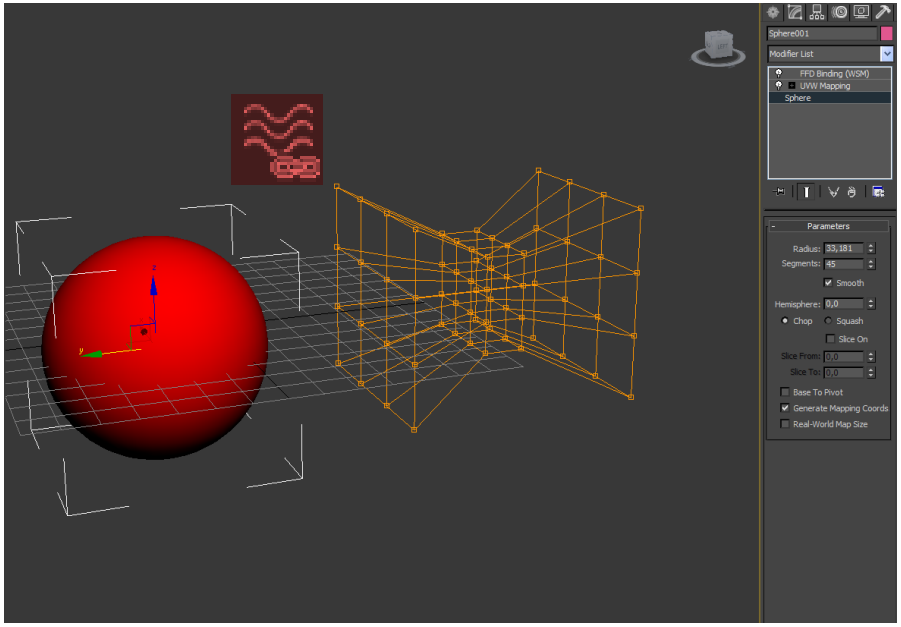
$$\mathbf{f}_{field} \propto \mathbf{f}(\mathbf{r})$$

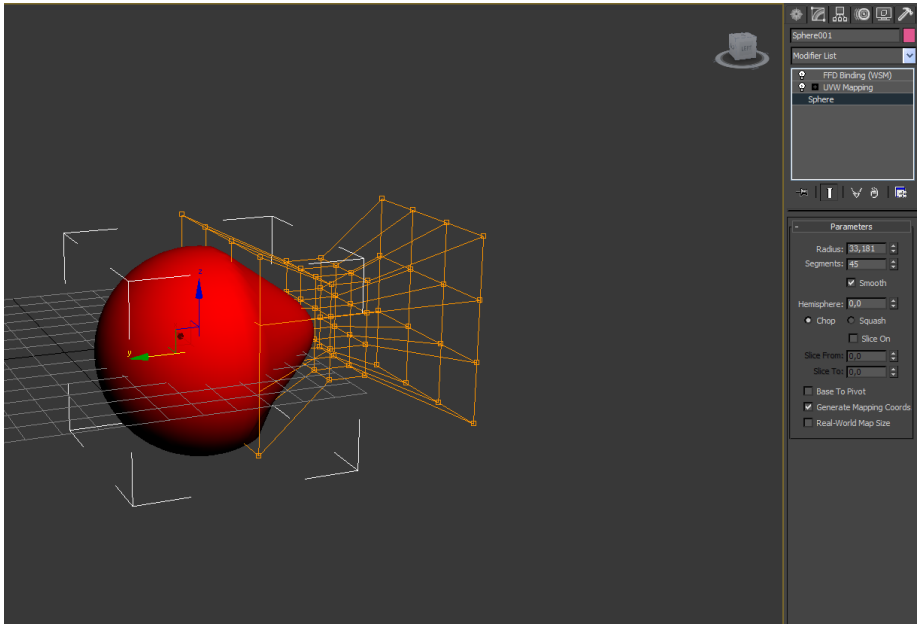
- Se pueden definir fuerzas como la **velocidad del aire** con una ecuación de campo similar y entonces usando fuerzas aerodinámicas calcular la fuerza final
- Usando esta **aproximación**, se pueden definir campos de **turbulencias**, **vórtices**, y otros **patrones de flujo**

Fuerzas

Campos de fuerzas

Geometry – Space Warps – Geometry/Deformable





Fuerzas

Colisión e Impulso

- Las fuerzas anteriores no controlan la **colisión** entre partículas
- Por eso se introduce el concepto de **impulso**, que puede ser definido como **la actuación de una gran fuerza actuando sobre un cuerpo en un pequeño instante de tiempo**

Fuerzas

Impulso

- El **impulso** es definido como el **cambio de momento** (cantidad de movimiento) de un cuerpo:

$$\mathbf{j} = \int \mathbf{f}\, dt = \Delta \mathbf{p}$$

- El **impulso** se comporta como una **fuerza**, pero en lugar de afectar a la aceleración, afecta directamente a la **velocidad**
- También cumple la tercera ley de Newton, por lo que los cuerpos pueden intercambiar impulsos iguales y opuestos
- Y al igual que las fuerzas también se puede calcular el **impulso total** como la suma de los **impulsos individuales**

Fuerzas

Impulso

- La **inclusión** del impulso añade cierta modificación a la simulación llevada a cabo

//Compute all forces and impulses

$$\mathbf{f} = \sum \mathbf{f}_i$$

$$\mathbf{j} = \sum \mathbf{j}_i$$

//Integrate to get new velocity & positi

$$\mathbf{v}' = \mathbf{v}_0 + \frac{1}{m}(\mathbf{f}\Delta t + \mathbf{j})$$

$$\mathbf{r}' = \mathbf{r}_0 + \mathbf{v}'\Delta t$$

Fuerzas

Colisiones

- La partícula tiene una velocidad **v** antes de la colisión
- La partícula colisiona contra la superficie con normal unitaria **n**
- Se busca el impulso **j** aplicado a la partícula tras la colisión

Fuerzas

Elasticidad

- Hay muchas teorías sobre colisiones
- Habitualmente se realizan ciertas simplificaciones (reducción complejidad problema)
- Se define una magnitud denominada **elasticidad** entre 0 y 1, que describe la energía devuelta tras la colisión
- Una **elasticidad** de 0 => indica que la velocidad tras la colisión es 0
- Una **elasticidad** de 1 => indica que la velocidad tras la colisión es exactamente la **misma** pero **opuesta** a la llevada antes de la colisión

Fuerzas

Colisiones

- Para simplificar se considera una colisión sin **fricción**
- El impulso tras la colisión será perpendicular al plano de colisión (por ejemplo normal superficie)

$$\vec{v}_{close} = \vec{v}_n$$

$$j = -(1+e)m v_{close}$$

Fuerzas

Combinación de fuerzas

- Todas las fuerzas descritas anteriormente pueden ser acopladas al sistema global de simulación fácilmente
- Hay que recordar que la fuerza total aplicada a una partícula se representa como la suma total de todas las fuerzas individuales
- En cada ‘frame’, se calculan todas las fuerzas en el sistema en ese intervalo de tiempo

Integración: sistemas partículas + fuerzas

Integración

- El cálculo de posiciones y velocidades a partir de aceleraciones se denomina **integración**
- En la práctica la fuerzas son complejas e imposibles de integrar analíticamente por eso se opta por un esquema de **integración numérica**
- El método ***`Particle::Update()`*** descrito anteriormente calcula una iteración a partir de su integración numérica. En particular, se usa el esquema **'forward Euler'**

Integración: sistemas partículas + fuerzas

Forward Euler Integration

- **'Forward Euler integration'** es el método más simple para llevar a cabo la integración numérica

$$x_{n+1} \Rightarrow x_n + \dot{x}_n \Delta t$$

- La posición de la partícula en el instante ***n+1*** se aproxima en función de un estado previo ***n*** de esa misma partícula

Integración: sistemas partículas + fuerzas

Forward Euler Integration

- Para las partículas, se integra doblemente para conocer su posición y velocidad

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{a}_n \Delta t$$

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \mathbf{v}_{n+1} \Delta t$$

desarrollando

$$\mathbf{r}_{n+1} = \mathbf{r}_n + (\mathbf{v}_n + \mathbf{a}_n \Delta t) \Delta t$$

$$= \mathbf{r}_n + \mathbf{v}_n \Delta t + \mathbf{a}_n (\Delta t)^2$$

Integración: sistemas partículas + fuerzas

Forward Euler Integration

- Se observa que

$$\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \mathbf{v}_t \Delta t + \mathbf{a}(t)$$

es muy similar al resultado que se tendría si se asumiera que la partícula está sometida a una aceleración uniforme durante un 'frame'

$$\mathbf{r}_{t+\Delta t} = \mathbf{r}_t + \mathbf{v}_t \Delta t + \frac{1}{2}\mathbf{a}(t)$$

Integración: sistemas partículas + fuerzas

Forward Euler Integration

- El método de **Euler** proporciona unos correctos resultados, siendo su uso ampliamente extendido
- Se comporta de forma muy estable en la mayoría de sistemas de partículas utilizados en **animación por computador**, pero para aplicaciones de **ingeniería** no es muy preciso ni el idóneo
- Se comporta de forma pobre en situaciones donde las fuerzas cambian rápidamente

Integración: sistemas partículas + fuerzas

Otros sistemas de integración

¿Esquemas más sofisticados de integración?

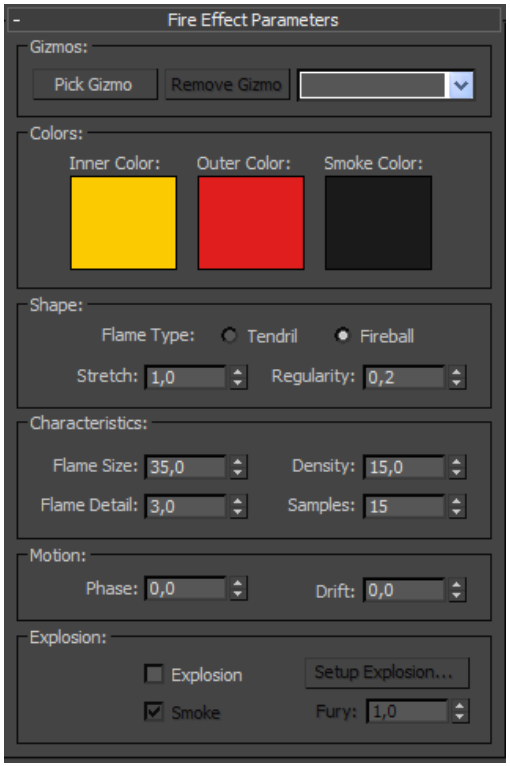
Tema 2: Modelado

1.- Introducción
2.- Modelos geométricos de representación
3.- Técnicas de modelado
4.- Transformaciones geométricas
5.- Deformadores
6.- Sistemas de partículas
7.- Fuerzas
8.- Efectos atmosféricos

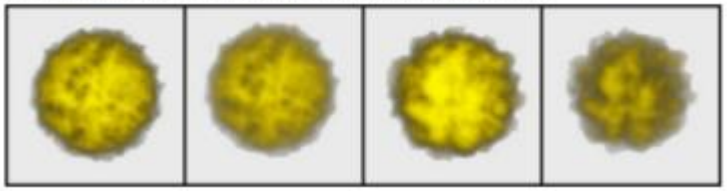
Efectos atmosféricos

Tipos

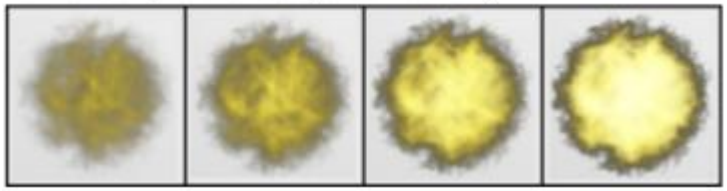
- **Fuego**



The Fire atmospheric effect can be either Tendril or Fireball shaped.



The Fire effect brightness is tied closely to the flame's Density value.

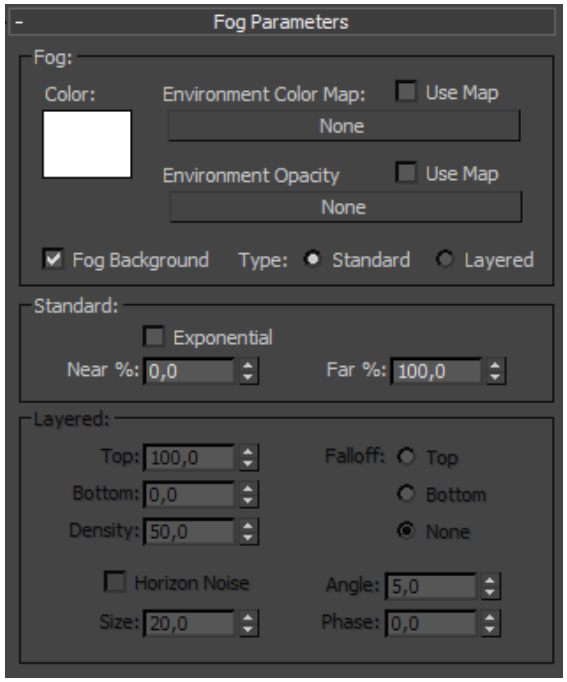


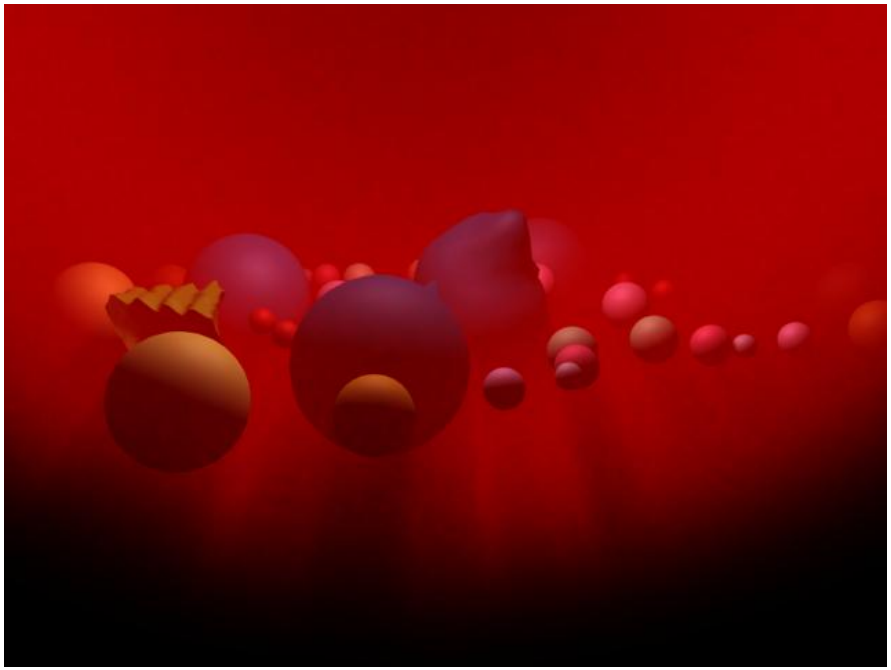
You can use the Fire atmospheric effect to create clouds.

Efectos atmosféricos

Tipos

- **Niebla**

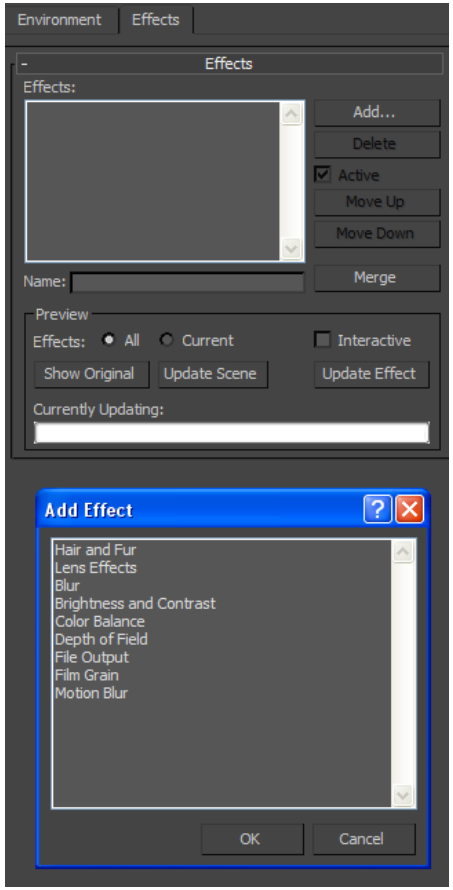




Efectos atmosféricos

Tipos

- **Otros**



Efectos atmosféricos

Tipos

- **Lens effect**