

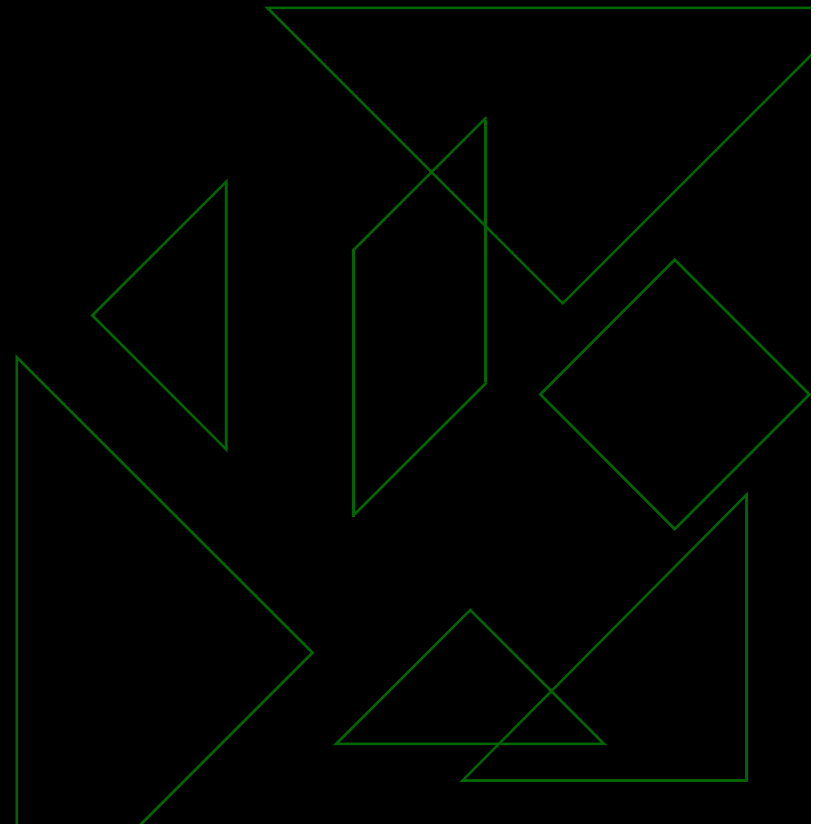
Sistemas de Partículas



Soraia Raupp Musse

Sistemas de Partículas

- ◆ CONCEITO:
Conjunto de partículas cujo comportamento evolui no tempo de acordo com regras algorítmicas com o objetivo de simular um fenômeno fuzzy



Sistemas de Partículas

- ◆ Usado em modelagem, animação e rendering (dependente da aplicação)
- ◆ Normalmente:
 - Coleção de partículas
 - Atributos definidos estocasticamente

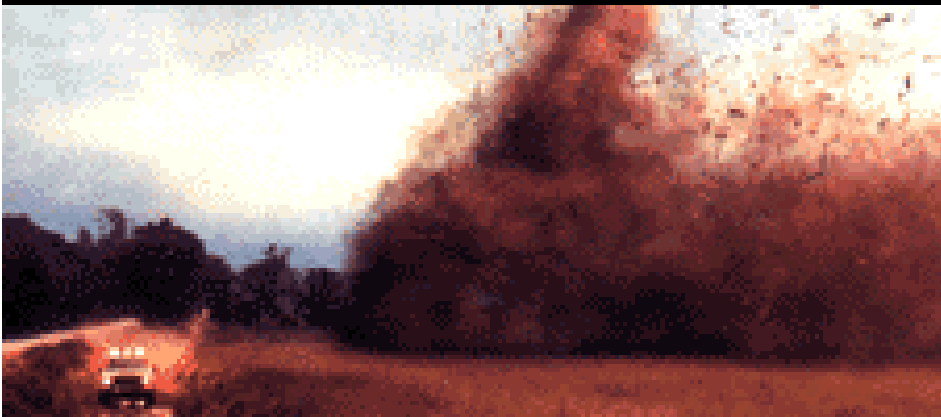
Histórico

- ◆ William Reeves (1983): primeira aplicação – Fogo e explosão



Aplicações

- ◆ Fenômenos naturais: explosões, fogos de artifício, nuvens, água
- ◆ Modelagem e deformação geométrica de superfícies



Atributos de uma partícula

- ◆ Massa (real)
- ◆ Posição (\mathbb{R}^3)
- ◆ Velocidade (\mathbb{R}^3)
- ◆ Aceleração (\mathbb{R}^3)
- ◆ Cor (RGB)
- ◆ Tempo de vida (frames ou tempo)
- ◆ Shape
- ◆ Tamanho
- ◆ Transparência



Atributos de uma partícula

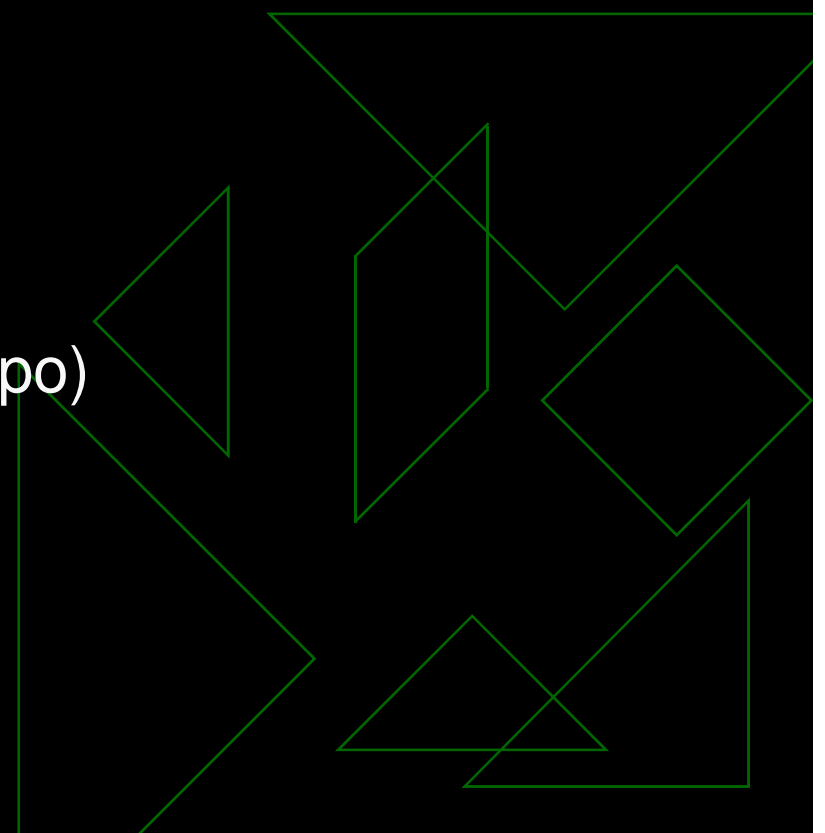
- ◆ Massa (real)
- ◆ Posição (\mathbb{R}^3)
- ◆ Velocidade (\mathbb{R}^3)
- ◆ Aceleração (\mathbb{R}^3)
- ◆ Cor (RGB)
- ◆ Tempo de vida (frames ou tempo)
- ◆ Shape
- ◆ Tamanho
- ◆ Transparência

Regras algorítmicas servem para variar esses atributos

Atributos de uma partícula

- ◆ Massa (real)
- ◆ Posição (\mathbb{R}^3)
- ◆ Velocidade (\mathbb{R}^3)
- ◆ Aceleração (\mathbb{R}^3)
- ◆ Cor (RGB)
- ◆ Tempo de vida (frames ou tempo)
- ◆ Shape
- ◆ Tamanho
- ◆ Transparência

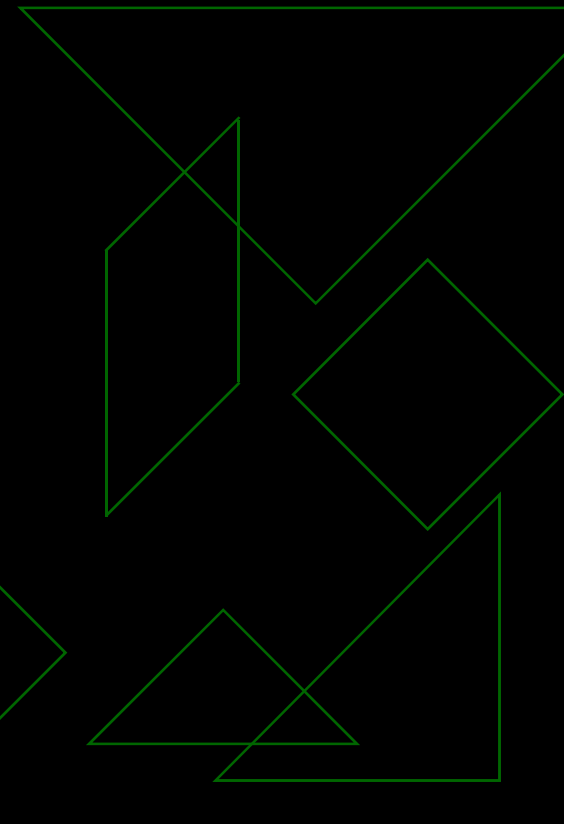
Movimento



Atributos de uma partícula

- ◆ Massa (real)
- ◆ Posição (\mathbb{R}^3)
- ◆ Velocidade (\mathbb{R}^3)
- ◆ Aceleração (\mathbb{R}^3)
- ◆ Tempo de vida (frames ou tempo)
- ◆ Cor (RGB)
- ◆ Shape
- ◆ Tamanho
- ◆ Transparência

Visualização



Dinâmica de Partículas

$$ma = F - \gamma.v$$

$$\gamma > 0$$

m=massa

a=aceleração

F=força

v=velocidade

γ =atrito proporcional à velocidade da partícula

Dinâmica de Partículas

- ◆ Sistemas desacoplados:
 - O movimento das partículas não é influenciado pelas outras partículas. Só depende de seu estado e interação com o ambiente
 - Vantagens:
 - ◆ Facilmente simulados
 - ◆ Complexidade na $O(n)$, onde n é o número de partículas vivas em t
 - ◆ Sistemas desacoplados são naturalmente paralelizáveis

Dinâmica de Partículas

- ◆ Sistemas acoplados:
 - O movimento das partículas depende fundamentalmente de:
 - ◆ interação entre os corpos
 - ◆ seu estado
 - ◆ interação com o ambiente
 - Vantagens:
 - ◆ Mais realista de acordo com os fenômenos reais

Dinâmica de Partículas

- ◆ O acoplamento pode ser:
 - Fixo, determinado por vínculos
 - Variável, determinado por proximidade
 - Total

Acoplamento Total: Todas as partículas interagem



Acoplamento Variável:



par pares de partículas espaciais
par pares de partículas

Dinâmica de Partículas

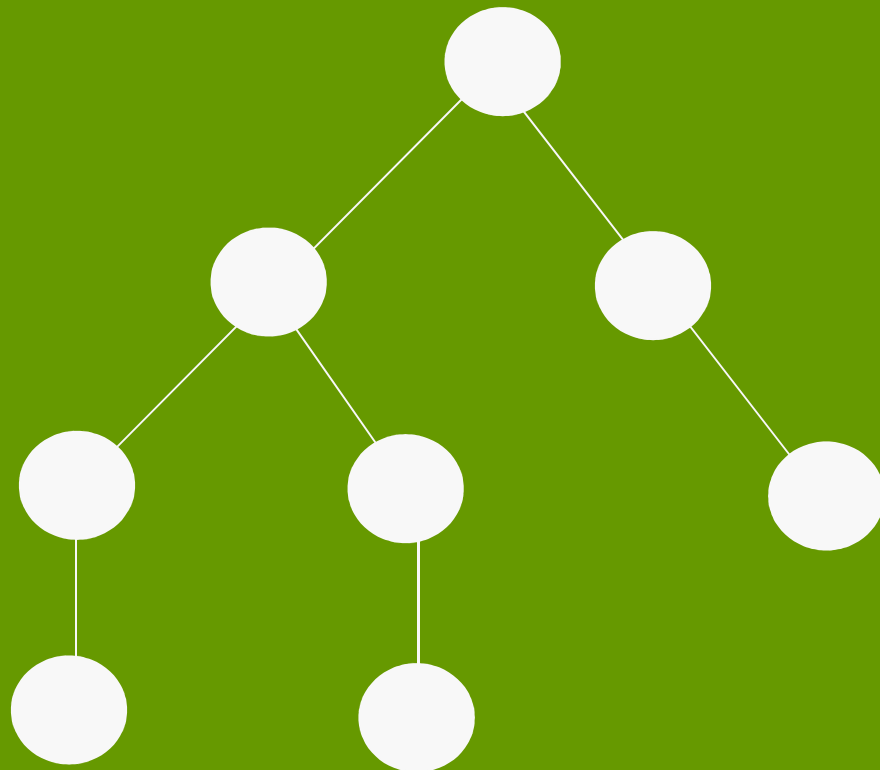
- ◆ Redução de complexidade em sistemas acoplados:
 - Decomposição celular do domínio
 - Separar as partículas por células
 - Calcular as interações entre partículas da mesma célula
 - Calcular as interações entre células vizinhas
- ◆ Complexidade $O(n)$ se:
 - Nro de células proporcional ao nro de partículas
 - Partículas uniformemente distribuídas
 - Partículas “que não se movem muito”

Modelo de Reeves (1983)

- ◆ Objetivo: modelar sistemas difusos, como nuvens, fumaça, água e fogo
- ◆ Porque as técnicas tradicionais de CG não são adaptadas a esses fenômenos?
- ◆ Evolução de partículas de acordo com regras algorítmicas incluindo aleatoriedade
- ◆ Porque possui componente estocástico?

Modelo de Reeves (1983)

- ◆ Vantagens [Reeves 1983]:
 - Partículas são geometricamente simples
 - Modelo procedural estocásticos (po
 - O modelo é dinâ
- ◆ As partículas po hierarquias em á sistemas de part
- ◆ Esses sistemas em tempos sepa

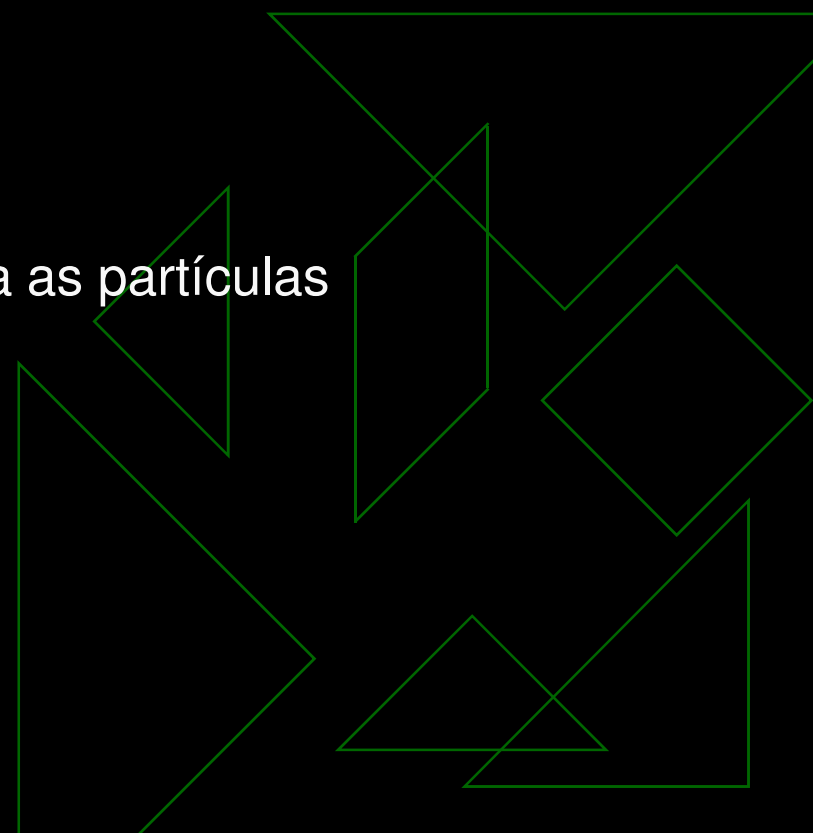


Modelo de Reeves (1983)

◆ Algoritmo

■ Para cada quadro

- ◆ Criar novas partículas
 - Incluí-las na hierarquia
 - Definir atributos
- ◆ Matar partículas velhas
- ◆ Gerar novos movimentos para as partículas
- ◆ Gerar uma imagem

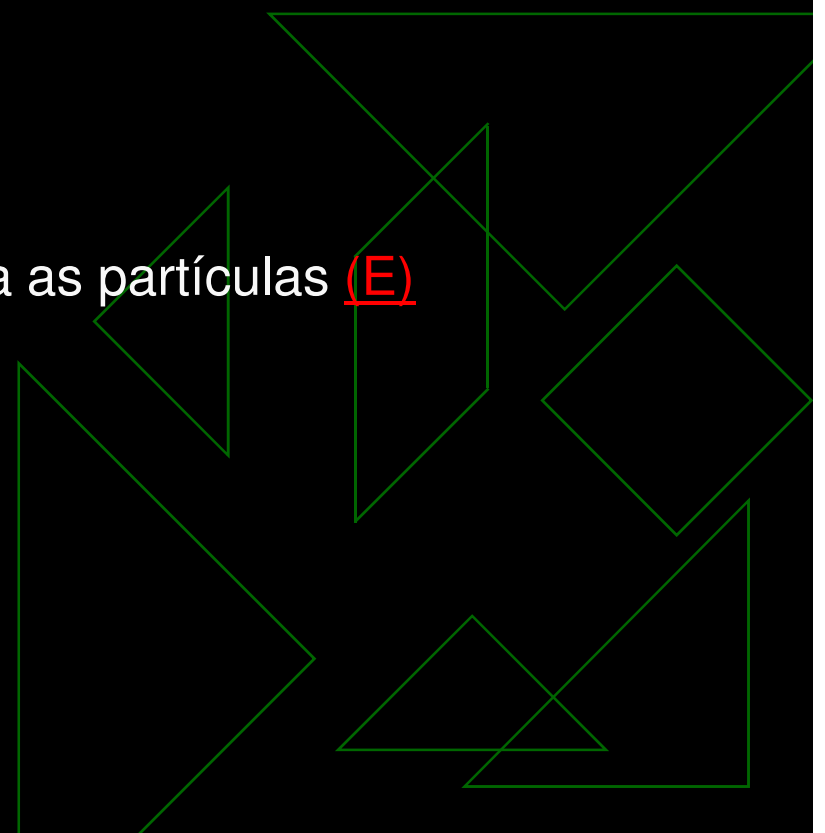


Modelo de Reeves (1983)

◆ Algoritmo

■ Para cada quadro

- ◆ Criar novas partículas
 - Incluí-las na hierarquia (E)
 - Definir atributos (E)
- ◆ Matar partículas velhas
- ◆ Gerar novos movimentos para as partículas (E)
- ◆ Gerar uma imagem



Modelo de Reeves (1983)

- ◆ (E) define parâmetros que controlam a faixa de validade dos atributos de forma, aparência e movimento
- ◆ Por exemplo: $n = m + vr$
 - n = número de partículas a serem criadas
 - m = média de valores
 - v = variância
 - r = variável aleatória distribuída no intervalo $[-1;1]$
- ◆ O usuário controla m e v

Modelo de Reeves (1983)

◆ Outros parâmetros:

- Posição
- Velocidade
- Forma
- Tamanho
- Cor
- Transparência
- Tempo de vida

Nova posição = dependente da
velocidade

Modelo de Reeves (1983)

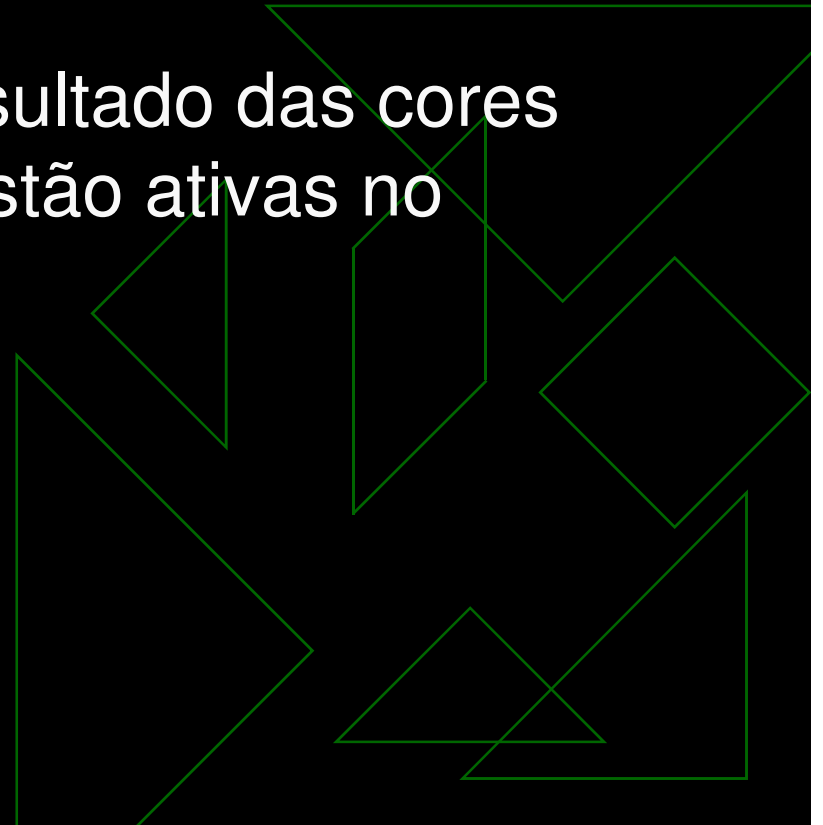
- ◆ Porque uma partícula morre?
 - Tempo de vida: Seu tempo de vida acabou
 - Baixa intensidade: sua cor não pode ser vista
 - Fuga do pai (na hierarquia); sai do controle

Modelo de Reeves (1983)

- ◆ Rendering
 - Simplificação possível:
 - ◆ Cada partícula é uma fonte pontual de luz (oclusão)
- ◆ Performance:
 - Dependendo do número de partículas a serem simuladas, não pode ser em tempo real

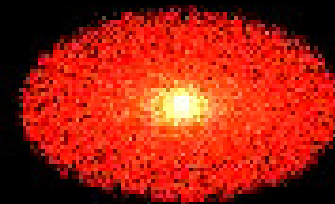
Exemplos

- ◆ Algoritmo de Reeves 1983
- ◆ Cada partícula é representada por um pixel na imagem gerada
- ◆ A cor de cada pixel é o resultado das cores de várias partículas que estão ativas no mesmo pixel
- ◆ 20000 partículas
- ◆ 20 a 50 frames



Exemplo 1

- ◆ Partículas são geradas numa estrutura em disco circular com velocidades iniciais. Quando o tempo de vida termina, elas são removidas

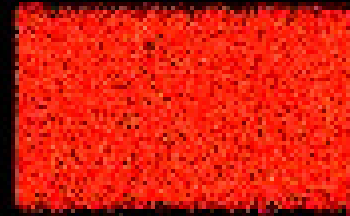


Exemplo 2

- ◆ Partículas são geradas todas próximas a um ponto no espaço com velocidades iniciais. Quando o tempo de vida termina, elas são removidas

Exemplo 3

- ◆ Partículas são geradas numa estrutura quadrada com velocidades iniciais. Quando o tempo de vida termina, elas são removidas



Exemplo 4

- ◆ Partículas são geradas numa estrutura toróide. Quando o tempo de vida termina, elas são removidas



Exemplo 5

- ◆ Partículas são geradas num ponto, porém quando o tempo de vida termina, elas são reiniciadas

Exemplo 6

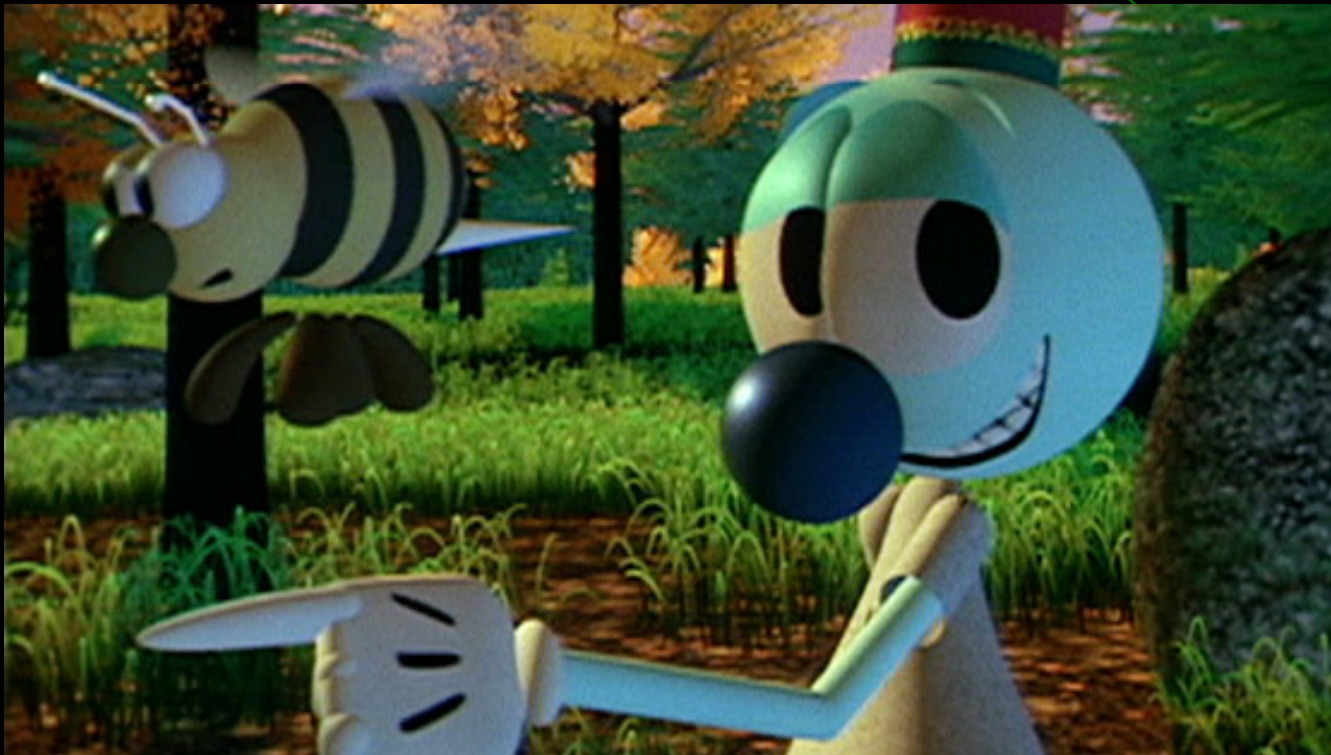
- ◆ Partículas são geradas num ponto que varia de posição e quando o tempo de vida termina, elas são reiniciadas

Exemplo 7

- ◆ Partículas são geradas em posições randômicas dentro de um quadrado. Quando terminam, não são regeneradas

Modelo do Reeves (1985)

- ◆ Árvores no short film: *The Adventures of Andre and Wally B.* (PIXAR) www.pixar.com



Modelo do Reeves (1985)

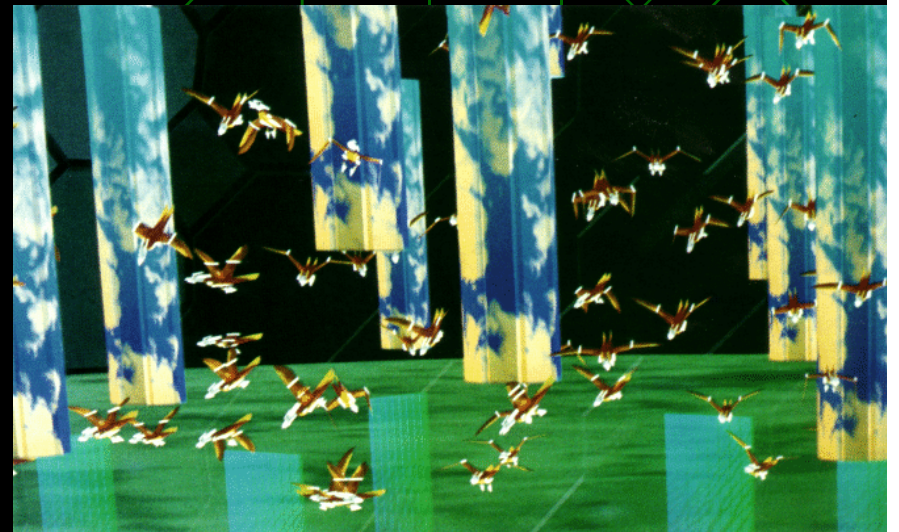
- ◆ Posição das árvores na floresta
- ◆ Cada árvore foi criada com Particle Systems
- ◆ Modificações em relação ao modelo original:
 - Rendering tradicional (RENDERMAN),
 - Geração dos shapes não era fixo, dependente para fazer folhas somente nas pontas dos galhos, e.g.

Modelo do Reynolds (1987)

- ◆ Utilização de sistemas de partículas para modelagem de flocks



Stanley and Stella



Modelo do Reynolds (1987)

- ◆ Objetivo: Simular bandos de pássaros, manadas de animais, cardumes, etc...
- ◆ [Stanley & Stella \(1988\)](#), The Lion King(1994), Batman Returns, The Hunchback of Notre Dame (1996), Hercules (1997), Mulan (1998)
- ◆ Boid (short termo para bird-object)
- ◆ Modelo mais utilizado em filmes e jogos

Modelo do Reynolds (1987)

- ◆ Diferenças entre Boids e Particle Systems:
 - Cada boid possui uma descrição geométrica 3D baseada em polígonos
 - Cada boid tem um SR local
 - Existe um número fixo de boids – eles não são criados ou destruídos
 - Métodos de rendering podem ser usados porque existem geralmente não muitos boids
 - O comportamento dos boids é dependente de seus estados internos e externos

Modelo do Reynolds (1987)

- ◆ Três regras locais:
 - Collision avoidance
 - Velocity matching
 - Flock centering
- ◆ Comportamento emergente:
 - Boids andam juntos
 - Com velocidades parecidas
 - Não se interpenetram



Modelo do Reynolds (1987)

◆ Três regras locais:

- Collision avoidance (CA)
- Velocity matching (VM)
- Flock centering (FC)

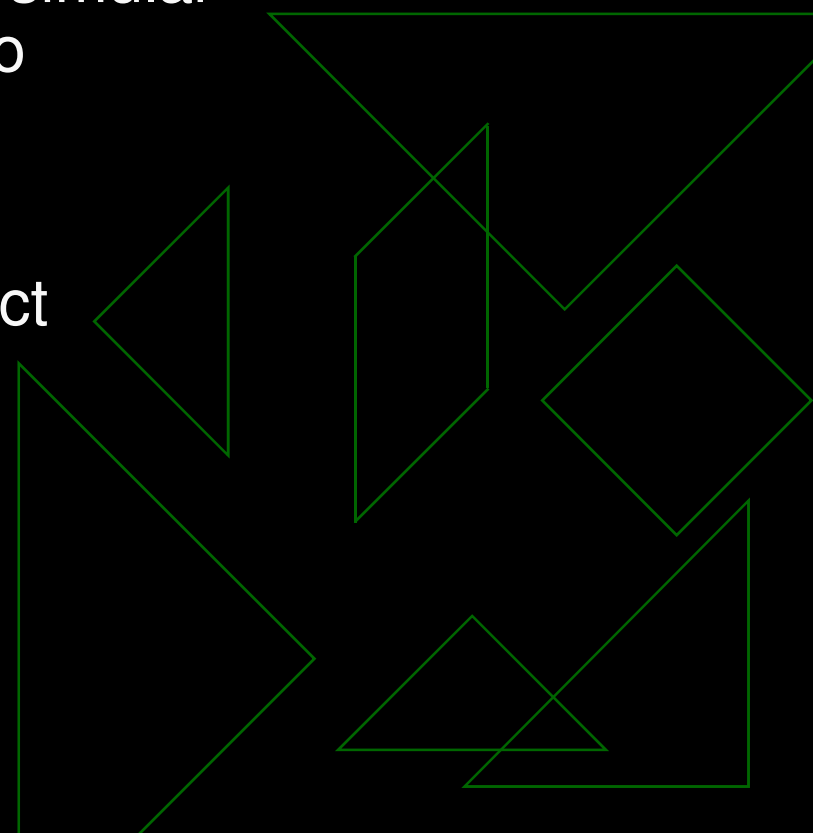
◆ **DISCUSSÃO:**

- O que aconteceria se $CA=0$, $VM=0$ e $FC=0$???
- Como podemos fazer para gerar um flock com superfícies maiores?
- Como podemos simular individualidades, e.g. Pássaros antipáticos, líderes, etc...

Flocks e Boids



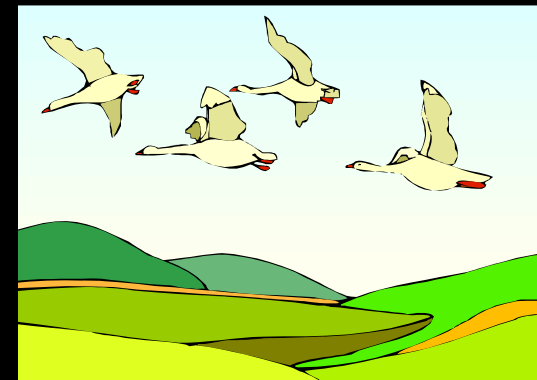
- ◆ Flock é um grupo de objetos que exibe uma classe geral de objetos alinhados, livres de colisão e que apresentam movimento agregado
- ◆ Para simular um flock, devemos simular comportamentos de um indivíduo
 - Percepção
 - Dinâmica de movimento
- ◆ Boid is a simulated bird-like object



Como simular um flock em Animação?

- Usar script não é boa idéia
- Particle system é muito simples
- Para tornar particle systems mais complexo
 - ◆ Percepção Local
 - ◆ Simulação baseada em Física
 - ◆ Comportamento de Flocking
- O resultado global é através da interação entre comportamentos locais individuais

Behavioral Animation



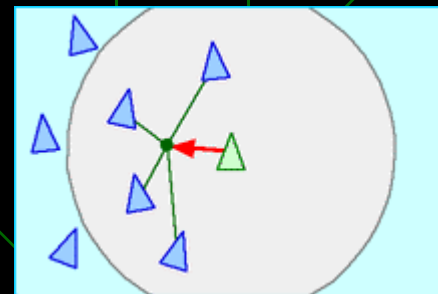
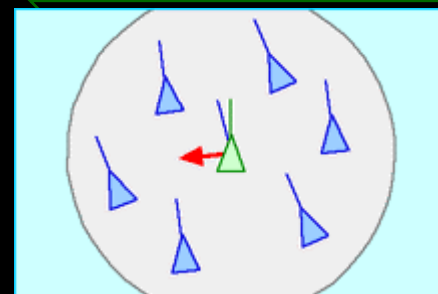
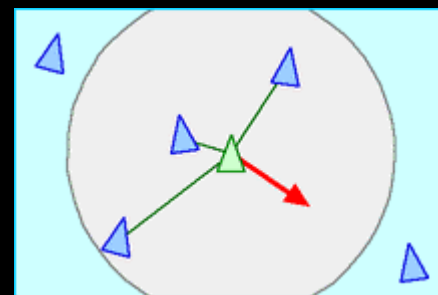
- ◆ Animador é um diretor de cinema
- ◆ A performance dos personagens é o resultado indireto das instruções do diretor aos atores
- ◆ Não saber o que vai acontecer com a simulação (onde são informados comportamentos e condições iniciais) pode trazer inesperadas e interessantes surpresas
- ◆ Problema: as vezes não é fácil descobrir porque eles estão fazendo o que estão fazendo... ("these darn boids seem to have a mind of their own!").

Flocks, Herds, and Schools

- ◆ O comportamento individual deve ser coordenado com os dos membros do flock
- ◆ Dois equilibrados e opostos comportamentos:
 - Desejo de ficar perto do centro do flock
 - Desejo de evitar colisão com o flock
- ◆ Indivíduos não prestam atenção aos comportamentos de cada pássaro
 - A percepção de um pássaro em relação ao resto do flock é localizada e filtrada:
 - ◆ 2 ou 3 vizinhos próximos
 - ◆ Resto do flock

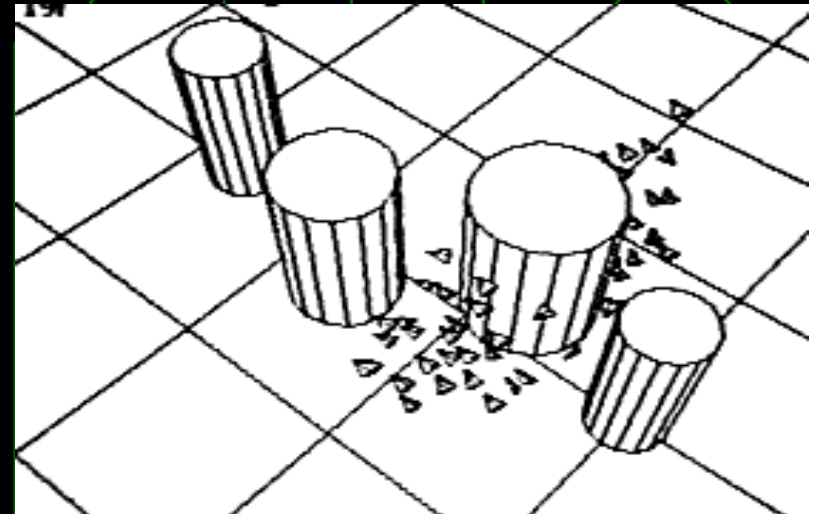
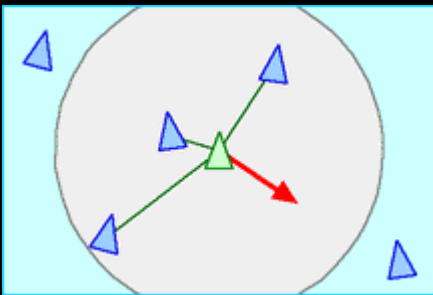
Flocks Simulados

- ◆ Collision Avoidance
 - Evitar colisão com os membros do flock
- ◆ Igualar velocidade
 - Tentam igualar as velocidades com seus vizinhos próximos
- ◆ Ir ao centro do flock
 - Tentam ir ao centro do flock

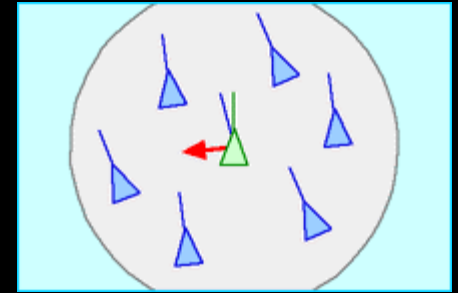


Collision Avoidance

- ◆ Devem evitar colisão com os outros
- ◆ A colisão estática é baseada na posição relativa dos membros do flock
- ◆ Ignora a velocidade

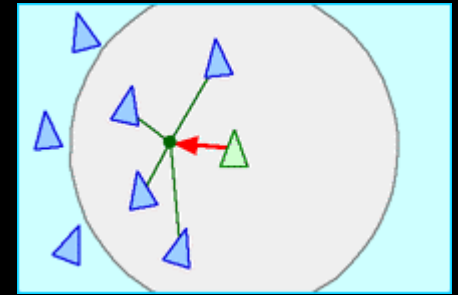


Igualar velocidade



- ◆ Percebe apenas a velocidade
 - Ignora a posição
- ◆ Sobre comportamentos de previsão de collision avoidance
 - If boid faz um bom trabalho com suas regras locais, não existiram colisões no futuro
- ◆ Collision avoidance estática serve para manter uma distância mínima de separação entre os boids – e o comportamento de igualar velocidades faz permanecer verdade

Ir ao centro do Flock

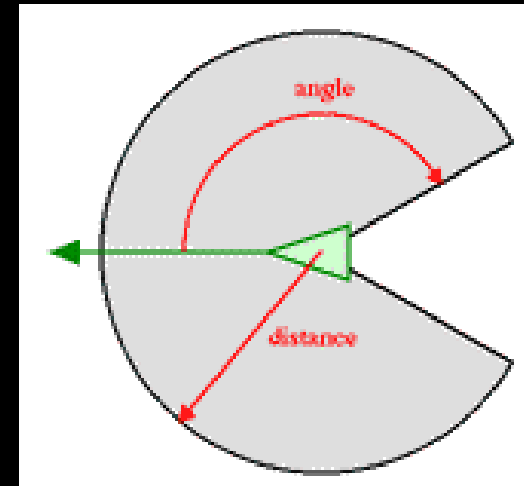


- ◆ O flock de cada boid é na verdade o centro dos boids mais próximos
- ◆ Isto faz com que cada boid voe na direção do centróide dos boids mais próximos
- ◆ Se um boid é perto do centro, este comportamento tem pouco efeito (desde que a densidade seja uniforme), mas se o boid está na borda, este comportamento tem grande efeito
- ◆ Permite o correto comportamento de splits

Percepção Simulada

- ◆ O modelo de percepção imita o modelo de pássaros reais
- ◆ Pássaros reais tem percepções imperfeitas— assim boids próximos podem ocultar pássaros que estão mais longe
- ◆ Movimento agregado dos flockings depende da sua percepção localizada e limitada
 - Exemplo do flock centering behaviour

Região de vizinhança



Cada boid tem acesso direto a toda a cena (modelo geométrico), no entanto o flocking requer que as reações sejam somente em relação aos flockmates, numa certa distância de vizinhança

A vizinhança é caracterizada pela distância (medida através do centro do boid) e um ângulo, medido através da direção de vôo do boid

Flockmates fora da vizinhança local são ignorados

A região de vizinhança define a região de influência dos boids

Outras aplicações

- ◆ Padrões de tráfego, peixes, crowds...



