

DCC192

2025/1



Desenvolvimento de Jogos Digitais

A9: Sistemas de Partículas

Prof. Lucas N. Ferreira

Plano de aula



- ▶ Sistemas de Partículas
 - ▶ Componentes
 - ▶ Detalhes de implementação
 - ▶ Exemplos
- ▶ TP2: Asteroids
 - ▶ Projéteis como sistemas de partículas
 - ▶ Forward vector

Sistemas de Partículas em Jogos Digitais



Geralmente, em jogos digitais queremos **modelar fenômenos naturais** orgânicos — aqueles em que uma malha sólida não consegue representar bem:



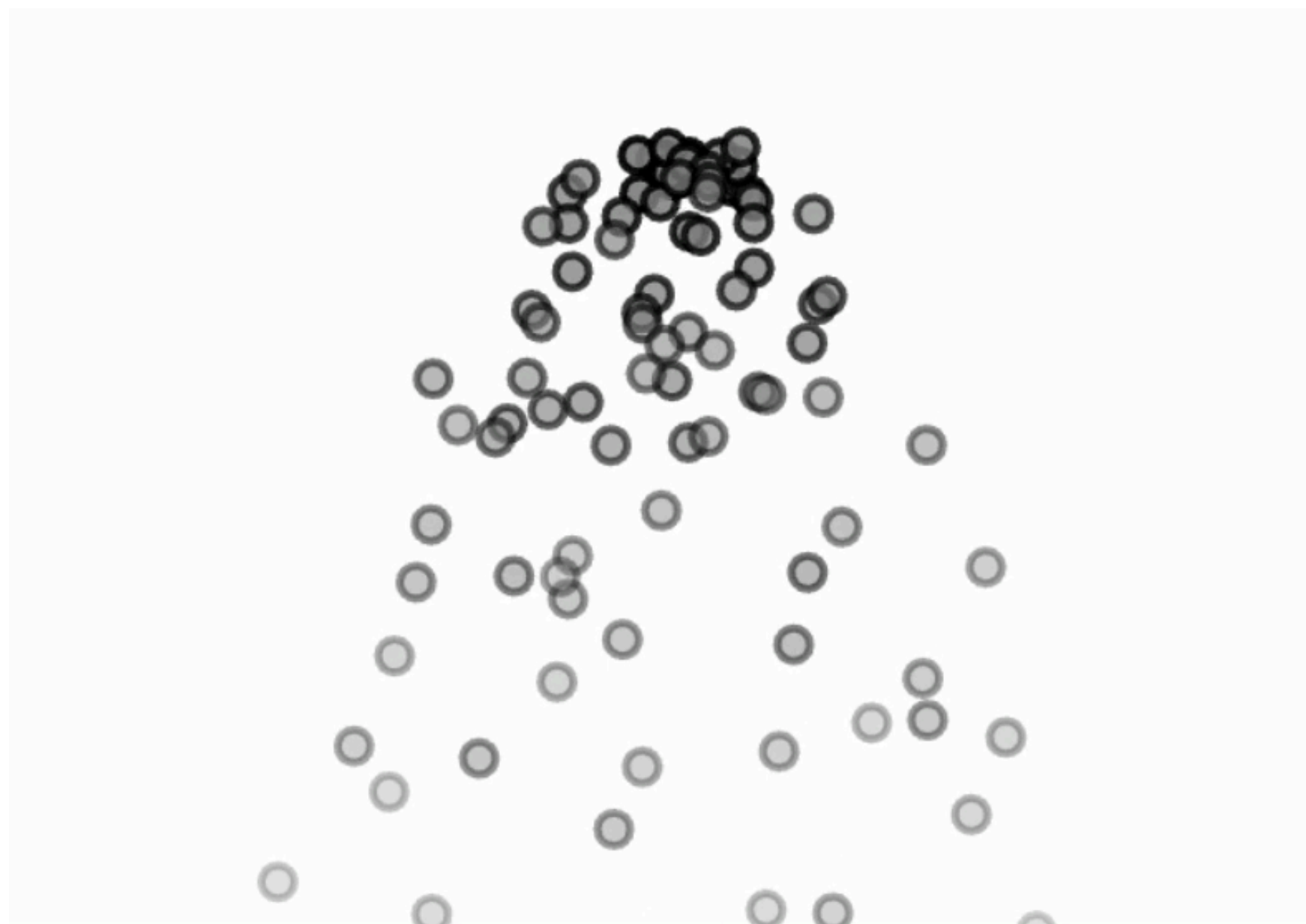
Por exemplo:

- ▶ Fogo
- ▶ Fumaça
- ▶ Explosões
- ▶ Tiros
- ▶ Estilhaços
- ▶ ...

Partículas, não malhas



Sistemas de partículas **modelam fenômenos naturais** complexos com uma coleção de pontos ou polígonos simples texturizados. Dessa forma, um sistema de partículas é composto por:



- ▶ **Partícula**

Objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo

- ▶ **Emissor de Partículas**

Uma estrutura de dados que gerencia add/del de partículas

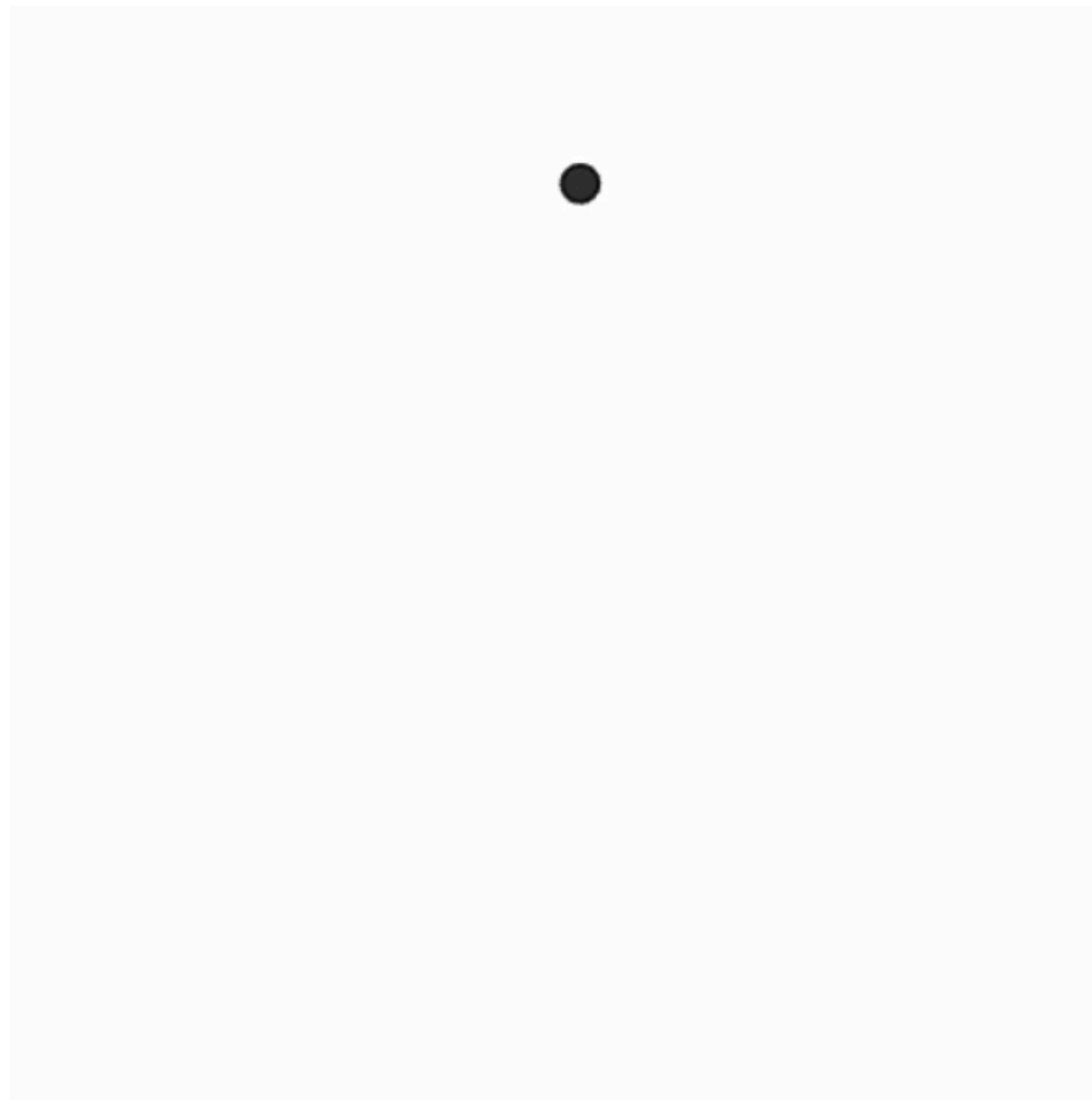
- ▶ **Sistemas de Emissores**

Uma estrutura de dados que gerencia múltiplos emissores

Partícula



Uma **partícula** é um objeto rígido sem rotação com tempo de vida limitado, i.e. morre após um certo tempo. Além das propriedades física (`pos`, `vel` e `acc`), uma partícula também possui:

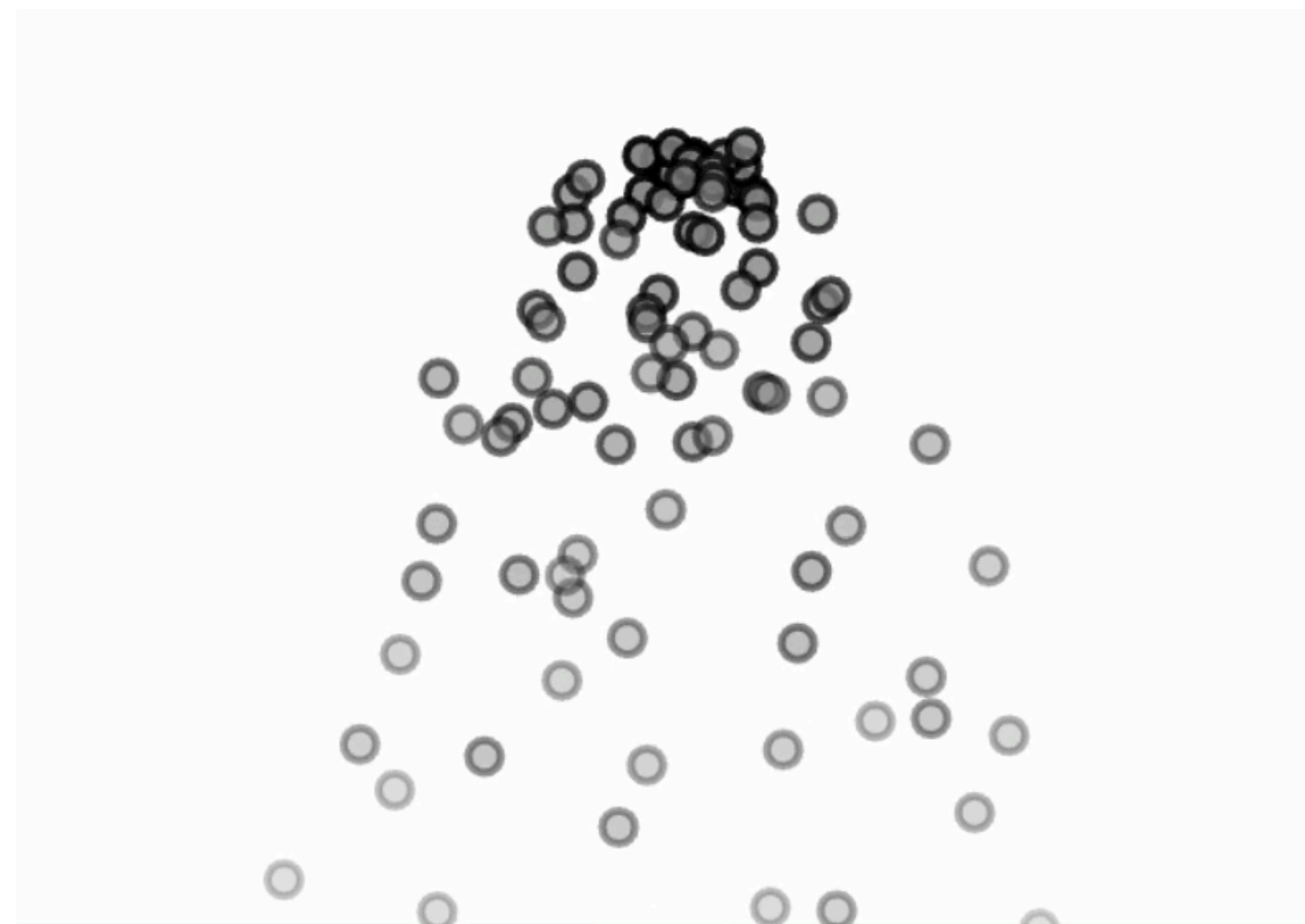


- ▶ **Lifetime**: tempo que a partícula permanecerá ativa
- ▶ **Color**: cor (RGBA) ou textura de renderização
- ▶ **Size**: tamanho da partícula em pixels
 - ▶ Circunferência → raio
 - ▶ Retângulo → largura x altura
- ▶ **IsDead**: estado de vida atual da partícula

Emissor de Partículas



Um **emissores de partículas** é uma estrutura de dados que gerencia a criação de remoção de partículas. As propriedades de uma emissor são as seguintes:



- ▶ **Particles**: lista de partículas do sistema
- ▶ **Position**: posição onde as partículas irão nascer
- ▶ **AddParticle()**: adiciona uma partícula
- ▶ **DelParticle(p)**: deleta uma partícula p

Sistemas de Emissores



Um **sistema de emissores** é uma estrutura de dados que gerencia a criação de remoção de emissores de partículas. As propriedades de uma sistema de emissores são as seguintes:

- ▶ **Emitters**: lista de emissores do sistema
- ▶ **AddEmitter()**: adiciona um emissor
- ▶ **DelEmitter(p)**: deleta um emissor

Normalmente, haverá uma série de sistemas de partículas trabalhando juntos para criar um único efeito visual. O sistema de emissores representa um único efeito visual.

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

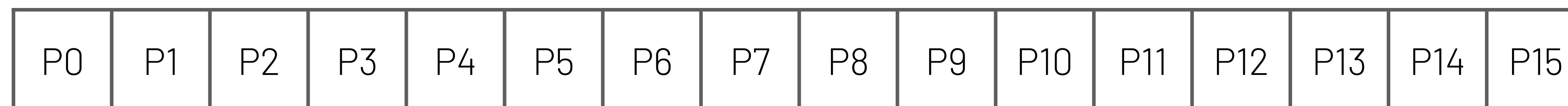
- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las



Lifetime - - - - - - - - - - - - - - -

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	1	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	3	1	3	-	-	-	-	-	-	-	-	-	-	-	-	-

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	2	-	2	-	-	-	-	-	-	-	-	-	-	-	-	-

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	1	3	1	-	-	-	-	-	-	-	-	-	-	-	-	-

Gerenciamento de Memória: Object Pool



Um sistema de partículas tipicamente possui muitas partículas que vivem muito pouco tempo.

- ▶ **Problema:** Alta frequência de alocação/desalocação de memória (fragmentação)
- ▶ **Solução:** Object Pooling
 - ▶ Alocar um *pool* de partículas durante a inicialização do emissor
 - ▶ Quando for adicionar uma partícula, procurar por partículas mortas e revivê-las

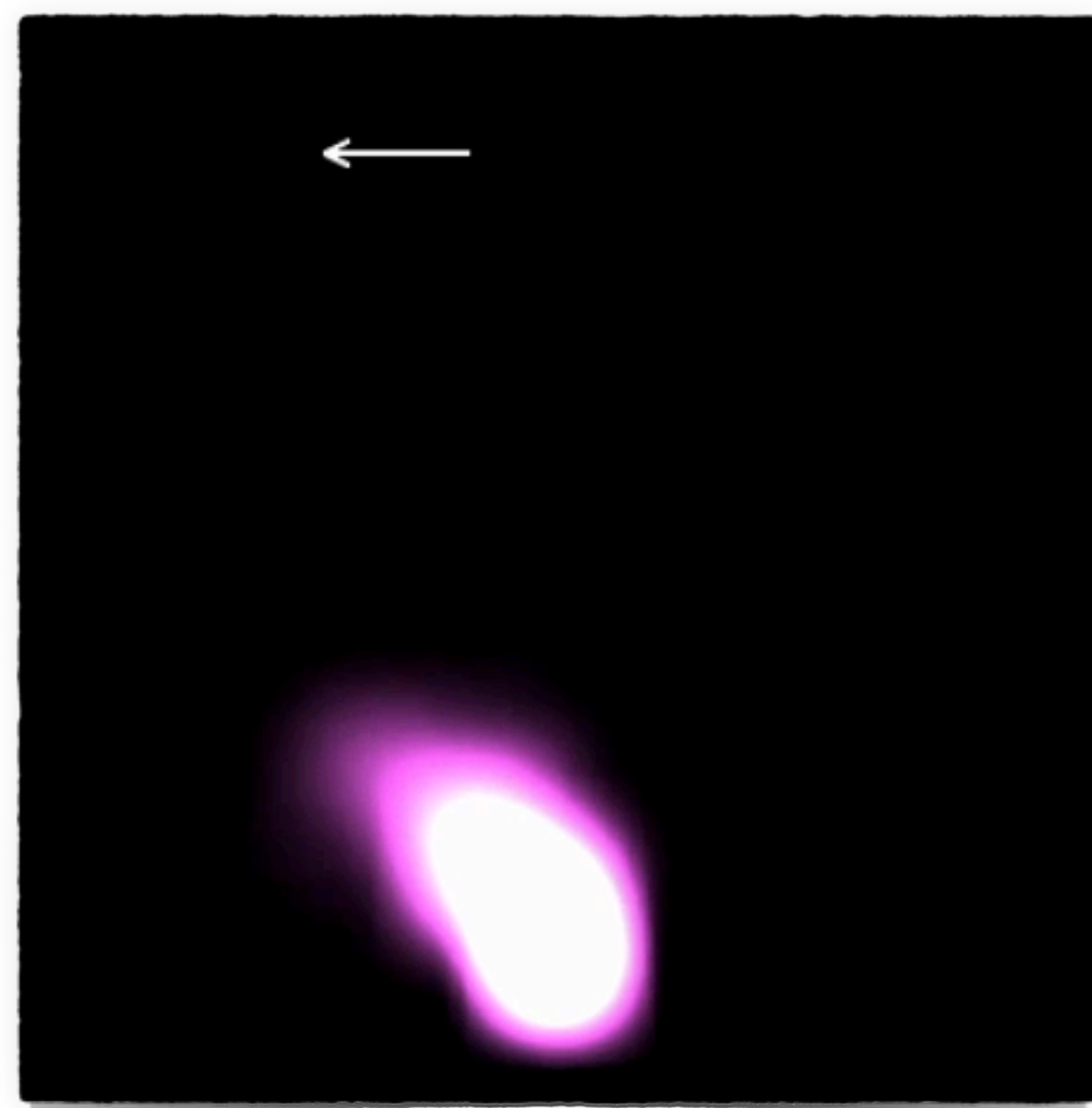
	P0	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
Lifetime	1	3	1	2	4	2	3	6	2	3	3	1	3	1	2	1

Se todas as partículas estiverem ocupadas, não será possível criar partículas por um tempo!

Modelando Comportamentos Orgânicos



Para **modelar comportamentos orgânicos**, podemos configurar diferentes parâmetros do sistema de partículas:



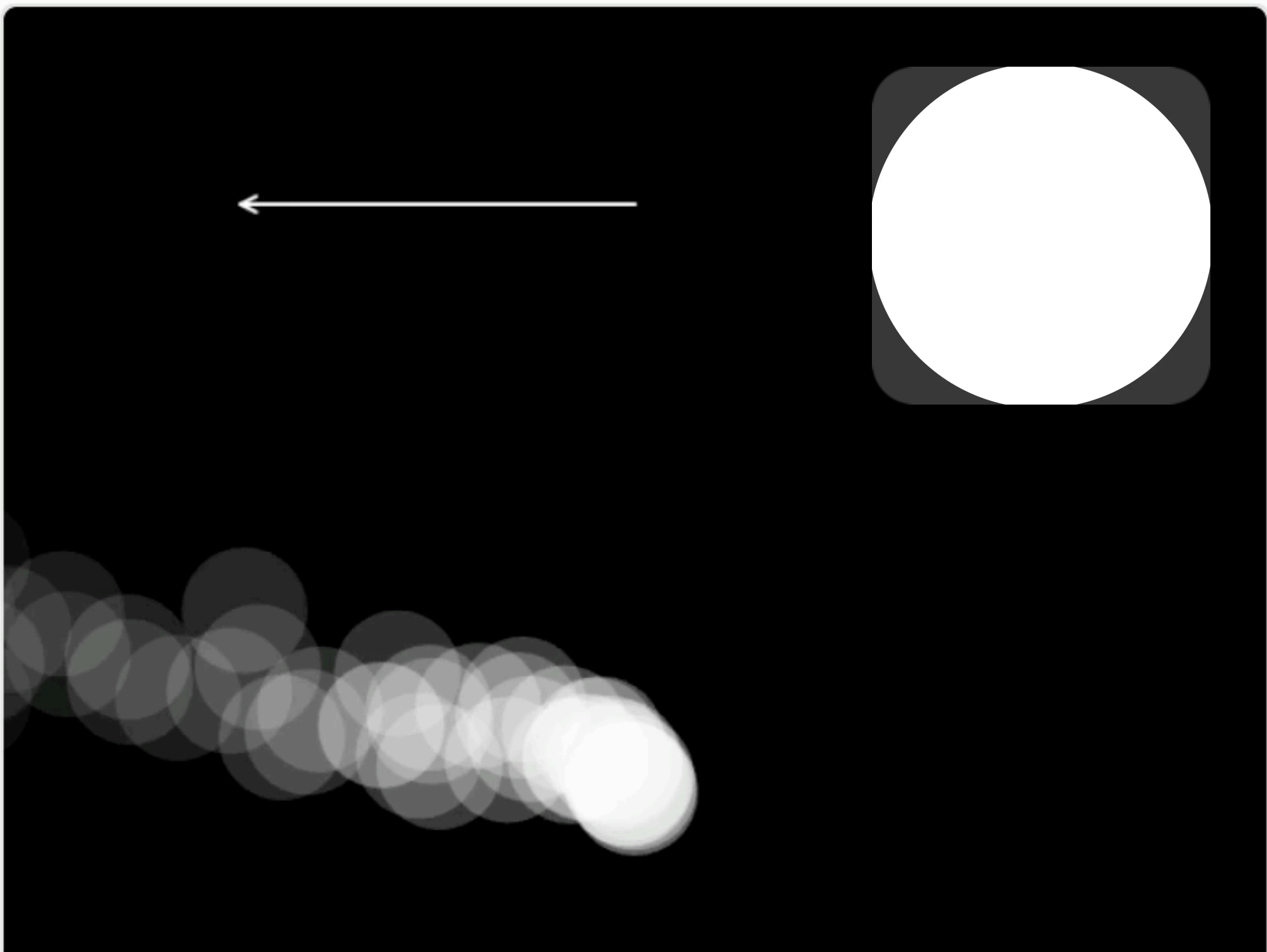
- ▶ Número, posição e tempo de vida de partículas
 - ▶ Tempo aleatório em um intervalo [min, max]
- ▶ Número e posição de emissores
- ▶ Frequência de emissão dos emissores
- ▶ ...

Além disso, podemos aplicar forças às partículas!

Exemplo 1: Fogo



Cor sólida com transparência

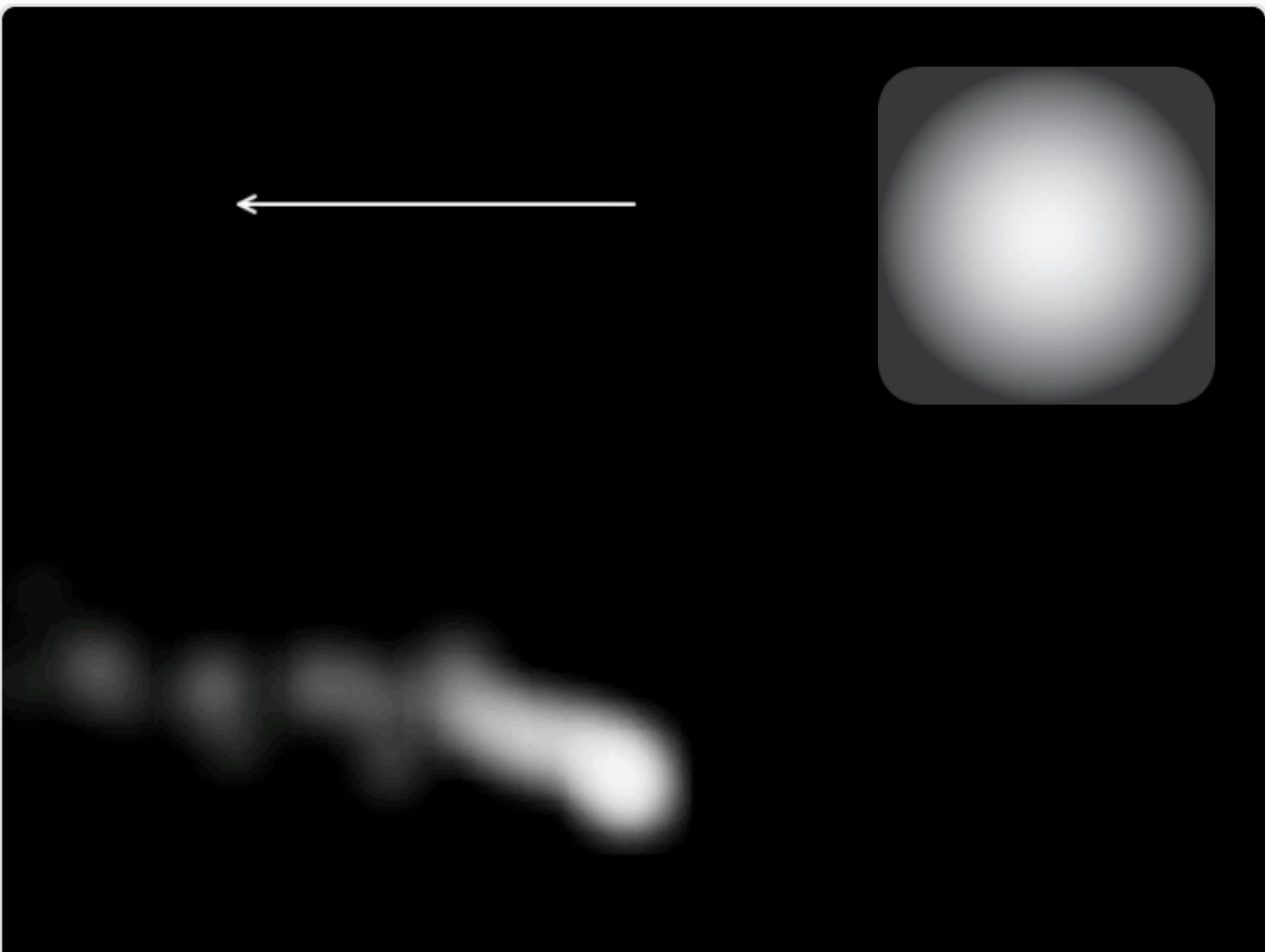


↺ Reset

⏸ Pause

[Open in Web Editor](#) ↗

Textura com fade out

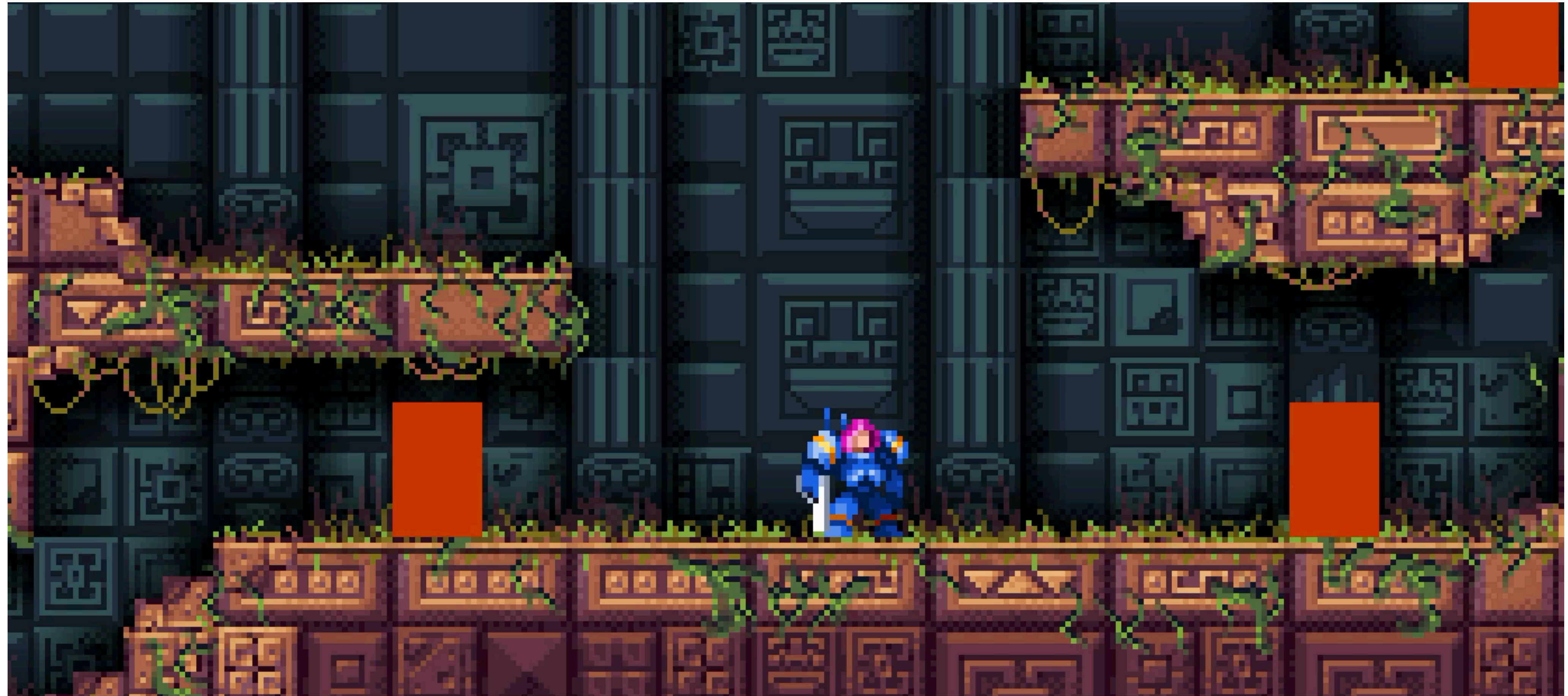


↺ Reset

⏸ Pause

[Open in Web Editor](#) ↗

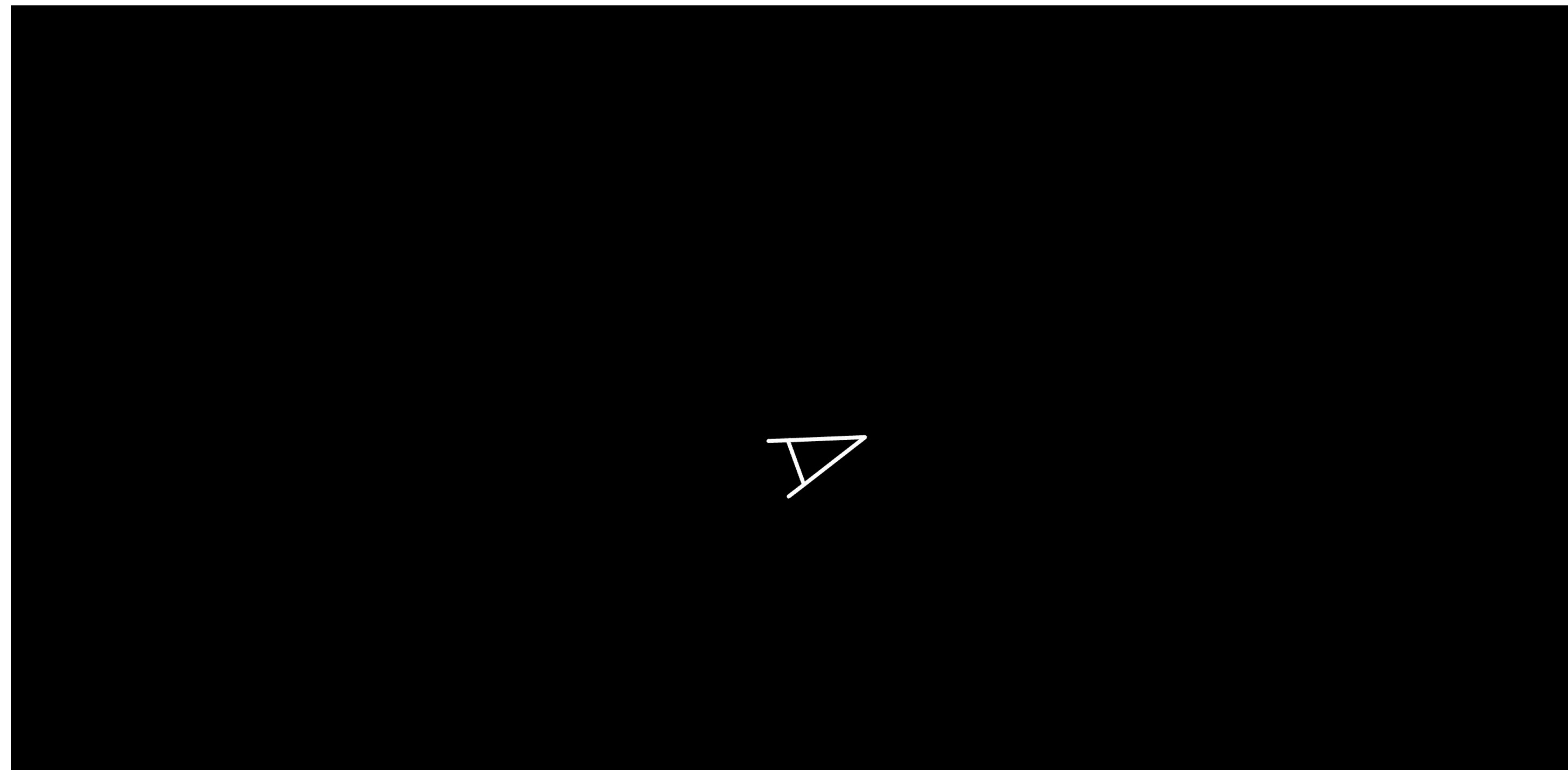
Exemplo 2: Projéteis



TP2: Asteroids



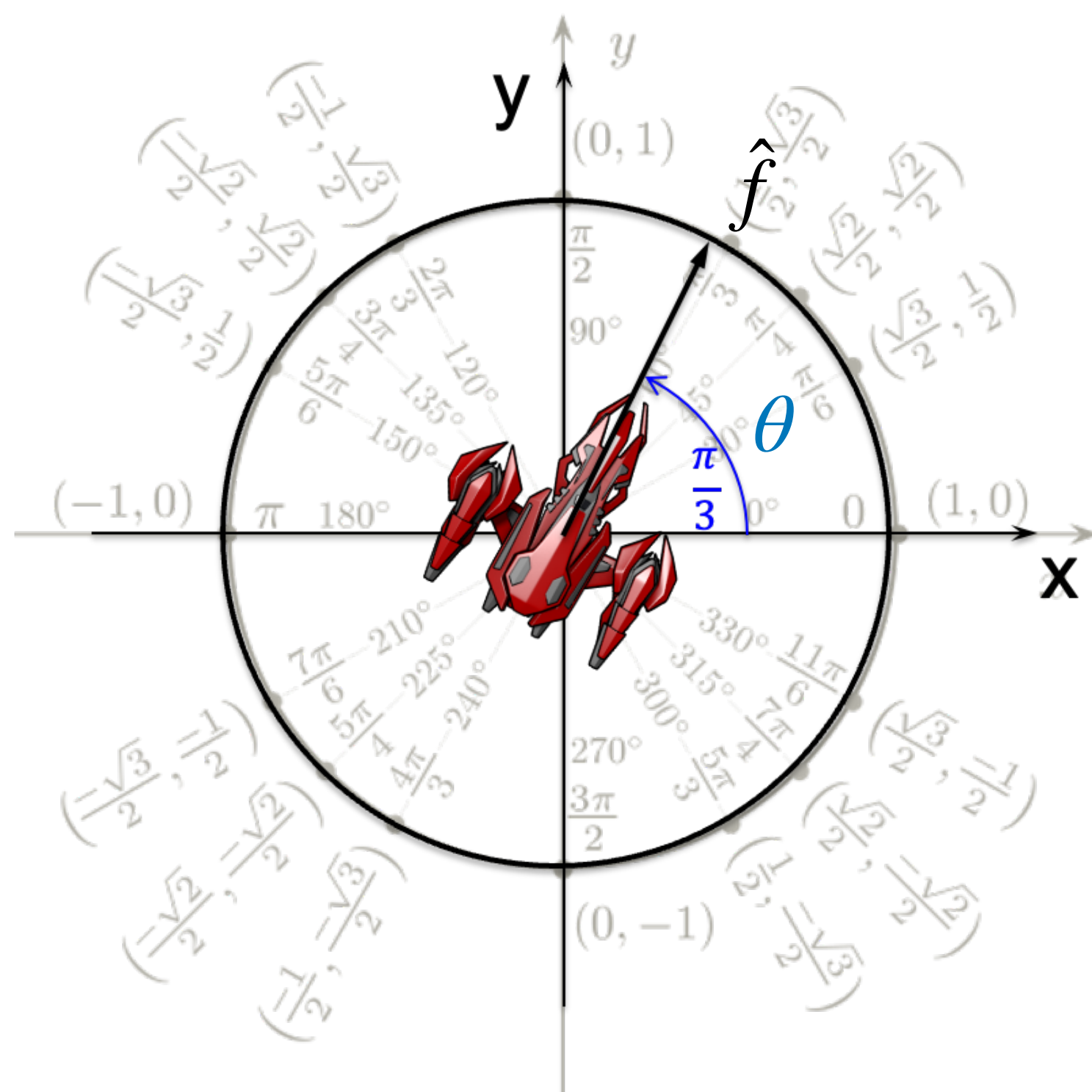
No **TP2: Asteroids**, os tiros que o jogador pode atirar da nave serão implementados com sistemas de partículas:



TP2: Asteroids



No **TP2: Asteroids**, será necessário calcular o *forward vector* \hat{f} , um vetor unitário que representa a direção da nave, a partir de um ângulo de rotação (ex., $\frac{\pi}{3}$)



► Como obter o vetor \hat{f} a partir do ângulo θ ?

$$f_x = \cos(\theta)$$

$$f_y = \sin(\theta)$$

TP2: Asteroids



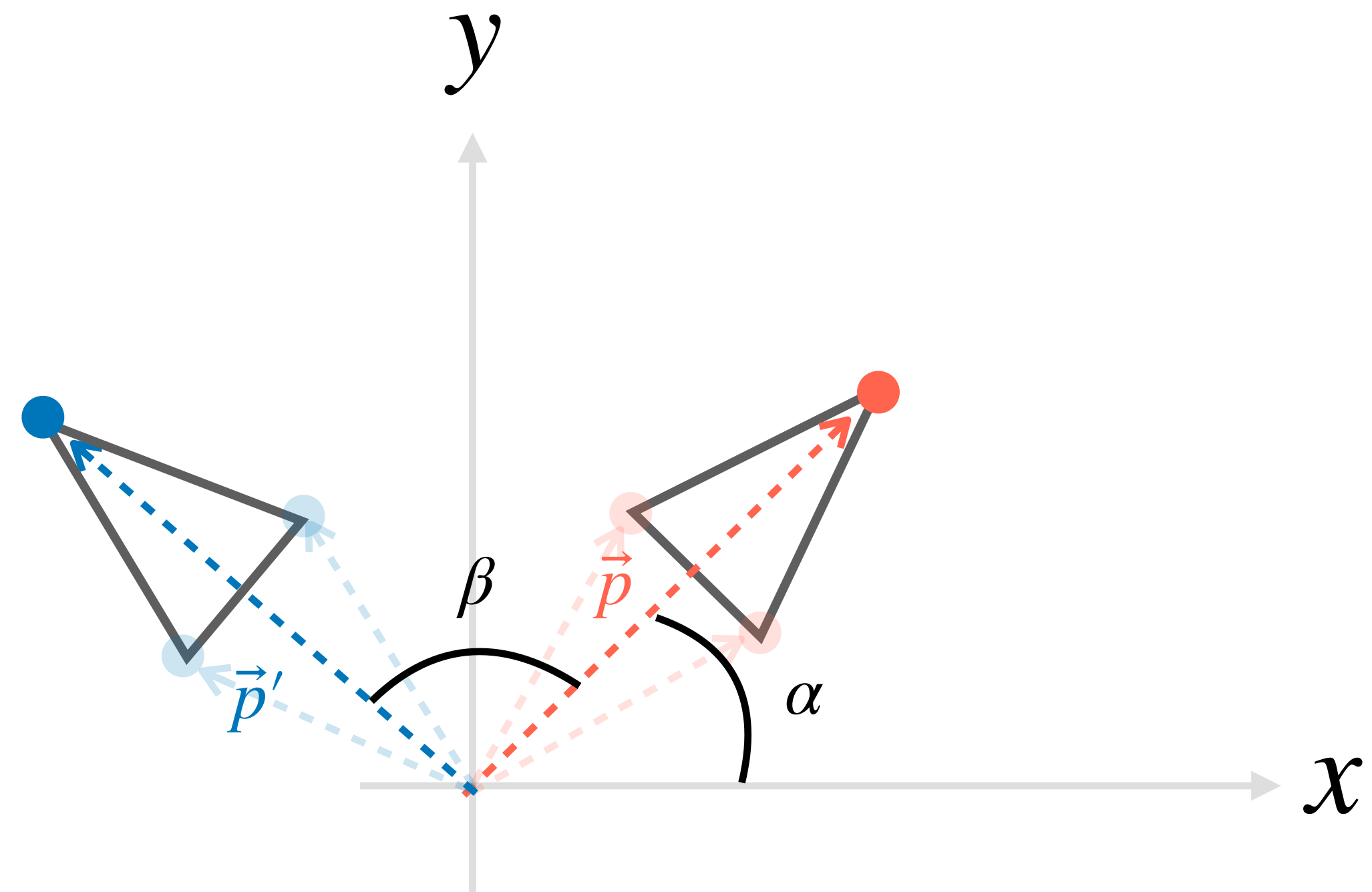
No **TP2: Asteroids**, será necessário rotacionar os vértices que representam a nave:

$$\vec{p'} = R \vec{p} = \begin{bmatrix} \cos \beta & -\sin \beta \\ \sin \beta & \cos \beta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\begin{aligned} x &= r \cdot \cos \alpha & y &= r \cdot \sin \alpha \\ x' &= r \cdot \cos(\alpha + \beta) & y' &= r \cdot \sin(\alpha + \beta) \end{aligned}$$

$$\begin{aligned} x' &= r \cdot \cos \alpha \cdot \cos \beta - r \cdot \sin \alpha \cdot \sin \beta \\ y' &= r \cdot \cos \alpha \cdot \sin \beta + r \cdot \sin \alpha \cdot \cos \beta \end{aligned}$$

$$\begin{aligned} x' &= x \cdot \cos \beta - y \cdot \sin \beta \\ y' &= x \cdot \sin \beta + y \cdot \cos \beta \end{aligned}$$



Próxima aula



A10: Gráficos 2D

- ▶ Sprites/Spritesheets
- ▶ Animações
- ▶ Tilemaps