Motor de jogos e Física

Etapas na simulação física de uma *game engine*

Slides por:
Gustavo Ferreira Ceccon (gustavo.ceccon@usp.br)

Este material é uma criação do
Time de Ensino de Desenvolvimento de Jogos Eletrônicos (TEDJE)
Filiado ao grupo de cultura e extensão Fellowship of the Game (FoG), vinculado ao ICMC - USP

Objetivos

- Introduzir física e a matemática de um motor de jogos
- Mostrar o processo da simulação física
  - Simulação e métodos de integração
  - Broadphase, Midphase e Narrowphase
  - Colisão e organização dos dados
  - Algoritmos básicos de colisão
  - Algoritmo de Impulso Sequencial

Índice

1. Introdução

1. Introdução

- Física no mundo real
  - Contínua, não há saltos de tempo
  - Contato, os objetos se tocam e não intersectam
- Física no mundo virtual
  - Discreta, a cada frame passa um dt
  - Intersecção, os objetos podem ocupar mesmo espaço

1. Introdução

- Física 2D
  - Colisores: quadrados, círculos
  - Algoritmos de colisão diferentes (SAT)
  - Transformações diferentes
    - R^2
    - Rotação com um grau de liberdade
- Física 3D
  - Colisores: cubos, esferas
  - Algoritmos de colisão diferentes (GJK)
  - Transformações diferentes
    - R^3
    - Rotação com três graus de liberdade

3. Simulação

- Tipos de objetos
  - Dinâmico com colisão: Mario
  - Dinâmico sem colisão: Moeda
  - Estático com colisão: Blocos
  - Estático sem colisão: Fundo

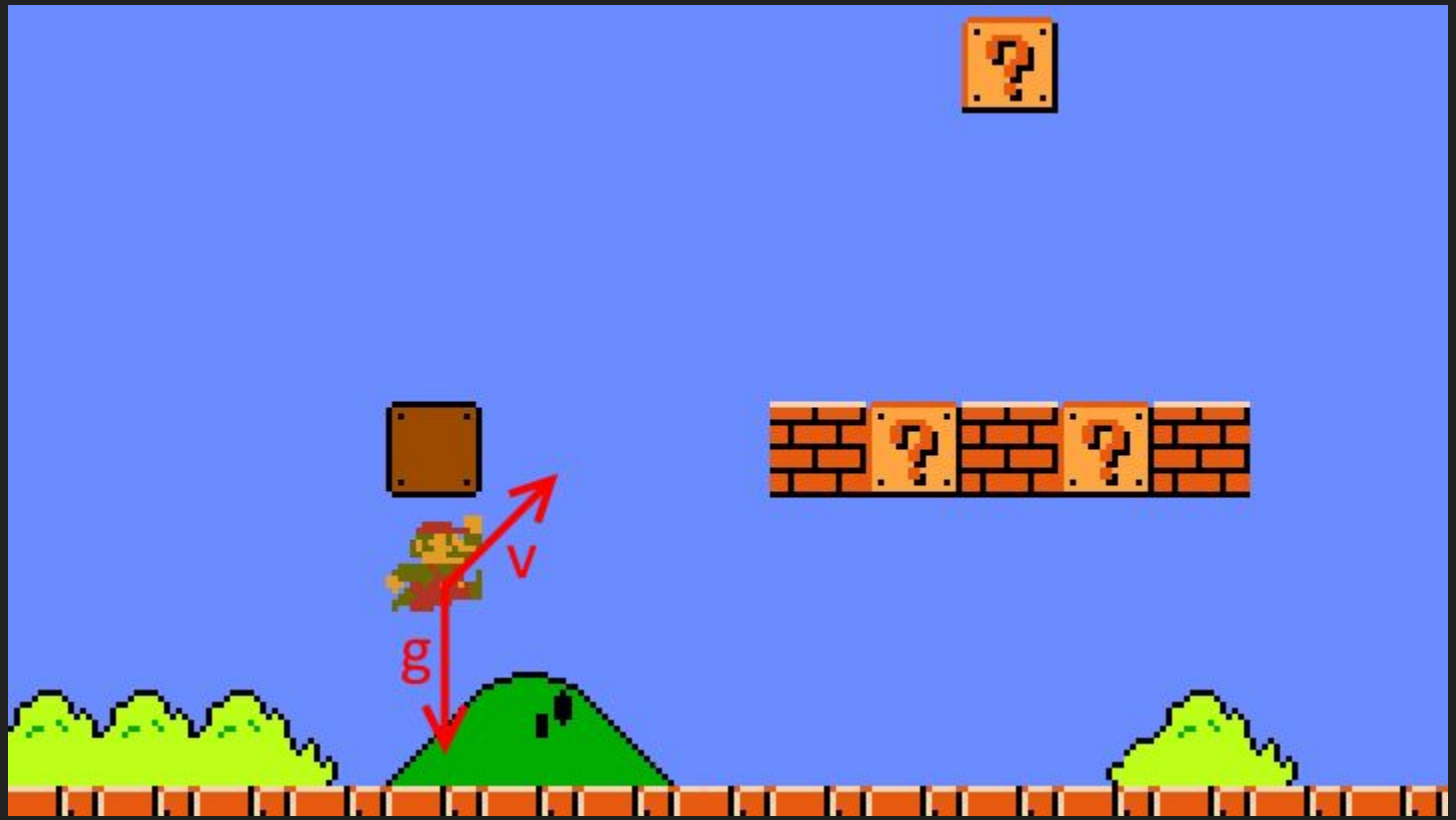
2. Etapas

2. Etapas

- Etapas do motor de física
  - Simulação
  - Detecção de Colisão
  - Resposta

2. Etapas

Simulação



Simulação

Detecção de Colisão

Resposta

Resposta

3. Simulação

3. Simulação

- Aplicar física newtoniana, que é suficiente para velocidades baixas
  - Aplicamos as forças para descobrir as acelerações
  - Aplicamos aceleração para descobrir a velocidade
  - Aplicamos a velocidade para descobrir a posição

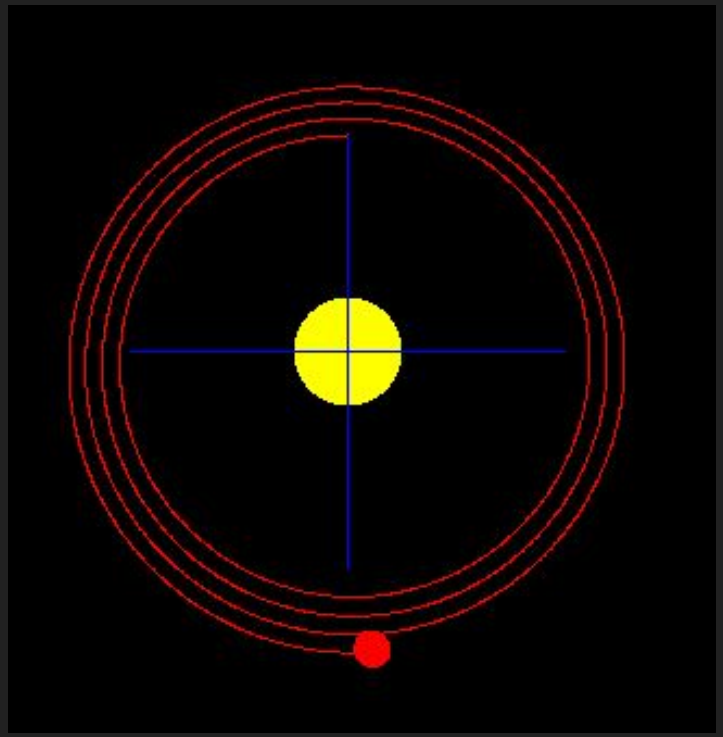
3. Simulação

- Não conseguimos aplicar as equações de ensino médio, porém elas são parecidas
  - Forças não são constantes, mundo discreto
- É preciso usar integrais para fazer os cálculos
  - Cálculo numérico aprendemos alguns métodos de integração, que são bem parecidos com esses

3. Simulação

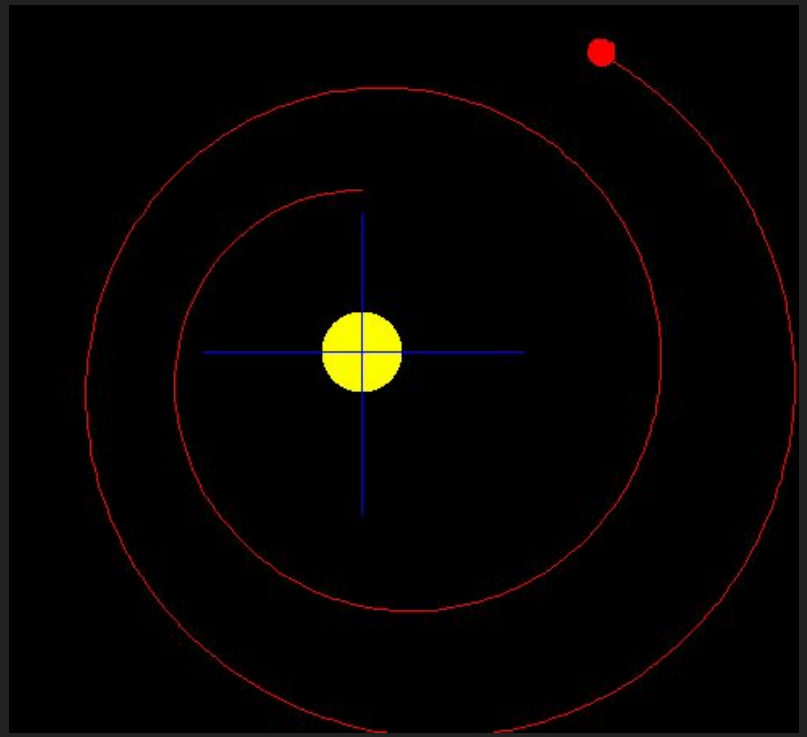
- Integração numérica é aproximar um valor de uma integral usando métodos computacionais (passos)
  - Diferentes métodos para diferentes tipos físicos
  - Euler e *Semi-Implicit* Euler (parecidos com ensino médio)
  - Verlet e *Time-Corrected* Verlet (pulam uma etapa)
  - Runge-Kutta 4ª ordem (mais acurado)

3. Simulação



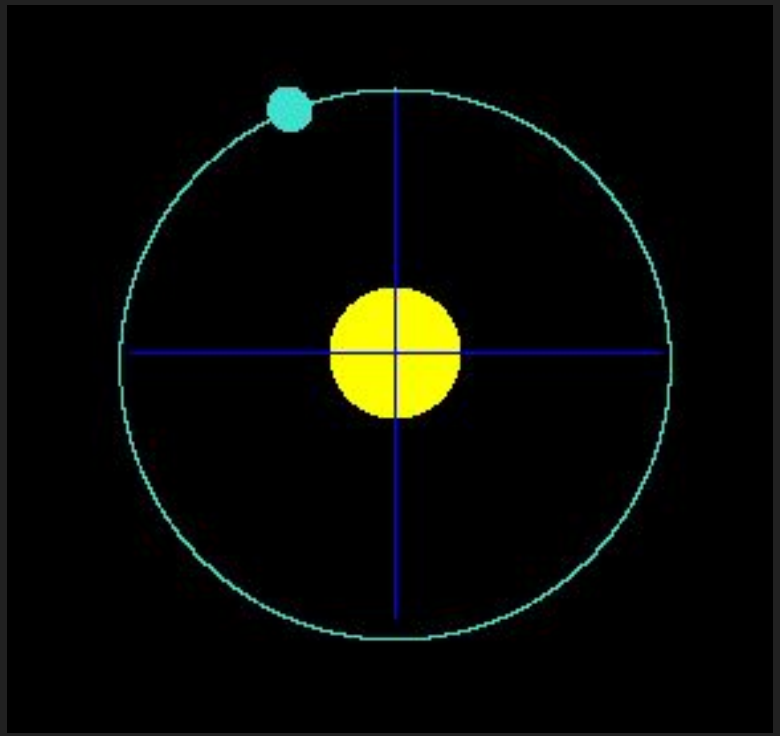
Euler 10 UPS

3. Simulação



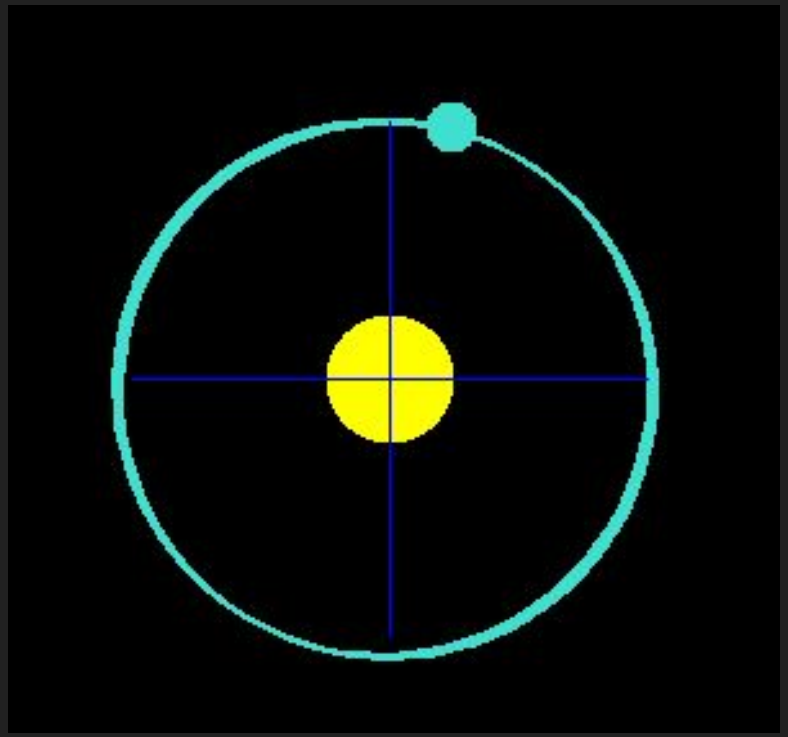
Euler 1 UPS

3. Simulação



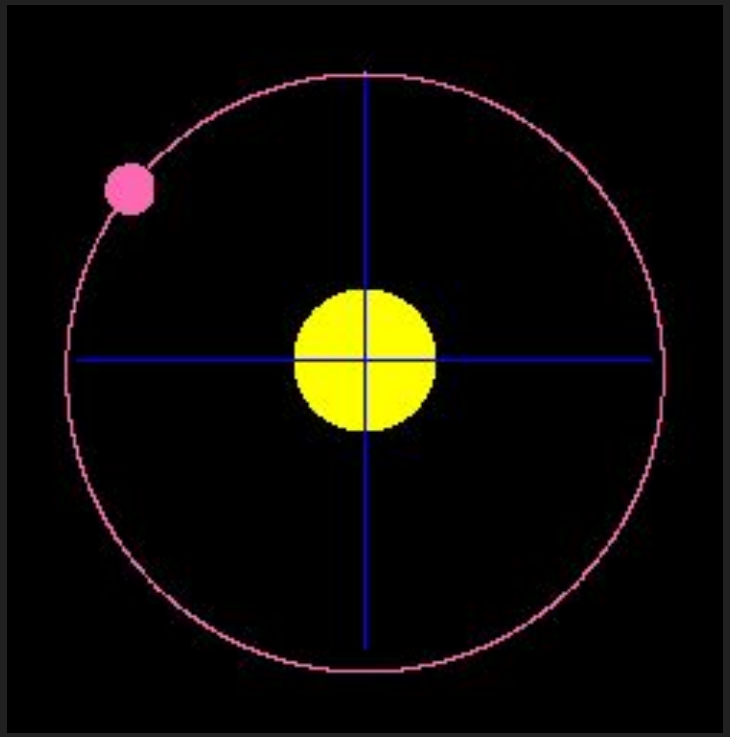
Verlet 10 UPS

3. Simulação



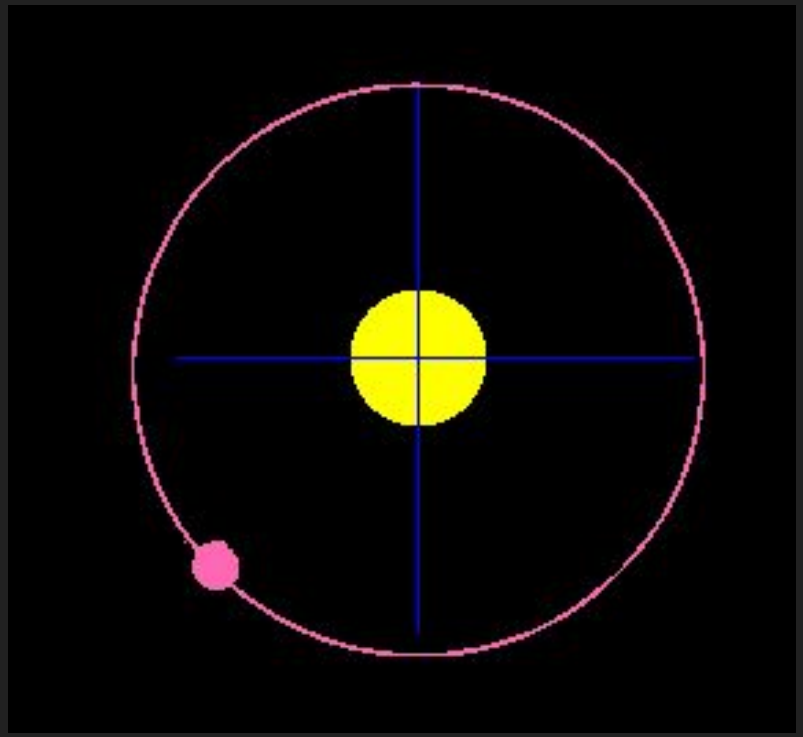
Verlet 1 UPS

3. Simulação



RK4 10 UPS

3. Simulação



RK4 1 UPS

3. Simulação

- Problemas possíveis
  - Forças variáveis
  - dt variável
- Problemas causam inconsistência na física
- Quanto maior a ordem do método, melhor a aproximação, porém maior custo computacional

4. Colisão

4. Colisão

- Precisamos testar a colisão de **n** objetos
  - Força bruta demoraria O(**n**²) para testar a colisão entre si todos os objetos
  - Precisamos diminuir o número de testes, além disso precisamos coletar as informações das colisões

4. Colisão

- Dividimos em três etapas a colisão
- Broad-phase
  - Separação brusca dos objetos impossíveis de colidir
- Mid-phase
  - Teste rápido de colisão entre objetos, usando colisores simples
- Narrow-phase
  - Teste preciso de colisão entre objetos, usando diferentes tipos de colisores (compostos) e algoritmos de colisão

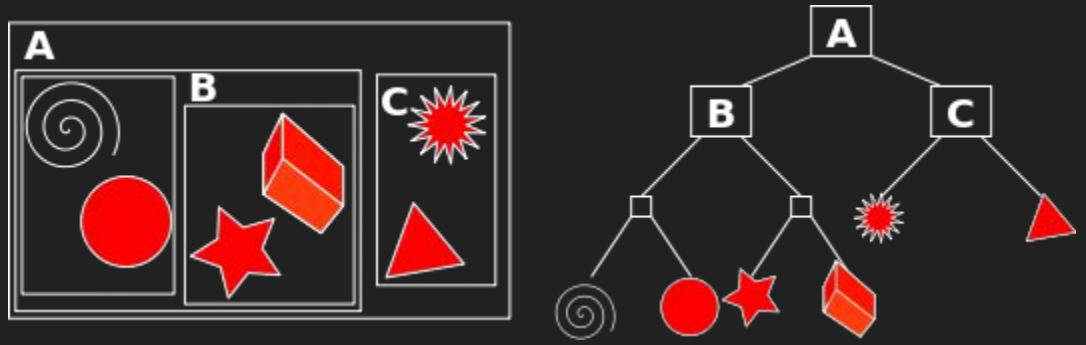
4. Colisão

- ➔ Broad-phase (algoritmos e colisores)
  - ◆ BVH - Bounding Volume Hierarchy
  - ◆ Spatial Partitioning
    - Quadtree e Octree
    - k-d tree
    - BSP - Binary Space-Partitioning Tree

4. Colisão

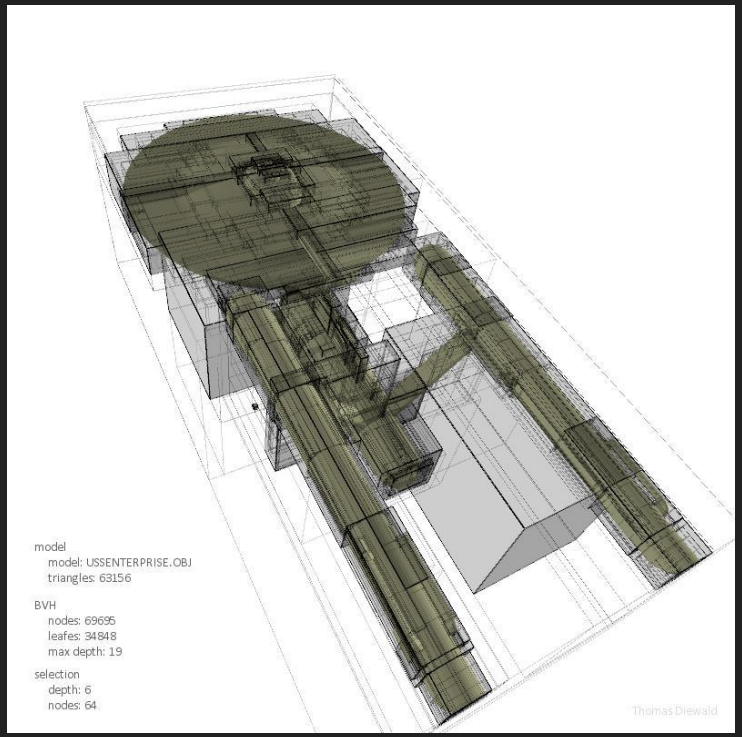
- BVH - Bounding Volume Hierarchy
  - Árvore de colisores de bounding volumes (como uma caixa ou uma esfera)
  - Nós são colisores aproximados de todos os colisores filhos
  - Quanto mais alta hierarquia, pior a aproximação
  - Usados tanto para teste quanto para representação

4. Colisão



Bounding Volume Hierarchy

4. Colisão



Bounding Volume Hierarchy

4. Colisão

- BVH - Bounding Volume Hierarchy
  - Estática: usada geralmente em um objeto para subdividir os colisores
  - Dinâmica: usada em múltiplos objetos (geralmente dinâmicos), calculada em tempo de execução
    - São pesadas para calcular todo frame

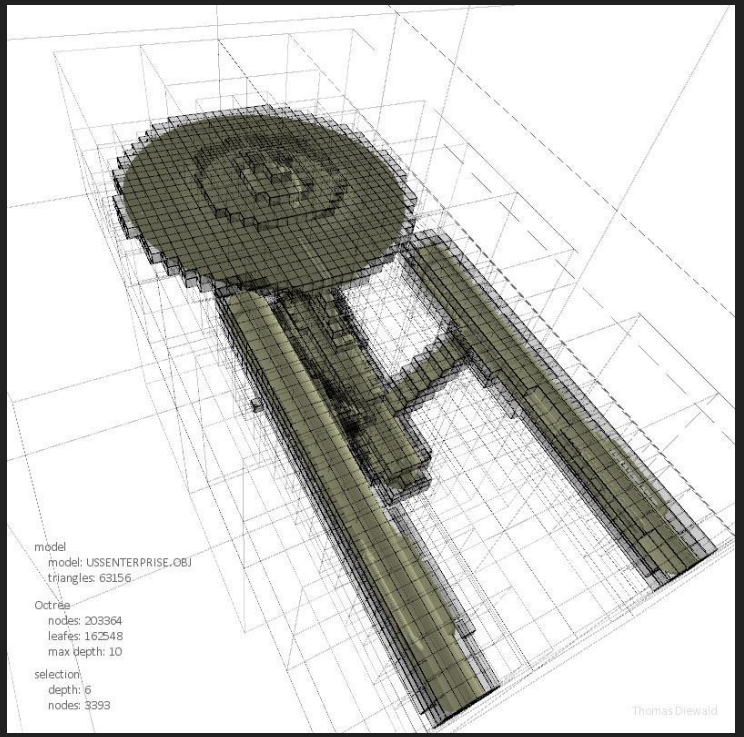
4. Colisão

- ➔ Spatial Partitioning
  - Dividir o espaço em partes onde pode haver colisão
  - Objetos dentro do mesmo espaço são checados posteriormente
  - Pode levar em consideração o game design, como paredes

4. Colisão

- Quadtree e Octree
  - Divisão binária simétrica, para cada eixo, dos colisores
  - São relativamente rápidas de calcular, podendo ser montadas em tempo de compilação
  - Diminuem consideravelmente o número de comparações necessárias

4. Colisão



Octree

4. Colisão

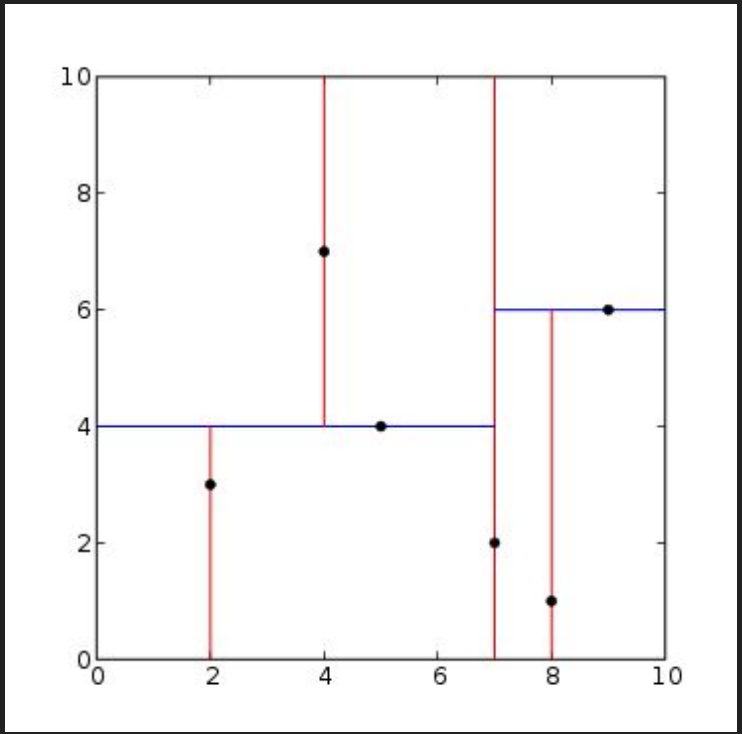
Quadtree

Quadtree 2

4. Colisão

- k-d tree
  - Generalização da quadtree e da octree
  - Divisão binária de tamanho arbitrário, para cada eixo
  - Maior custo de computação, porém maior precisão
  - Quadtree e Octree podem conter objetos em múltiplos espaços, k-d tree pode resolver isso

4. Colisão

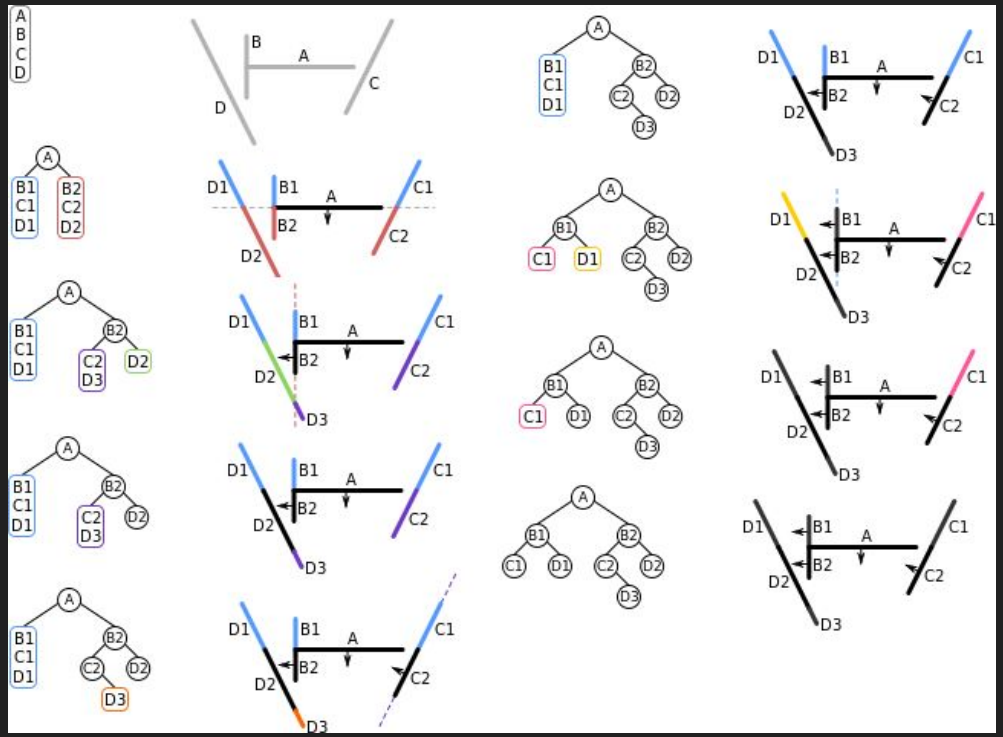


k-d tree

4. Colisão

- BSP (Binary Space-Partitioning) tree
  - Generalização ainda maior da k-d tree, onde os colisores são divididos de forma arbitrária
  - Muito pesada de calcular, porém muito eficaz para certos casos
  - Usada em objetos estáticos, como um cenário urbano (objetos geralmente são paralelos entre si)
  - Usada em renderização (ordenação dos objetos), ao invés do Z-buffer, técnica de computação gráfica

4. Colisão



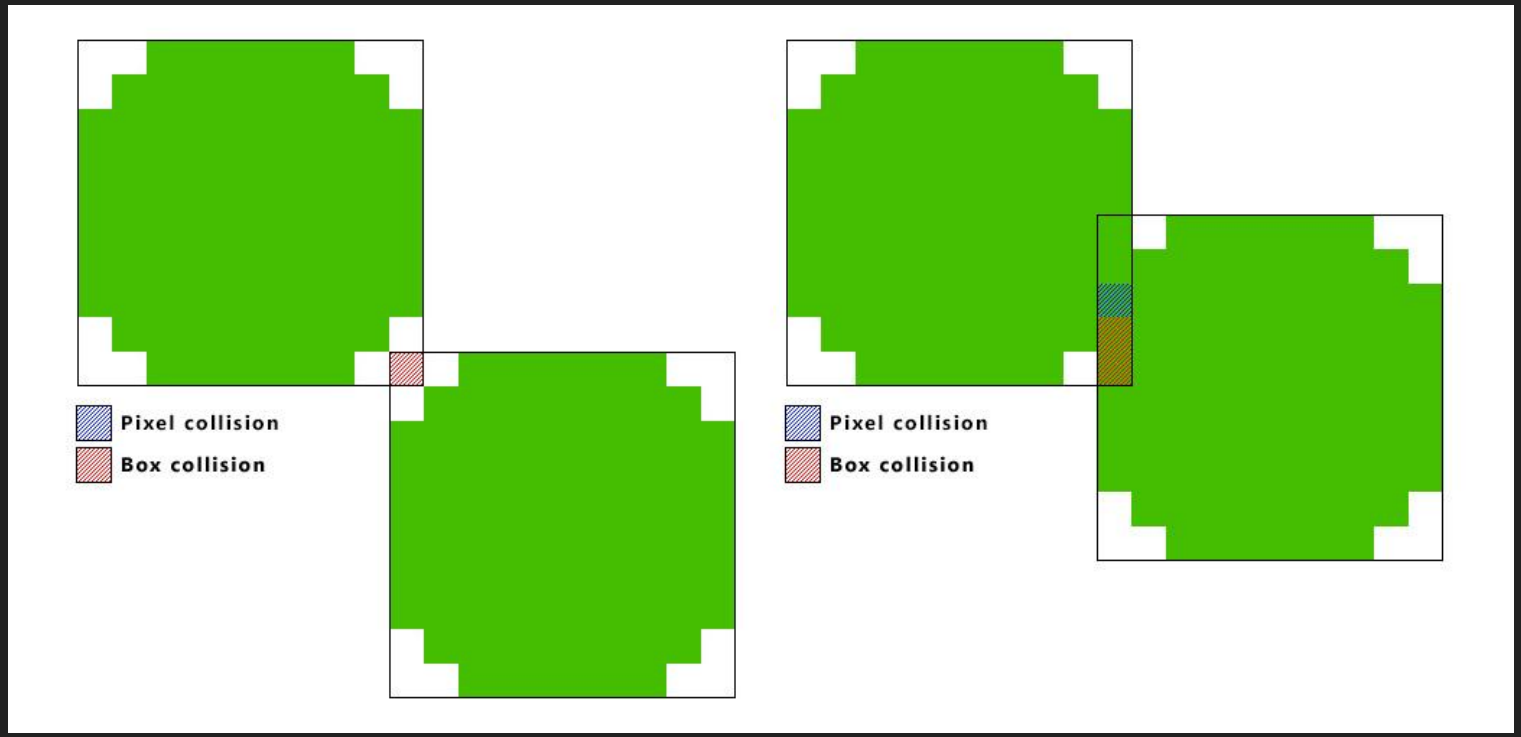
BSP tree

5. Tipos de Colisão

5. Tipos de Colisão

- ➔ Objeto 2D ou 3D
- ➔ Pixel-level
- ➔ Formas primitivas
- ➔ Formas côncavas e convexas
- ➔ Raycast
- ➔ SAT - Separating Axis Theorem
- ➔ GJK - Gilbert–Johnson–Keerthi

5. Tipos de Colisão



Colisão *pixel-level*

5. Tipos de Colisão

- Formas primitivas
  - Esfera-Esfera
  - AABB - *Axis Aligned Bounding Box*
  - AABB-AABB
  - AABB-Esfera

5. Tipos de Colisão

Exemplo

5. Tipos de Colisão

- Formas côncavas
  - Dividir em formas convexas
  - Não funciona com os algoritmos usados
  - Côncavo para convexo
    - Decomposição e triangulação

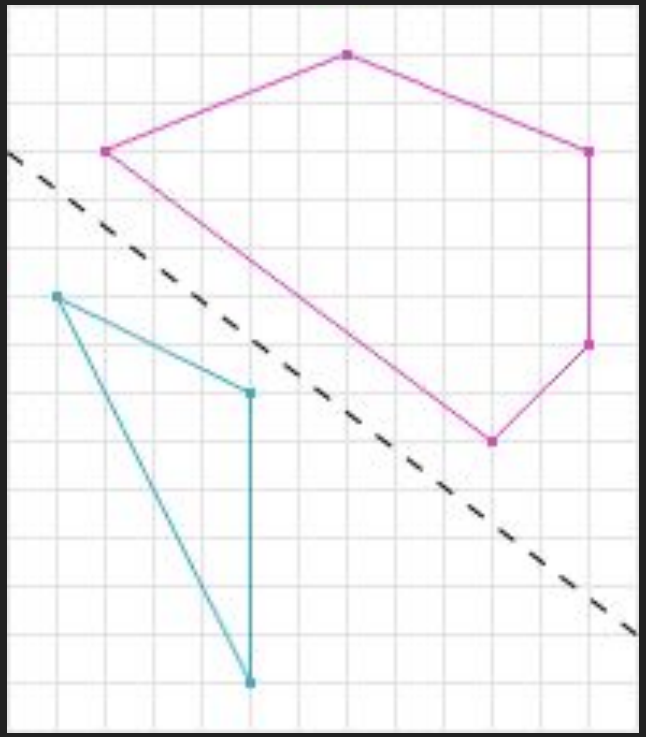
5. Tipos de Colisão

- Raycast
  - Raio de colisão
  - Útil em aproximação de objetos rápidos
    - Projétil
  - Muito pesado normalmente
    - Broadphase é necessário para reduzir o tempo
  - Pode testar até o primeiro alvo ou todos os objetos

5. Tipos de Colisão

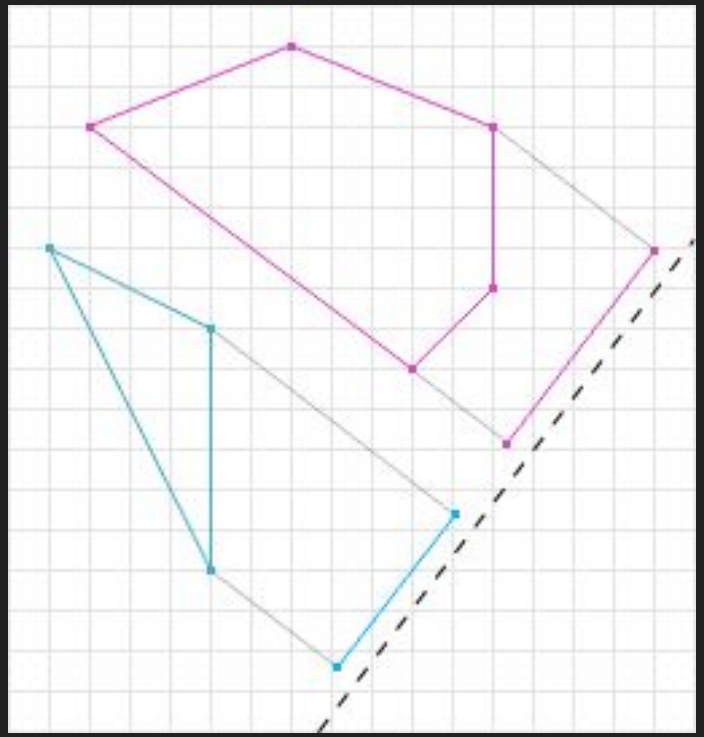
- SAT - Separating Axis Theorem
  - A ideia é testar a colisão em diferentes eixos dos objetos, projetando eles em uma reta ou em um plano
  - Algumas bibliotecas de física 2D (Box2D) implementam o SAT e deixam o usuário escolher qual algoritmo usar
  - Complexidade aumenta com a quantidade de vértices

5. Tipos de Colisão



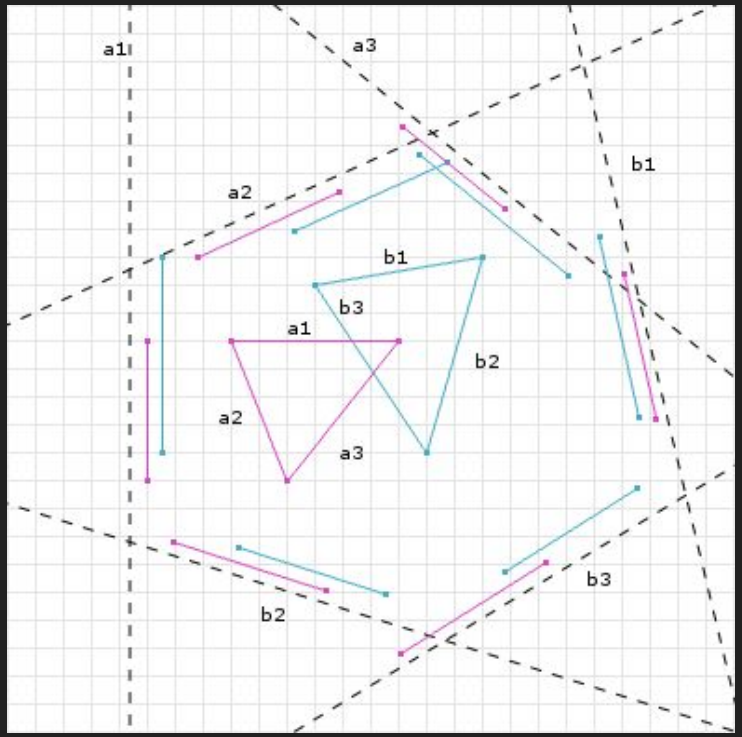
SAT - *Separating Axis Theorem*

5. Tipos de Colisão



SAT - *Separating Axis Theorem*

5. Tipos de Colisão

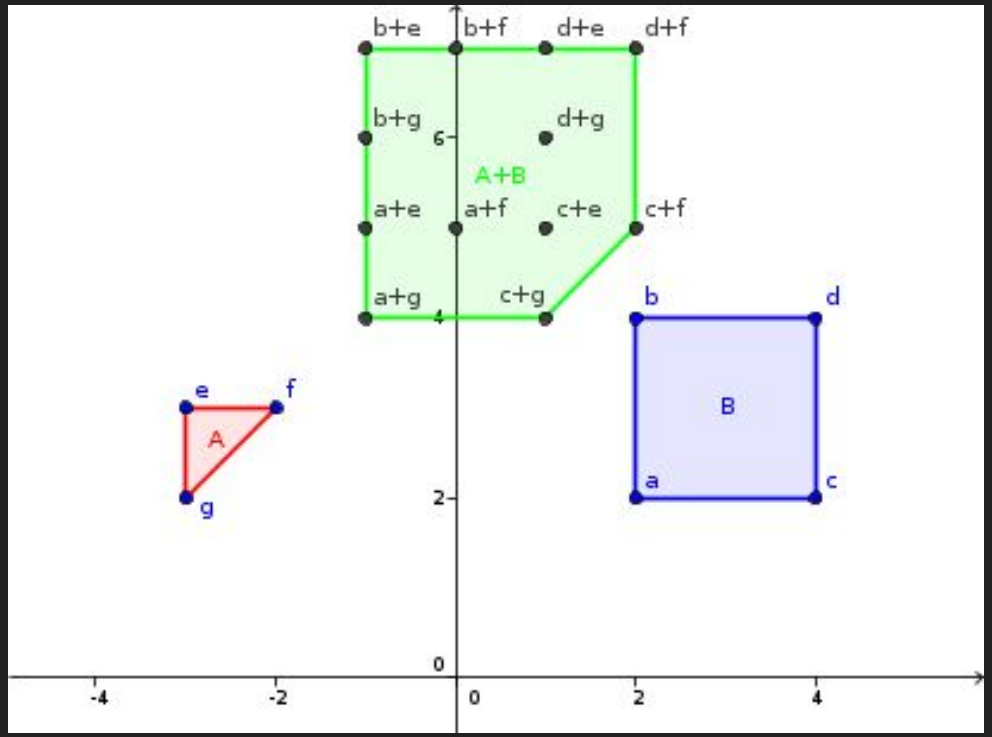


SAT - *Separating Axis Theorem*

5. Tipos de Colisão

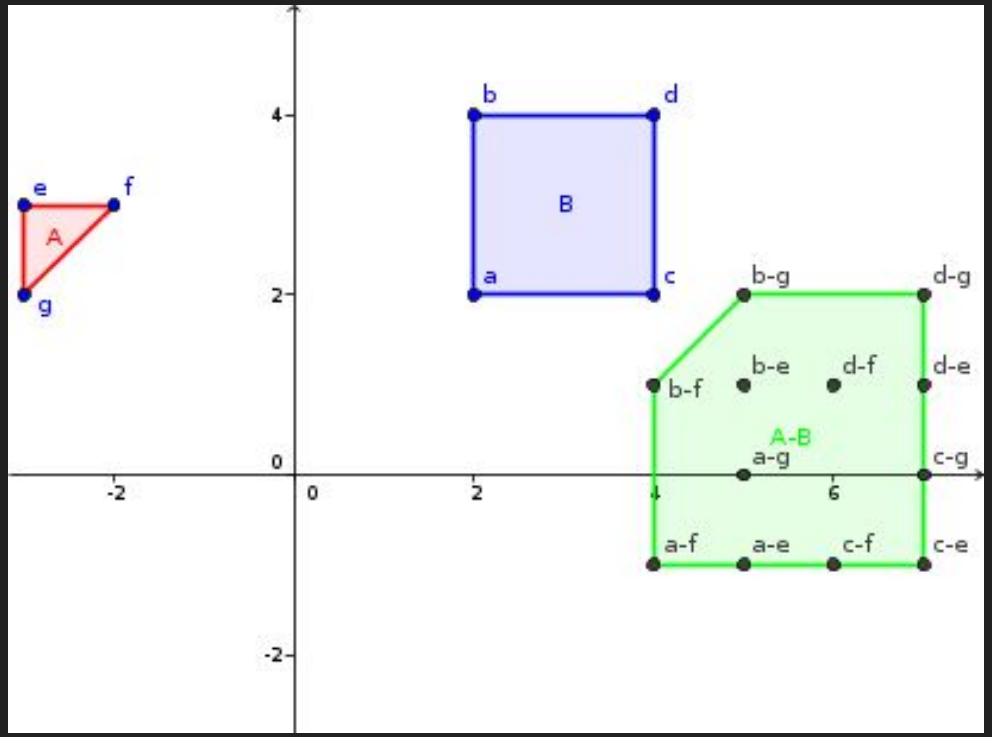
- GJK - Gilbert–Johnson–Keerthi
  - Nomeado aos autores do algoritmo
  - É mais rápido na maior parte dos casos
  - GJK pode ser reaproveitado para descobrir mais informações
    - Distância e pontos mais próximos
    - Informações de colisão, usando extensões do algoritmo (EPA)

5. Tipos de Colisão



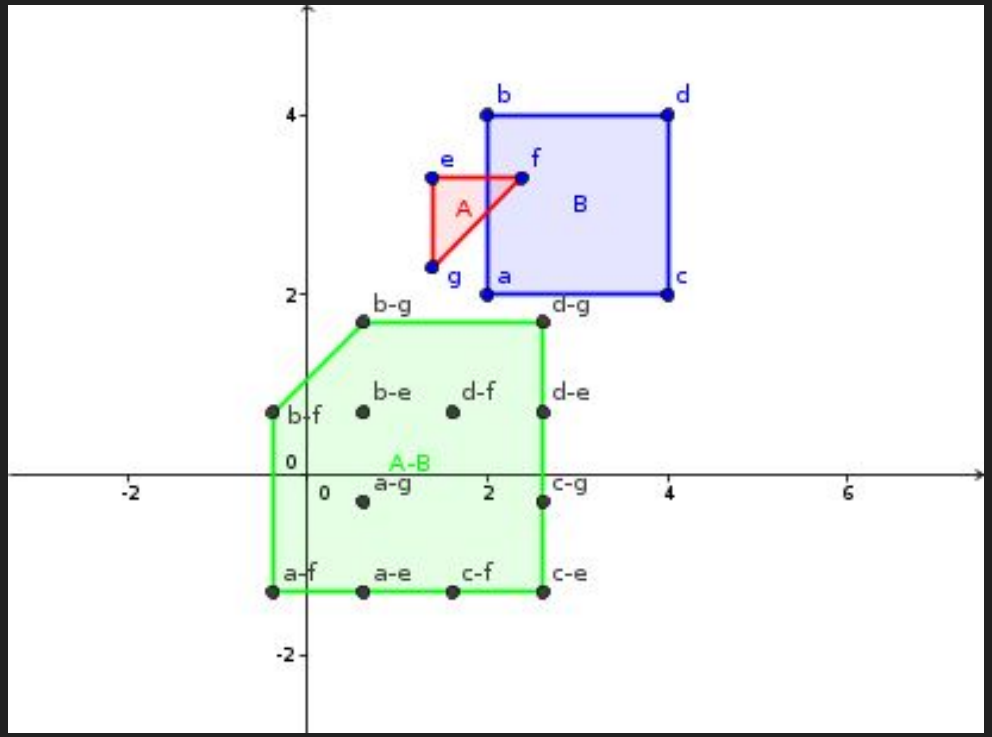
Soma de Minkowski

5. Tipos de Colisão



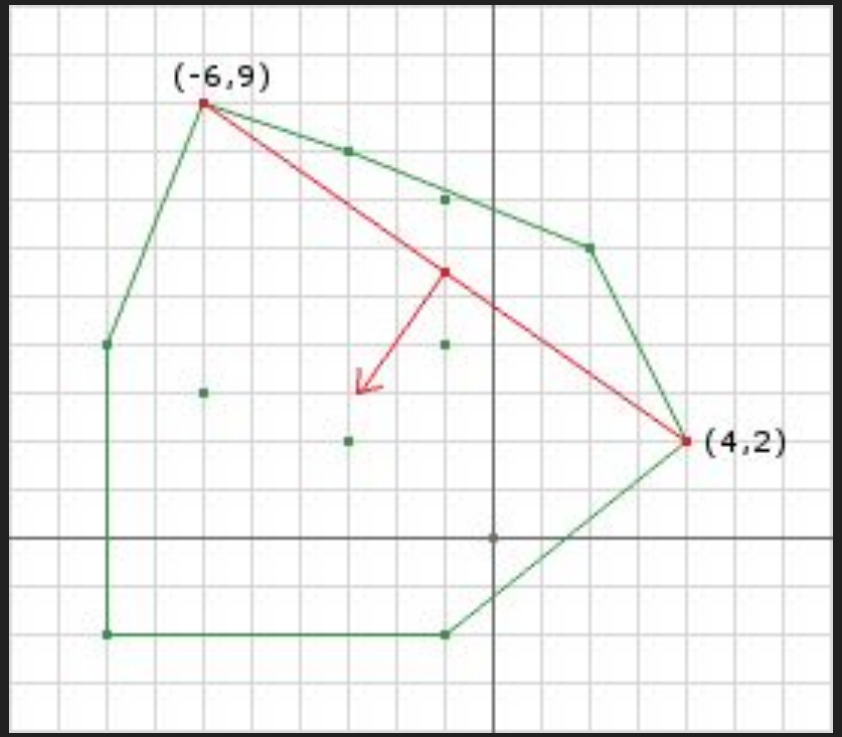
Subtração de Minkowski

5. Tipos de Colisão



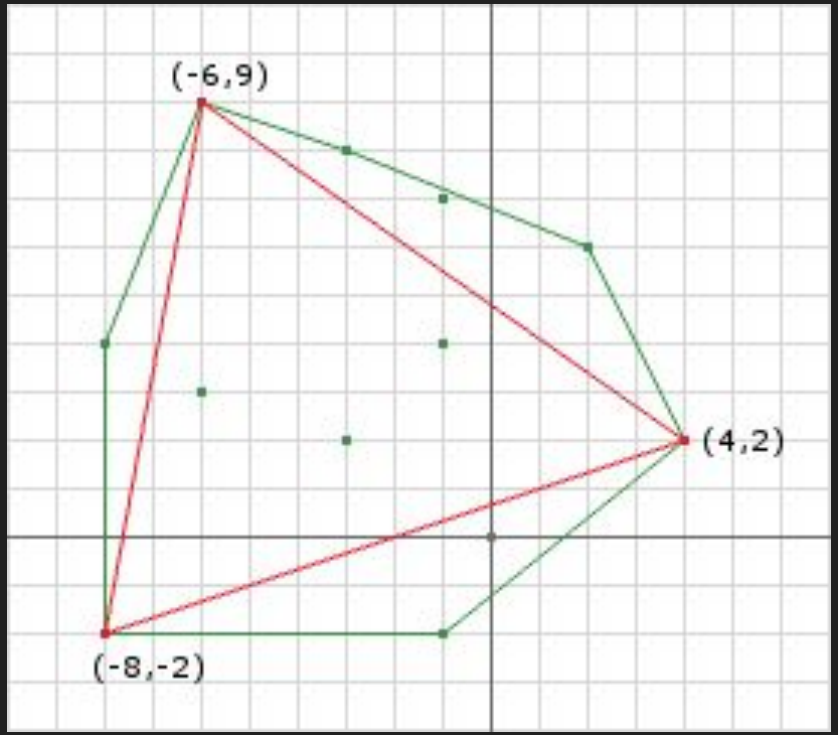
Subtração de Minkowski

5. Tipos de Colisão



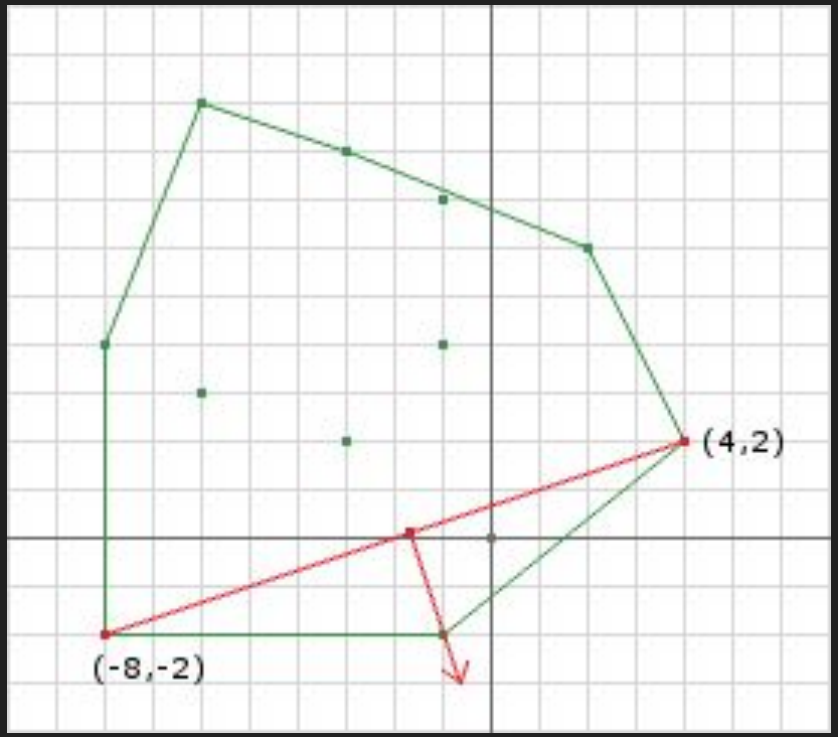
GJK - Gilbert–Johnson–Keerthi

5. Tipos de Colisão



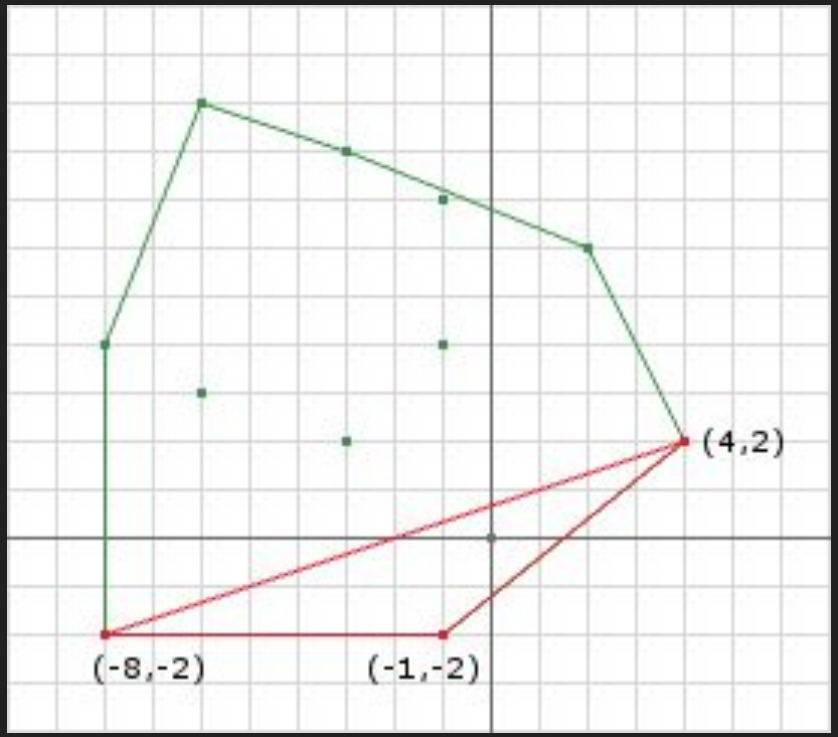
GJK - Gilbert–Johnson–Keerthi

5. Tipos de Colisão



GJK - Gilbert–Johnson–Keerthi

5. Tipos de Colisão



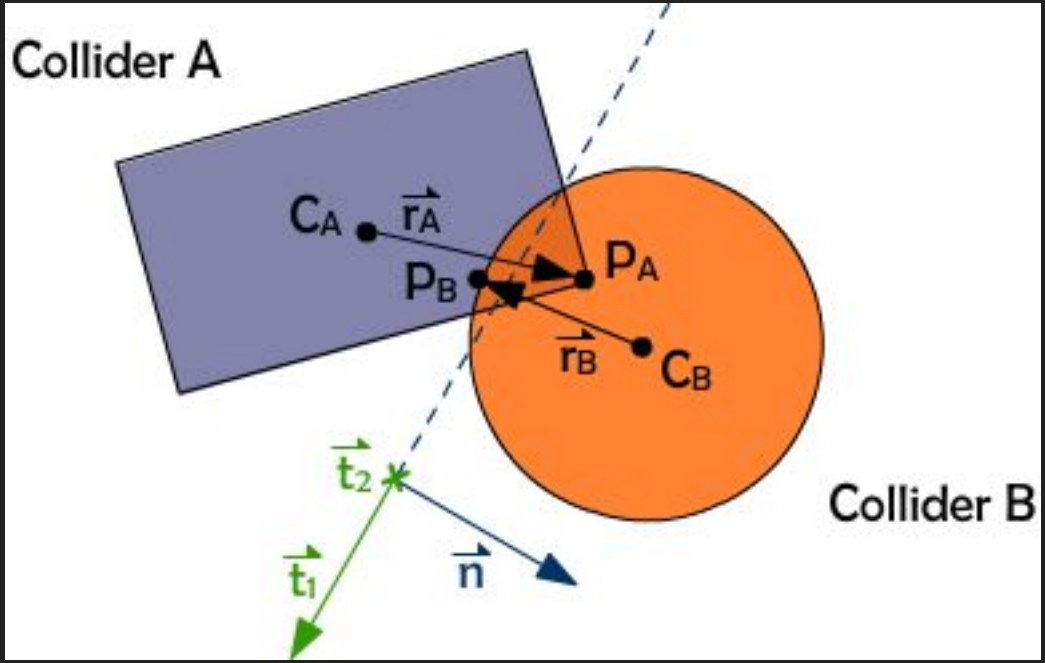
GJK - Gilbert–Johnson–Keerthi

6. Resposta

6. Resposta

- Precisamos coletar diversas informações da colisão para resolver as intersecções
  - Normal
  - Pontos de contato
  - Profundidade

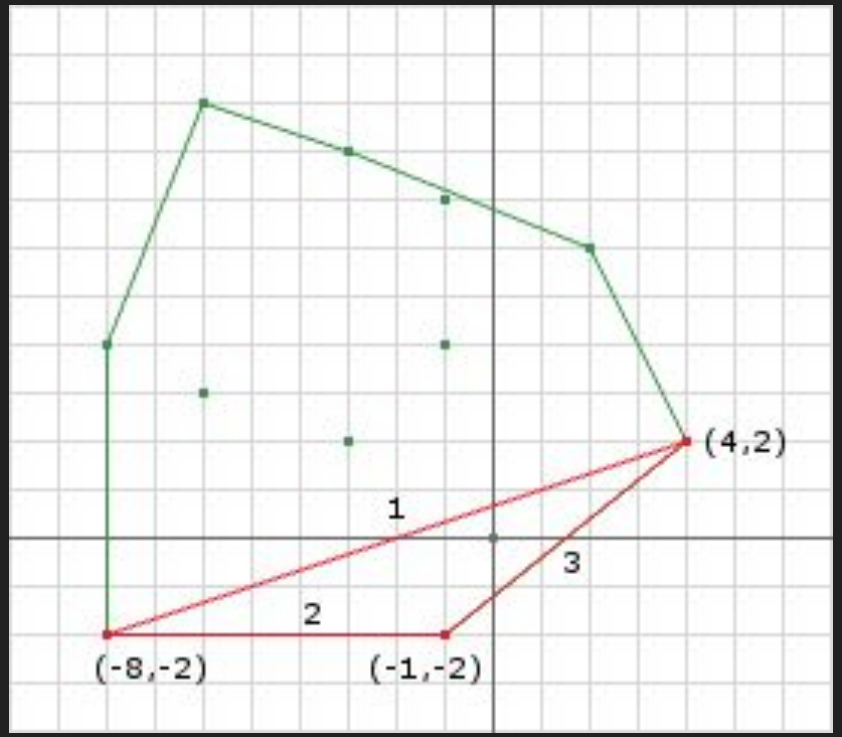
6. Resposta


Collider A
C_A
$\vec{r}_A$
P_B
P_A
$\vec{r}_B$
C_B
$\vec{t}_2$
Collider B
$\vec{t}_1$
$\vec{n}$

6. Resposta

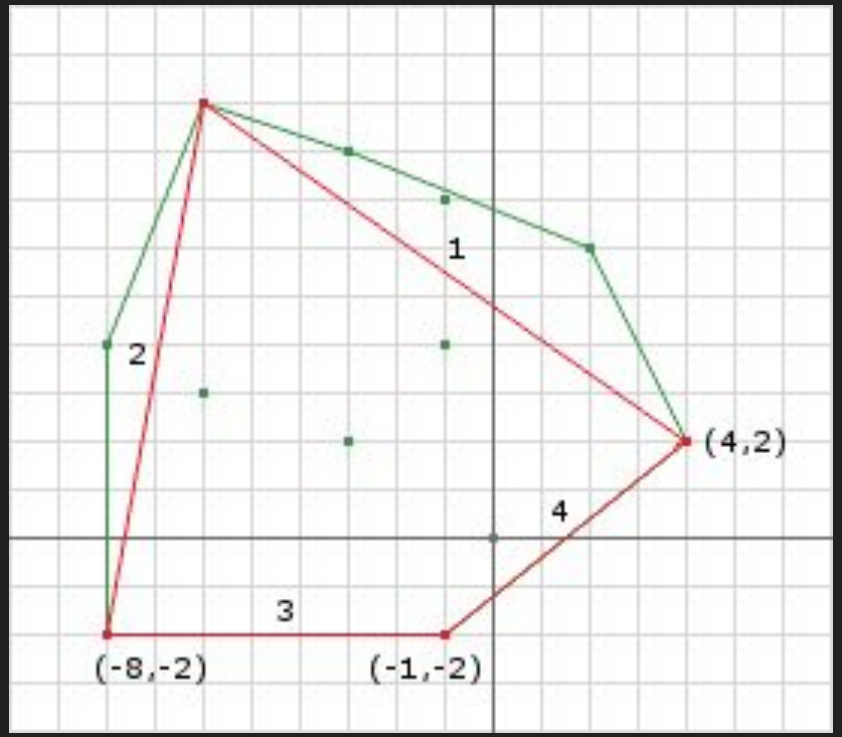
- EPA - Expanding Polytope Algorithm
  - Pega o simplex final do GJK e expande em direção às retas e planos mais próximos
  - A aresta ou plano final mais próximo do polígono diz
    - Profundidade da intersecção
    - Normal da intersecção

5. Tipos de Colisão



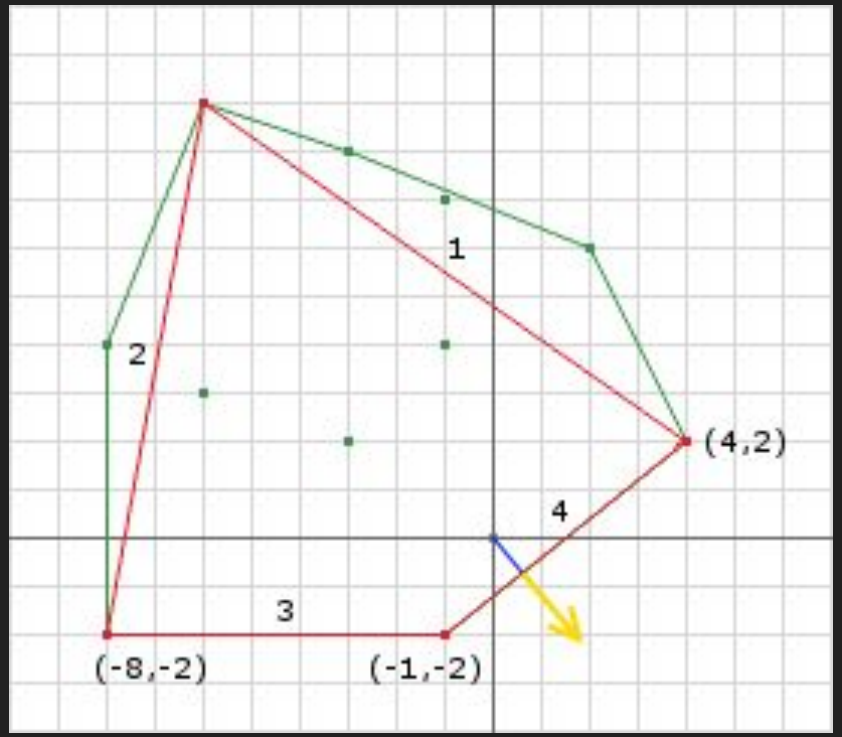
EPA - *Expanding Polytope Algorithm*

5. Tipos de Colisão



EPA - *Expanding Polytope Algorithm*

5. Tipos de Colisão



EPA - *Expanding Polytope Algorithm*

6. Resposta

- Sequential Impulse Solver é o algoritmo que resolve as intersecções dos objetos, ele calcula o quão o objeto deve se afastar, direção etc.
  - Resolve constraints (restrições) como limite de ângulo, modelos físicos (elasticidade, flutuação etc.)
  - Encontra o impulso, força de repulsão dos objetos e o novo vetor velocidade

6. Resposta

Exemplo

7. Colisão Contínua

7. Colisão Contínua

- ➔ Objetos muito rápidos podem quebrar a física, por exemplo projéteis
  - ◆ Detectando colisão de objetos rápidos
    - Múltiplas iterações (tempo entre o dt)
    - Estender o colisor para interceptar os objetos entre os tempos
  - ◆ Precisamos determinando tempo de impacto para resolver a física, muitas vezes desnecessário, por exemplo num projétil

Dúvidas?

Referências

Referências

[1]http://www.dyn4j.org/
[2]http://allenchou.net/game-physics-series/
[3]http://codeflow.org/entries/2010/aug/28/integration-by-example-euler-vs-verlet-vs-runge-kutta/
[4]https://benjaminhorn.io/code/pixel-accurate-collision-detection-with-javascript-and-canvas/
[5]www.jeffreythompson.org/collision-detection/table_of_contents.php
[6]https://www.toptal.com/game/video-game-physics-part-ii-collision-detection-for-solid-objects
[7]https://research.ncl.ac.uk/game/mastersdegree/gametechnologies/physics8collisionmanifolds/Tutorial%208%20-%20Collision%20Manifolds.pdf
[8]http://judis.me/wordpress/sequential-impulse-solver/
[9]http://box2d.org/
[10]http://thomasdiewald.com/blog/?p=1488
[11]https://en.wikipedia.org/wiki/Binary_space_partitioning
[12]http://www.gdcvault.com/play/1020583/Animation-Bootcamp-An-Indie-Approach
[13]https://www.toptal.com/game/video-game-physics-part-ii-collision-detection-for-solid-objects
[14] (The Morgan Kaufmann Series in Interactive 3-D Technology) Christer Ericson-Real-Time Collision Detection-Morgan Kaufmann (2005)