Detecção de Colisões entre Pares de Poliedros Rígidos Aplicada ao Projeto ASIMOV

por

Anderson Maciel

Conteúdo desta apresentação:

Parte I

- Introdução
  - Objetivos e Motivação
  - A Detecção de Colisões
- Boyse (1979) [Checagem de Intersecção
- Uchiki (1983) [Ocupação de Espaço]
- Moore (1988) [Cyrus-Beck]
- Lin (1992) [Regiões Voronoi].

Conteúdo desta apresentação: Parte II

- Garcia-Alonso (1994) [Four-Step]
  - A estrutura do método
  - Os quatro passos
- ASIMOV (o ambiente de simulação)
- Escolha e implementação do método
- Os resultados e as conclusões.

Parte I

Motivação e Objetivos

Objetos (virtuais/reais) não podem se interpenetrar

Solução visual ➡ Solução automática

Trabalhos acadêmicos e software comercial.

Investigar algoritmos de DC existentes

Escolher o mais adequado às estruturas do ASIMOV

Implementá-lo no simulador do ASIMOV.

Detecção de Colisões?

Penetração entre objetos ➡ ALTO CUSTO

Variações possíveis para o problema:

- Descobrir o momento da colisão
- Encontrar o ponto de colisão
- Obter o vetor de direção da colisão
- Responder a uma situação de colisão
- Detectar possível colisão antes que ela ocorra
- Poliedros ou superfícies, rígidos ou flexíveis
- Convexos ou não convexos
- Genérico → Muitos casos especiais.

Boyse (1979)

- Objetos sólidos e superfícies paramétricas
- Representados por poliedros
(Faces→Arestas→Vértices)
- Dois tipos de checagem de interferência:
  - Intersecção entre objetos em posições fixas
  - Colisões entre objetos que se movem ao longo de trajetórias definidas.

Estática

Relações de Interferência:

$$A \cap B = \varnothing$$

$$A \cap B = A \quad (A \subseteq B)$$

$$A \cap B = B \quad (B \subseteq A)$$

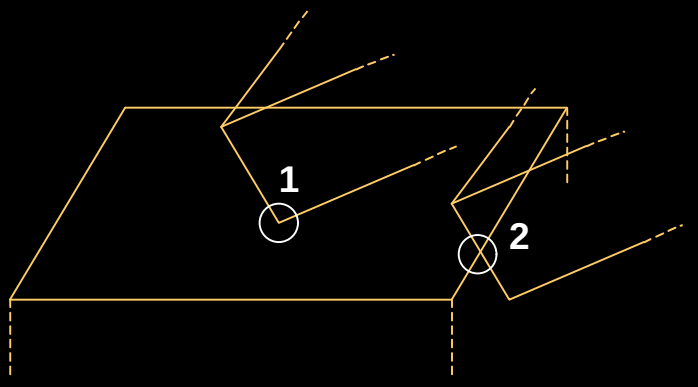
$$A \cap B = C \quad (C \neq \varnothing)$$

$$(C \neq A)$$

$$(C \neq B)$$

Dinâmica

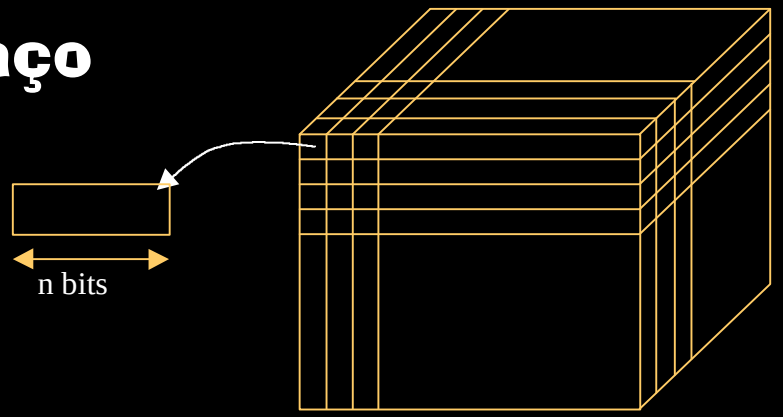
Posição inicial de não interferência



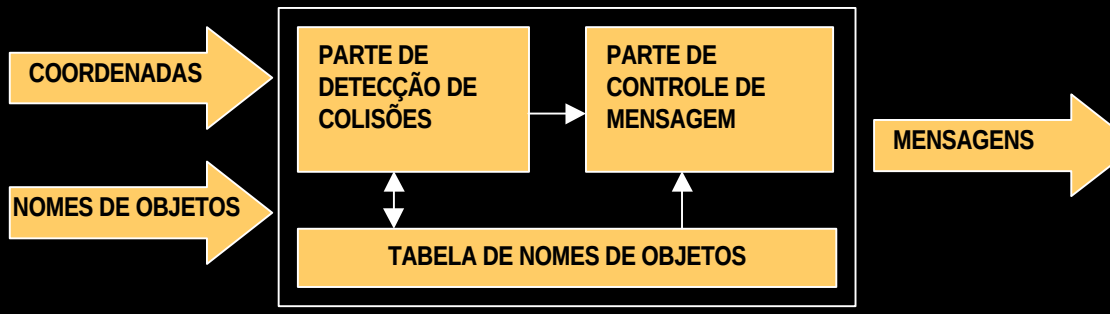
1. *Aresta contata interior da face*
2. *Aresta contata borda da face*

Uchiki (1983)

Ocupação do Espaço PEARY



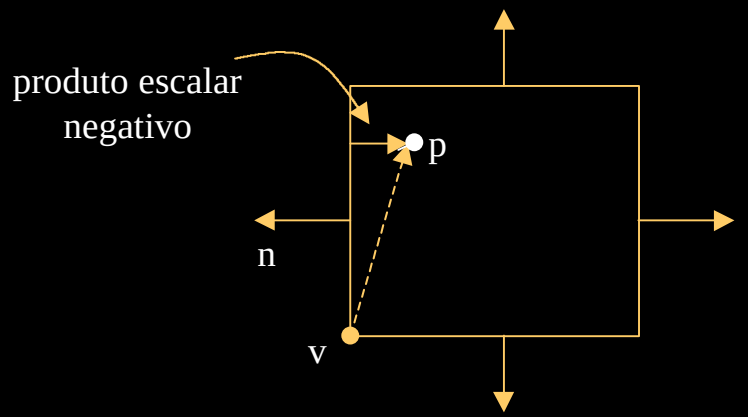
Detector de colisões



Resposta imediata X Restrição de memória.

Moore (1988)

Algoritmo de recorte de Cyrus-Beck



Representação (faces poligonais→arestas→vértices)

Testa pontos representativos de B contra A (B no SRA) e de A contra B (A no SRB).

Vértices

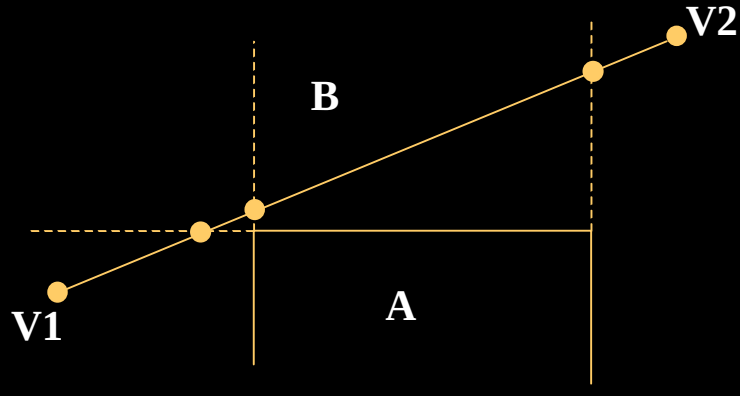
Comparar cada vértice de B com cada face de A

Se $(v_i - u_{j1}) \cdot n_j < 0$ *então* v_i *está dentro da face*

Se algum vértice está dentro de todas as faces de B ele está dentro de B =(colisão)

Arestas

Dividir a aresta em pequen segmentos

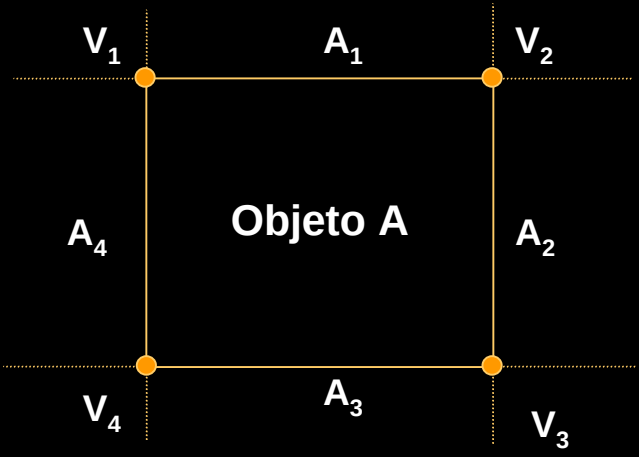


$$d_i = (v_i - u_{k1}) \cdot n_k \qquad t = \frac{|d_i|}{|d_i| + |d_j|}$$

$$d_j = (v_j - u_{k1}) \cdot n_k \qquad P = vi + t(v_j - v_i)$$

Testar o ponto médio de cada segmento contra as faces (análogo aos vértices

Lin (1992)



Algoritmo de distância

Regiões *Voronoi*

Teste de Aplicabilidade
(Verifica se um ponto está em um região Voronoi)

Tempo limite para colisão (fila)

Programação de horários

Pode trabalhar com objetos não-convexos

Executa em tempo constante (independente da complexidade dos objetos).

Parte II

Garcia-Alonso (1994)

Refinamento (cada passo é menor do que o anterior)

Quatro Passos

1. Matriz de **participação** em colisões
2. Interferência entre pares de **containers**
3. Interferência entre **voxels**
4. Interferência entre pares de **faces**

1 - Matriz de Participação

Gerada no início da simulação

Contém todos os pares de objetos e um flag

ID Obj #1	ID Obj #2	Flag
1	2	TRUE
2	3	FALSE
2	4	TRUE

Checa flag de cada par.

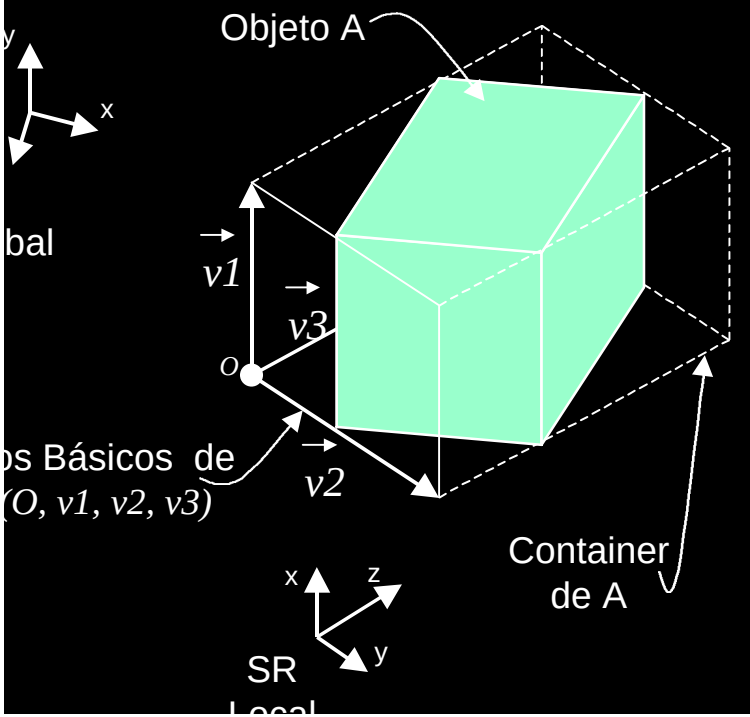
Executa (ou não) o teste entre containers

Principais Vantagens:

- Mesma Peça
- Usuário não deseja testar
- Peças conectadas

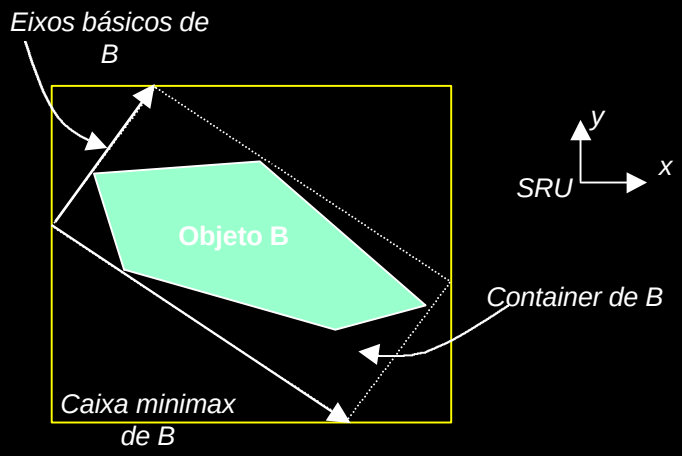
2 - Teste entre Containers

Calcular eixos básicos para todos os objetos



Mudar eixos básicos para o SRU

Encontrar caixa $minimax$



Aplicar teste $minimax$

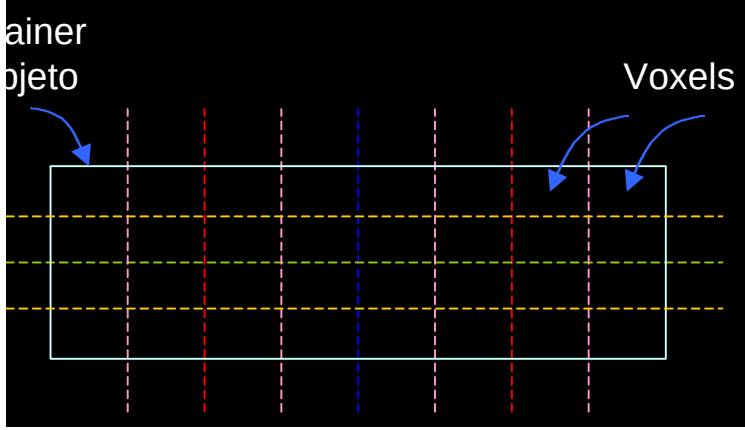
$$MIN(A) > MAX(B)$$

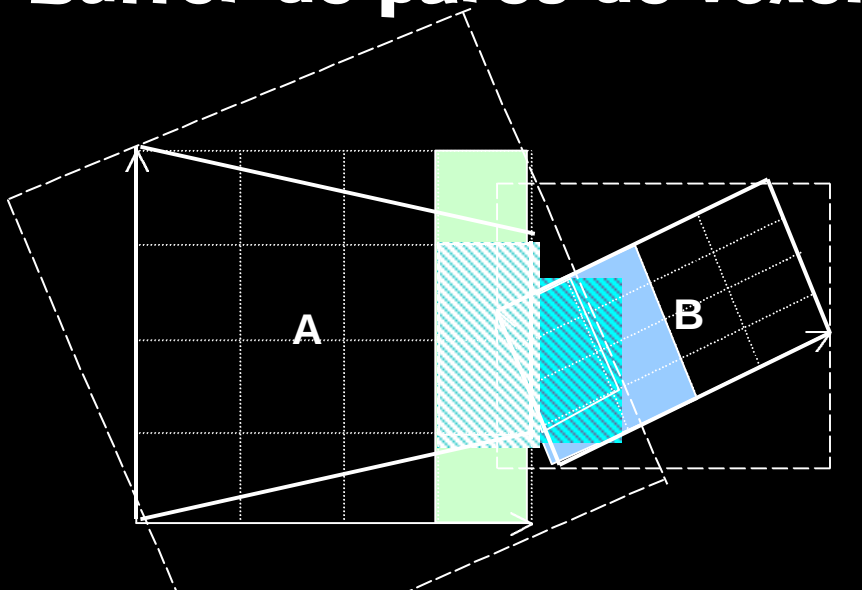
$$MAX(A) < MIN(B)$$

3 - Interferência entre Voxels

- Pequenas caixas idênticas
- Voxelização gerar matriz de voxel
- Ponteiro para lista de faces

- Gerar $T_{[A \to B]}$ e $T_{[B \to A]}$
- Calcular minimax de B em A e de A em B
- Matrizes de interferência
- Buffer de pares de voxel





4 - Intersecção de Faces

Elementos pré gerados pelos outros passos

- Descrição geométrica dos objetos
- Matriz de voxels
- Buffer de voxels

Gerar lista de pares de faces

Aplicar método para intersecção geométrica de faces

Se algum par intersecciona → Colisão ocorreu no ponto de intersecção.

Algoritmo de Intersecção
Aresta X Aresta

Método de Goldman

(interf. entre arestas definidas por 2 pts

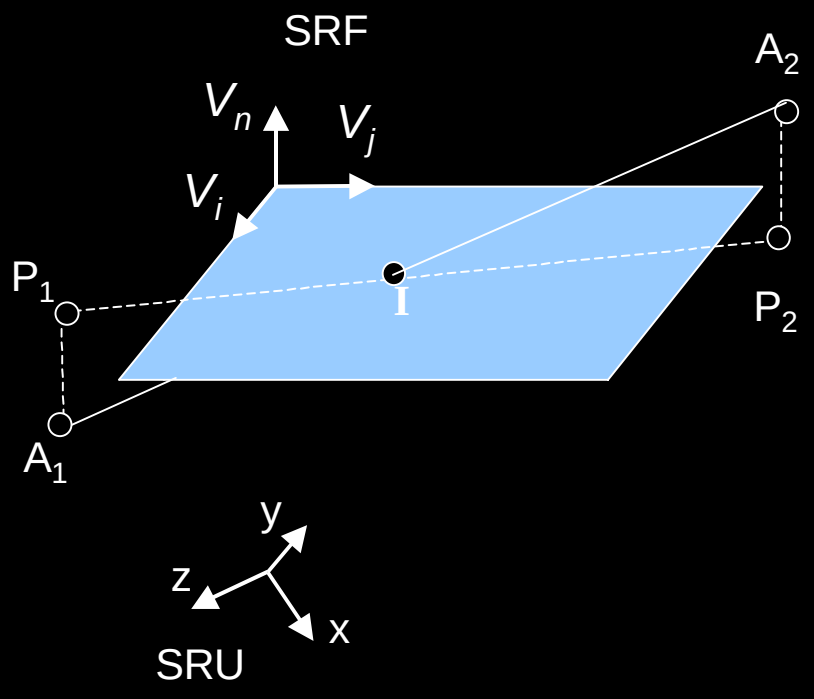
Problema #1 ➡ Voxel(6 faces) X Face

Problema #2 ➡ Face X Face

Aresta X Face ➡ Aresta X Aresta ??

➡ Projeção!!

Algoritmo de Goldman



Calcular distâncias T e S

$$T = \frac{\begin{vmatrix} V_{B1} - V_{A1} \\ V_{B2} - V_{B1} \\ V \times V' \end{vmatrix}}{(V \cdot V') \times (V \cdot V')} \qquad S = \frac{\begin{vmatrix} V_{B1} - V_{A1} \\ V_{A2} - V_{A1} \\ V \times V' \end{vmatrix}}{(V \cdot V') \times (V \cdot$$

Descobrir pontos mai próximos

Se ponto igual, então o ponto de intersecçã

Calcular Projeção

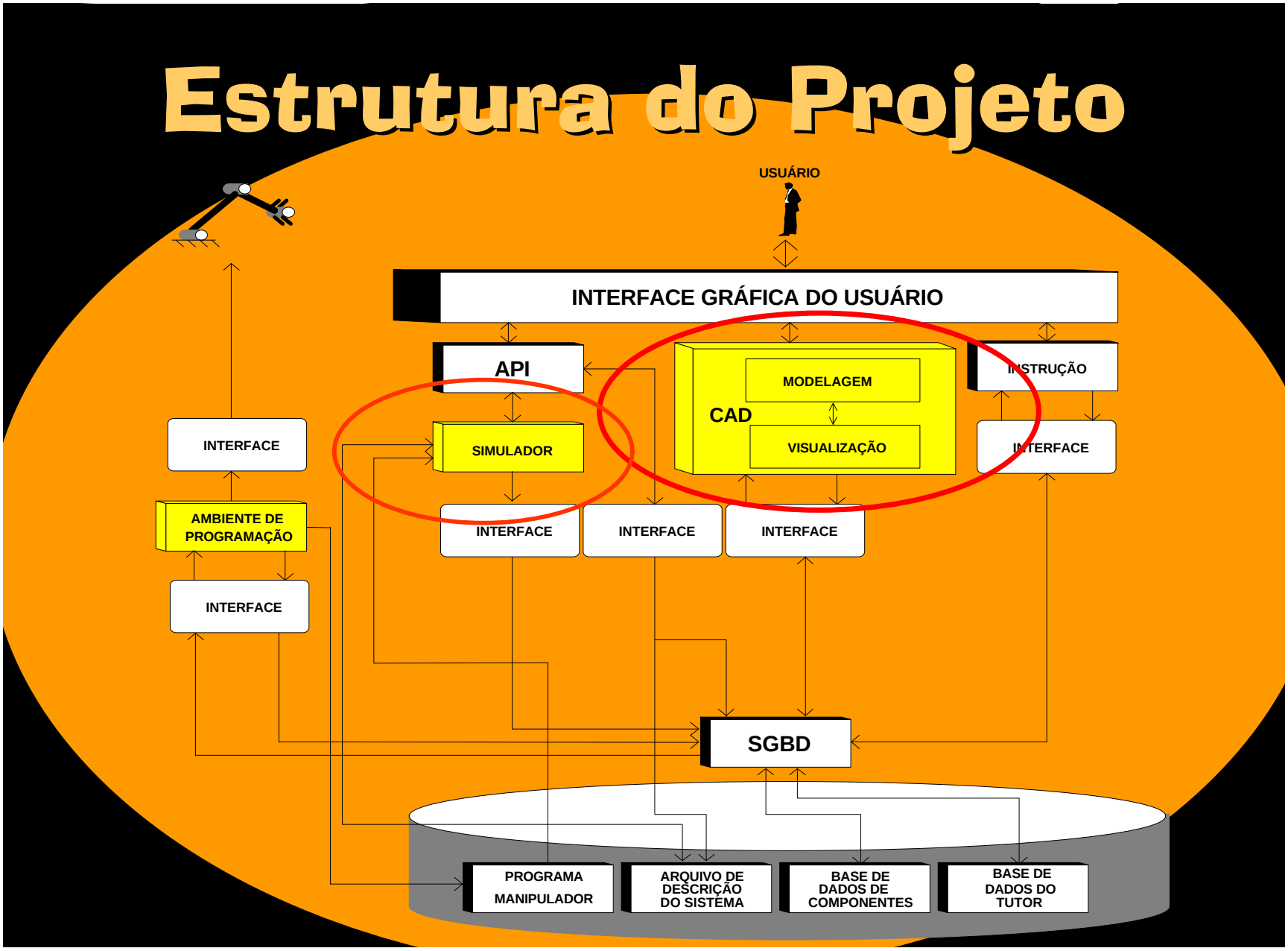
O Projeto ASIMOV

Hospeda a implementação do método escolhido

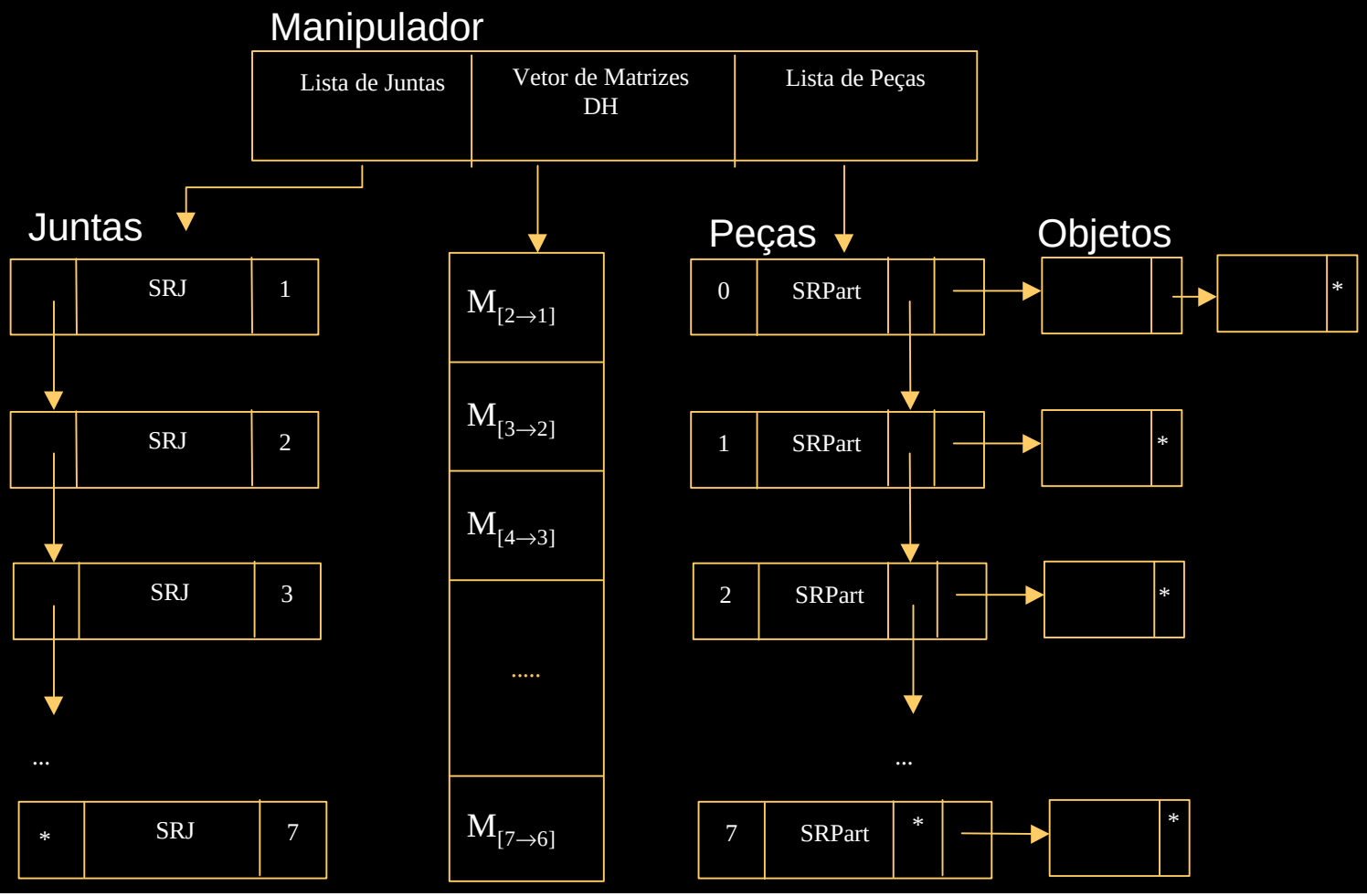
Ambiente educacional integrado para o projeto e simulação de Manipuladores Mecânicos

Qualidade e baixo custo

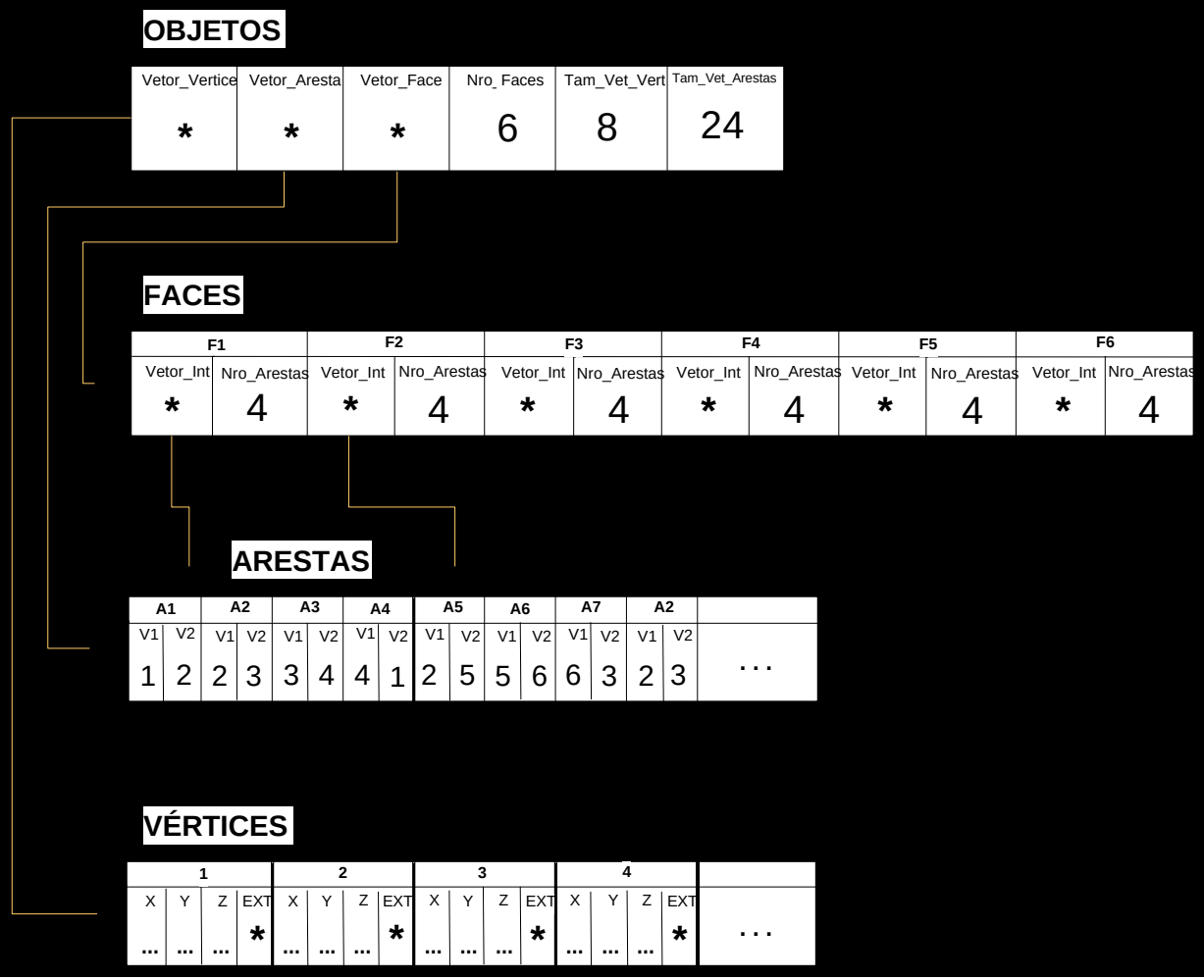
Estrutura do Projeto



Estrutura Hierárquica



Estrutura Geométrica



A Escolha do Método

Necessidades do ASIMOV

Estrutura de representação

Funcionamento do simulador do ASIMOV

Tempo Real.

Implementação

Ambiente Windows

Linguagem C++

Padrões e metodologia do ASIMOV

Elementos adicionados à estrutura.

Resultados

Software capaz de detectar colisões em tempo real no PC.

Uma imagem vale por mil palavras ...

(Execute o programa Simulacao.exe do CD Detecção de Colisões para ver um vídeo demonstrativo)

Conclusões

Tendência a otimizar tempo

Métodos se adaptam a aplicações específicas ➧ não existe um geral

Tempo-real ➧ aspectos pouco explorados

- Superfícies curvas
- Enormes quantidades de objetos
- Coerência espacial entre quadros
- Contato ≠ Interferência