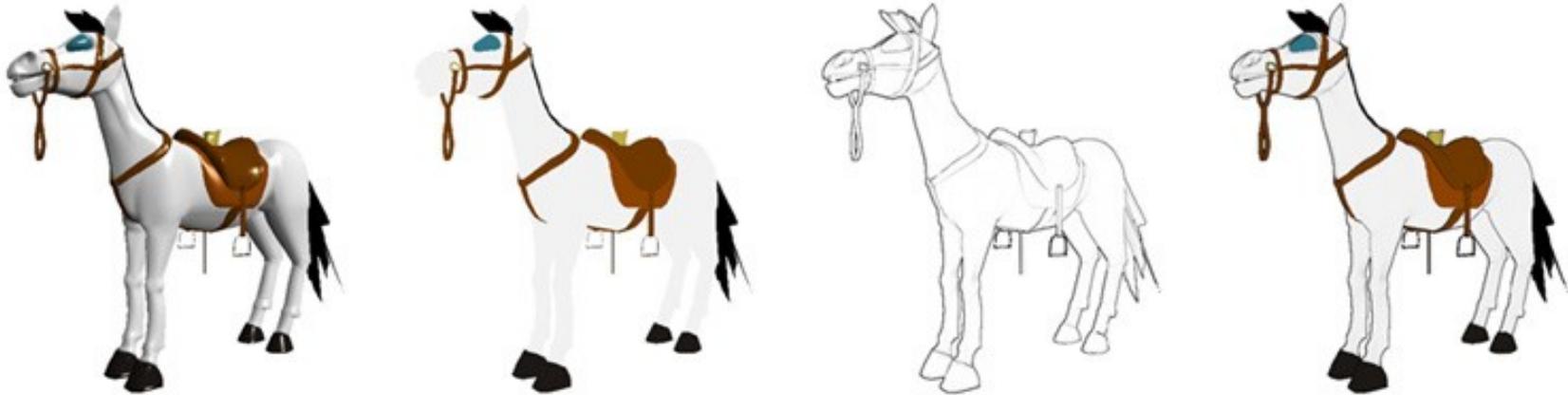


Renderização de cenas tridimensionais não foto-realistas explorando o hardware programável



- Bruno Pereira Evangelista
- Alessandro Ribeiro da Silva
- Marcelo de Souza Nery (Orientador)
- Rosilane Ribeiro Mota (Orientadora)



PUC Minas

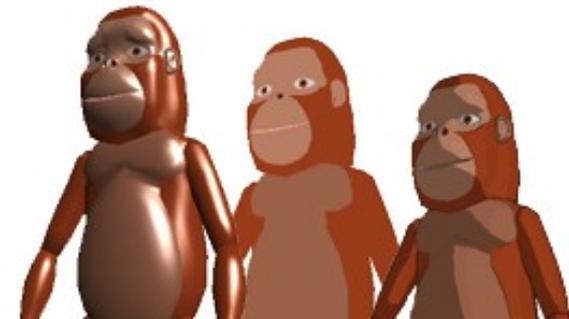
- **Estilo artístico**
- **Utilizado em várias aplicações**
 - **Ilustrações técnicas**
 - **Desenhos animados**
- **Possui várias sub-áreas**
 - **Cartoon**
 - **Hatching**

- **Objetivo**
 - **Gerar imagens que não possam ser diferenciadas de desenhos feitos a mão**
- **Uso limitado de tons**
- **Sombreamento uniforme**
- **Presença de bordas**
- **Traços bruscos**

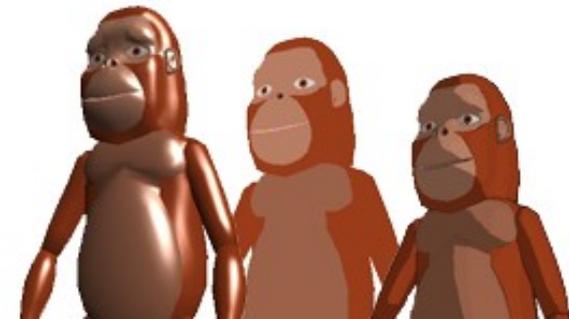


- **Gerar imagens com aparência de *cartoons* a partir de cenas tridimensionais**
 - **Imagens de alta qualidade**
 - **Geradas em tempo real**

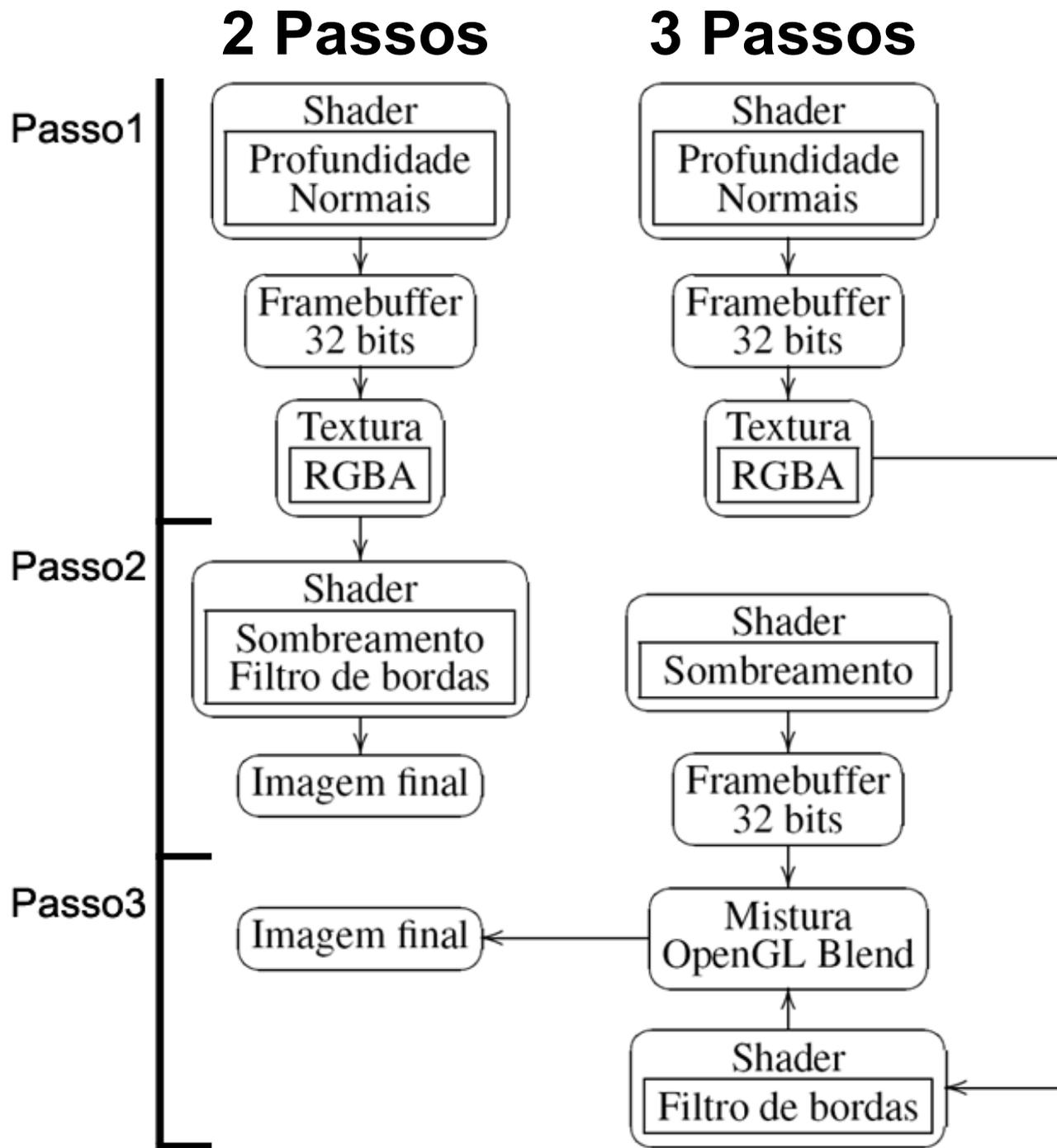
- **Proposição de 2 métodos para renderização de *cartoons***
 - **Método 2 passos**
 - **Método 3 passos**
- **Trabalho no domínio da imagem**
 - **Maior qualidade e precisão das bordas**
 - **Não é necessário nenhum pré-processamento**



- Exploração do hardware gráfico programável (*shaders*)
 - Linguagem de shaders do OpenGL (GLSL)

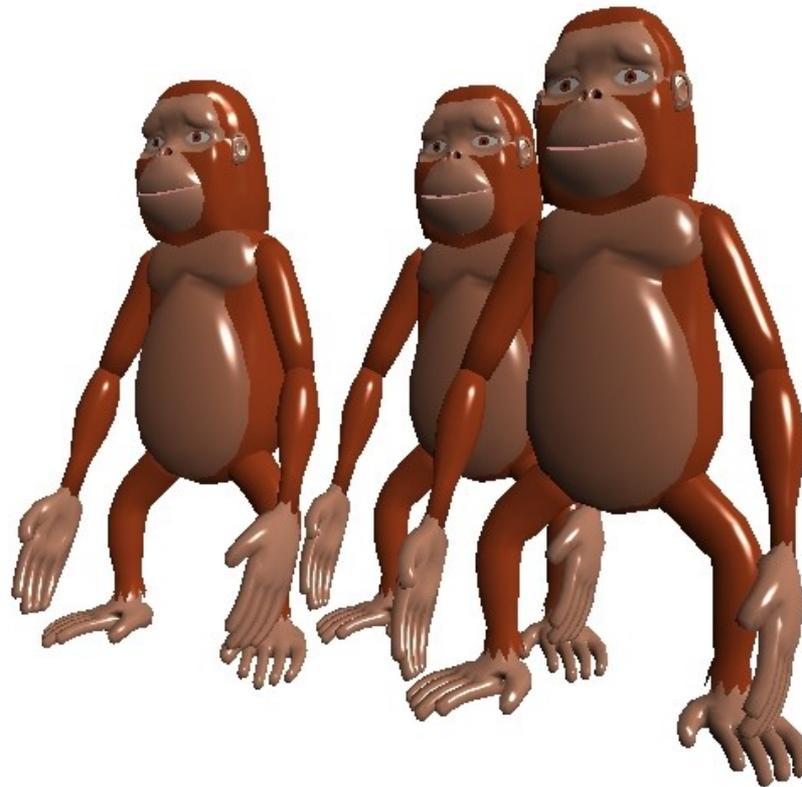


- 1 - Obtenção dos mapas de profundidade e normais da cena
- 2 - Renderização da cena utilizando sombreamento uniforme e totalização desejada
- 3 - Detecção de bordas a partir dos mapas de profundidade e normais da cena
- 4 - Geração da cena final



Macacos Groyl

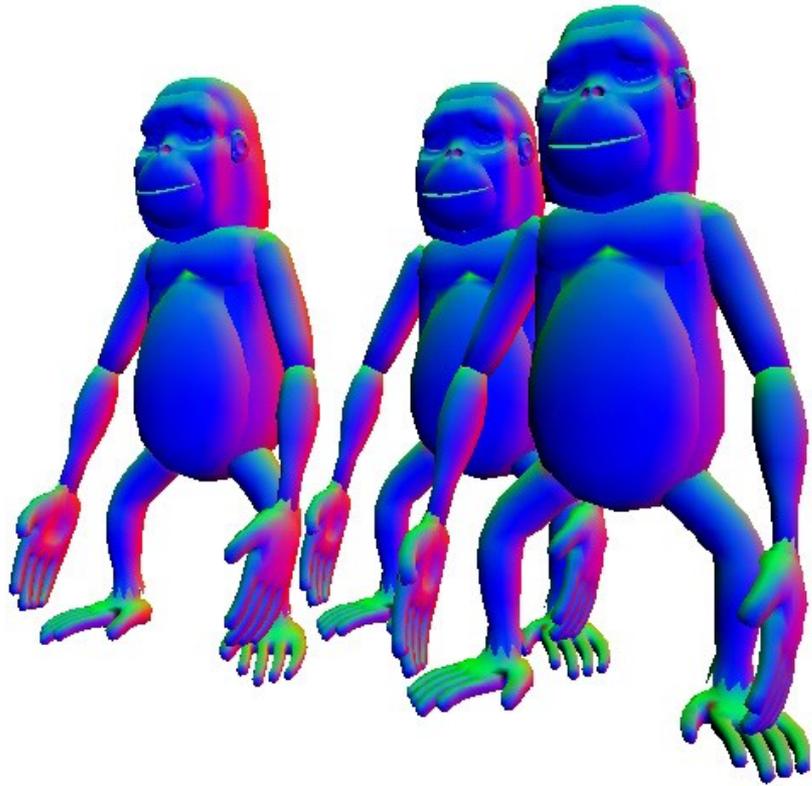
Cada macaco possui 6310 triângulos



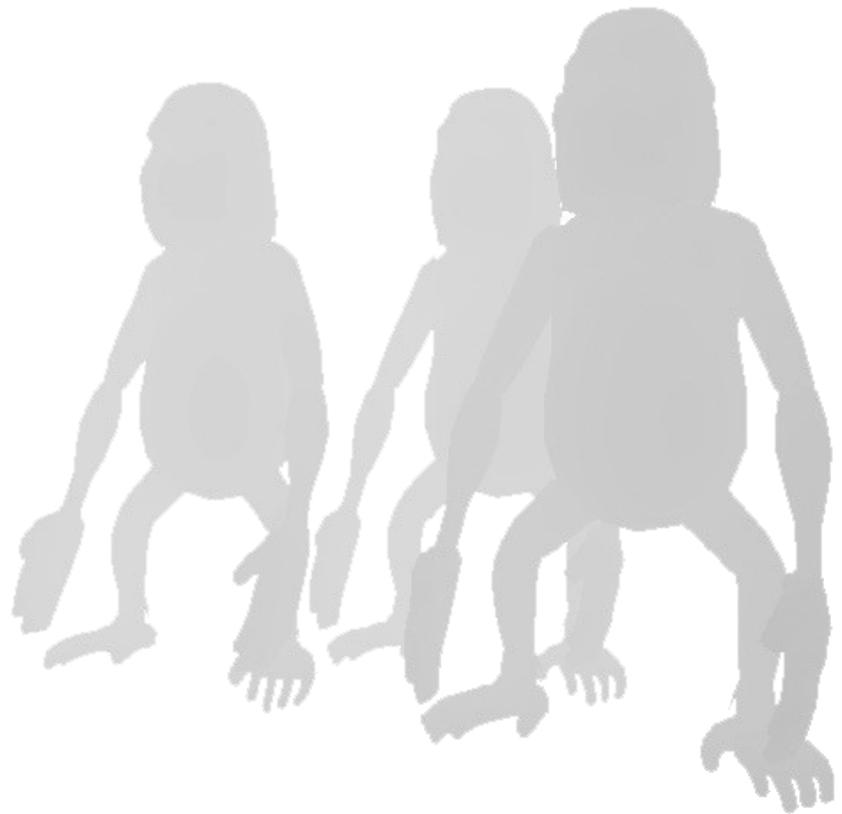
Renderização com sombreamento de Phong

- **Mapa de normais representado pelas componentes de cor RGB**
- **Mapa de profundidade representado pela componente de cor *alpha***
- **Cada componente possui a precisão de 8 bits**
- **Valores são quantizados e normalizados entre 0 e 1**

Normais

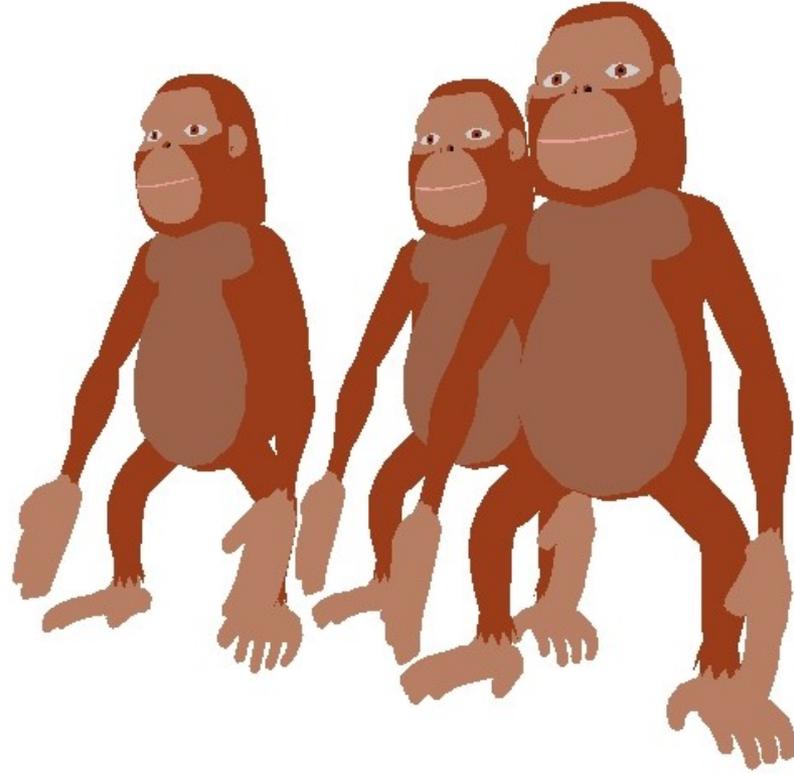


Profundidade



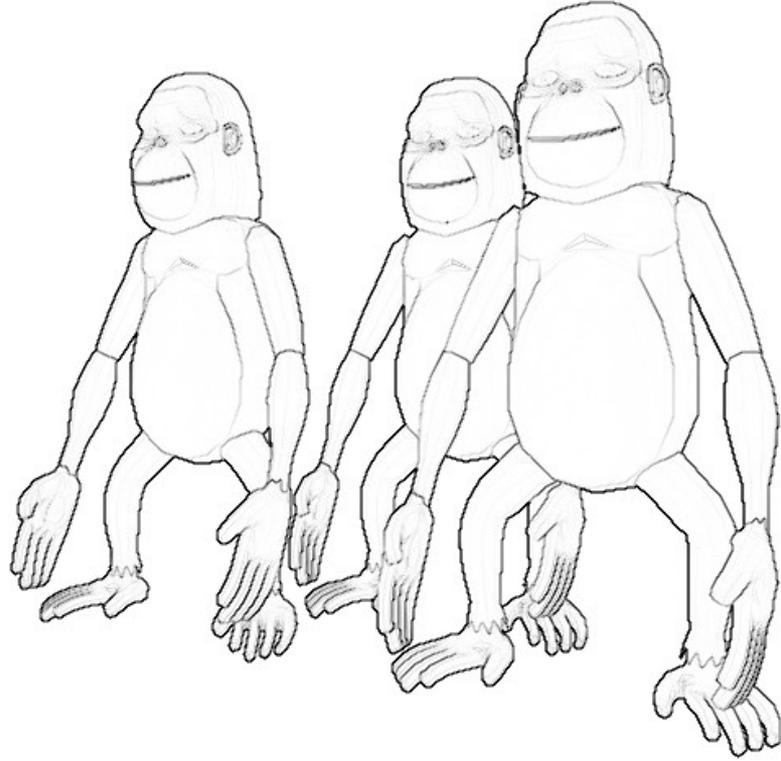
- **Sombreamento uniforme utilizando um número limitado de tons (1 a 3)**
 - **Componentes ambiente, difusa e especular da luz**
 - **Sombras próprias (self-shadows)**
- **Não existe suavização**

2 - Renderização



- **Utiliza a imagem gerada no passo 1**
- **Aplica um filtro de detecção de descontinuidade na imagem**
 - **Filtro Laplaciano com um número menor de acessos a memória de textura**

3 - Bordas



- **Mistura das imagens geradas nos itens 2 e 3, gerando a imagem final**
- **Uso do *pipeline* padrão do OpenGL para efetuar a mistura**

Imagem original

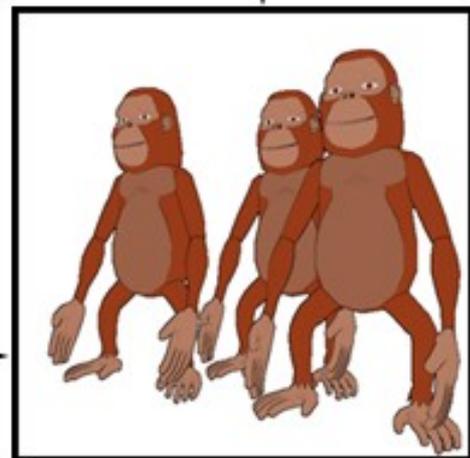
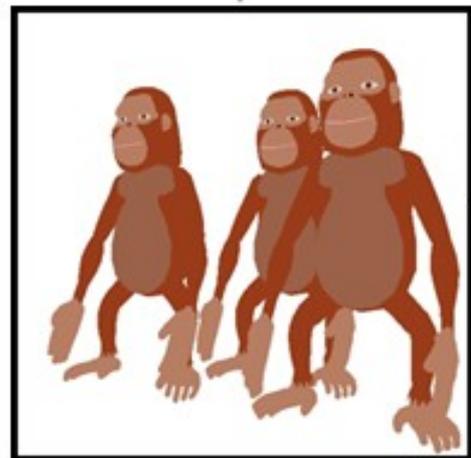
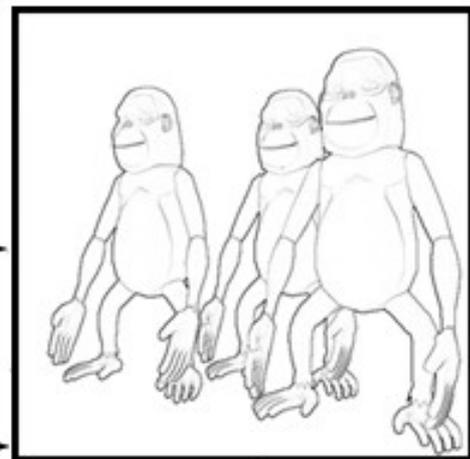
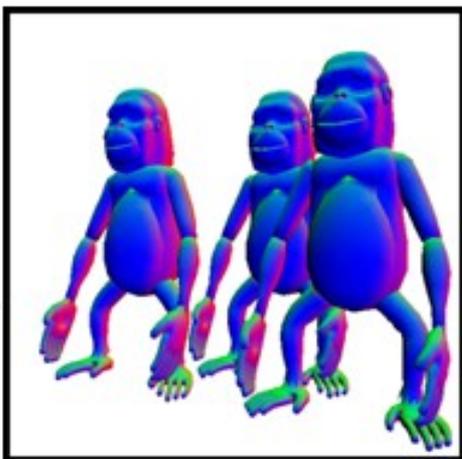
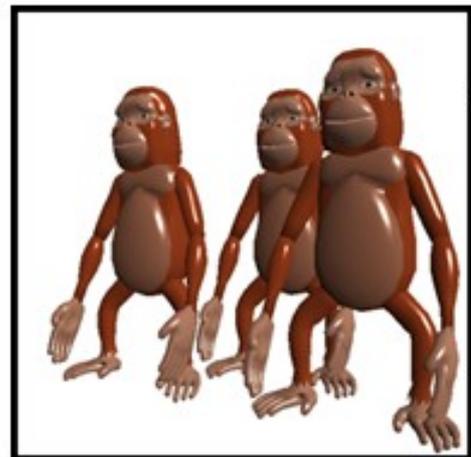


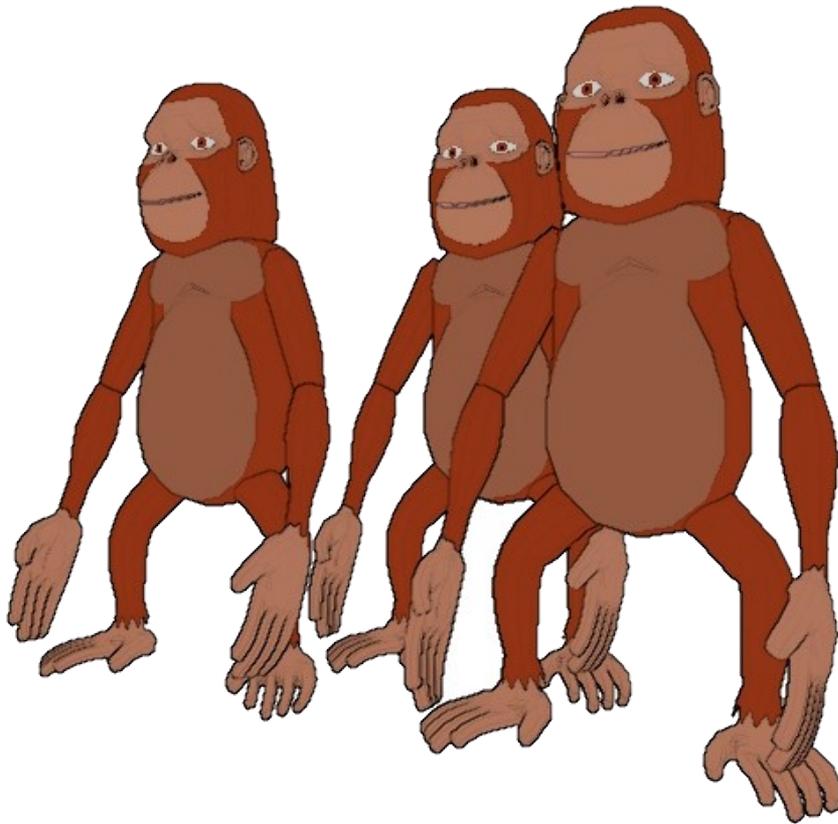
Imagem final

- **Uso de uma linguagem de *shader* de alto nível – GLSL (OpenGL Shading Language)**
 - **Evolução dos compiladores das placas de vídeo**
 - **Maiores possibilidades**
- **Menor número de passos**
- **Alto desempenho***
 - **Primeiros trabalhos demoravam cerca de 6 minutos para gerar cada imagem**

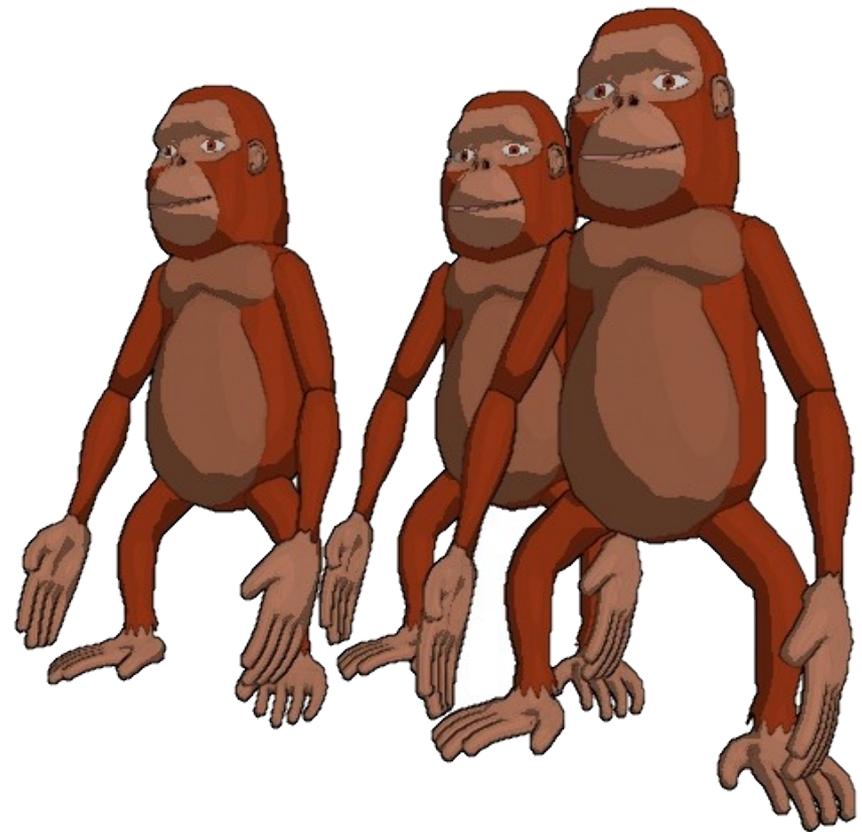
*** Não foi possível comparar com a técnica mais recente, proposta por Mitchell**

Renderização de *cartoons* com bordas utilizando um número variado de tons

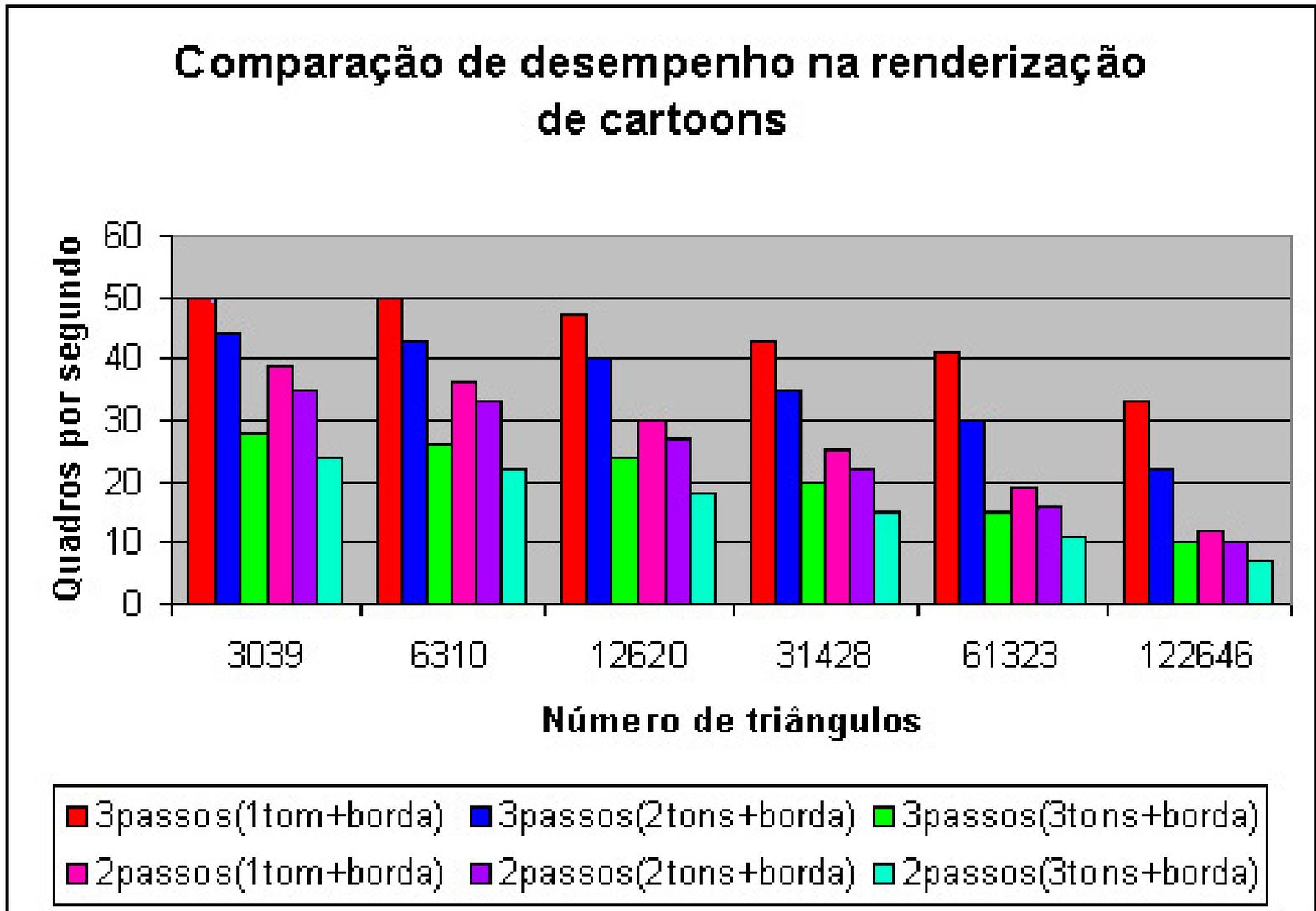
Resultado



Um tom



Três tons



Processador 1GHz – 512 RAM
GeForce FX 5200

- **Método de 3 passos sempre obteve um desempenho superior ao de 2 passos**
- **Validação utilizando o NVShaderPerf**
 - **Suporte somente ao shader de fragmentos**
 - **Considera tempos iguais para todas as instruções**

- **Falta de informações para comparação de desempenho**
- **Poucas ferramentas disponíveis**
- **Versões dos *drivers* de vídeo**

- **Todas as cenas foram renderizadas na resolução de 512x512 pixels, utilizando modelos livres da 3D Café(www.3dcafe.com)**
- **Utilizando as técnicas propostas é possível desenhar cartoons em tempo real, com uma alta qualidade gráfica**
- **A renderização de cartoons vêm sendo utilizada em várias aplicações, incluindo jogos**

• Perguntas ???

Bruno Pereira Evangelista
bpevangelista@yahoo.com.br

Alessandro Ribeiro da Silva
spdoido@yahoo.com.br

Marcelo de Souza Nery
msnery@pucminas.br

Rosilane Ribeiro Mota
rosilane@pucminas.br

