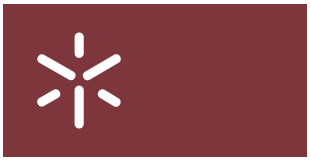

Computação Gráfica - TP4

Normais e Coordenadas de Textura

11.06.2025
André Carvalho a100818,
Flávio Sousa a100715,
Vicente de Carvalho Castro a91677,

Índice

1. Introdução

Nesta fase do projeto de Computação Gráfica, foram implementadas funcionalidades essenciais para aumentar o realismo visual das cenas geradas pela aplicação. O foco principal esteve na geração de normais e coordenadas de textura para cada vértice dos modelos geométricos, elementos fundamentais para a correta aplicação de iluminação e texturas.

Com estas adições, o motor 3D passa a suportar iluminação e texturização, interpretando e aplicando as informações de normais e coordenadas de textura contidas nos ficheiros de modelo. Além disso, foi realizada a extensão do ficheiro XML de configuração para permitir a definição dos componentes de material — incluindo as cores difusa, especular, emissiva e ambiente, bem como o fator de brilho (shininess).

Outra componente crítica abordada nesta fase foi a introdução de fontes de luz, com suporte para diferentes tipos, como luzes direcionais e posicionais, cada uma exigindo diferentes parâmetros no XML.

Para demonstrar a aplicação prática de todas estas funcionalidades, foi desenvolvida uma cena de demonstração: um sistema solar animado com aplicação de texturas e iluminação realista, destacando a integração harmoniosa entre geometria, materiais e luzes no ambiente 3D.

2. Implementação

2.1. Engine

A transição para a Fase 4 representou um salto qualitativo no desenvolvimento do motor gráfico, introduzindo funcionalidades avançadas que permitiram criar cenas tridimensionais mais realistas e eficientes. Esta fase trouxe consigo um conjunto significativo de melhorias que transformaram profundamente a arquitetura e capacidades do sistema.

Um dos avanços mais notáveis foi a reformulação completa da estrutura de dados para representação de modelos. Enquanto nas fases anteriores os vértices eram armazenados de forma simplificada, na Fase 4 foi implementada uma estrutura Vertex mais completa, incorporando não apenas as coordenadas espaciais (x, y, z), mas também normais (nx, ny, nz) e coordenadas de textura (u, v). Esta evolução permitiu calcular efeitos de iluminação mais realistas e implementar mapeamento UV para texturas, algo impossível nas versões anteriores.

O sistema de iluminação foi completamente redesenhado, passando a suportar múltiplas fontes de luz configuráveis, tanto pontuais como direcionais. Cada luz pode agora ser definida com parâmetros específicos de cor ambiente, difusa e especular, criando interações luminosas mais ricas e realistas com os objetos da

cena. A implementação aproveita as normais dos vértices para calcular corretamente os efeitos de iluminação, seguindo o modelo de shading de Phong.

A introdução do suporte a texturas através da biblioteca STB Image marcou outro marco importante. O motor gráfico passou a poder carregar texturas a partir de ficheiros de imagem comuns (como PNG e JPEG), aplicando-as aos modelos através do mapeamento UV. Foram implementados mecanismos de filtragem avançada, incluindo mipmapping para melhorar a qualidade visual em diferentes distâncias, e controle sobre o modo de repetição das texturas.

Do ponto de vista de desempenho, a adoção de Vertex Buffer Objects (VBOs) trouxe ganhos significativos. Ao transferir os dados geométricos para a memória da GPU de forma persistente, reduziu-se drasticamente a sobrecarga causada pelo envio contínuo de vértices da CPU para a GPU em cada frame. Esta otimização, combinada com o uso de arrays de vértices em vez do modo de renderização imediata (glBegin/glEnd), permitiu alcançar taxas de renderização muito mais elevadas.

A hierarquia de cena foi reforçada com um sistema de grupos mais robusto e flexível, capaz de lidar com transformações aninhadas de forma eficiente. Através do uso combinado de glPushMatrix e glPopMatrix, o motor gráfico pode agora aplicar transformações hierárquicas (rotações, translações e escalas) mantendo a relação espacial entre objetos de forma consistente.

O parser de XML foi ampliado para suportar a leitura de todos estes novos elementos - materiais, texturas, luzes e hierarquias complexas - mantendo uma estrutura limpa e organizada. Esta melhoria permitiu que cenas mais complexas pudessem ser descritas e carregadas de forma intuitiva através de ficheiros de configuração.

A câmera virtual ganhou maior flexibilidade, com parâmetros de projeção em perspectiva (campo de visão, planos near e far) totalmente configuráveis. O posicionamento e orientação da câmera podem ser ajustados com precisão através da função gluLookAt, permitindo composições visuais mais elaboradas.

Em síntese, a Fase 4 consolidou o motor gráfico como uma plataforma capaz de renderizar cenas 3D complexas com níveis satisfatórios de realismo visual e desempenho. As estruturas implementadas nesta fase criaram as fundações sólidas necessárias para futuras expansões, como a implementação de técnicas mais avançadas de iluminação, sombras ou efeitos pós-processamento. O salto qualitativo em relação às fases anteriores é evidente tanto nas capacidades técnicas quanto na qualidade visual alcançável.

3. Generate

Na fase atual do projeto, implementamos um sistema automatizado para geração de ficheiros 3D que supera as limitações das abordagens anteriores. O nosso

gerador de modelos produz agora ficheiros .3d completos, contendo não apenas as coordenadas geométricas dos vértices, mas também as normais e coordenadas de textura necessárias para uma renderização avançada.

O processo de geração inicia-se com a definição matemática dos objetos primitivos. Para cada forma geométrica - sejam esferas, cones, caixas ou outros sólidos - calculamos precisamente a posição de cada vértice no espaço 3D. No caso de uma esfera, por exemplo, utilizamos coordenadas esféricas convertidas para cartesianas, garantindo uma distribuição uniforme dos vértices.

As normais são calculadas de forma diferenciada conforme a geometria do objeto. Para superfícies curvas como esferas, a normal em cada vértice corresponde exatamente ao vetor que vai do centro do objeto até esse vértice, normalizado. Já para superfícies planas como as faces de um cubo, todas os vértices de uma mesma face compartilham a mesma normal, perpendicular ao plano da face. Este cálculo preciso das normais é fundamental para o correto funcionamento do modelo de iluminação implementado.

O mapeamento de texturas é resolvido através da atribuição de coordenadas UV adequadas a cada vértice. No caso de objetos como o plano ou a caixa, utilizamos um mapeamento proporcional simples. Para formas mais complexas como a esfera ou o cone, implementamos mapeamentos específicos que minimizam distorções - no caso da esfera, por exemplo, utilizamos uma projeção equiretangular que mapeia a textura como um mapa-múndi.

O formato do ficheiro .3d gerado foi cuidadosamente estruturado para ser tanto eficiente quanto legível. Cada linha representa um vértice completo, com os valores separados por vírgulas na seguinte ordem: coordenadas x,y,z, componentes nx,ny,nz da normal, e finalmente as coordenadas u,v de textura. Esta organização permite uma leitura direta pelo motor de renderização sem necessidade de conversões adicionais.

A implementação atual inclui geradores para diversos primitivos geométricos:
- Planos (úteis para terrenos ou paredes)
- Caixas (com dimensões personalizáveis)
- Esferas (com controlo do número de paralelos e meridianos)
- Cones (com altura e raio ajustáveis)
- Cilindros (com tampa superior e inferior)

Cada um destes geradores produz automaticamente a triangulação adequada da superfície, garantindo que todas as faces sejam renderizadas corretamente. A triangulação é particularmente importante para objetos curvos, onde o número de divisões afeta diretamente a suavidade da superfície renderizada.

Este sistema de geração automatizada representa um avanço significativo em relação às fases anteriores do projeto. Não apenas simplifica o processo de criação de modelos 3D, como também garante a correta parametrização das normais

e coordenadas de textura - elementos essenciais para a qualidade visual final. Todos os ficheiros gerados são imediatamente compatíveis com o nosso motor de renderização, pronto para serem texturizados e iluminados de forma realista.

4. Testes

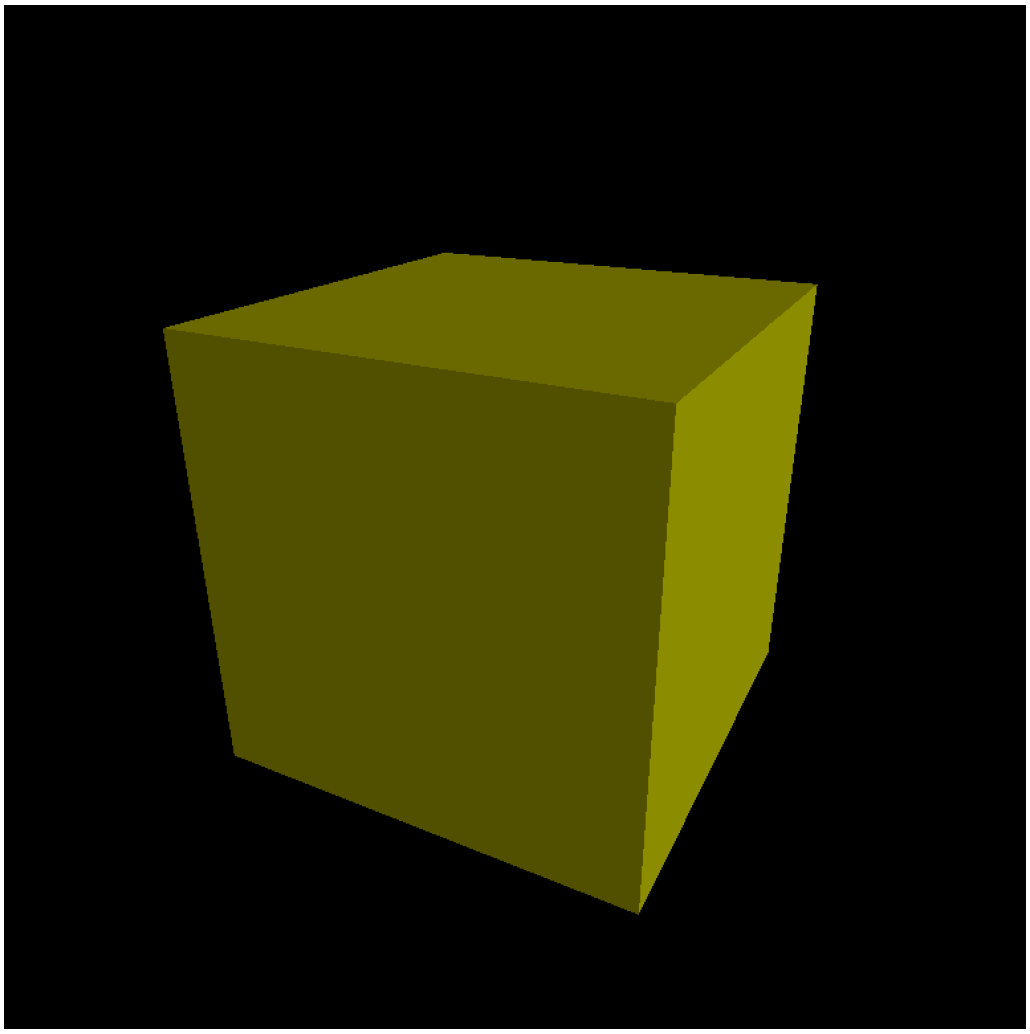

Figura 1: Figura 1: Test_4_1

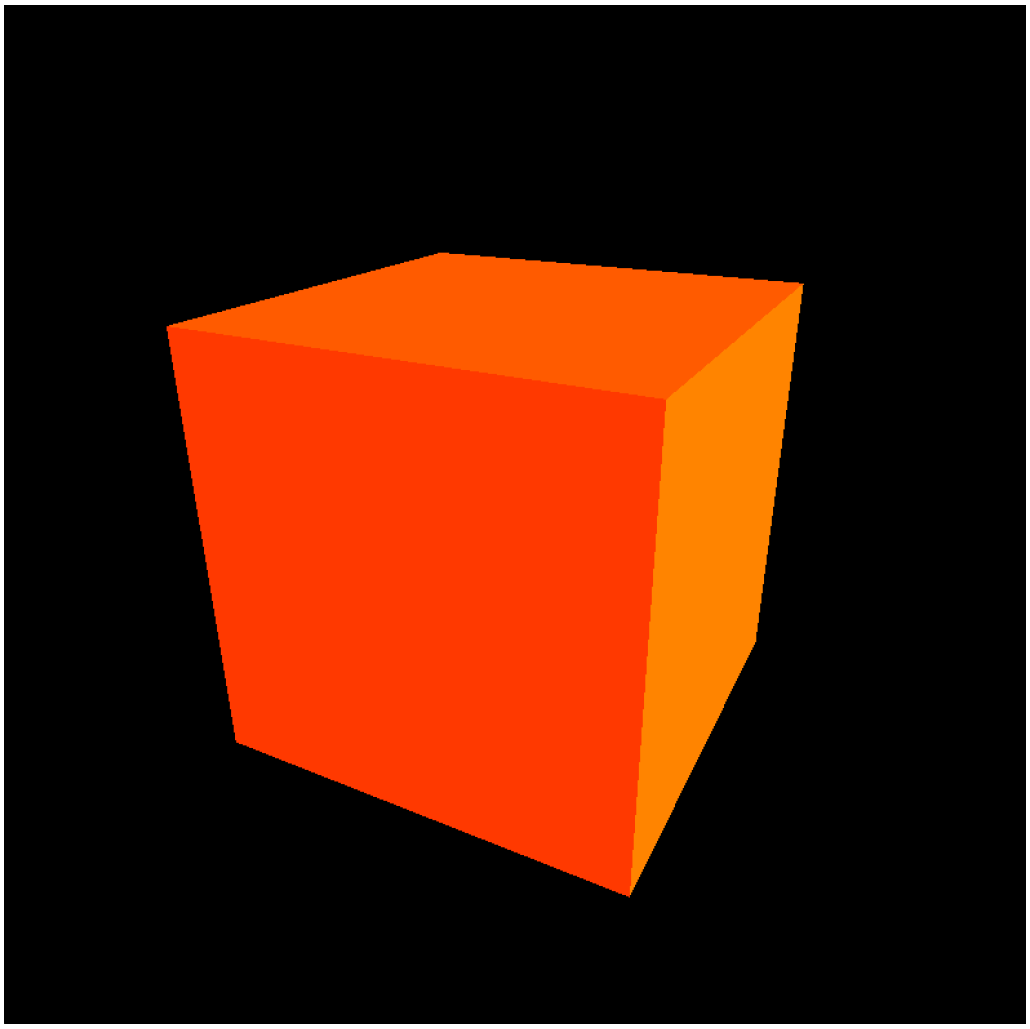

Figura 2: Figura 1: Test_4_2

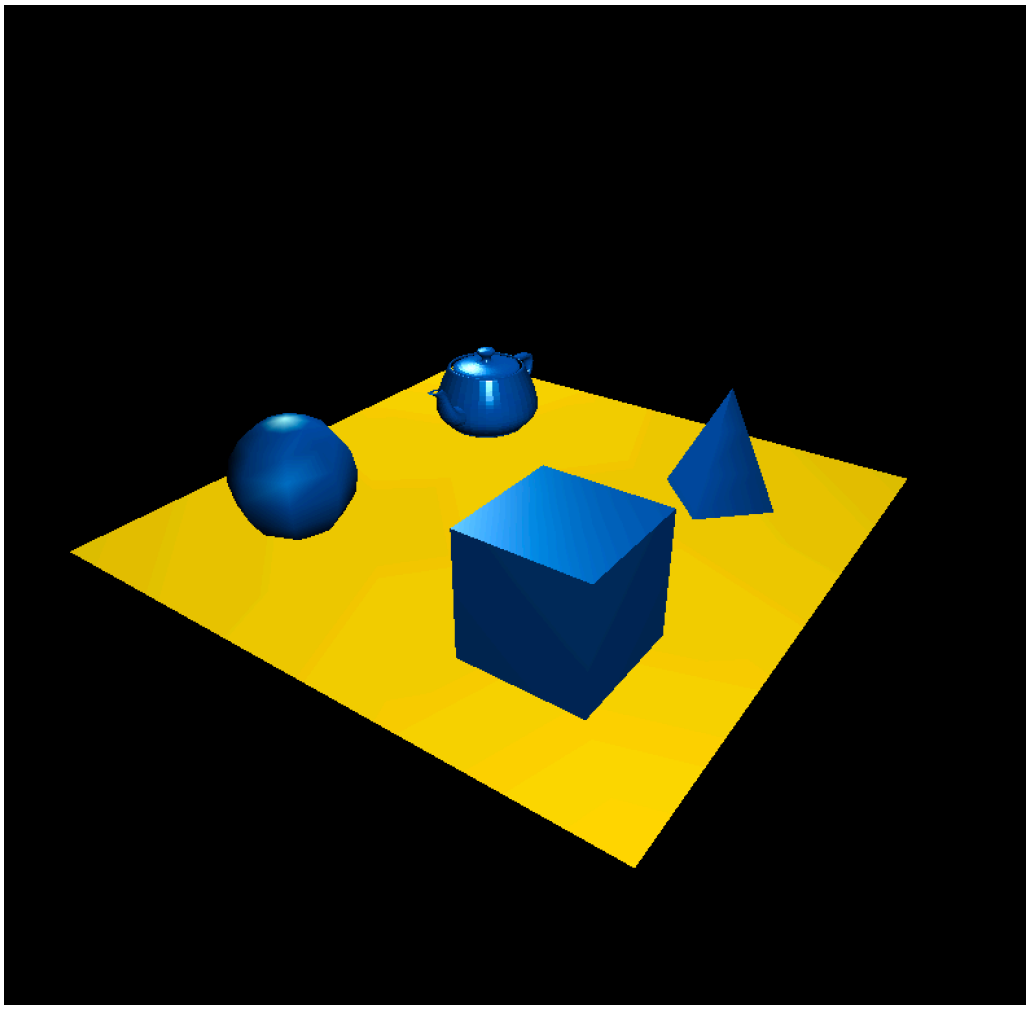

Figura 3: Figura 1: Test_4_3

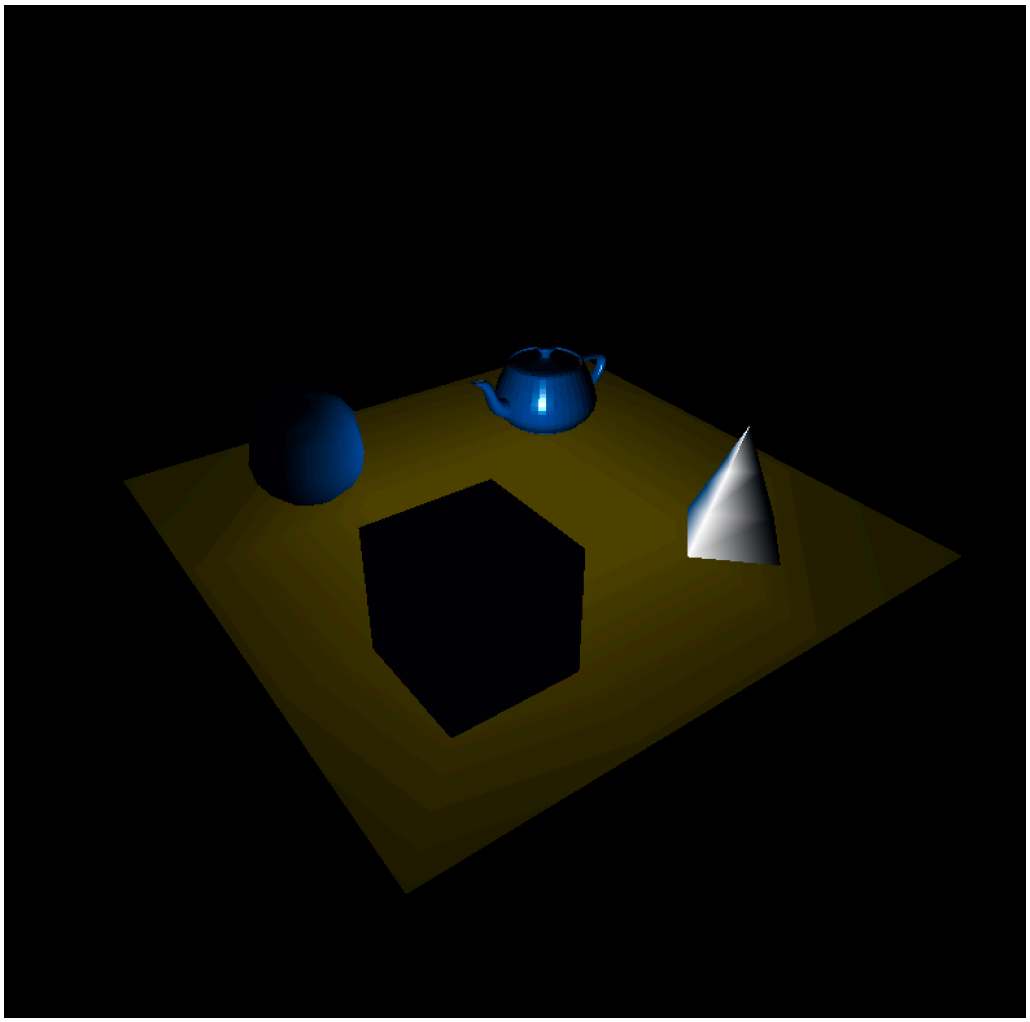

Figura 4: Figura 1: Test_4_4

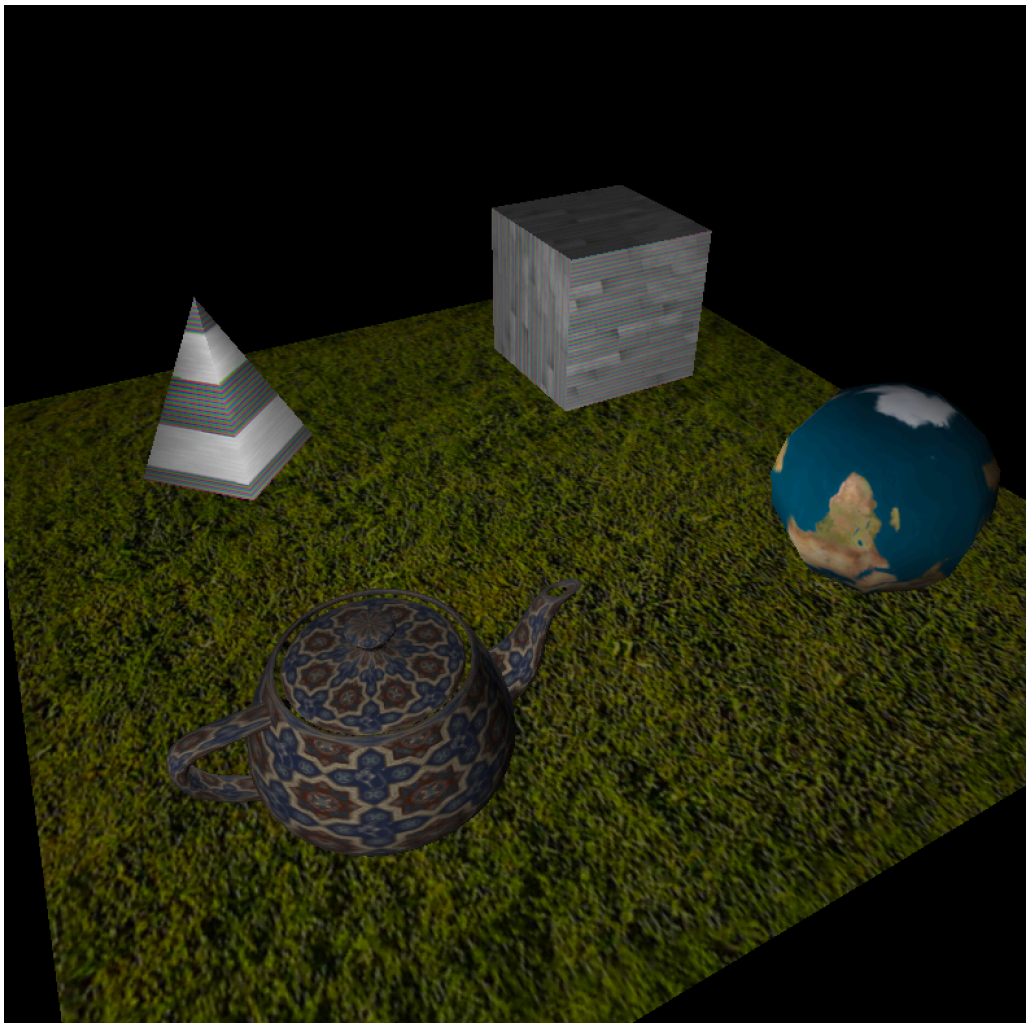

Figura 5: Figura 1: Test_4_6

Figura 6: Figura 1: Solar System

5. Conclusão

O desenvolvimento deste motor gráfico 3D, culminando na Fase 4, representou um avanço significativo na capacidade de renderização de cenas tridimensionais complexas e visualmente ricas. Ao longo das diferentes etapas do projeto, conseguimos implementar funcionalidades essenciais que transformaram um sistema básico de exibição de modelos em um motor gráfico completo, com suporte a iluminação dinâmica, texturização, materiais físicos e otimização de desempenho.

Um dos principais destaques foi a introdução do gerador automático de modelos 3D, que permite criar objetos primitivos (como esferas, cubos, cones e cilindros) com normais e coordenadas de textura calculadas matematicamente. Essa automatização não apenas agilizou o processo de modelagem, mas também garantiu precisão nos cálculos de iluminação e mapeamento UV, fundamentais para um resultado visual realista.

Além disso, a implementação de Vertex Buffer Objects (VBOs) trouxe uma melhoria substancial no desempenho, transferindo o processamento geométrico para a GPU e reduzindo o overhead da CPU. Combinado com o sistema de materiais e luzes configuráveis, que inclui propriedades como difusão, reflexão especular e brilho, o motor gráfico alcançou um nível de realismo adequado para simulações e visualizações avançadas.

A estrutura baseada em hierarquia de cena e transformações aninhadas permitiu a composição de ambientes complexos, enquanto o parser de XML facilitou a configuração de cenas através de arquivos externos, tornando o sistema mais flexível e modular.