

CENTRO FEDERAL DE EDUCAÇÃO TECNOLÓGICA DE MINAS GERAIS

ENGENHARIA DA COMPUTAÇÃO

RELATÓRIO TÉCNICO - PROJETO FINAL DE COMPUTAÇÃO GRÁFICA

---

## **Modelagem e animação interativa 3D: Navegação de um barco a remo em alto mar**

---

*Aluno:*

Lucas de Souza Gontijo

*Orientador:*

Prof. Dr. Higor Alexandre Duarte Mascarenhas

14 de Dezembro, 2025



# 1 Introdução

A Computação Gráfica é uma área da Ciência da Computação dedicada à geração, manipulação e visualização de imagens por meio de sistemas computacionais [1]. Ela é um dos pilares tecnológicos das indústrias de filmes e jogos digitais, sendo responsável pela criação de ambientes virtuais, personagens, efeitos visuais e simulações realistas. Em produções cinematográficas, técnicas como modelagem tridimensional, animação, iluminação e renderização permitem a construção de cenas complexas que seriam inviáveis ou excessivamente custosas no mundo real. De forma semelhante, nos jogos digitais, a Computação Gráfica viabiliza a renderização em tempo real de cenários interativos, garantindo desempenho, imersão e fidelidade visual.

Além do entretenimento, a Computação Gráfica tornou-se indispensável no cotidiano das pessoas, especialmente na divulgação de informações, na acessibilidade e na visualização de conteúdos. Interfaces gráficas intuitivas, infográficos, mapas interativos e simulações visuais facilitam a compreensão de dados complexos e ampliam o acesso à informação por diferentes públicos. Em contextos comerciais, educacionais e informativos, o uso de recursos gráficos melhora a comunicação, otimiza processos e aumenta a praticidade na interação com sistemas digitais, consolidando a Computação Gráfica como uma tecnologia essencial no dia a dia moderno [2].

Na disciplina de Computação Gráfica, foram apresentados os principais conceitos fundamentais dessa área, incluindo as teorias que orientam as transformações geométricas, a modelagem de objetos tridimensionais, o uso de câmeras virtuais, shaders, texturas e sombras, além de aspectos técnicos relacionados a *softwares* voltados para modelagem e simulação 3D. Esses conteúdos possibilitaram compreender de forma ampla como a Computação Gráfica é aplicada atualmente, principalmente nas grandes produções cinematográficas e na indústria de jogos digitais.

Para demonstrar que o conteúdo da disciplina estudado foi fixado, o trabalho final da disciplina, descrito neste relatório, propõe aos alunos o desenvolvimento de um projeto gráfico envolvendo modelagem, animação e aplicação das técnicas aprendidas ao longo do curso. Conforme as instruções fornecidas, cada estudante deveria elaborar um projeto tridimensional utilizando ferramentas como *Blender*, *Unity 3D*, *Unreal Engine* ou outra equivalente. O projeto deveria incluir a criação de um objeto, cena ou animação 3D, contendo algum nível de interatividade ou movimento, de modo a apresentar técnicas vistas em aula.

## 1.1 Justificativa da escolha

Na computação gráfica, principalmente em jogos, a construção de um sistema aquático é apontada, principalmente em blogs como *Redit* ou através de vídeos no *Youtube*, sendo um bom medidor de qualidade da obra [3] visto a complexidade do desenvolvimento. Jogos como *GTA V* (2013) são pautados ainda hoje pela qualidade física e gráfica da água desenvolvida pelo estúdio *Rockstar Games*. Sendo assim, foi de extrema curiosidade entender na prática o que torna a água essa referência de qualidade e entender quais as dificuldades por trás do desenvolvimento desse tipo de sistema. Assim, foi proposto o desenvolvimento de uma animação gráfica interativa na qual o ambiente aquático (nesse caso o oceano) é o principal compositor da cena. Além disso, a construção de um sistema como esse envolve vários conceitos abordados durante a disciplina de Computação Gráfica, o que colabora para a riqueza do trabalho.

## 1.2 Objetivos

O objetivo principal deste projeto era o de desenvolver uma cena interativa, similar a um jogo, onde cada aluno pudesse não só escolher a metodologia na qual trabalharia, mas também criar modelos 3D ou personalizá-los de modo a inseri-los em cena posteriormente. Ao propor um modelo de navegação em mar aberto, o objetivo principal no contexto de computação gráfica foi: reproduzir a física e gráficos (*shader*) realistas para a água; personalizar um modelo 3D de um pequeno barco; modelar completamente o remo e construir sua animação; aplicar texturas ao céu e inserir uma iluminação para deixar toda a cena algo mais crível.

## 1.3 Objetivos Secundários

1. Utilizar um *software* de modelagem e/ou um motor de jogo para desenvolver uma animação interativa.
2. Desenvolver ou personalizar um modelo 3D, que reberá animações e texturas, e deverá possuir funcionalidades que poderão ser acionadas pelo usuário.
3. Aplicar conceitos da disciplina de Computação Gráfica para demonstrar o entendimento do conteúdo.

## 2 Metodologia

O processo metodológico do trabalho apresentado foi dividido em 6 etapas principais: (I) escolha das ferramentas utilizadas; (II) compreensão de como estúdios grandiosos desenvolveram a água e interações dela com outros objetos em seus jogos; (III) utilização de *shaders* e criação da física da água; (IV) personalização de um modelo 3D de um pequeno barco e definição de *probes* para funcionar o *buoyancy*, como é visto na literatura, ou empuxo da água; (V) construção da cena e animação/interação, de remos e fundo do barco, com a água; (VI) aplicação de texturas dos objetos e do céu, assim como adição de iluminação simulando o sol no motor de jogos.

### Fase I

A justificativa do projeto mostra como foi possível escolher o tema, e seria necessário entender as melhores ferramentas para se trabalhar nesse caso. Após avaliar a os motores de jogos *Unreal Engine*, *Unity* e *Godot Engine*, a escolha da 3ª opção deve-se, principalmente, ao seu caráter *open-source*, à leveza do ambiente de desenvolvimento, a similaridade de sua linguagem com a que é vista em *Python* e a integração com formatos de arquivos gráficos. Além disso, o motor oferece suporte nativo a renderização 2D e 3D, utilização de *shaders*, sistema de física integrado e compatibilidade com padrões amplamente utilizados, como *glTF*. Para modelagem e personalização de modelos 3D, o *Blender* foi a opção escolhida principalmente por contar com uma vasta documentação e tutoriais disponíveis para entusiastas no assunto.

### Fase II

Dada a afirmação no tópico Justificativa, de que a água é um fator que muitas vezes impõe complexidade a produção de jogos, foi de grande ajuda procurar por trabalhos em que esse sistema é

utilizado e assim, aproveitar técnicas consolidadas para entender e agilizar o desenvolvimento da física da água e também a colisão dela com outros objetos. Em um documento relatando detalhadamente as técnicas utilizadas na produção do oceano presente no jogo Assassin's Creed III (2012) [4], a equipe Ubisoft mostra os passos para trazer realismo a um jogo que envolve batalha naval 3D e as escolhas acertadas durante sua produção.

Para o trabalho em questão, claro, não foi necessário e, muito menos, viável utilizar toda a gama de técnicas discutidas no documento. Apesar disso, ele orienta também em questões consideradas simples como o movimento da água simulando ondas, e também como podem ser selecionados os pontos de "prova" (*probe*) do barco ou modelo 3D que interage com a água, para que o sistema de flutuação funcione acompanhando os movimentos da água. Na imagem a seguir, a ilustração desse mecanismo de flutuação:



Figura 1: Esferas de prova fixadas ao barco para simular flutuação no mar; imagem retirada de *fxguide*.

### Fase III

A terceira fase do projeto concentrou-se na utilização de *shaders* para a simulação visual da água e na criação de uma aproximação para sua física. Os *shaders* empregados não foram desenvolvidos do zero, mas obtidos a partir de materiais disponíveis publicamente em repositórios e sites especializados, sendo posteriormente adaptados para atender às necessidades do projeto. Esses *shaders* foram responsáveis pelo deslocamento dos vértices da malha da água, simulando o movimento das ondas ao longo do tempo. Paralelamente, a física da água foi implementada por meio da amostragem de valores de ruído utilizados no *shader*, permitindo estimar a altura da superfície em pontos específicos e sincronizar o comportamento visual com a interação física de outros objetos presentes na cena.

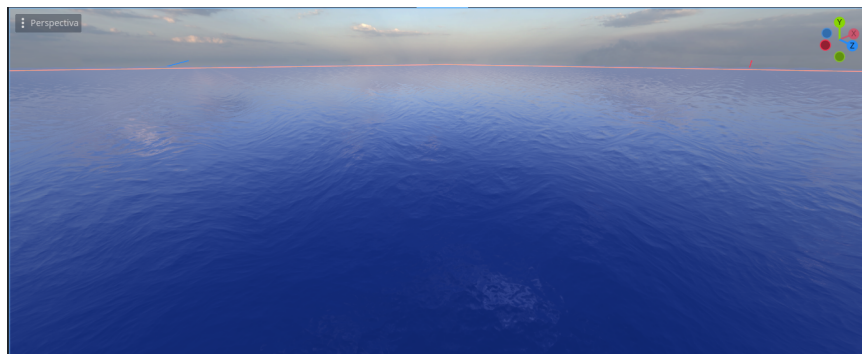


Figura 2: *Shader* de água com movimentos de ondas simulando um oceano.

## Fase IV

Na quarta fase, realizou-se a modelagem e personalização de um modelo tridimensional simplificado de um pequeno barco, juntamente com seus remos, utilizando o *Blender* e um conjunto de tutoriais disponíveis na plataforma YouTube. O modelo foi desenvolvido com uma abordagem *low poly*, visando reduzir a complexidade geométrica e garantir melhor desempenho durante a simulação. Após a finalização, o modelo foi importado para o *Godot Engine*, onde foram definidos pontos de amostragem, conhecidos como *probes*, distribuídos ao longo da estrutura do barco. Esses *probes* foram utilizados para calcular o empuxo exercido pela água, permitindo a simulação do fenômeno de *buoyancy*, conforme descrito na literatura, ou simplesmente empuxo, garantindo um comportamento mais estável e coerente do objeto flutuante. Para isso, foram espalhadas 7 *probes* de forma uniforme ao longo do casco do barco.

Ainda nesta fase, os remos foram animados por meio do uso de *keyframes*, definindo transformações de translação ao longo do tempo para simular o movimento de remada a partir de interpolações. Essa animação foi realizada de forma independente do casco do barco, permitindo maior controle sobre o movimento e facilitando a posterior integração com a interação física entre os remos e a superfície da água.

Além disso, os remos apresentam comportamento individual, no qual as teclas A e Q são responsáveis pelo controle do remo esquerdo, sendo a tecla A associada ao avanço do barco e a tecla Q ao recuo. De forma análoga, as teclas D e E controlam o remo direito, onde a tecla D está relacionada ao avanço e a tecla E ao recuo. Essa funcionalidade foi inspirada no sistema de remada presente no jogo *Sea of Thieves*, desenvolvido pela *Microsoft*.

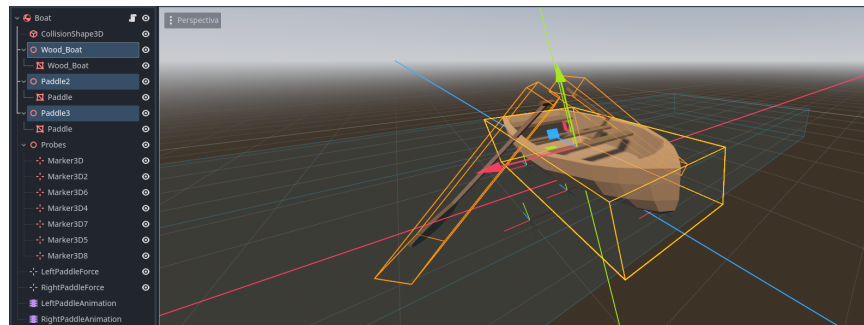


Figura 3: Modelo de barco *low poly* já importado na *Godot Engine*, com remos independentes e sete *probes* distribuídas ao longo de sua estrutura.

## Fase VI

Na última fase do projeto, foram aplicadas as texturas aos objetos da cena, bem como configurado o ambiente visual geral. As texturas utilizadas foram obtidas a partir dos recursos disponibilizados pelo próprio *Godot Engine* e fora dele, sendo aplicadas tanto ao barco quanto aos demais elementos do cenário. Também foi adicionada uma textura ara HDRi (*High Dynamic Range Image* - Imagem de Alta Faixa Dinâmica) ao céu com nuvens em 2D, encontradas na plataforma *Poly Haven* [5], servindo como base para a ambientação da cena. A iluminação foi configurada de forma a simular a presença do sol, com a inserção de luz direcional 3D, buscando coerência entre a direção da luz e a textura utilizada, de modo a reforçar o realismo visual e a percepção espacial do ambiente.

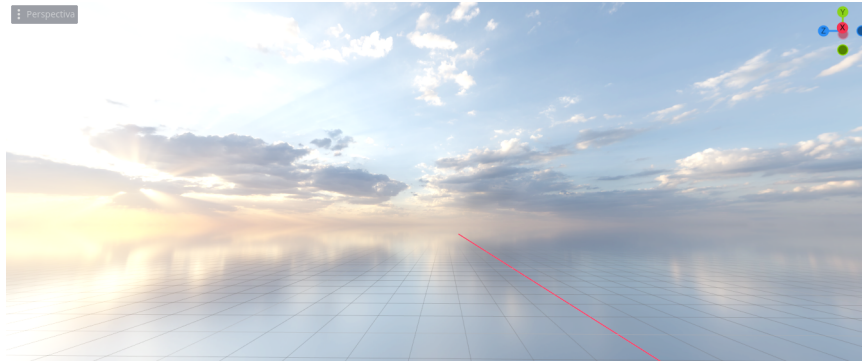


Figura 4: Textura HDRI do céu e iluminação direcional 3D aplicada a cena.

### 3 Resultados

A simulação desenvolvida apresentou uma superfície de água animada por meio de *shaders*, com movimentação contínua das ondas ao longo do tempo. O modelo do barco respondeu à altura da água calculada a partir das *probes*, mantendo-se flutuante e acompanhando as variações da superfície. A interação entre o barco e a água ocorreu de forma consistente, permitindo observar o efeito do empuxo durante a execução da cena.

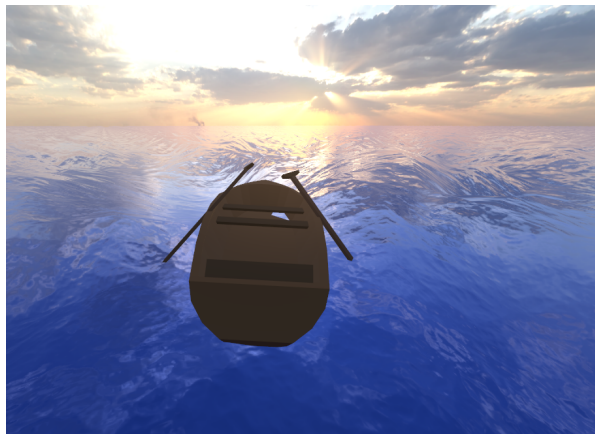


Figura 5: Simulação final da cena

Os remos animados por *keyframes* responderam corretamente às entradas do teclado, possibilitando o avanço e o recuo do barco de acordo com as ações do usuário. As texturas aplicadas aos objetos e a iluminação direcional configurada para simular o sol resultaram em uma ambientação visual coerente com a proposta da cena.

### 4 Conclusão

De modo geral, o sistema desenvolvido contribui para consolidar os conteúdos abordados ao longo da disciplina de Computação Gráfica. Além da parte técnica relacionada à utilização do motor gráfico e de *shaders*, foi possível compreender, de forma sucinta, algumas das técnicas empregadas em sistemas complexos de simulação de oceanos, como as descritas na documentação do *fxguide* referente ao

jogo *Assassin's Creed III*, e adaptá-las para um modelo mais simplificado, conforme apresentado neste relatório.

Apesar dos resultados gerais positivos do projeto, alguns problemas tornam-se perceptíveis durante a execução da simulação, como a transposição da água pelos limites internos do barco em determinados momentos e a ausência de um sistema que considere a profundidade da água, bem como o enriquecimento da iluminação para detalhar essas regiões e até mesmo mudança no comportamento da maré. Ainda assim, considerando o escopo da disciplina, o trabalho desenvolvido atendeu às expectativas propostas e marca a conclusão das atividades realizadas.

## Referências

- [1] “O que é computação gráfica.” [https://lapix.ufsc.br/ensino/computacao-grafica/programa/1.1.-introducao-e-conceitos-basicos/#O\\_que\\_e\\_computacao\\_Grafica](https://lapix.ufsc.br/ensino/computacao-grafica/programa/1.1.-introducao-e-conceitos-basicos/#O_que_e_computacao_Grafica). Acessado em: 10 dez. 2025.
- [2] “Como a computação gráfica transformou a indústria da publicidade nos últimos anos.” <https://jornalraza.com/geral/como-a-computacao-grafica-transformou-a-industria-da-publicidade-nos-ultimos-anos/>. Acessado em: 10 dez. 2025.
- [3] “Por que é tão difícil simular água nos jogos?.” <https://www.youtube.com/watch?v=UFhAGX1pzLg&t=285s>. Acessado em: 10 dez. 2025.
- [4] “Assassin's creed iii: The tech behind (or beneath) the action.” <https://www.fxguide.com/ffeatured/assassins-creed-iii-the-tech-behind-or-beneath-the-action/>. Acessado em: 08 dez. 2025.
- [5] “Hdri / kloppenheim 06 (pure sky).” [https://polyhaven.com/a/kloppenheim\\_06\\_puresky](https://polyhaven.com/a/kloppenheim_06_puresky). Acessado em: 08 dez. 2025.