Relatório Final

Filipi Biazoto

Jean Pasquini

Espírito Santo do Pinhal, SP
Maio/2025

Sumário

Resumo

Este projeto consiste no desenvolvimento de um jogo multiplayer local em 2D, de estratégia por turnos com base em física, onde os jogadores controlam tanques em um ambiente pixelado e destrutível. O jogo foi planejado e desenvolvido utilizando boas práticas de design e programação, com um foco especial em jogabilidade, imersão visual e mecânicas estratégicas. Ao longo do projeto, foram aplicadas metodologias ágeis e ferramentas como Trello e GitHub para organização e versionamento do código.

As decisões de design foram orientadas por um Game Design Document (GDD) detalhado, que guiou todas as etapas do projeto, desde a concepção da mecânica central até a implementação de funcionalidades como câmera dinâmica, interface intuitiva e ambientação sonora.

O uso de sprites em pixel art e trilha sonora 8-bit reforça o estilo retrô proposto, enquanto os efeitos climáticos, destruição de terreno e animações no cenário contribuem para uma atmosfera viva e imersiva. Ao final, o jogo oferece não apenas uma competição divertida entre jogadores, mas também um produto completo que reflete organização, colaboração e aplicação prática de conceitos teóricos aprendidos ao longo do curso.

1. Introdução

O presente trabalho tem como objetivo apresentar o processo de criação de um jogo digital com base em um Game Design Document (GDD). O jogo proposto é um título multiplayer local com batalhas entre tanques em um mundo futurista com estética retrô. A proposta envolve desafios técnicos, artísticos e de design, englobando desde a concepção das regras até a implementação prática do sistema.

A ambientação do jogo se passa em um universo alternativo dominado por conflitos territoriais e avanços tecnológicos, onde diferentes facções utilizam tanques de guerra aprimorados para conquistar poder e influência. O estilo visual em pixel art e a trilha sonora inspirada em sons 8-bit reforçam a identidade nostálgica do projeto, enquanto as mecânicas de física realista, destruição de terreno e turnos estratégicos proporcionam profundidade tática e rejogabilidade.

Durante o desenvolvimento, foram utilizadas metodologias ágeis para garantir a organização e eficiência da equipe, com o Trello sendo adotado para o gerenciamento de tarefas e o GitHub para controle de versão e colaboração no código. O projeto também contemplou aspectos fundamentais do design de jogos, como balanceamento de personagens, feedback visual e sonoro, fluidez da interface e clareza nas regras. Assim, o resultado final é um produto coeso e funcional que integra teoria e prática em uma experiência de jogo envolvente e estratégica.

2. Metodologia

Para o desenvolvimento do jogo, adotamos metodologias ágeis com divisão de tarefas em sprints semanais. Ferramentas como Trello foram utilizadas para planejamento e acompanhamento do progresso, enquanto o GitHub foi usado para controle de versão do código. O projeto iniciou-se com a elaboração do GDD, onde todas as ideias, mecânicas e elementos visuais foram descritos detalhadamente. A seguir, desenvolvemos os sistemas de movimentação, combate, física, interface e interatividade com base nesse documento.

Cada etapa do desenvolvimento foi dividida em ciclos iterativos, permitindo testes e aprimoramentos constantes. Inicialmente, foram implementadas estruturas básicas como o menu de entrada, a movimentação dos tanques e a geração procedural de mapas a partir de matrizes. Em seguida, evoluímos para a criação dos sistemas de colisão, gravidade e destruição de terreno, assegurando que a jogabilidade fosse dinâmica e coerente com a proposta tática.

O sistema de combate foi desenvolvido com base em física realista, exigindo o cálculo da trajetória dos projéteis com base em ângulo e força. Para isso, implementamos ferramentas de mira e controle do canhão, além da interação dos disparos com os blocos do cenário, incluindo explosões em área e modificação do terreno.

A interface do usuário (HUD) e os menus também foram construídos de forma modular, visando facilitar o acesso às informações e comandos essenciais durante a partida. Por fim, realizamos testes para validar a estabilidade do jogo, corrigimos bugs, ajustamos sprites e sons, e finalizamos com a criação do menu de créditos e polimento geral do produto.

Essa abordagem permitiu um desenvolvimento organizado, colaborativo e focado na entrega de um jogo funcional e imersivo, respeitando os princípios estabelecidos no planejamento inicial.

3. Implementação

A implementação foi dividida em diversas etapas:

- Criação do menu principal e estrutura básica do jogo;
- Desenvolvimento do sistema de movimentação dos tanques com física e colisão;
- Implementação do sistema de mira e disparo com cálculo de ângulo e força;
- Geração procedural do mapa com terrenos destrutíveis e elementos indestrutíveis;
- Criação de três modelos de tanques com atributos distintos;
- Implementação do sistema de turnos com limite de tempo por jogador;
- Interface de HUD com status dos jogadores, tempo de turno e indicadores de ação;
- Trilha sonora e efeitos sonoros inseridos conforme o tema pixelado de guerra;
- Sistema de posicionamento inicial dos tanques, com pontos pré-definidos para garantir o equilíbrio da partida;
- Mecânica de derrota por queda na água, ampliando as estratégias possíveis por meio da destruição do terreno;
- Câmera dinâmica, que foca no jogador ativo e permite zoom com o scroll do mouse, proporcionando melhor visibilidade e imersão;
- Menus de seleção de personagens e preparação de partida, com informações detalhadas dos tanques e opções de configuração do jogo;
- Sistema de explosão em área, capaz de alterar blocos do cenário e causar dano baseado no raio de impacto;
- Controle completo por teclado e mouse, com comandos intuitivos tanto nos menus quanto durante a jogabilidade.

4. Código

MATRIZ DE OBJETOS / CENÁRIO

- CONSTRUÇÃO DE CENÁRIO

Foi utilizado para construção do cenário uma matriz onde são pré-colocados diversos números em resolução de (...)x(...) quadrados. Para melhorar a visualização de nossos desenvolvimentos, criamos uma planilha no google sheets que simula um "tilemap" porém para objetos.

- DEFINIÇÃO DE OBJETO

Cada número colocado na matriz era passado por uma verificação de condição e criaria um objeto para cada um.

- MANIPULAÇÃO

Cada objeto passaria a conter seu próprio tamanho, função e cor.

PERSONAGENS

- COLISÃO

Na colisão de objetos interativos foi passado por uma verificação de posições tanto do personagem quanto do bloco objeto posicionado, caso seu posicionamento batesse com ambos o personagem era automaticamente posto ao seu local contrário do objeto.

- GRAVIDADE

Criadas variáveis para conseguir manipular o controle de queda do personagem em relação ao chão. Caso houvesse contato com o chão o personagem teria seu movimento posição y = 0, caso contrário uma variável do tipo gravidade (para determinarmos um valor que vai acrescentando) diminuiria a posição Y conforme o valor da variável.

- CANHÃO / TIRO

Para fazer a simulação de um cano de um tanque onde ele se movimenta juntamente com o tanque porém tem uma interação diferente, foi criado um novo objeto onde permanece conectado ao centro do personagem, pegando o personagem.y e personagem.x o tamanho de x e y do personagem e dividindo por 2. Depois disso foi implementado a ação onde o personagem realiza o disparo que ao pressionar em tal tecla, ativa uma barra controlada por uma variável denominada força e aumentando por uma outra variável aos poucos, onde após o jogador soltar a tecla contabiliza o

disparo criando se um objeto do tipo projétil que possui uma série de variáveis como: x, y, tamanho x, tamanho y e influência da força do jogador. Possuindo o mesmo tipo de gravidade do personagem para ele ir fazendo a parábola.

- MOVIMENTAÇÃO

Utilizado a própria funções do love2d para contabilizar as teclas do teclado e fazendo devida função para cada botão, como:

- andar para direita: “D” - fazer com que o x do jogador aumente continuamente enquanto pressionado
- andar para esquerda: “A” - Fazer com que o x do jogador diminui continuamente enquanto pressionado
- mudar angulação anti-horário: “W” - Fazer uma função onde o cano do canhão é alterado continuamente enquanto pressionado
- mudar angulação horário: “S” - Fazer uma função onde o cano do canhão é alterado continuamente enquanto pressionado
- disparar: “F” - Fazer com que a força do personagem seja acumulada continuamente enquanto pressionado respeitando após chegar a força máxima.
- pular: “Space” - Fazer com que o Y do personagem seja aumentado gradualmente até 10, após isso a força da gravidade é aplicado sobre o personagem.

- CÂMERA DINÂMICA

Foi implementada uma câmera dinâmica que segue o personagem de forma suave, usando LERP. Basicamente, em vez da câmera ficar colada no personagem ou se mover bruscamente quando ele anda, eu usei uma interpolação linear para fazer a câmera 'chegar' até a posição do personagem aos poucos reduzindo gradualmente seu x e y (Contudo uma criação de objeto apenas para a câmera). Funciona assim: a cada frame, a câmera olha onde o personagem está, calcula essa posição como sendo o destino, e aí ela se move em direção a esse ponto, mas suavemente.

- DESTRUIÇÃO DE CENÁRIO

Para destruir o cenário o projétil criado na ação de disparar é feito o mesmo passo para colisão de um personagem com objeto, contudo se o projétil se colidir com objeto é feito uma verificação do local de origem a uma área de 100x100 de todos os objetos que tiverem essa localização sofrerem uma ação.

- Personagem - sofre ação de dano e knockback
- Dano - Calculado com o valor da variável de dano do personagem
- Knockback - Verificando o valor contrário da origem atingida do projétil e a posição do jogador, aumentando ou diminuindo gradualmente o X e Y do jogador fazendo uma simulação de knockback.
- Objeto - excluído do mapa e sumindo sua interação

CENÁRIO VISUAL

- TILES / SPRITES

Para construção de tiles e sprites foram postos em suas próprias classes de objetos e personagens uma variável onde armazena a imagem correta ou uma série de sprites rolando por um determinado momento para simulação de movimento, de modo que cada ação tem uma condição diferente para cada sprite.

- PLACE TILES - TILE MAP

Para facilitar e agilizar os processos de tiles no mapa todo, fizemos uma planilha semelhante ao da matriz de objetos onde são colocados os tiles que desejamos, contudo conseguimos visualizar antes mesmo de colocarmos em prática no jogo.

MENUS

- PRINCIPAL

Todos os botões criados com simples comandos do próprio love2d como criar retângulo e preenchê-lo com uma determinada cor, além de ser utilizado fonte de terceiros. Feito uma função para cada botão como mudar o estado do game, onde dependendo do estado muda o Load do game para tela da gameplay, configuração etc.

- CRÉDITOS

Realizado um tela com um texto enorme splitado diminuindo aos poucos o seu y, para demonstrar os criadores e realizações colocadas no trabalho feito

- CONFIGURAÇÕES

Realizado o mesmo preenchimento de botões do menu principal.

TRILHAS SONORAS / EFEITOS SONOROS

- VOLUME SEPARADOS

Para conseguirmos controlar os valores dos volumes criamos 2 variáveis globais para administrar os sons, uma para Efeito Sonoro e outra para Trilha Sonora, onde é aplicado os sons conforme a ação é realizada pelo player ou troca de cenário por exemplo. Contudo possui uma função onde apenas colocamos o caminho do som que é ocorrido juntamente com a variável global do som, administrada pelo próprio jogador no menu de configurações.

- ALTERAÇÃO DE MÚSICAS

Para facilitar alterações de música e efeitos sonoros foram realizadas uma função onde diminui o som gradualmente de uma música para outra, contudo ela vai reduzindo o som até chegar a 0 e retornando com a música pré-selecionada no valor do som atual.

REGRAS DE JOGO

- TEMPO DE JOGADA

É computado a todo momento um timer que é passado na função de update, pois ela vai retirando o tempo conforme ocorre as ações. No total 20 segundos, porém com uma finalidade de caso o jogador atirar, fazer com que o time fique 4s para sua pós ação.

- TURNO

Os turnos são decididos pelos jogadores em ordem player 1, player 2, player 3 e player4, onde caso o usuário fizer o disparo e passar os 4 segundos passa o seu turno para o próximo jogador ou o jogador não disparar e passar os 20s ou morrer durante o processo de seu disparo.

- +2 JOGADORES SIMULTÂNEOS

Como os players são apenas objetos para essa realização, apenas são criados mais objetos do tipo player que vão receber identidades diferentes.

HUD

- VIDA

Posto uma imagem png de moldura calculando o local conforme quantos jogadores há na partida, dividir conforme quantos jogadores há. Para aparecer a vida atual do personagem é colocado atrás da moldura um preenchimento de retângulo conforme a porcentagem de vida atual do personagem sendo que 100 é o tamanho atual da imagem de moldura.

- TURNO

Apenas uma label abaixo do tempo mostrando qual jogador é a vez, pegando da variável onde administra de quem é o turno. Além disso, ao passar o turno aparece bem explícito o jogador da vez.

- TEMPO

Foi criado um mini relógio que vai diminuindo conforme o tempo mostrando a quantidade de tempo sobrando e a ação de um relógio no anti-horário até acabar o tempo.

- FORÇA DO TIRO

Uma imagem png de moldura que faz a mesma ação da vida, ou seja é preenchida conforme a variável da força do personagem é pressionada.

5. Resultados

O projeto resultou na criação de um jogo 2D multiplayer local que combina estratégia por turnos com física realista, em um ambiente destrutível e estilizado em pixel art. A partir das diretrizes definidas no GDD, todas as funcionalidades essenciais foram implementadas com sucesso, incluindo:

- Sistema de movimentação física dos tanques com colisão e gravidade;
- Cálculo de disparo com ângulo e força, influenciado pela gravidade;
- Terreno parcialmente destrutível, com impacto direto na estratégia de jogo;
- Sistema de turnos com limite de tempo, promovendo decisões rápidas e táticas;
- Três tanques distintos, cada um com atributos próprios (vida, velocidade e poder de fogo);
- Interface intuitiva (HUD) com informações claras de turno, vida e força de ataque;
- Câmera dinâmica com foco no jogador ativo e ajuste de zoom;
- Ambientação sonora com trilha 8-bit e efeitos de combate;
- Menus de seleção de personagem e preparação estratégica antes das partidas;
- Mecânica de eliminação por queda na água, ampliando a complexidade tática.

Os testes realizados demonstraram que o jogo oferece uma experiência fluida, divertida e estratégica, com uma curva de aprendizado acessível e boa jogabilidade. O sistema modular e o uso de boas práticas de programação também facilitaram o polimento final do jogo, correção de bugs e implementação de ajustes visuais e sonoros.

6. Conclusões

A partir da proposta apresentada no GDD, o desenvolvimento do jogo foi bem-sucedido ao transformar conceitos teóricos em uma aplicação prática robusta. A utilização de metodologias ágeis, a divisão clara de tarefas e o uso de ferramentas como Trello e GitHub contribuíram para um processo eficiente, colaborativo e organizado.

O jogo final entrega uma experiência coerente com sua proposta: um combate tático entre tanques, onde cada decisão importa. A destruição de terreno, o uso estratégico do mapa e a variação entre personagens criam partidas únicas a cada rodada. A estética retrô e a trilha sonora reforçam a imersão no universo proposto, enquanto a interface clara garante acessibilidade a todos os jogadores.

Além de atingir os objetivos acadêmicos, o projeto também evidenciou o amadurecimento técnico e criativo dos desenvolvedores, que conseguiram integrar programação, design visual, sonora e de gameplay de forma harmoniosa. O jogo serve, portanto, como um marco importante na formação e portfólio dos autores, bem como uma base sólida para futuras expansões e melhorias.

7. Bibliografia

- Programando em LUA - Roberto Ierusalimschy
- Löve for Lua Game Programming - Damilare Darmie Akinlaja

Anexo 1. Game Design Document

Anexo 2. Código Fonte