CGR - Relatório do Trabalho Final

Kauã Hopfer[1]

[1]Departamento de Computação – Universidade do Estado de Santa Catarina (UDESC)
Joinville – SC – Brazil

Resumo. *Este é o relatório de desenvolvimento do trabalho final da discplina de Computação Gráfica (CGR) ministrada pelo Prof. André Tavares em 2024/2.*

1. Sobre o desenvolvimento do jogo

O desenvolvimento do jogo *Air Combat* foi realizado com uso da linguagem Python e da biblioteca `pygame`. O objetivo foi criar um jogo 2D em que o jogador controla um avião, enfrentando inimigos enquanto atira mísseis e tenta sobreviver pelo maior tempo possível. O projeto incluiu funcionalidades como gerenciamento de telas, movimentação e colisão de inimigos, controle de som, e a adição de um cronômetro que incrementa a dificuldade ao longo do tempo.

Este relatório documenta as principais dificuldades encontradas e a sequência de tarefas realizadas, incluindo o tempo aproximado e esforço para cada atividade.

2. Sequência de Tarefas

Tarefa 1: Implementação inicial do jogo com `pygame`

Descrição: Configuração inicial do ambiente de desenvolvimento com `pygame`, criação da janela do jogo, tela de fundo e implementação básica do avião do jogador.
Dificuldades: Ajustar os controles de movimentação para garantir que fossem responsivos.
Tempo: $\approx$ 2h.

Tarefa 2: Movimentação e lógica dos inimigos

Descrição: Implementação da movimentação dos inimigos, incluindo geração aleatória de posições e comportamento ao atingir as bordas da tela.
Dificuldades: Inicialmente, os inimigos estavam se movendo muito rápido e tendo muito *overlap*.
Tempo: $\approx$ 1h.

Tarefa 3: Controle de som

Descrição: Configuração do som no jogo, utilizando `pygame.mixer` para reproduzir a música de fundo e os efeitos sonoros, como explosões.
Dificuldades: Garantir que os volumes entre música e efeitos fossem proporcionais e ajustar como e quando ocorre os sons.
Tempo: $\approx$ 2h.

Tarefa 4: Adição do cronômetro e incremento de dificuldade

Descrição: Implementação de um cronômetro que registra o tempo de jogo e aumenta a velocidade dos inimigos a cada 10 segundos. O tempo também é exibido na tela de *game over* para mostrar o tempo de sobrevivência do jogador. **Dificuldades:** O cronômetro estava contando o tempo mais rápido do que deveria, exigindo ajustes no cálculo do `delta_time`. **Tempo:** $\approx$ 3h.

Tarefa 5: Tela de início e *game over*

Descrição: Criação de telas de início e *game over*, com centralização de elementos e inclusão de botões para reiniciar ou sair do jogo.
Dificuldades: O alinhamento das imagens não estava correto, causando sobreposições na tela de *game over*. Foi necessário redimensionar as imagens e fazer os ajustes no código.
Tempo: $\approx$ 2h.

Tarefa 6: Ajuste de movimentação e responsividade

Descrição: Ajustes na lógica de movimentação para corrigir o problema de atraso ao pressionar teclas e garantir que o avião respondesse imediatamente às ações do jogador.
Dificuldades: Configurar corretamente `pygame.key.get_pressed()` e eliminar a interferência de eventos desnecessários que antes estavam sendo testados como `KEYDOWN`/`KEYUP`.
Tempo: $\approx$ 3h.

Tarefa 7: Refinamento final e teste

Descrição: Teste geral do jogo, incluindo ajustes finais no controle de som, balanceamento de dificuldade e validação de funcionalidades como cronômetro e telas de início/*game over*.
Dificuldades: Pequenos ajustes no código, no alinhamento de elementos e teste de compatibilidade do jogo em diferentes resoluções.
Tempo: $\approx$ 2h.

3. Principais Dificuldades

- **Movimentação Responsiva:** Ajustar os controles do avião para que fossem responsivos. Inicialmente, o avião continuava se movendo mesmo após o jogador soltar as teclas.
- **Cronômetro:** O cronômetro apresentava inconsistências ao usar `delta_time`, contando o tempo mais rápido do que deveria. Foi necessário ajustar o cálculo com base no método `clock.tick()`.