Porte do jogo *Traveling Will* para o *Nintendo Game Boy Advance*

Igor Ribeiro Barbosa Duarte e Vítor Barbosa de Araujo

Faculdade Gama
Universidade de Brasília
Brasil

10 de dezembro de 2018

Objetivo geral

- O objetivo geral deste trabalho é reescrever o jogo *Traveling Will*, desenvolvido originalmente para PC na disciplina de Introdução aos Jogos Eletrônicos, para o *Nintendo Game Boy Advance*

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento de níveis do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento do *level design* das fases do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento do *level design* das fases do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento do *level design* das fases do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento do *level design* das fases do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Objetivos específicos

Os objetivos específicos são:

- Comprimir imagens e músicas do jogo original para reduzir o uso de memória;
- Criar módulos para renderização de imagens e texto;
- Criar módulos para manipulação de *inputs* dos botões e carregamento de áudio;
- Criar módulos para detecção de colisões e manipulação de eventos;
- Criar métodos para carregamento do *level design* das fases do jogo;
- Executar e testar o jogo desenvolvido na plataforma escolhida.

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- *avconv*, *libpng* e *modplug-tracker*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Ferramentas

Para o porte do jogo foram utilizadas as seguintes ferramentas:

- *C++*
- *devkitARM*
- *GIMP*
- *GRIT*
- `avconv`, `librosa` e `modplug tracker`
- *VisualBoyAdvance-M*
- *Nintendo DS*
- *EZFlash II*

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: inicializar e tocar músicas e efeitos sonoros;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: iniciar e parar músicas de fundo;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: iniciar e parar músicas de fundo;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: iniciar e parar músicas de fundo;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: iniciar e parar músicas de fundo;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Desenvolvimento da *engine*

Para este trabalho foi desenvolvida a `gbengine`.

- Módulos implementados: *input*, vídeo, gerenciador de memória, áudio e física;
- **Módulo de input**: detectar pressionamento dos botões do GBA.
- **Módulo de vídeo**: renderizar, animar e atualizar imagens;
- **Módulo gerenciador de memória**: garantir alocação segura e eficiente de memória;
- **Módulo de áudio**: iniciar e parar músicas de fundo;
- **Módulo de física**: simular a ação de gravidade e detectar colisões entre objetos do jogo;

Módulo de input

- Detectar pressionamento dos botões
- *Bitmask* onde o valor 0 indica pressionamento

Módulo de input

```
1  #include "input.h"
2
3  volatile unsigned int *buttons_mem = (volatile unsigned int *) 0
       x04000130;
4
5  void check_buttons_states() {
6    for(int i = 0; i < N_BUTTON; i++) {
7      pressed_state[i] = !((*buttons_mem) & (1 << i));
8    }
9  }
10
11 bool pressed(int button) {
12   return pressed_state[button];
13 }
```

Figura: Código da classe *input*

Módulo de input

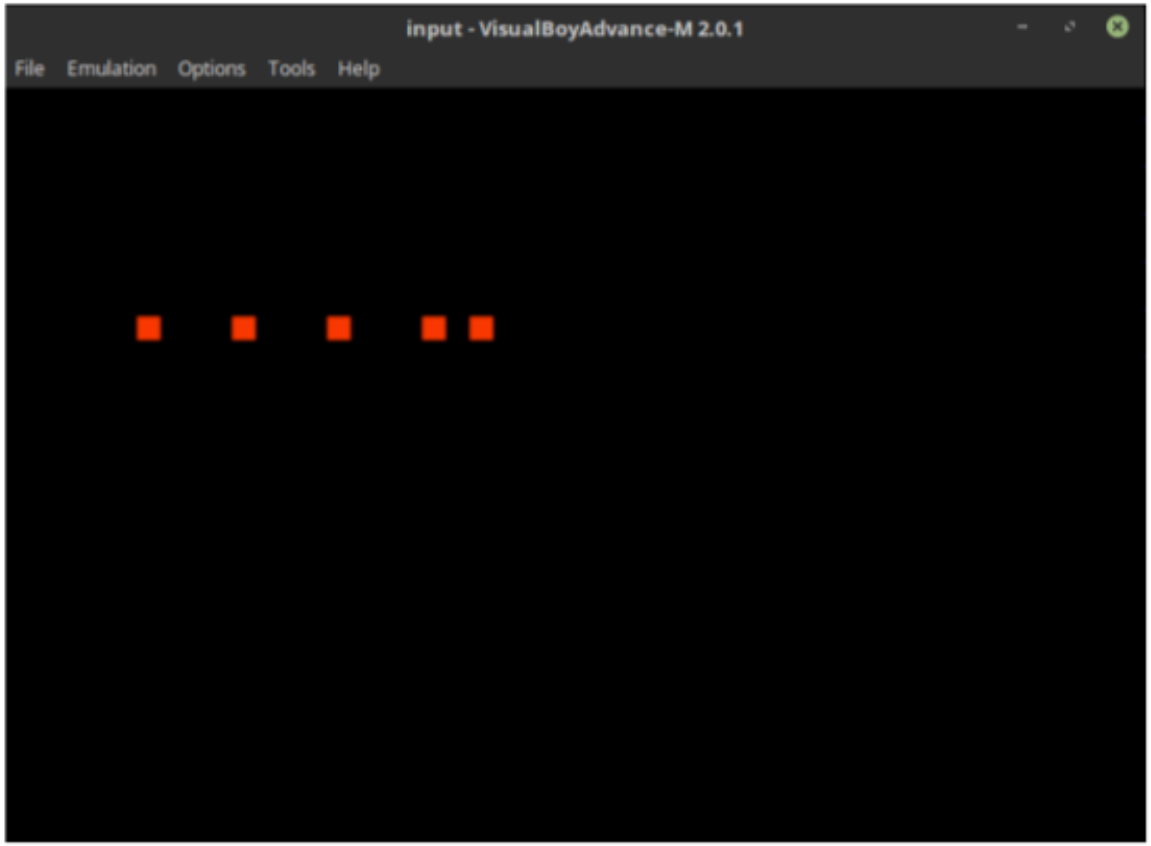

Figura: *Demo* do *input*

Módulo de input

```
9  int main() {
10   reset_dispcnt();
11   set_video_mode(3);
12   enable_background(2);
13
14   while(1) {
15     check_buttons_states();
16
17     for(int i=0;i<=9;i++){
18       int padding = i + 10;
19
20       if (pressing(1 << i)) {
21         for(int x=-2;x<=2;x++) {
22           for(int y=-2;y<=2;y++) {
23             vid_mem[(50 + x) * 240 + (20 + y + i * 10)] = RED;
24           }
25         }
26       }
27       else {
28         for(int x=-2;x<=2;x++) {
29           for(int y=-2;y<=2;y++) {
30             vid_mem[(50 + x) * 240 + (20 + y + i * 10)] = 0;
31           }
32         }
33       }
34     }
35   }
```

Figura: Código do *demo* de *input*

Módulo de vídeo

Módulo gerenciador de memória

Módulo de física

Módulo utilitário

Adaptação das músicas do jogo

Abstração de níveis e objetos do jogo

Adaptação das imagens do jogo

Construção dos níveis do jogo

Transição entre os níveis do jogo

Comparação entre o porte e o jogo original

Considerações Finais

Principais impedimentos durante a execução do trabalho:

- Entender detalhes do *hardware do GBA*;
- Testar o jogo no *console*;
- Adaptação de imagens e músicas do jogo;

Considerações Finais

Principais impedimentos durante a execução do trabalho:

- Entender detalhes do *hardware do GBA*;
- Testar o jogo no *console*;
- Adaptação de imagens e músicas do jogo;

Considerações Finais

Principais impedimentos durante a execução do trabalho:

- Entender detalhes do *hardware do GBA*;
- Testar o jogo no *console*;
- Adaptação de imagens e músicas do jogo;

Considerações Finais

Pontos de melhoria pós-execução do trabalho:

- Melhor priorização das tarefas a serem executadas;
- Testes mais frequentes no *console*

Considerações Finais

Pontos de melhoria pós-execução do trabalho:

- Melhor priorização das tarefas a serem executadas;
- Testes mais frequentes no *console*

Considerações Finais

É possível portar o jogo Traveling Will, desenvolvido para PC pelos autores deste trabalho, para o Nintendo Gameboy Advance, no contexto de um trabalho de conclusão de curso, com performance e jogabilidade próximos da versão para computador?
R: **Sim, é possível.**

Trabalhos futuros

- Melhorar módulo de áudio para carregar efeitos sonoros e pausar músicas;
- Implementar carregamento e utilização de fontes;
- Adicionar elementos de HUD e seleção de fases;
- Salvar o estado do jogo em memória;

Trabalhos futuros

- Melhorar módulo de áudio para carregar efeitos sonoros e pausar músicas;
- Implementar carregamento e utilização de fontes;
- Adicionar elementos de HUD e seleção de fases;
- Salvar o estado do jogo em memória;
- Implementar um desfragmentador de memória na classe `MemoryManager`

Trabalhos futuros

- Melhorar módulo de áudio para carregar efeitos sonoros e pausar músicas;
- Implementar carregamento e utilização de fontes;
- Adicionar elementos de HUD e seleção de fases;
- Salvar o estado do jogo em memória;
- Implementar um desfragmentador de memória na classe `MemoryManager`

Trabalhos futuros

- Melhorar módulo de áudio para carregar efeitos sonoros e pausar músicas;
- Implementar carregamento e utilização de fontes;
- Adicionar elementos de HUD e seleção de fases;
- Salvar o estado do jogo em memória;
- Implementar um desfragmentador de memória na classe `MemoryManager`

Trabalhos futuros

- Melhorar módulo de áudio para carregar efeitos sonoros e pausar músicas;
- Implementar carregamento e utilização de fontes;
- Adicionar elementos de HUD e seleção de fases;
- Salvar o estado do jogo em memória;
- Implementar um desfragmentador de memória na classe `MemoryManager`

Considerações Finais

Obrigado!