# Dragon Souls of Senac: Touhou Edition

**Danilo Makoto Ikuta**[1], **Gabriel Nopper Silva Roddrigues**[1]

[1]Centro Universitário Senacário Senac
04.696-000 – São Paulo – SP – Brasil

danilo0195@hotmail.com, ganopper@gmail.com

***Resumo.*** *Thread exception: text input here should be of type ¡String¿, but is ¡void¿ Thread Error: 11*

***Abstract.*** *Thread exception: text input here should be of type ¡String¿, but is ¡void¿ Thread Error: 11*

## 1. Introdução

Este relatório traz os resultados do Projeto Interativo III(PI III), de Bacharelado em Ciancia da Computação do Centro Universitário Senac, 2014. O PI III traz como proposta a construção de um jogo em linguagem c com auxilio das bibliotecas allegro, para construção grafica, e OpenCV para captura de video. Este jogo, deve ter apenas uma camera como meio de interação jogador-programa.
Cabe aos alunos esta interação jogador-programa via algoritmos de tratamento de visão computacional, e os relacionar com a interface também gerada via allegro.

## 2. Revisão de Literatura

### 2.1. A idéia

O próprio nome dado ao jogo indica uma composição de bases para a idéia. A idéia do jogo efetivamente surgiu com Dragon Quest, para plataforma Wii. Neste jogo, utiliza-se o WiiMote (O controle especofico do console) como a espada, e o nun-chuck(A contraparte esquerda do controle) como um escudo. No jogo, o jogador é colocado e um caminho no qual se anda constantemente. Ao longo deste caminho, entretanto, surgem diversos monstros que ele deve derrotar, e no final de cada caminho, um chefe. A idéia é similar - é dada uma fase, monstros nascem, mas ao invés de completar tempo, um numero especifico de monstros derrotados é necessário. Ainda sim, no final, um chefe também será encontrado.
O nome DSoSTE (Dragon Souls of Senac: Touhou edition) é dado baseado nos 3 jogos inspiração para o projeto. Destaquemos os outros 2:
Dark Souls: Dark Souls é um jogo recente (2011, 2014 o segundo) que ficou rapidamente famoso deviado a alguns fatores. O primeiro destes, e potencialmente o que faz a fama do jogo, é uma extrema dificuldade. A jogabilidade ”sem perdão”e a falta de recursos para cura ou instante seguros tornou o jogo famoso em curto espaço de tempo. O outro fator é uma combinação de graficos e musica que traz um tema sombrio ao jogo. Tentamos trazer ambos fatores ao jogo: Um clima ora sombrio ora de tensão, com uma dificuldade grande e escassez intencional de recursos.
Touhou: Touhou é um jogo não-comercial feito por programadores apenas por diversão.

Trata-se de um jogo plataforma de tiro, similar ao classico Space Invaders, com extremas diferenças. Primeiro, o movimento é livre para qualuqer lado em direções no eixo X e Y. Além disso, os inimigos se movem independentemente, os tiros são mirados no jogador, e com frequencia eles são tiros multiplos, lotando a tela de tiros. Tentamos trazer a quantidade enorme de tiros em tela do Touhou, para nosso jogo.

### 2.2. Adaptação

Em se tratando de visão computacional, é necessário diferenciar o sistema em relação ao Dragon Quest.
Primeiro, observa-se que no jogo original, utiliza-se como controle o Wii Mote e o Nun-chuck, controles centrados por infravermelho em um aparelho captor de alta performance. isto permite além de uma precisão extrema, a habilidade de, apenas com botões pressionados, acionar a espada ou escudo. Com a visão computacional, entretanto não há controles - isto demanda certa adaptação.
Primeiro, foi personalizado a lógica de acerto. Com uma subtração de fundo, cria-se um hitbox igual ao corpo da pessoa. Maioria dos ataques acerta o jogador apenas se desferido sobre aquela hitbox. caso contrário, o ataque ”erra”, permitindo que o jogador desvie.
Além disso, não há uma forma adequada de controle para balancear escudo e espada, originalmente restrito a apenas um ligado em tela. Por isso, o tamanho do escudo é reelevantemente menor que em Dragon Quest, e o corte da espada também não pode acertar sobre o escudo.
Também não há uma forma de pause ou inventário por limitações de controle. Portanto, substitui-se magias de cura ou uso de itens, por algo diferente. Em geral a escolha obvia para visão computacional é a de monstros derrubarem ”itens”de vida adquiriveis ao toque. Entretanto, esta escolha complicaria muito a usabilidade, gerando três focos distintos em uma mesma tela ao jogador.
Por isso, optou-se pela estratégia de ganho de vida no acerto. A espada do jogador seria uma espada mágica, que drena a força vital dos inimigos acertados, lentamente recuperando vida baseado em dano total.

### 2.3. Referencias completas

#### 2.3.1. Fase 1

A fase 1, ”Beach Side Woods”não faz referencia a nada. O chefe da fase 1, Old Sorcerer Marlingone é uma referencia ao chefe do PI6 de 2013 do BCC-Senac.

#### 2.3.2. Fase 2

A fase 2, ”Ruined Village”não faz referencia a nada. O chefe da fase 2, Troll Chieftain Huehueda faz referencia a algo desconhecido.

#### 2.3.3. Fase 3

A fase 3, Frozen Lake, faz referencia à fase ”Frozen Lake”, de Touhou 6. O chefe da fase 3, ice Fairy Cirno faz referencia à chefe da Fase Frozen Lake de Touhou 6, Cirno.

#### 2.3.4. Fase 4

A fase 4, Drangleic Castle, faz referencia a um castelo de Dark Souls II. O chefe da fase 4, Ancient Dragon, faz referencia ao chefe de Dark Souls da fase ”Aldia’s Keep”.

#### 2.3.5. Fase 5

A fase 5, Old Dwarven Mines, faz referencia a Moria, das obras de J.R.R.Tolkien. O chefe da fase 5, Fire Demon Balrog, faz referencia ao Balrog, do universo de J.R.R.Tolkien.

#### 2.3.6. Fase 6

A fase 6, Undersea Palace, faz referencia à penultima (ou ultima) fase de Chrono Trigger. O chefe final do jogo faz referencia a lavos, ultimo chefe de Chrono Trigger.

## 3. Desenvolvimento

## 4. O Código

Para fazer o uso da visão computacional, para controlar a espada e o escudo, foi escolhido o método de identificação por cor e quanto à identificação e segmentação entre jogador e fundo(background), foi utilizado a técnica de subtração de fundo.

### 4.1. Identificação de Espada e Escudo

Ao utilizar a separação por cor com o sistema RGB(padrão da câmera) para tentar identificar a espada pela cor vermelha, notou-se que nem sempre os resultados saiam como o esperado, pois quando havia muita variação de luz, os resultados saiam com alguma distorção, já que essa diferença de iluminação afeta todos os parâmetros de cor de dado pixel. Era preciso parametrizar a quantidade de luz. Ao testar a identificação por cor, antes convertendo os dados de entrada da câmera para o sistema HSV(Hue, Saturation, Value) através da função rgbToHsv. Com isso, foi possível utilizar uma fonte de luz vermelha como espada com precisão satisfatória. Aproveitando a conversão, são checados os parâmetros para a identificação do escudo, de cor verde.
Tanto para espada quanto escudo, calcula-se o ponto médio dos pixels identificados de cada um, e a partir de uma quantidade mínima de pixels identificados no frame, esses dados são utilizados para determinar as posições atuais da espada e do escudo.
Para identificar e guardar o rastro dos ataques, dado um certo intervalo mínimo de tempo, a posição da espada é armazenada numa fila, cujos pontos são processados em pares pela função drawAtk para determinar as posições inicial e final de cada execução da função al_draw_line. A cada vez que a função drawAtk é chamada toda a fila é percorrida, pois para os pares de pontos que foram adicionados antes, aplica-se um efeito de *fade-out* ao

reduzir o valor de alpha da cor da linha desenhada por al_draw_line. O ponto mais antigo da fila sempre é removido toda vez que a contagem de elementos for maior que o limite determinado (no caso, 10 elementos).
O caso do escudo é mais simples: a posição determina o ponto central da imagem que representará o escudo.

### 4.2. Separação de fundo

Toda vez que o jogo é executado, alguns segundos são reservados à obtenção do fundo(de preferência estático), sem o jogador. Algumas imagens do fundo são convertidas para HSV. A média para cada pixel dessas imagens são armazenadas numa matriz, a qual será comparado cada frame para determinar se dado pixel pertence ao fundo ou ao jogador. Os pixels identificados como sendo do jogador são armazenados em uma matriz com uma cor azulada, da qual é obtida um ALLEGRO_IMAGE, as partes que não forem dessa cor são transformadas em transparência através da função al_convert_mask_to_alpha e desenhadas na tela. O parâmetro utilizado atualmente no jogo para diferenciar jogador e fundo é verificar se a diferença de luminosidade(value) entre pixel verificado no momento e o pixel de mesma posição na matriz de background é maior que 25%.

### 4.3. Código do Jogo

## 5. Resultados

## 6. Considerações Finais

## Referências