	<p>Centro Tecnológico Departamento de Informática</p>
<p>Disciplina: Computação Gráfica</p>	<p>Código: INF09282 e INF09284</p>
<p>Prof. Thiago Oliveira dos Santos</p>	

## Trabalho 2D

### 1 Introdução

Esse trabalho tem como objetivo avaliar o conhecimento dos alunos em relação computação gráfica interativa 2D.

Para isso, o aluno deverá implementar um **jogo de rolagem**, ou seja, que visualiza o jogador e a arena pela lateral. O jogo será composto por um jogador e uma arena com alguns elementos (obstáculos e oponentes) que interagirão com o jogador. O personagem do jogador será controlado pelo teclado e pelo mouse, e poderá pular e atirar. **O objetivo do jogador é atravessar a arena e chegar ao outro lado (iniciando na esquerda e finalizando na direita) sem ser atingido.** O trabalho deverá ser implementado em C++ (ou C) usando as bibliotecas gráficas OpenGL e GLUT (freeglut).

### 2 Especificação das Funcionalidades

Ao rodar, o programa deverá ler e interpretar os elementos da arena do arquivo do tipo SVG informado pela linha de comando. A arena será composta por uma série de elementos (ver Figura 1): uma arena representada sempre por um retângulo azul; um círculo verde representando sempre o personagem do jogador e círculos vermelhos representando sempre os oponentes (controlados pelo computador). Um arquivo SVG será fornecido como exemplo juntamente com a descrição do trabalho, porém é responsabilidade do aluno testar outros arquivos com configurações (posições e tamanhos) diferentes para os elementos. **A leitura do SVG poderá ser feita utilizando-se um parser para XML. Sugiro utilizar a Tinyxml que é simples e pode ser enviada juntamente com o código para ser compilada. Use o Inkscape para visualizar a imagem da arena.**

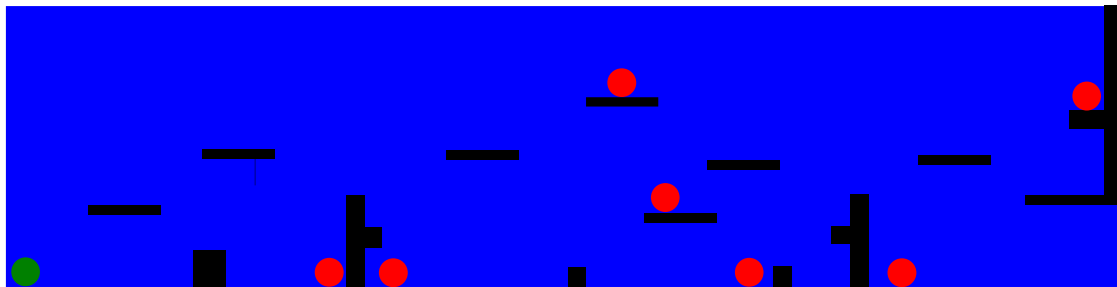



Figura 1: Arquivo SVG representando a arena em azul, obstáculos em preto, jogador em verde e oponentes em vermelho.

#### Desenho

Os elementos lidos do arquivo SVG vão servir somente para desenhar a configuração inicial do jogo e não para desenhar os personagens completos, portanto ele conterá somente os elementos e cores descritos anteriormente e seguindo aquela definição. **A arena será retangular sendo a dimensão horizontal sempre a maior. A janela de visualização será quadra (com altura e largura iguais a menor dimensão da arena) e será exibida em uma janela de 500x500 pixel do sistema operacional. Ela verá apenas uma porção da parte horizontal da arena. O foco da janela de visualização será o jogador, portanto o centro do eixo horizontal da janela estará sempre alinhado com a coordenada horizontal do jogador (ver Figura 2).** Não é necessário tratar o *resize*.

	Centro Tecnológico Departamento de Informática	
Disciplina: Computação Gráfica		Código: INF09282 e INF09284
Prof. Thiago Oliveira dos Santos		

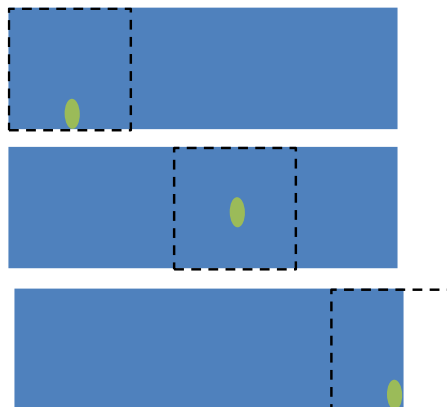


Figura 2: Exemplo da janela de visualização (quadrado pontilhado) acompanhando o jogador (ilustrado pelo elemento verde) ao longo da arena (retângulo azul). O centro da base da janela está alinhado com a coordenada horizontal do jogador.

Os círculos verde e vermelho servirão somente para saber onde e com qual tamanho desenhar respectivamente os personagens do jogador e dos oponentes (utilizar as cores do círculo para diferenciar os personagens ao desenhá-los). Mais especificamente, os círculos representarão a posição inicial dos personagens e estarão sempre dentro da arena. O diâmetro do círculo definirá a altura dos personagens. Cada personagem será formado por uma cabeça, um tronco, um braço e duas pernas com articulação de quadril e de joelho. Ver um exemplo do jogador na Figura 3. O personagem deve ser escalado proporcionalmente de acordo com a altura definida pelo seu círculo inicial.

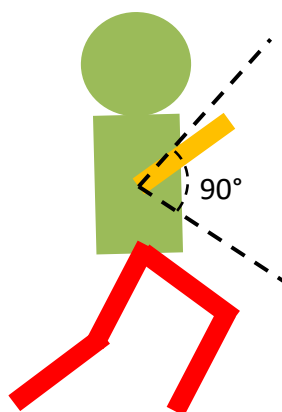



Figura 3: Ilustração do personagem composto necessariamente por uma cabeça (em verde), um tronco (em verde), um braço (em amarelo), duas pernas (em vermelho). O braço pode girar 90 graus, considerando 45 para cima e 45 para baixo. O personagem pode variar ligeiramente de acordo com a criatividade do aluno. Contudo, os elementos e juntas descritos nesta figura devem estar presentes.

#### Andar

O controle do movimento de andar será feito com teclas *a* e *d* do teclado. As teclas *a* e *d* servirão para mover o jogador para esquerda e para direita da arena respectivamente. Ao andar, as pernas deverão se mover de forma a representar o movimento humano, considerando as duas articulações (de quadril e de joelho). A velocidade de movimento do jogador deverá ser calibrada para permitir uma boa jogabilidade independentemente do tipo de máquina.

#### Pulo

O jogador deverá ser capaz de pular, ou seja, ao apertar o botão direito do mouse em qualquer lugar da arena, ele deverá pular. Ele não deverá pular se já estiver executando um pulo. O pulo deverá ser de no

	<p>Centro Tecnológico Departamento de Informática</p>
<p>Disciplina: Computação Gráfica</p>	<p>Código: INF09282 e INF09284</p>
<p>Prof. Thiago Oliveira dos Santos</p>	

máximo 3 vezes a altura do personagem e se o botão de pular for liberado antes de atingir a altura máxima, o jogador deverá começar a descer. Isso permitirá controlar a altura do pulo com o botão. O pulo permitirá o personagem subir nos obstáculos. Uma vez em cima do obstáculo, o jogador deverá se movimentar normalmente, pular ou cair ao sair dele. O pulo deve durar aproximadamente 4 segundos, quando feito na altura máxima, 2 para subir e 2 para descer. O jogador ainda poderá se movimentar lateralmente enquanto estiver pulando. Os valores de tempo dos pulos foram estimados aqui, se precisarmos ajustar, podemos conversar sobre os melhores valores.

#### *Colisão*

Os personagens não deverão sair da arena (o tronco não deve sair) e não deverão ocupar um mesmo espaço. Para não ocupar o mesmo espaço, a colisão deverá ser calculada considerando um retângulo imaginário da largura do tronco e da altura do personagem. Perceba que o retângulo imaginário não deve ser desenhado. A colisão com os obstáculos e com tiro também deverão ser tratadas pelo mesmo retângulo imaginário do jogador, inclusive para permitir respectivamente que o jogador ande em cima dos obstáculos e morra ao ser atingido. Caso o jogador pule e caia na cabeça de um oponente, esse deverá ser tratado como uma plataforma (ou seja, o jogador poderá andar na cabeça do oponente).

#### *Atirar*

O jogador poderá atirar, sendo um tiro disparado a cada clique do botão esquerdo do mouse. O braço do jogador será controlado pelo movimento vertical do mouse. Ao mover para cima, o braço girará para cima, sendo análogo quando mover para baixo. O movimento de giro do braço deverá obedecer aos limites definidos na Figura 3. A sensibilidade para ir de um limite ao outro pode ser ajustada para um movimento do mouse com a mão. Isto é, calibre o movimento para evitar ter que mover o mouse várias vezes para baixo até chegar ao limite ou não conseguir parar com o braço nas posições intermediárias. O jogador também poderá atirar enquanto estiver pulando. O tiro deverá sumir ao atingir um obstáculo ou sair da arena e deverá matar (fazer sumir) ao atingir um personagem. A velocidade do tiro deverá ser duas vezes a do jogador.

#### *Movimento do oponente*

O oponente deverá ter algum movimento “aleatório” para atacar o jogador. Ele deverá atirar de tempos em tempos em direção ao jogador e se mover de um lado para o outro na plataforma onde estiver (sem cair), porém respeitando as ações de movimento estabelecidas para andar e atirar (por exemplo, tratar as colisões).

#### *Fim de jogo*

O jogo acaba quando o jogador é atingido (neste caso, perdendo) ou quando ele chega no final da arena a direita (neste caso, ganhando). Uma mensagem deverá ser exibida no centro da tela indicando se o jogador ganhou ou perdeu. Ao teclar a tecla *r* o jogo deve recomeçar.

#### *Geral*


O aluno deverá tratar os movimentos para funcionar de forma equivalente em máquinas diferentes (ou seja, tratar o tempo decorrido entre chamadas *idle*), conforme visto em sala. Ele deverá também utilizar os conceitos de *double buffer* e variável de estado das teclas para interação com teclado (usando o evento *KeyUp* como visto em aula). A utilização de conceitos de modularização (e.g. usando classes para representar os objetos da cena) facilitará a implementação do trabalho seguinte.

## 3 Regras Gerais

O trabalho deverá ser feito individualmente. Trabalhos identificados como fraudulentos serão punidos com nota zero e os envolvidos poderão ter que responder processo na universidade. Casos típicos de fraude incluem, mas não se restringem à cópia de trabalhos, ou parte deles, assim como trabalhos feitos por terceiros.

### 3.1 Entrega do Trabalho

O código deverá ser entregue pelo Google Classroom dentro do prazo definido. Trabalhos entregues após a data estabelecida não serão avaliados.

	<p>Centro Tecnológico Departamento de Informática</p>
<p>Disciplina: Computação Gráfica</p>	<p>Código: INF09282 e INF09284</p>
<p>Prof. Thiago Oliveira dos Santos</p>	


A entrega do trabalho deverá seguir estritamente as regras a seguir. O não cumprimento **inviabilizará a correção do trabalho**.

- **Arquivo zippado** (com o nome do autor, ex. **FulanoDaSilva.zip**) contendo todos os arquivos necessários para a compilação do trabalho;
- **Não enviar** arquivos já compilados, inclusive bibliotecas!
- O arquivo zip deverá necessariamente conter um **makefile** que implemente as seguintes diretivas **"make clean"** para limpar arquivos já compilados, **"make all"** para compilar e gerar o executável. O executável deverá ser chamado **trabalhocg**.

Lembre-se que a localização do arquivo da arena será passada via linha de comando e, portanto, não se deve assumir que haverá um arquivo desses na pasta do executável. Seja cuidadoso ao testar o seu programa, isto é, não teste com o arquivo no diretório do programa, pois você pode esquecer de testá-lo em outro lugar posteriormente.

## 3.2 Apresentação do Trabalho

O trabalho terá 30 minutos para ser apresentado para a turma. A apresentação será feita para a turma no laboratório. As apresentações ocorrerão no horário da aula e em uma data posterior à de entrega, conforme cronograma das aulas. Durante o tempo de apresentação, o aluno deverá mostrar e testar todas as funcionalidades requeridas do trabalho. O trabalho (arquivos) a ser utilizado na apresentação deverá ser o mesmo enviado para o professor, e será fornecido pelo professor na hora da apresentação. A ordem de apresentação será sorteada durante a aula, portanto, todos os alunos devem estar preparados para apresentar o trabalho durante o período de apresentações. Os alunos devem estar preparados para responder possíveis perguntas sobre o trabalho. Prepare-se para fazer a apresentação dentro do seu tempo. **Pontos só serão dados para funcionalidades apresentadas**, isto é, a audiência deverá ser capaz de ver e perceber o resultado produzido pela funcionalidade implementada no jogo. Cabe aos alunos, portanto, criarem atalhos (para habilitar e desabilitar funcionalidades, por exemplo, movimento do oponente) no trabalho para facilitar a apresentação das funcionalidades.

	Centro Tecnológico Departamento de Informática	
Disciplina: Computação Gráfica		Código: INF09282 e INF09284
Prof. Thiago Oliveira dos Santos		

### 3.3 Pontuação

A avaliação do trabalho seguirá a tabela abaixo, com bug sendo descontados caso a caso.

Itens	Sub-Itens	Feito	Observações	Pontos	Nota
Desenho	Arena e personagens			1,0	
Andar	Andar corretamente			2,0	
Pular	Pulo e controle do botão			2,0	
	Subir nos obstáculos				
Colisão	Não sair da arena			0,5	
	Não invadir o oponente			0,5	
	Andar e cair dos obstáculos			1,0	
Atirar	Tiro certo			0,75	
	Movimento do braço			0,75	
Fim de Jogo	Mensagens e reinício			0,5	
Movimento do oponente	Andar corretamente			0,5	
	Atirar corretamente			0,5	

## 4 Erratas

Qualquer alteração nas regras do trabalho será comunicada durante a aula e no portal. É de responsabilidade do aluno frequentar as aulas e manter-se atualizado.