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

## Trabalho 3D

### 1 Introdução

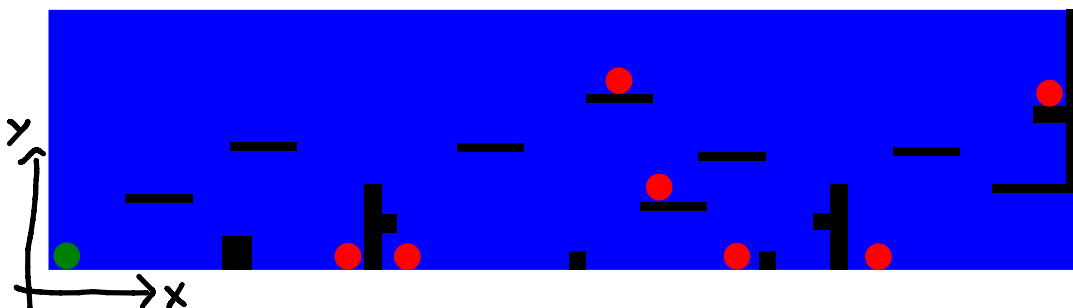
Este trabalho tem como objetivo fixar as técnicas de computação gráfica 3D adaptando o trabalho anterior, T2D, para 3 coordenadas.

O objetivo geral do jogo será parecido com o do trabalho 2D e, também, terá os mesmos elementos de jogo, porém com algumas funcionalidades específicas do ambiente 3D.

O aluno deverá implementar um programa que transforme o trabalho 2D em 3D. 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.*

Após ler as informações do arquivo de configurações (equivalentes ao T2D), o programa deverá carregar os elementos da arena do arquivo do tipo SVG respectivo e colocar um jogador verde ao invés de um círculo verde, um jogador vermelho ao invés de um círculo vermelho, além da arena como definidos no trabalho anterior. A janela de visualização deverá ter 500x500 pixels. O jogo deverá ter as mesmas funcionalidades do T2D, exceto as que forem redefinidas abaixo.

#### Arena

Assim como no trabalho 2D, o programa deverá criar uma arena virtual, porém desta vez em 3D. **O plano x-y (ver Figura 1) terá informações idênticas às lidas do arquivo "svg" (assim como o trabalho anterior).** A largura da arena, em z, deverá ser 1/2 vezes a altura da arena, em y. O chão da arena será representado no plano z-x. Os obstáculos deverão se estender de um lado ao outro da arena, ou seja, ao longo de z. Eles formarão muros ou plataformas dentro da arena. Os personagens deverão ser iniciados em sua coordenada x e y (conforme representado na Figura 1) seguindo as especificadas do svg, sendo a coordenada z um valor aleatório que o coloque dentro da arena.

	<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>	

### *Jogador*

O jogador deverá ter cabeça, braços, corpo e pernas representados em 3D. Utilize a criatividade para construir o jogador! O sistema de colisão será parecido com o do trabalho anterior, porém agora representado por um cilindro da altura do jogador e com raio suficiente para conter tronco do jogador. A colisão deverá ser calculada considerando cilindro imaginário envolvendo o jogador. Perceba que o cilindro é “virtual”, ele serve apenas para calcular a colisão e não deve ser mostrado na tela. Neste trabalho, a arena terá um formato e paralelepípedo e o jogador não poderá sair da arena. Portanto, a colisão também deverá ser calculada para as paredes da arena.

### *Andar*

O controle do movimento de andar será feito com teclas *a*, *d*, *w* e *s* do teclado. As teclas *a* e *d* servirão respectivamente para girar o jogador no sentido anti-horário e horário quando visto de cima da arena. As teclas *a* e *d* servirão respectivamente para mover o jogador para frente e para trás. Ao andar para frente e para trás, as pernas deverão se mover de forma a representar o movimento humano, considerando as duas articulações (de quadril e de joelho), como no trabalho anterior, porém agora em 3D. A velocidade de movimento do jogador deverá ser calibrada para permitir uma boa jogabilidade independente do tipo de máquina.

### *Pulo*

O jogador deverá ser capaz de pular, conforme especificado no trabalho anterior, porém utilizando a barra de espaço. Ele poderá apenas mover-se para frente e para trás quando estiver pulando.

### *Atirar*

O jogador poderá atirar assim como no trabalho anterior, porém agora no espaço 3D, ou seja, o braço pode fazer o movimento de um cone imaginário e a bala deve seguir na direção em que a arma estiver apontando. Os controles da arma (braço) seguem o do trabalho anterior.

### *Movimento do Oponente*

O jogador oponente deverá se movimentar como no trabalho anterior, ou seja, de forma “aleatória” e atacando o jogador de tempos em tempos.

### *Jogo em geral*


O jogo em geral deverá seguir as funcionalidades do T2D, e.g., ganhar, perder, mostrar mensagens, reiniciar, etc.

### *Aparência do Jogo*

Deverão ser utilizados conceitos de iluminação e textura. O jogo deverá conter pelo menos um modelo de luz na arena (pontual ou direcional). Além disso, o jogo deverá ter um modo noturno (fazer a troca de modos com a tecla “n”) em que todas as luzes da arena são apagadas, sendo ligado somente uma lanterna na arma (braço) do jogador apontando na mesma direção da arma. A lanterna será representada por uma iluminação spot que deverá ser vista na arena quando iluminada. As paredes, o chão e o teto da arena deverão ser texturizados, assim como o jogador. O aluno está livre para escolher as texturas e utilizar luzes adicionais. Use a criatividade!

### *Câmeras*

O jogo deverá implementar 3 tipos de visões que poderão ser trocadas com os botões numéricos do teclado (1, 2 e 3). O botão 1 (opção padrão) deverá acionar uma câmera perspectiva posicionada no olho do jogador e olhando para frente. O botão 2 deverá acionar uma câmera na arma do jogador, ou seja, como se fosse uma mira. Ela será uma câmera perspectiva posicionada em cima da arma, olhando na direção do tiro (paralelamente ao cano da arma) e com o up perpendicular a arma. Com essa visão, seria possível ver parte da arma, assim como o que estiver a frente dela. Opcionalmente, você poderá entrar temporariamente nesta câmera (a partir de qualquer outra) enquanto o botão direito do mouse estiver pressionado. O botão 3 deverá acionar uma câmera perspectiva posicionada inicialmente atrás do jogador (a uma distância grande suficiente para ver todo o jogador por uma terceira pessoa) e a uma altura superior à do jogador, e olhando para o centro do jogador (up apontando para o teto). Essa última câmera poderá ser rotacionada (360 graus em torno do jogador e  $\pm 60$  graus para cima e para baixo do centro do

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

jogado) quando pressionada a tecla *x*, resultando em um movimento esférico em volta do jogador. As teclas de + e – controlarão o seu zoom.

#### *Bônus 1 – Minimapa*

Mapa de posição, dos jogadores. Utilizar uma câmera ortogonal para desenhar um minimapa da arena descrevendo a sua posição (verde) e a posição dos oponentes (vermelho). Este mapa representará a arena vista de lado, como no trabalho 2D. Ele deve ser transparente para não ofuscar a visão original do jogo (ou seja, utilizar apenas linhas para representar a arena). Utilizar o mesmo conceito da impressão de texto no canto da tela. O mapa deve ficar fixo no canto inferior direito e ocupar 1/4 da largura da janela (ou seja, 1/16 da área).

#### *Bônus 2 – Modelos Avançados*

Utilizar modelos avançados de jogador e suas partes (ver exemplos na Figura 2). O aluno está livre para utilizar modelos 3D e suas partes feitos no Blender ou baixados da internet, ou editado com ambos. Um exemplo de site com modelos é o do mixamo (<https://www.mixamo.com/>). A qualidade dos modelos será julgada caso a caso. Atenção, modelos muito pesados podem deixar o jogo muito lento e isso não é desejável. Escolha de forma a ter um jogo fluido. Para quem for usar modelos avançados, será permitido usar a arma acoplada ao ombro para não ter que movimentar o braço do modelo. Neste caso, as funcionalidades referentes ao braço descritas nos itens anteriores deverão ser implementadas para a arma (ou seja, controle com o mouse, atirar, etc).



*Figura 2: Exemplos de modelos avançados.*


**OBSERVAÇÕES:** O aluno poderá incluir (e deverá, se for a única maneira de mostrar uma funcionalidade) parâmetros e teclas adicionais para facilitar a apresentação do trabalho. Por exemplo, o aluno pode criar uma tecla para habilitar e desabilitar uma determinada funcionalidade, para mostrar que ela funciona. As funcionalidades só serão pontuadas se elas forem vistas durante a apresentação, isto é, falar que colocou a luz não basta, é necessário mostrar o seu efeito e explicar coerentemente. Esperar 10 minutos para ver o um personagem ser atingindo não é factível. Planeje a sua apresentação. O aluno deverá utilizar os mesmos conceitos já exigidos no trabalho anterior. Um arquivo exemplo será distribuído juntamente com essa especificação. **Inclua um README.txt explicando os atalhos e funcionalidades adicionais.**

## **3 Regras Gerais**

O trabalho poderá ser feito em dupla, exceto para os alunos das pós-graduação. Trabalhos identificados como fraudulentos serão punidos com nota zero. Casos típicos de fraude incluem, mas não se restringem às cópias de trabalhos (independente de quem fez e quem forneceu), ou de parte dele, assim como trabalhos feitos por terceiros (inclusive por programas de geração de código). Cada membro da dupla deverá obrigatoriamente conhecer todo o conteúdo e código do trabalho.

### **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.

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

A entrega do trabalho deverá seguir estritamente as regras a seguir. O não cumprimento **inviabilizará a correção do trabalho** que, por sua vez, receberá nota zero.

- Arquivo zippado (com o nome do(s) autor(es), ex. FulanoDaSilva.zip ou **FulanoDaSilva\_CiclanoSantos.zip**) contendo todos os arquivos necessários para a compilação do trabalho (não assumir que haverá bibliotecas adicionais instaladas);
- **Não enviar arquivos já compilados, nem mesmo de 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.

## 4 Pontuação


O trabalho será pontuado conforme a tabela dada na última folha desse documento e resumida abaixo. Bugs serão descontados caso a caso. Observe que existem duas funções bônus no trabalho, ou seja, 2 pontos extras. Os pontos dessas questões bônus serão utilizados para completar a nota desse trabalho ou do trabalho anterior que não tenham atingido a nota máxima 10.

Funcionalidade	Pontuação
Base do jogo	2
Andar	1
Pulo	1
Atirar	1
Aparência do jogo (iluminação e textura)	2
Câmeras	3
Bônus 1 – minimapa	1
Bônus 2 – modelos avançados	1


### 4.1 Apresentação do Trabalho

O grupo terá **20 minutos** para apresentar seu trabalho para a turma. A apresentação será feita no laboratório, sendo que todos os componentes do grupo devem estar preparados para apresentar o trabalho. As apresentações ocorrerão no horário da aula e em uma data posterior à de entrega. Durante o tempo de apresentação, o aluno deverá mostrar e testar (mostrando o funcionamento) 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 (20 min.). **Pontos só serão obtidos por 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, criar atalhos no trabalho para facilitar a apresentação das funcionalidades.

## 5 Erratas

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

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

	<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>	

**Tabela de pontos**

Nome do aluno:	
Nome do aluno:	

Itens	Sub-Itens	Feito	Observações	Pontos	Nota
Base do jogo	Movimentos do oponente			0,5	
	Colisões			0,5	
	Jogo (morrer, ganhar, mensagem, etc.)			1,0	
Jogador 3D	Andar			1,0	
	Pulo			1,0	
	Atirar			1,0	
Aparência do jogo	Iluminação 1			0,5	
	Holofote			0,5	
	Textura			1,0	
Câmeras	Câmera 1			1,0	
	Câmera 2			1,0	
	Câmera 3			1,0	
Bônus 1	Minimapa			1,0	
Bônus 2	Modelos avançados			1,0	