Disciplina: Computação e Representação Gráfica

Prof. Me. Pedro Henrique Vieira de Oliveira Azevedo

***Divertimento Computacional 3***

# Objetivo e Instruções Gerais

Esse trabalho tem como objetivo integrar os conceitos dos trabalhos anteriores em um pequeno jogo. O jogo terá como objetivo completar uma volta na pista no sentido anti-horário sem tomar um tiro de seus adversários. Para isso você poderá matar seus adversários com tiro. O joga acaba quando você completa 5 voltas, ou quando você morre.

O aluno deverá implementar um programa que incremente o trabalho curto 3 (TC3). Nesse trabalho, os componentes da arena serão dinâmicos, ou seja, os inimigos deverão se mover de um lado para o outro de forma automática. Ganha-se o jogo caso o carro do jogador consiga completar cinco voltas completas na pista. O trabalho deverá ser implementado em C++ (ou C) usando as bibliotecas gráficas OpenGL e GLUT (freeglut).

# Especificação das Funcionalidades

Ao rodar, o programa deverá ler, de um arquivo de configurações (denominado “config.xml”), as configurações necessárias para suas tarefas. O arquivo de configurações deverá estar no formato xml e será fornecido juntamente com a especificação do trabalho. A localização do arquivo “config.xml” será fornecida pela linha de comando ao chamar o programa. Por exemplo, se o arquivo estiver dentro de uma pasta chamada “Test1” localizada na raiz, basta chamar o programa com “/Test1/” como argumento (outros exemplos de caminhos possíveis “../Test1/”, “../../Test1/”, etc.). As informações contidas nesse arquivo servirão para ler o arquivo SVG contendo as informações da arena.

Além das tags já especificadas no trabalho curto 2 e 3, o arquivo de configurações deverá conter uma sub-tag específica para descrever os parâmetros inimigos, denominada <inimigo>. A tag <inimigo> terá, além de um atributo de velocidade de movimentação e velocidade do tiro, um atributo “freqTiro” para determinar a frequência dos tiros dos adversários, isto é, quantos tiros os adversários darão por instante de tempo. Ele será dado em número de tiros por milissegundos. Percebam que os valores dados como exemplo foram escolhidos aleatoriamente e portanto podem não representar valores ótimos para teste.

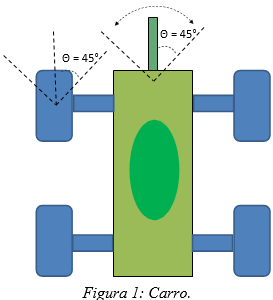
Abaixo temos um exemplo do arquivo “config.xml”:

|  |
| --- |
| <aplicacao>    <arquivoDaArena nome="arena" tipo="svg" caminho="../../Test1/"></arquivoDaArena>  <carro velTiro="0.2"></carro>  <inimigo freqTiro=“0.0001” velInimigo=”0.1” velTiro=”0.3”></inimigo>    </aplicacao> |

Após ler as informações do arquivo de configurações, o programa deverá carregar os elementos da arena do arquivo do tipo SVG respectivo e desenha-las na tela (assim como feito no trabalho curto 2), exceto o círculo do personagem do jogador que terá o carro da Figura 1, ao invés de um círculo.

**O Carro**

O carro deverá ser capaz de se movimentar pela arena e detectar as colisões já implementadas no trabalho anterior. Ele iniciará na posição definida pelo centro do círculo verde na arena e terá o tamanho aproximado do círculo que o representa, ou seja, o carro deverá estar contido no círculo. O carro será composto por 4 rodas com seus respectivos eixos, um corpo, e um canhão, ver Figura 1 como exemplo.



*Tiro*

Todos os inimigos e também o carro, poderão atirar. O carro do jogador terá o tiro controlado conforme o TC3, e os inimigos deverão atirar de tempos em tempos atirar (utilizar o parâmetro “freqTiro”). Para saber se um tiro acertou um carro, basta verificar se o círculo envolvendo o carro toca o círculo que define o tiro. Perceba que esse círculo é “virtual”, ele serve apenas para calcular a colisão e não deve ser mostrado na tela. Tiro amigo não mata! Ou seja, os carros inimigos não podem se matar. O tiro do inimigo deverá SEMPRE mirar no carro do jogador!

*Morte*

Caso um tiro do jogador acerte o inimigo, o mesmo deverá morrer, ou seja, sumir da tela. Depois de um período de tempo a ser definido pelo programador, o mesmo deverá ressurgir e voltar a atirar.

*Finalização do Jogo*

No final do jogo, uma mensagem deverá ser impressa na tela dizendo se você ganhou ou perdeu. O jogador ganha se completar cinco voltas, e perde se levar um tiro.

# Regras Gerais

O trabalho deverá ser feito em duplas. Essas duplas serão as mesmas para todos os trabalhos computacionais até o final do semestre. Trabalhos identificados como fraudulentos serão punidos com nota zero. Casos típicos de fraude incluem, mas não se restringem à cópias de trabalhos, ou parte dele, assim como trabalhos feitos por terceiros. Caso seja necessário confirmar o conhecimento do aluno a respeito do código entregue, o professor poderá pedir ao aluno para apresentar o trabalho oralmente em um momento posterior. A nota da apresentação servirá para ponderar a nota obtida no trabalho.

## 3.1. Entrega do Trabalho

O código deverá ser enviado para o seguinte email:

[**pedro.hvo.azevedo@gmail.com**](mailto:pedro.hvo.azevedo@gmail.com)

Até:

**23:59 do dia 16 de Julho de 2017**

Trabalhos entregues após a data estabelecida não serão corrigidos! Lembre-se que a internet pode cair, o cachorro pode comer seu trabalho, seu trabalho pode molhar e outros acidentes mais. Cuidado com envios em cima da hora!

A entrega do trabalho deverá seguir estritamente as regras a seguir. O não cumprimento acarretará na **não correção do trabalho** e respectivamente na atribuição da nota zero.

* Assunto da mensagem: [CG-2017-1] <tipo do trabalho>. Onde, <tipo do trabalho> pode ser TC1, TC2, TC3 e representa respectivamente trabalho computacional 1, 2, 3, etc , ou TF para o trabalho final;
* Anexo da mensagem: arquivo zipado (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 diretório 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 config.xml 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. Pontuação

O trabalho será pontuado conforme a tabela abaixo. Bugs serão descontados caso a caso.

|  |  |
| --- | --- |
| **Funcionalidade** | **Nota** |
| Funcionalidades do TC3 | 2 |
| Tiro dos Inimigos | 3 |
| Morte e Ressurgimento dos Inimigos | 3 |
| Mensagem de Finalização | 2 |

# Erratas

Qualquer alteração nas regras do trabalho serão comunicadas em sala. É de responsabilidade do aluno frequentar as aulas e manter-se atualizado!