Universidade Federal de Santa Maria
Curso de Ciência da Computação
Disciplina: Computação Gráfica
Primeiro Semestre de 2017
Prof. Cesar Tadeu Pozzer
Data: 24/04/2017

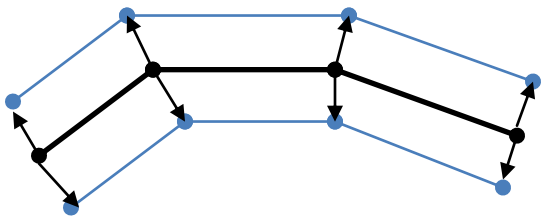
Trabalho 2- Transformações Geométricas e Curvas

Implemente um veículo que se locomove em um espaço 2D sobre uma estrada definida pelo usuário, simulando um jogo de corrida de veículos.

Ferramentas: Linguagem C++ com a API Qt (Versão 5.7.1), ferramenta QT Creator, utilizando as funções de desenho da Canvas2D – demo CanvasQT, compilando com MinGW (disponível na versão 16.01 da IDE Code::Blocks). https://www.qt.io/developers/. **Não podem ser utilizadas bibliotecas auxiliares.** Não pode ser usada a API OpenGL.

A estrada é composta por vários pontos de controle interpolados via alguma curva paramétrica (Bezier ou B-Spline). Os pontos de controle devem ser definidos pelo usuário, como o uso do mouse. À medida que o usuário insere os pontos de controle, deve-se ir desenhando o grafo de controle correspondente. Quando o usuário acabar a edição, a estrada curva deve ser gerada.

A estrada deve ter uma largura definida pelo usuário, e é gerada pela expansão do grafo de controle perpendicularmente em relação a sua direção, como mostrado na figura. A estrada deve formar um circuito fechado.



O veículo deve ter uma velocidade controlada, bem como direção de deslocamento. A geometria do veículo pode ser bem simples, como uma simples seta, que possa dar noção de sua direção.

O usuário pode interagir guiando o veículo com o uso das setas direcionais, e com outras duas para mudar a velocidade (e.g. “a” para acelerar e “s” para frear). Pode-se adicionar teclas para movimentos laterais. Pode-se utilizar o mouse também.

Deve-se controlar o ângulo máximo de rotação do veículo. O movimento do veículo deve ser suave e contínuo, independente do usuário estar ou não pressionando alguma tecla. Quanto maior a velocidade, menor deve ser o ângulo de giro.

Critérios que serão avaliados:

- (3 pts) Classes em C++ para definição de vetores e transformações geométricas
- (1.5 pts) Sistema do controle do veículo
- (1.5 pts) Interatividade – não usar o refresh do teclado para mover o veículo.
- (3 pts) Corretude das curvas geradas

Critérios avançados (Bônus):

- Recurso de edição em tempo real dos pontos de controle e automaticamente desenho da pista. (Até 1pt)
- Determinação se o veículo está andando em cima da pista e respectiva contagem de pontos. (Até 1pt)
- Vetores indicando as "forças" aplicadas ao veículo, como aceleração, força centrípeta, posição em relação a pista, etc. (Até 1pt)
- Adicione objetos ao longo da pista que devem ser coletados para ganhar pontuação (semelhante o jogo Temple Run). (Até 1pt)
- Definir outros veículos autônomos andando na pista para simular uma competição de carros de corrida. (Até 3pt)
- Coloque recurso para disparo de projéteis que podem tirar os adversários da pista. Tratamento de colisão de tiros. Pode-se aproximar os alvos como círculos. (Até 2pt) (dependente do anterior)
- Placar do jogo (Até 1pt) (dependente dos 2 anteriores)
- Parametrização do nível de IA dos veículos adversários (Até 1pt dependente do 5º critério)
- Etc.

O trabalho deve apresentar uma lista de instruções, explicando de forma como o usuário deve interagir com o programa. Enumere no início do código fonte (arquivo main.cpp) os quesitos que foram implementados.

Data e Formato de Entrega:

- Data: 09/05 até as 23:59.
- No email e no cabeçalho do arquivo, devem conter o nome completo e matricula do aluno. O arquivo deve ser enviado para pozzer3@gmail.com e rtrindade@inf.ufsm.br com o subject "CG T2".
- O programa deve ser enviado em um arquivo compactado **fulano.rar** (fulano = login ou nome do aluno). Dentro deste arquivo deve haver um **diretório com o mesmo nome do arquivo** e dentro deste diretório os arquivos do trabalho. Deve-se enviar fontes, projeto Qt Creator. Não envie arquivos com extensão pdb, ilk, idb, obj, etc. **Envie somente o que for necessário para compilação**. Não devem ser enviadas as libs.
- Ex: o arquivo pozzer.rar deve conter um diretório chamado pozzer, e dentro do diretório devem estar os arquivos do trabalho.

Critério de Avaliação:

- Documentação: descrever no cabeçalho de cada arquivo a ideia geral do código e detalhes específicos de partes que mereçam uma explicação – não comente por exemplo o que faz b++.
- README.txt: incluir um arquivo "README.txt" contendo informações sobre quais funcionalidades foram implementadas (requisitos e extras).

- Pontualidade: Trabalhos não entregues na data não serão avaliados e receberão nota zero.
- Legibilidade: nome de variáveis, estruturação do código.
- Clareza: facilidade de compreensão – evite códigos complexos e desnecessários. Adote a solução mais simples possível.
- Funcionalidade: o programa deve satisfazer todos os requisitos. Programas que não compilarem ou que não atenderem nenhum requisito receberão nota 0 (zero).

Você pode discutir estratégias e ajudar o colega na implementação, porém evite passar código fonte. Programas semelhantes terão a nota 0 (zero).