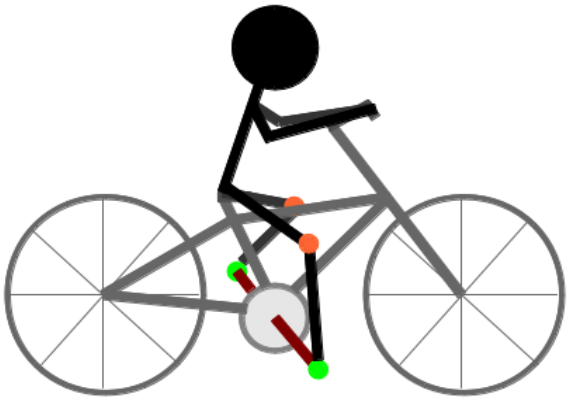
Universidade Federal de Santa Maria
Curso de Ciência da Computação
Disciplina: Computação Gráfica
Primeiro Semestre de 2013
Prof. Cesar Tadeu Pozzer
Data: 25/04/2013

Trabalho *2*- Transformações Geométricas

Descrição:

Implemente em C++, usando a API SCV v. 4.2.1 ou a Canvas 2D, um personagem simples que tem a capacidade de pedalar uma bicicleta. A imagem abaixo ilustra como este personagem pode ser.



A bicicleta não necessariamente deverá mover-se a cada pedalada do personagem, porém, deve haver os movimentos das pernas do personagem, dos pedais e das rodas da bicicleta assim como ocorre na realidade. Você deve criar estruturas de dados que facilitem as transformações geométricas e que representem cada uma das partes deste personagem e da bicicleta.

Você pode procurar na *web* vídeos e animações que ilustrem os movimentos das penas de um ciclista para auxiliá-lo no desenvolvimento do programa. Há uma série de jogos *online* que tem como base esse tipo de personagem. Você pode acessar este *link* <http://www.quero-jogar.com/bicicleta/mestre-bmx.htm> para ver um exemplo.

Caso você implemente algum adicional, o trabalho deve apresentar uma **listagem**, explicando a forma como o usuário deve interagir com o programa. Enumere no início do código fonte (arquivo main.cpp) os quesitos que foram implementados. O trabalho tem peso 9.0 se não forem feitos tópicos adicionais.

Dicas:

- Definam as coordenadas da bicicleta em relação a origem. Utilizem um vetor ou estrutura, por exemplo. A geometria pode ser bem básica.

- Quando forem desenhar a bicicleta, apliquem transformações sobre esses dados e criem um novo vetor/estrutura que vai se desenhado. Não alterem o vetor/estrutura original
- Se forem combinar transformações, concatenem numa matriz e apos apliquem essa matriz sobre os dados a serem exibidos. Se tiver muito complicado usar matrizes, façam cálculos sequenciais. O uso de matriz da bônus.

O trabalho deve ser individual. Em caso de dúvidas, entrem em contato com o monitor (Thiago Trugillo) ou comigo.

Adicionais

- Fazer com que a bicicleta movimente-se de diferentes formas (como, por exemplo, andar sobre uma roda, pular, etc.) com ou sem comandos do usuário.
- Implementar não só o personagem e a bicicleta, mas também um cenário com um chão (plano, com rampas ou irregular), *background* e obstáculos. Neste caso, faça com o que o cenário mude o comportamento do personagem como, por exemplo, haja trepidação ao passar a roda da bicicleta por um buraco.
- Use técnicas como Parallax para criar uma ideia de movimento no cenário.
- Crie uma simulação mais realista da bicicleta – com um sistema de transmissão de movimento por correia – e utilize leis da física para definir o quanto a roda é rotacionada por pedalada. Você pode procurar materiais na *internet* ou livros que ilustrem tais leis. Para uma exemplificação, você pode acessar este *link* <http://www.youtube.com/watch?v=nznd3Rqi9Q0>.
- Utilizar concatenação de matrizes de transformação
- Simular o movimento realista da perna em relação ao pedal.
- Etc.

Data e Formato de Entrega:

- Data: 23/05/2013.
- 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 thiago@inf.ufsm.br com o *subject* "CG T2". Deve-se enviar fontes e o projeto para o compilador Code::Blocks.
- 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 somente os fontes e projeto Code::Blocks (disponível na pasta userProject do SCV). **Envie somente o que for necessário para compilação**. Não deve ser enviada a lib do SCV e nem DLLs. Deve-se enviar o projeto UserProject, juntamente com os códigos fonte. Caso for usada alguma ferramenta de geração de código, os arquivos devem ser copiados para userProject\src.
- **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++.

- **pontualidade**: Trabalhos não entregues na data não serão avaliados e receberão nota zero.
- **legibilidade**: nome de variáveis, estruturação do código. O código digital a ser entregue deve ter ***3 espaços de identação*** e não deve possuir tabulações.
- **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).