Modelagem Integrada

Escola de Ciências e Tecnologia
UFRN

Projeto: Desenvolvimento de software para simulação de oscilações mecânicas

2015.2 - Turma 01 D
Docente: Sérgio Luiz Eduardo Ferreira da Silva

Discente: Alexandre José Justino de França
Matrícula: 2010037014

Natal/RN
Set/2015

Resumo

O propósito deste projeto é desenvolver um software de auxílio aos alunos do componente Modelagem Integrada, de forma a facilitar a compreensão do assunto tratado na primeira unidade do curso. Durante este relatório, serão tratados tópicos sobre as tecnologias utilizadas, o sistema massa-mola, as principais funções utilizadas no software e por fim uma lista de possíveis futuras melhorias. Concluiu-se com o projeto que ele tem potencial para cumprir o seu objetivo, de forma que o próprio desenvolvedor atingiu uma melhor compreensão sobre oscilações mecânicas através da interação com o “produto” final.

Introdução

Alguns cursos das ciências exatas apresentam conceitos, certas vezes abstratos, que podem ser difíceis de se entender e visualizar. Certos conceitos estudados a cerca da modelagem de sistemas com oscilações mecânicas podem não ser absorvidos de forma natural por alguns. O projeto proposto consiste em desenvolver um software que simule oscilações mecânicas a fim de ajudar o aluno do curso de Modelagem Integrada a melhor visualizar o efeito que a alteração das propriedades causa no sistema, em tempo real.

Tecnologias Utilizadas

Desde a chegada do *HTML5* (2014), o desenvolvimento de animações e gráficos para a internet tem crescido bastante. Isso ocorre porque o *HTML5* trouxe consigo um novo elemento, chamado "*canvas*", no qual é possível imprimir gráficos e figuras geométricas. Para manipular ou desenhar algo no "*canvas*", utiliza-se geralmente a linguagem *JavaScript*, que é interpretada e executada por um navegador de internet. Todos os navegadores atuais já possuem suporte ao *HTML5*.

A fim de escrever o código do programa, foi utilizado o ambiente de desenvolvimento integrado (IDE) "*PhpStorm*". Entretanto, pode-se utilizar qualquer editor de texto (inclusive o "*Bloco de Notas*", do *Windows*). A escolha da IDE é importante e pode facilitar a vida do desenvolvedor de várias maneiras. O *PhpStorm* possui uma licença grátis para estudantes de 1 ano.

A escolha dessas tecnologias foi motivada por curiosidade de entender como funciona a criação de uma animação utilizando os novos conceitos do *HTML5*.

O gerenciamento de versões (*VCS*) também foi levado em consideração durante o desenvolvimento do projeto. Esse conceito é muito utilizado hoje em dia e muito importante, pois gerencia de forma muito simples todas as versões do programa, permitindo que o desenvolvedor retome uma antiga versão se necessário, por exemplo. Para tal, foi utilizado o software *Git*.

Modelagem do sistema “massa-mola”

O primeiro desafio do projeto é desenvolver uma simulação para o sistema massa-mola. Necessita-se, obviamente, modelar uma equação que represente esse sistema mecânico. Após definir o modelo matemático do sistema, chegamos à seguinte Equação Diferencial Ordinária (EDO):

$$m\ddot{x} + kx = 0$$

Porém, precisamos da solução dessa EDO para que o programa possa definir a posição da massa em cada instante de tempo durante a simulação. Existem diversas técnicas disponíveis para chegar à essa resposta, porém, elas não entram no escopo desse projeto. Utilizamos, então, a ferramente online *WolframAlpha.com* para obter a equação que representa essa EDO de forma aproximada. Obtemos a seguinte solução:

Differential equation solution:

$$x(t) = c_2 \sin\left(\frac{\sqrt{k}\ t}{\sqrt{m}}\right) + c_1 \cos\left(\frac{\sqrt{k}\ t}{\sqrt{m}}\right)$$

No código do programa, essa equação é representada da seguinte maneira:

```
xp * Math.sin(Math.sqrt(kMola) * tempo / Math.sqrt(massa)) + x0 * Math.cos(Math.sqrt(kMola) * tempo / Math.sqrt(massa));
```

Onde *xp* ($\dot{X}$) e *x0* (X_0) representam a velocidade inicial e a posição inicial, respectivamente (esses valores serão definidos pelo usuário antes do início da simulação).

Com essa equação em mãos, devemos iniciar o desenvolvimento do software.

Utilização do elemento “canvas”

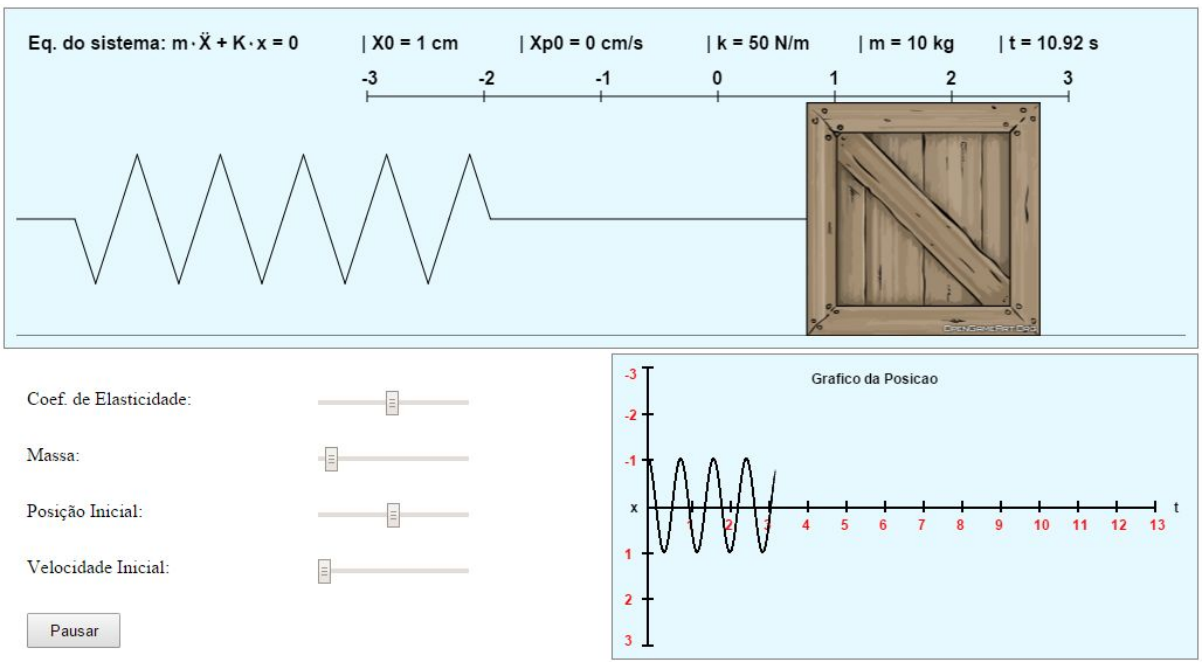
Iremos utilizar dois “*canvas*” para o nosso projeto. O primeiro para a animação da oscilação do sistema, o segundo para a criação de um gráfico posição por tempo. Precisamos, inicialmente, declarar os nossos “*canvas*” no código *HTML*:

```
<canvas id="canvasSimulacao" width="1000" height="270">
  Navegador não suporta HTML5
</canvas>
<canvas id="canvasGrafico" width="480" height="240">
  Navegador não suporta HTML5
</canvas>
```

Podemos visualizar que dentro de cada elemento, existe a seguinte frase “Navegador não suporta *HTML5*”. Essa mensagem será mostrada somente caso o usuário tente iniciar o programa utilizando um navegador antigo, que não tem suporte ao *HTML5*.

Cada canvas possui um *id* diferente, que servirá para referenciar o objeto no código JavaScript. Pode-se também verificar as dimensões de cada “*canvas*” no código, o primeiro sendo 1000x270 px (pixels) e o segundo 480x240 px. Nossa massa (caixote) possui dimensões 200x200 px.

O “*canvas*” é um elemento no qual podemos desenhar linhas, círculos, retângulos, pontos e também escrever na tela (em sua tradução literal para o Português, “*canvas*” significa “tela”, como a de um quadro de pintura). Na imagem a seguir podemos visualizar a interface gráfica do programa:



Porém, não é possível fazer esses elementos se moverem na tela. Para movimentar esses elementos e criar uma animação, precisamos escrever um programa que limpe a tela, apagando todos os elementos que foram desenhados, e desenhe-os novamente em uma posição diferente, muito próxima da anterior. Esse processo precisa ser feito muito rapidamente, de modo que os olhos humanos vejam essa transição como um movimento natural, sem perceber que o quadro está sendo apagado diversas vezes por segundo.
Para nosso projeto, isso será feito a cada 30 milisegundos (0,030 segundos), o que nos deixa com uma taxa de 33,33 frames por segundo (FPS). A escolha desse valor deve ser feita de tal forma que não seja exigido muitos recursos do computador (quanto maior o FPS, mais se exige do computador) e que ao mesmo tempo não deixe o usuário perceber que a frequência de atualização dos frames está muito baixa.

Para tal, utilizamos a seguinte linha de código:

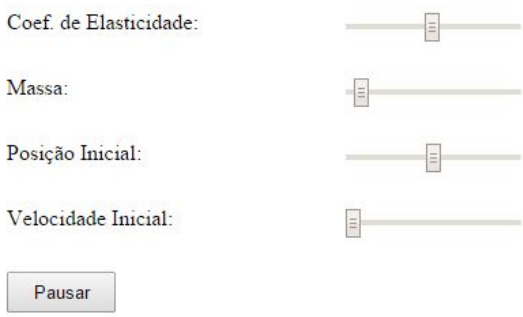
```
drawInterval = setInterval(draw, tempoDeAtualizacaoDoCanvas);
```

Onde:

```
tempoDeAtualizacaoDoCanvas = 30;
```

Onde a função "*setInterval*" é responsável por executar a função "*draw*" a cada intervalo de 30 milisegundos.

Para que o usuário possa alterar as características do sistema, foram adicionados os seguinte controles na interface:



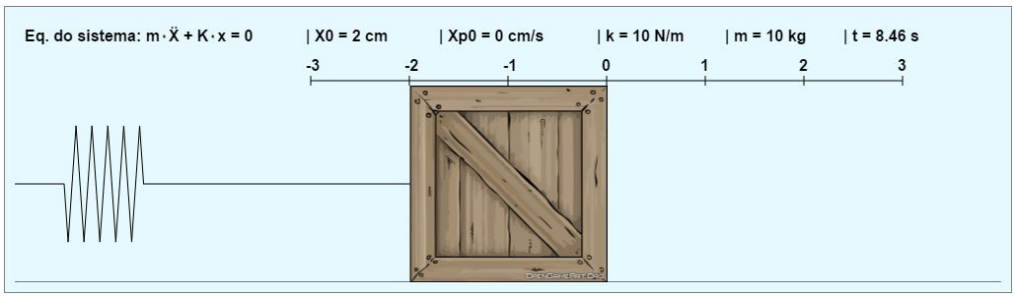
Esses controles permitem visualizar o efeito da mudança dos valores na vibração do sistema. O aluno poderá, por exemplo, visualizar que ao aumentar o coeficiente de elasticidade da mola a massa irá vibrar mais rapidamente (menor período), pois a força F_k puxa/empurra a massa com maior intensidade.

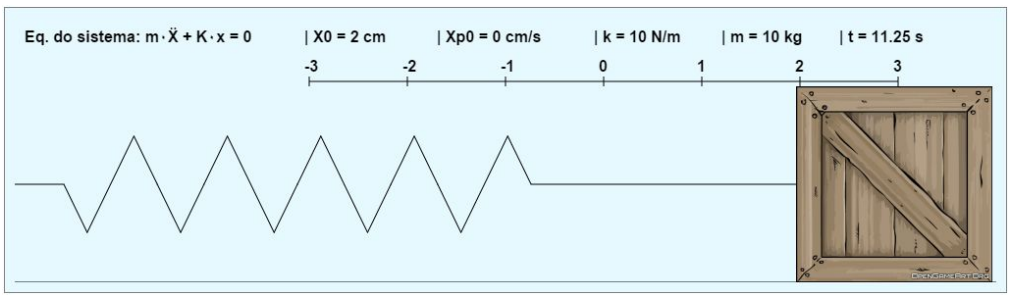
Função “*draw*”

A função “*draw*” é a grande responsável por desenhar todos os elementos na tela do “*canvas*”. De início, seu primeiro comando é desenhar um retângulo que ocupe toda a extensão do “*canvas*”, para cobrir todos os outros elementos e limpar a tela. Sua segunda instrução é desenhar a massa em sua posição atual.

Para cada elemento que queremos desenhar na tela, é necessário informar qual a coordenada X e Y (em pixels) de início e de término. Então, é necessário, mapear a largura e altura do “*canvas*” de forma que se assemelhe à uma distância em escala real. Essa foi a parte mais complicada da tarefa de desenhar a animação, principalmente para desenhar a mola, pois sua forma varia com a posição da massa.

Nas imagens a seguir, podemos visualizar a variação do formato da mola com relação à posição da massa:





Foi necessário criar uma equação (em função de x) para definir a posição das arestas de cada aspira da mola em cada instante de tempo, a fim de desenhar a mola corretamente.

Função “*tick*”

A função “*tick*” é responsável por manter o registro do tempo no programa, como se fosse o “tic toc” do relógio (daí o nome da função). Também é responsável por calcular a nova posição da massa a cada variação de tempo.

```
function tick(){
    tempo += tempoDeAtualizacaoDoCanvas / 1000;
    massaPosicaoX = massaPosicaoXOriginal + xp *
Math.sin(Math.sqrt(kMola) * tempo / Math.sqrt(massa)) + x0 *
Math.cos(Math.sqrt(kMola) * tempo / Math.sqrt(massa));
}
```

A variável “*tempo*” representa o tempo atual (em segundos) e é sempre incrementada em 0,03 segundos (30 milisegundos). Logo em seguida é calculado a nova posição da massa, a qual será utilizada na função “*draw*” para inserir o caixote na animação, desenhar a mola e plotar os pontos no gráfico.

Todo esse processo se repete a cada 30 milisegundos, trazendo a animação à vida!

Possíveis melhorias

Por limitação do tempo para conclusão do projeto, apenas a funcionalidade básica foi implementada, deixando espaço para possíveis futuras melhorias. Entre elas, encontram-se:

- Inclusão de novos sistemas mecânicos
    - Sistema na vertical
    - Sistema massa-mola-amortecedor
    - Sistema com 2 GDL
- Apresentar, no gráfico, o período de vibração e a amplitude atingida
- Desenhar os vetores F_k e $\dot{X}$ em tempo real na animação

Conclusões

Concluo, ao concluir este projeto, que não só conhecimentos na área de desenvolvimento de software foram adquiridos, mas também foi possível aprofundar alguns conhecimentos na área de Modelagem Integrada, a fim de melhorar a qualidade do software.
O desenvolvimento desse projeto me ajudou a descobrir e aprender sobre uma importante tecnologia, que é o "*canvas*". Muitas vezes, programadores, como eu, almejam desenvolver algo que contenha elementos móveis e interação em seus projetos, porém, nem sempre sabem como fazê-lo. Creio que esse conhecimento abrirá outras oportunidades no futuro.

O resultado final do projeto pode ser acessado através do link:
http://rawgit.com/alexandretok/oscilacoes-mecanicas/master/massa-mola.html
ou
https://goo.gl/cUVkV7 (case-sensitive)

Os arquivos e códigos do projeto podem ser acessados através do link:
https://github.com/alexandretok/oscilacoes-mecanicas
ou
https://goo.gl/QZGv7k (case-sensitive)