Animação de Corda baseada em Física

Daniel Guimarães – matrícula

Mariana Barreto – 1820673

Matheus Levi - matrícula

## Introdução

O projeto busca realizar uma simulação de um pedaço de corda em 2D considerando os efeitos gravitacionais. A corda será representada por uma grade de partículas com barras de restrição utilizando o método númerico conhecido como Integração de Verlet. Para tal método, foi utilizada a relaxação para tratar as restrições.

## Desenvolvimento

Para realizar a simulação, foi escolhida a linguagem de programação *Python*. Além disso, o pacote *pygame* foi utilizado para a visualização da corda e a biblioteca *numpy* foi utilizada para criar vetores necessários para implementação do método. Os vetores foram criados como *numpy.array* ao invés de listas para facilitar as operações vetoriais.

Em primeiro lugar, todas as partículas são inicializadas. Cada partícula contém as informações da posição anterior da partícula, posição atual da partícula e um *status* que indica se a partícula está fixa ou não. Quando as partículas inicializadas, todas elas não são fixas com exceção da primeira. A posição anterior de cada partícula é calculada a partir de:

previous\_pos = np.array([0.1 \* i, tam\_corda – dist\_minima \* i + 10])

Ou seja, a coordenada x será equivalente a 10% da posição da partícula no vetor, enquanto a coordenada y será equivalente à diferença entre o tamanho da corda e a distância mínima estabelecida entre as partículas. Tanto o tamanho da corda quanto a distância mínima são parâmetros que podem ser alterados.

Depois de criar as partículas, as distâncias iniciais entre essas partículas são calculadas e guardadas em um vetor e, a partir da posição anterior obtida anteriormente, a posição atual é calculada. O passo de Euler é tomado para obter essa primeira posição, isto é:

current\_pos = particles[i].previous\_pos + h \* v0

A posição atual inicial de cada partícula corresponderá à posição anterior dessa partícula acrescida do passo de integração multiplicado pela velocidade inicial. A velocidade inicial **v0** utilizada foi de (1, -5). [ver se isso não foi alterado, ou o motivo]

Por fim, o relaxamento inicial é feito. Um relaxamento inicial é feito entre cada partícula e sua partícula com posição imediatamente superior. Por exemplo, se temos apenas duas partículas pA e pB com uma distância dA, o relaxamento é feito em cima de pA e pB considerando dA. Já o segundo relaxamento feito considera uma partícula de posição posterior a pB. Ou seja, se fosse acrescida uma partícula pC em uma posição imediatamente posterior a pB com uma distância dB, o segundo relaxamento seria feito em cima de pA, pC considerando a distância entre essas duas partículas (dA + dB).

Para cada relaxamento, é calculada distância entre as posições atuais de cada partícula. Um ajuste é feito em cima da diferença dessa distância e da distância considerada para o relaxamento. A direção de cada partícula é calculada dividindo as duas distâncias. Dessa forma, a nova posição das partículas serão incrementadas de acordo com:

point1.current\_pos = (adjust / 2)\*direction

point2.current\_pos = (adjust / 2)\*(-direction)

Nesta seção, deve-se descrever como foi feito o desenvolvimento do projeto. Deve-se apresentar com mais detalhes o problema em questão e descrever os métodos numéricos que foram empregados.

Se os alunos consultarem livros [1] e/ou artigos [2], os mesmos devem ser referenciados. O mesmo vale para consultas feitas na internet: a fonte deve ser referenciada [3].

Se optar por apresentar figuras, as mesmas devem ter legendas e devem ser referenciadas no texto. Por exemplo, a Figura 1 ilustra o funcionamento do Método de Newton-Raphson.

Macintosh HD:Users:celes:DI:AN:2018.2:slides:4 - raizes:figs:nr.pdf

Figura 1: Método de Newton-Raphson.

A apresentação de trechos de códigos longos deve ser evitada. Dê preferência a apresentação de pseudo-código concisos, se achar necessário. Trechos de código pequenos, que ilustram detalhes importantes também podem ser apresentados na linguagem real de programação. A apresentação de (pseudo-) código pode ser *inline*:

/\* Exemplo de código inline \*/

double valor (double x);

Se o trecho de código for grande, é melhor apresentá-lo e referenciá-lo como uma figura.

## Resultados e Análise

Esta é a seção *mais importante* do relatório. Nela, deve-se apresentar e analisar os resultados, tomando como inspiração as perguntas listadas no enunciado. É fundamental que os alunos baseiem suas análises em experimentos computacionais de fato realizados. Se optarem por dispor resultados em tabelas, as mesmas devem ser apresentadas e referenciadas como figuras. Se tiver muitos resultados, escolha os mais relevantes para não ultrapassar o limite de 6 páginas. Não se esqueça que cada resultado apresentado deve ser seguido por uma análise. Boa sorte!

## Referência

1. Nome dos Autores, *Título do Livro*, edição, Editora, ano.
2. Nome dos Autorea, “Título do Artigo”, *Nome do Veículo onde foi Publicado,* páginas, ano.
3. Nome dos Autores, “Título do Artigo”, url do site, data da consulta.