**BEAMSEQ: Software para estudo de Equilíbrio e Esforços em Vigas Isostáticas**

*Tito A. Gross1, Gilberto Gomes2*

**Resumo**: Este trabalho tem por objetivo a construção de um software em linguagem aberta pela metodologia da Programação Orientada a Objetos (POO) para o estudo de vigas isostáticas. Além disso, apresenta o propósito de ser uma ferramenta de *E-learning*, que mescla a tecnologia com o aprendizado na área de engenharia civil. A linguagem utilizada para o desenvolvimento do programa foi a *Python*, devido a sua facilidade de aprendizado (linguagem de alto nível), por ser uma linguagem de *Software* Livre e Código aberto. Voltado para a aplicação em sala de aula para disciplinas básicas de Resistência dos Materiais, o programa teve como parte da sua metodologia, uma interação com o usuário bastante intuitiva e rápida, oferecendo ajuda para uma navegação simples. Entre essas, existem dicas da função exercida por cada botão contido na interface gráfica de usuário (IGU), ajudas de como montar uma estrutura válida (isostática e com as forças aplicadas sobre ela), *menus* para se desfazer e refazer ações, janelas que descrevem as características básicas do *software* e *links* para web-sites famosos que tem como objetivo ensinar o aluno sobre a matéria abordada. Os resultados que o programa espera apresentar são os de cálculo de reações em apoio e o de montagem dos diagramas de esforços solicitantes, mostrando o passo-a-passo da resolução feita pelo programa, e as apresentando de uma forma análoga aos cálculos feitos por um estudante da área em seu caderno. O software é totalmente aberto para o acréscimo de novas funções e o aprimoramento das funções existentes, sendo um programa colaborativo entre a comunidade.

**PALAVRAS-CHAVE**: Vigas; Resistência dos Materiais; *E-learning*; Python; Interface Gráfica

1Gilberto Gomes, Dept. Engenharia Civil e Ambiental, Universidade de Brasília, CEP 70910-900, Brasília/DF, Brasil, ggomes@unb.br 2Estudante de Iniciação Científica ProIC/DPG/UnB – Universidade de Brasília, Brasília/DF, Brasil

**BEAMSEQ: Software for Study of Equilibrium and Effort in Isostatic Beams**

**Abstract** - The objective of this work is the construction of software in open language using the Object Oriented Programming (OOP) methodology for the study of isostatic beams. In addition, it also presents the purpose of being an E-learning tool, which mixes technology with learning in the area of ​​civil engineering. The language used for the development of the program was Python, due to its ease of learning (high level language) and for being a Free Software and Open Source language. Aimed at classroom application for basic Strength of Materials disciplines, the program had as part of its methodology a very intuitive and quick user interaction, offering help for simple navigation. Among these, there are tips on the function of each button contained in the graphical interface, help with how to assemble a valid structure (isostatic and with the forces applied on it), menus to undo and redo actions, windows describing the basic characteristics of the software and links to famous web-sites that aims to teach the student about the subject addressed. The results that the program expects to present are the calculation of reactions in support and the assembly of the requesting effort diagrams, showing the step-by-step of the resolution made by the program, and presenting them in a way analogous to the calculations made by a student of the area in his notebook. The software is fully open for adding new functions and enhancing existing functions, being a collaborative program among the community.

**Keywords**: Beams; Strength of Materials; E-learning; Python; Graphic Interface

**1 INTRODUÇÃO**

Desenvolver e adaptar técnicas de Computação Gráfica, de Modelagem Geométrica e Computacional e de Estrutura de Dados para confecção de programas orientados a objetos (POO) que permitam a simulação computacional dos problemas da engenharia através dos métodos numéricos tem sido uma resposta eficiente para a solução de problemas reais no ambiente profissional de engenharia (GOMES, 2000).

Há, também, um movimento atual para conciliar tecnologia com o ensino nos diversos níveis que mostra resultados positivos em diversos países. Exemplo disso ocorreu nos Estados Unidos, onde estudantes dos anos finais criaram anuários escolares digitais e os mostraram para estudantes mais novos, o que resultou em alunos mais comprometidos com os estudos (BBC, 2018).

Ainda assim, observa-se uma demanda alta e não atendida no desenvolvimento de softwares voltados para o aprendizado em todas as áreas de ensino no ambiente acadêmico (SEABRA, 2008.).

É possível enxergar, portanto, um potencial grande no desenvolvimento de plataformas que busquem ajudar no aprendizado dentro da engenharia civil, principalmente em disciplinas que tem enfoque em cálculos. Esse estudo provoca de imediato um debate sobre o aprendizado na área de engenharia e chama a atenção para a área de programação (RUSCHEL, *et al*, 2013).

Nesse contexto, este trabalho busca atender parte da demanda por processos informatizados na solução de problemas de disciplinas básicas de resistência dos materiais e, desta forma, propõe o desenvolvimento de um software denominado BEAMSEQ para estudo e ensino-aprendizagem de vigas isostáticas.

**2 MATERIAL E MÉTODOS**

**2.1 Estudo da linguagem e do método de confecção**

A primeira parte do trabalho contou com um estudo bem detalhado da linguagem utilizada no código. Atentou-se para o fato de Python ser uma linguagem de alto nível, interpretada, totalmente orientado a objetos com um modelo de desenvolvimento aberto e comunitário (PYTHON, 2018), o que ajuda na construção de bibliotecas para o uso de todos. Assim, essa é muito usada para a criação de programas dentro da engenharia, devido a facilidade de uso e acessibilidade.

O estudo do paradigma da POO foi complementar ao próprio aprendizado do *python*, já que esta é uma linguagem voltada para essa forma de confecção. Assim, apesar de não haver necessidade de conhecimento de POO para se programar em Python, a própria ideia da linguagem e seus objetos, classes, atributos e métodos usam conceitos desse paradigma. Esse método, portanto, se mostrou bastante eficaz, também devido a sua estrutura hierárquica, que facilita a utilização de métodos previamente criados em diferentes classes (BOOCH, 1994), o que proporcionou mais velocidade na atividade de programação.

O programa propôs, também, uma revisão teórica dos conceitos de equilíbrio interno, que foram realizados através dos livros de resistência dos materiais. Esse estudo, assim como tratado na maioria dos livros da área, é feito pela ideia de que cada ponto (externo ou interno aos elementos estruturais) devem estar em equilíbrio. Assim, fazendo-se um corte qualquer em uma viga, o somatório de esforços nesse ponto deve ser igual a 0 (HIBBELER, 2011; BEER, F.P. e JOHNSTON, R.E. e EISENBERG, E.R, 2006), como mostra a figura 1 e a Equação (1).



Figura 1: Representação dos esforços internos em uma seção S

$\left\{\begin{array}{c}ΣF\_{x}=0 ⇒ ΣF\_{ext\_{x}}+N=0\\ΣF\_{x}=0 ⇒ ΣF\_{ext\_{y}}+V=0\\ΣM\left(S\right)=0 ⇒ M\_{ext}+ M=0\end{array}\right.$ (1)

**2.2 Interface gráfica do usuário e tratamento dos dados fornecidos**

Como observado no estudo da linguagem de Python, existem diversas bibliotecas com ferramentas avançadas e práticas para que se torne mais rápida a possibilidade de usar a linguagem. Uma das bibliotecas de interface mais conhecidas e com mais referências é a *PyQt*, que conta, entre outras coisas, com *Widgets* com formatos pré-definidos que contem sinais conectados a eles para que se execute o que quiser a partir da interação destes com o usuário.

Dessa forma, utilizou-se o aplicativo *QtDesigner* para a montagem da geometria da interface e com a ferramenta *pyuic* gerou-se um documento de formato .*py* que, quando executado, apresenta a interface em seu estado primário.

Em outro documento do mesmo ambiente virtual, foram criadas as implementações geradas pela interação com o usuário. Estas surgem de classes e métodos derivados do primeiro arquivo, que é importado pelo segundo nas suas primeiras linhas de código.

Os cálculos matemáticos mais complexos foram realizados pela biblioteca *numpy*, que é completa para o tratamento de sistemas lineares. A interação com o usuário e a apresentação dos resultados

**2.3 Apresentação dos resultados**

Como o objetivo final do programa é calcular as reações de apoio e os diagramas de esforços solicitantes, foi criada uma classe em um terceiro arquivo que recebe, para cada viga desenhada, os métodos do elemento gráfico da linha, os pontos inicial e final, as cargas a que elas estão submetidas (que se transfere a uma viga pela avaliação da distância até seu elemento gráfico na área de desenho) e as restrições em suas extremidades (se está conectada a outra viga, a um apoio, ou se está livre).

Com esta classe é possível calcular os esforços internos em cada uma delas separadamente, além de se obter as reações externas.

**3 RESULTADOS E DISCUSSÃO**

A parte mais detalhada do programa foi a sua interação com o usuário. Para que se recebesse os dados da estrutura proposta, foi construída uma interface gráfica com uma área de desenho contida em uma aba dentro da janela principal, que está representada na figura 3.1. Essa mesma janela contém outras três abas para disponibilizar os três diagramas de esforços solicitantes para estruturas bi-dimensionais (esforço normal, esforço cortante e momento fletor) que ocupam o mesmo espaço geométrico que a primeira aba, e o usuário define qual delas deve estar em primeiro plano e visível.



Figura2: Interface Gráfica do programa

A janela principal ainda conta com botões que alternam entre as funções previstas para a área de desenho (que se encontram no lado esquerdo da IGU) e botões que apresentam os resultados após o usuário terminar de definir sua estrutura, além das restrições e carregamentos desta. Estes últimos botões só executam a função a que foram designados se a estrutura for válida (isostática e com um sistema linear possível e determinado), em virtude da existência de um bloco de código condicional do tipo “se”, como exemplifica o pedaço de código abaixo.

if len(GraphicsScene.number\_of\_variables) == 3:

 if self.scene.restrictions[0] == 1:
 dist = (-self.scene.y + self.scene.y2)/100
 x = [[1, 1, 0], [0, 0, 1], [dist, 0, 0]]
 a = np.asarray(x)

 Momentos = np.matmul(self.force, self.distance)
 SomatorioMomentos = np.matrix.trace(Momentos)
 y = [SomatorioX, SomatorioY, SomatorioMomentos]
 b = np.asarray(y)
 resultados = np.linalg.solve(a, b)

As aplicações que podem ser executadas na área de desenho são as de construção dos elementos gráficos (semelhante ao diagrama de corpo livre) de apoios de primeiro, segundo e terceiro gênero, vigas e carregamentos do tipo força pontual, força distribuída e momento pontual, além de apagar estes. Todas estas utilidades são executadas com eventos do *mouse* e podem ser vistas enquanto são realizadas. Um exemplo disso é a função em que o usuário define a linha que representa uma viga, que inicia o item gráfico após o primeiro clique e faz o segundo ponto deste seguir o movimento do mouse até que este seja definido pelo segundo clique. Um exemplo disso está representado nas figuras 3, 4 e 5, que mostram as etapas para a adição de uma viga, de uma força pontual e de uma carga distribuída (com a adição dos seus principais métodos a listas ocorrendo no *back-end*, para que estas sejam passadas a uma classe que faz o tratamento dos dados para as soluções numéricas).

  

b

a

 

c

d

Figura 3: Etapas da construção de uma viga na área de desenho: (a) seleciona-se a função de desenho de vigas; (b) *mouse* próximo a uma das extremidades de uma viga já desenhada mostra elementos gráficos que indicam aproximação para o primeiro *click*; (c) linha liga o primeiro clique até o a posição do *mouse* até que haja outro clique; (d) após o segundo *click* a viga é definida, podendo automaticamente horizontal ou vertical se a tecla *shift* for selecionada enquanto a viga é desenhada.

 

a

b

 

d

c

Figura 4: Etapas da adição de uma força pontual a uma estrutura bi-apoiada: (a) seleciona-se a função de adição de forças pontuais; (b) abre-se uma aba para definição das magnitudes das forças vertical e horizontal que serão adicionadas; (c) *mouse* próximo a uma das extremidades de uma viga já desenhada mostra elementos gráficos que indicam aproximação para o *click*; (d) a força é adicionada.

 

b

a

 

d

c

Figura 5: Etapas para acréscimo de força distribuída: (a) seleciona-se a função de adição de forças pontuais; (b)abre-se uma aba para definição das magnitudes das forças inicial e final, além da decisão da direção da força (horizontal ou vertical) ; (c) após o primeiro *click*, a força acompanha o movimento do *mouse*; (d) a força é totalmente definida após o segundo *click*.

Para a praticidade de uso e facilidade de aprendizado, as funções citadas contam com finalidades usuais em softwares profissionais, como o uso do refazer e desfazer, elementos gráficos que indicam quando o mouse está próximo o suficiente de um ponto para que, quando ocorra o clique, o elemento gráfico que está em construção seja atraído para se conectar a um elemento gráfico já construído e a presença de linhas de edição que mostram a posição do mouse em relação ao sistema de eixos coordenados da área de desenho (ou em relação ao primeiro clique nas funções de desenhar vigas e cargas distribuídas). Há ainda dicas da funcionalidade de cada botão que ficam visíveis quando o mouse passa por cima destes e atalhos de teclado para funções básicas do programa.

Em casos em que o programa não consegue executar uma utilidade, este busca informar ao usuário de um erro provável, com a aparição de janelas de informação. Como exemplo disso temos a tentativa de execução de adquirir as reações de apoio para uma viga não isostática, na qual o programa irá relatar que apenas prevê o cálculo de estruturas isostáticas e mostrar quantas reações o diagrama construído apresenta.

Ao solicitar a solução do problema, o programa busca mostrar o passo-a-passo da solução feita, sempre procurando ser análogo ao que seria feito em um caderno de um estudante da área, como mostra a figura 6.

 

b

a



c

Figura 6: Exemplo numérico (a) construção gráfica de um diagrama de corpo livre para um sistema de carregamentos simples; (b) tela de saída das reações de apoio; (c) aba do gráfico de diagrama dos momentos fletores.

**4 CONSIDERAÇÕES FINAIS**

O trabalho tem como objetivo, além da solução numérica de problemas propostos pelos usuários, trazer o aprendizado de engenharia civil e de computação mais próximo aos estudantes. Dessa forma, todo o código é aberto e foi feito para se ter facilidade de aprimorar funções já existentes ou criar novas, como a construção do gráfico de flechas ou o cálculo de tensões para determinada seção. Assim, ele busca ser um programa colaborativo entre a comunidade.

O programa contem limitações como a de números de barras por nó (só se podem conectar duas vigas a cada nó) e a falta de ligações de segundo gênero entre vigas. Esses ações, no entanto, estão atualmente sendo acrescentadas ao programa para futuros projetos.

**REFERÊNCIAS**

Gomes, G., “Estrutura de Dados para Representação de Modelos Bidimensionais de Elementos de Contorno”, Dissertação de Mestrado, Universidade de Brasília, Brasil, 2000.

Gomes, G., TRUSS\_GI: Interface Gráfica para Pré-Processamento de Treliças Planas. 2000 (Software sem patente).

Luiz, M., ASCHER, D., “Aprendendo Python. Editora Bookman, 2007.

BRUECK, D., TANNER, S., “Python 2.1 Bible. Editora Hungry Minds, 2001.

Booch, G., “Object-Oriented Analysis and Design with Applications”, The Benjamin/Cumming Publishing Company, Inc., 1994.

R.C. Hibbeler, "Engenharia Mecânica - Estática", 8ª Edição, LTC.

BEER, F.P. e JOHNSTON, R.E. e EISENBERG, E.R. “Mecânica Vetorial para Engenheiros. Vol. Estática”. Ed. MacGraw-Hill. 7ª edição. S. P. 2006.

SEABRA, Carlos. O computador na criação de ambientes interativos de aprendizagem. **Em Aberto**, v. 12, n. 57, 2008.).

Ruschel, Regina Coeli, Max Lira Veras Xavier de Andrade, e Marcelo de Morais. "O ensino de BIM no Brasil: onde estamos?." *Ambiente Construído* (2013).

Python Software Foundation.  2018. Disponível em: <<https://www.python.org/psf/>>. Acesso em: jul. 2018.

BBC News. 2018. Disponível em: <<https://www.bbc.com/portuguese/noticias/2014/12/141202_tecnologia_educacao_pai>>. Acesso em: jul. 2018