CIV 2801 – Fundamentos de Computação Gráfica Aplicada – 2024.2

3º Trabalho: Cross02 – Interface gráfica do programa Cross

Desenho do modelo no programa do processo de Cross para vigas contínuas em MATLAB

Data da entrega: 27/set/2024

Este trabalho é continuação do programa que implementa o processo de Cross para vigas contínuas com cargas uniformemente distribuídas em cada vão. O objetivo do trabalho é implementar a função de desenho em um *canvas* do modelo da viga contínua com apoios e cargas. O desenho da configuração deformada do modelo, do diagrama de momentos fletores e a criação da tabela com os passos do processo de Cross serão tratados nos próximos trabalhos.

Complemente a classe **`CrossDraw`** do programa em MATLAB fornecido na homepage da disciplina (procure por terceiro trabalho):
https://www.tecgraf.puc-rio.br/ftp_pub/lfm/civ2801-242-trab3.zip.

Uma versão pré-compilada (**`pcode`**) da solução do trabalho também é fornecida no site da disciplina: https://www.tecgraf.puc-rio.br/ftp_pub/lfm/civ2801-242-trab3_pcode.zip.

No link https://www.tecgraf.puc-rio.br/~lfm/compgraf-242/cross02/main.html tem uma documentação do código do programa do trabalho.

O entendimento do código do programa é parte do trabalho.

Especificação

1. O arquivo **`CrossGUI.mlapp`** do segundo trabalho (com o diálogo de interface criado pelo *App Designer* na pasta **`gui`**) deve se modificado para incluir o novo *canvas* para desenho do modelo da viga contínua (com apoios, carregamentos e linhas de cota). Sugestão: faça as modificações baseadas no arquivo **`CrossGUI_code.m`** fornecido.
2. O arquivo **`CrossSolver.m`** com a solução do primeiro trabalho deve ser aproveitado no presente trabalho.
3. O arquivo **`CrossControl.m`** foi modificado em relação ao segundo trabalho. Entretanto, para completá-lo basta copiar a solução adotada naquele trabalho para o este arquivo. É importante notar que o método privado **`CrossControl.updateDraw`** da classe **`CrossControl`** associa o método **`CrossDraw.model`** da classe **`CrossDraw`** como função para redesenho do modelo da viga contínua no *canvas* e associa o método **`CrossDraw.crossBoundBox`** dessa mesma classe como função para retorno do retângulo envolvente (*bounding box*) do modelo.
4. O programa contém novas classes para desenho do modelo no *canvas*: é a hierarquia de classes **`Canvas`**, **`Canvas_2D`** e **`CanvasCross`** no contexto de Programação Orientada e Objetos. Essas classes estão prontas e não precisam ser modificadas no trabalho.
5. Os métodos **`protected CrossDraw.beam`**, **`CrossDraw.memberLoad`**, e **`CrossDraw.dimLine`**, da classe **`CrossDraw`** no arquivo **`CrossDraw.m`** devem ser implementados neste trabalho. As funções auxiliares **`square`**, **`triangle`**, e **`arrow2D`** dessa classe (já implementadas) devem ser utilizadas.
6. Enviar via e-mail um arquivo .zip contendo os arquivos **`CrossGUI.mlapp`**, **`CrossSolver.m`**, **`CrossControl.m`** e **`CrossDraw.m`** com a solução do trabalho.
7. Enviar via e-mail um arquivo texto em PDF mostrando apenas as funções que foram complementadas no arquivo **`CrossDraw.m`**.

Observação
O método **`CrossDraw.crossBoundBox`** (que não precisa ser completado) retorna o retângulo envolvente (*bounding box*) do modelo da viga contínua do processo de Cross. Esse método define um *bounding box* do modelo tal que na direção horizontal os limites de coordenadas são **`Xmin = 0`** e **`Xmax = totalLen`**, em que **`totalLen`** é o comprimento total da viga contínua, e na direção vertical os limites do *bounding box* são **`Ymin = -totalLen * 0.05`** e **`Ymax = +totalLen * 0.05`**. Observa-se que o tamanho do *bounding box* na direção vertical retornado por esse método corresponde a 10% do comprimento total da viga contínua.

Nas notas de aula ("Desenho de primitivas vetoriais em Canvas no MATLAB") existe uma explicação de como o *bounding box* de um modelo é ajustado para o retângulo de visualização (chamado de *window*). O *window* define a região do espaço de modelagem que é visualizada em um *canvas* (cujo retângulo é chamado *viewport*). A transformação de coordenadas do espaço de modelagem para o espaço do *canvas* é chamada de transformação *window-viewport*.

O *window* é definido ajustando as dimensões do retângulo do *bounding box*. Esse ajuste é feito pelo método **`fit2view`** da classe **`Canvas_2D`** com dois objetivos. O primeiro é aumentar o retângulo do *bounding box* para dar uma folga, evitando que o desenho do modelo não fique colado nas bordas do canvas. O segundo objetivo do ajuste é alterar a razão **`Y/X`** entre as dimensões **`Y`** (vertical) e **`X`** (horizontal) do *bounding box* de tal maneira que o retângulo do *window* tenha a mesma proporção entre base e altura do retângulo do *viewport* (retângulo do *canvas*). Isso é feito para que a escala do desenho na direção horizontal seja igual à escala na direção vertical.

Além disso, como os limites **`Ymin`** e **`Ymax`** do *bounding box* são iguais em módulo, o retângulo do *windows* será ajustado mantendo **`Ymin`** e **`Ymax`** iguais em módulo. A consequência disso é que o eixo da viga contínua (com **`Y = 0`**) vai sempre ficar centrado no *canvas* na direção vertical.

O ajuste do *window* é sempre feito aumentando uma das dimensões do *bounding box*, isto é, ou a dimensão **`X`** é aumentada ou a dimensão **`Y`** é aumentada, dependendo da razão entre base e altura do retângulo do *window* em relação ao retângulo do *canvas* (*viewport*).

Se o carregamento da viga contínua ou as linhas de cota estiverem muito afastados do eixo da viga na direção vertical, o ajuste do *window* seria feito aumentando sua dimensão horizontal. Nesse caso, o modelo teria a escala reduzida no desenho na direção horizontal, o que não é desejável.

Para que esse problema não ocorra, o método **`CrossDraw.crossBoundBox`** força que a dimensão aumentada do *window* seja sempre na direção **`Y`**. Na direção **`X`**, o *window* vai ser só aumentado da folga para que o desenho da viga contínua não fique colado nas bordas laterais do canvas. Essa é a razão porque esse método define a dimensão do *bounding box* na direção vertical igual a 10% do comprimento total da viga contínua.

Entretanto, é preciso tomar cuidado para que o desenho do modelo com carregamento, valores do carregamento e linhas de cota não tenha nenhuma linha ou texto fora dos limites **`Ymin`** e **`Ymax`** do canvas, pois qualquer linha ou texto desenhado fora desses limites não vai aparecer na imagem do canvas.

Por esse motivo, nas propriedades da classe **`CrossDraw`**, o tamanho do carregamento é definido como uma percentagem da metade da dimensão vertical do *window*. As linhas de cota e os valores dos carregamentos também não podem ultrapassar os limites **`Ymin`** e **`Ymax`**. Esse cuidado tem de ser tomado.