Trabalho Prático 1
Disciplina de Computação Gráfica – 2024

Instituto Federal de Educação, Ciência e Tecnologia do Norte de Minas Gerais – Campus Montes Claros

Tema:

Desenvolva um programa que seja capaz de ler um arquivo contendo informações a respeito dos objetos a serem desenhados, da *window* e da *viewport*, e gere um arquivo de saída contendo os objetos no sistema de coordenadas de *viewport.* Além disso, seu sistema também deverá permitir a movimentação da window no mundo usando as teclas direcionais do teclado.

Descrição:

Para desenvolver este trabalho prático, vocês terão que ler um arquivo texto contendo todas as informações de entrada necessárias (será dado um modelo de arquivo texto em xml para que vocês possam ter ideia do formato básico exigido) e, como saída, deverão exibir graficamente os objetos descritos no arquivo de entrada, possibilitar salvar a descrição dos dados da viewport, ou seja, as coordenadas transformadas dos pontos do mundo para o sistema de coordenadas da viewport, inclusive os pontos que não estão visíveis.

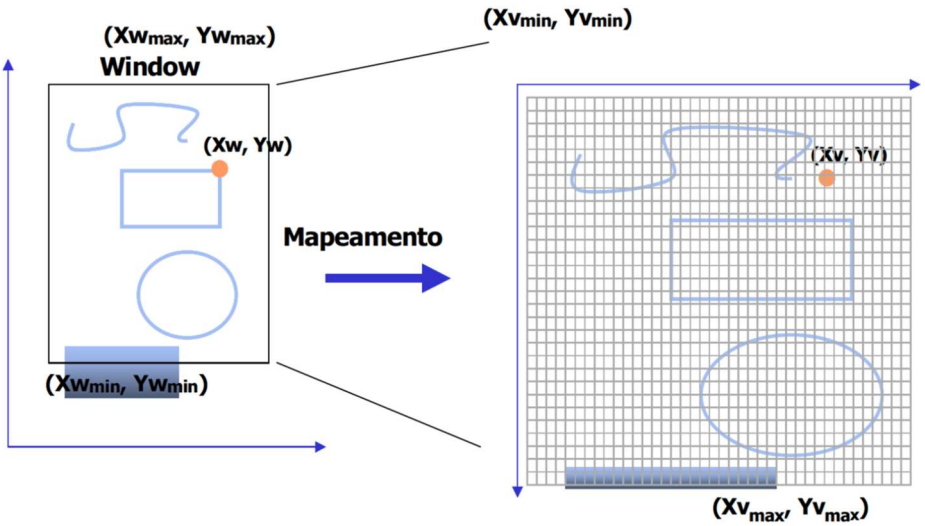

Figura 1: Representação gráfica da transformada de viewport.

Adicione ao seu sistema de visualização um *minimapa* do mundo para ter uma visualização ampliada da cena e entender melhor sobre a localização da *window* neste espaço.

Permita que o usuário possa interagir com seu programa movimentando a *window* no mundo usando as teclas direcionais do teclado. Cada vez que uma tecla por pressionada, a *window* irá se movimentar em **1 unidade** na direção correspondente e todas as coordenadas de *viewport* dos elementos do mundo deverão ser recalculadas para refletir a nova posição da *window*. Para já irem se acostumando com o processo de aplicação de transformações lineares, gerem uma matriz de transformação que é modificada cada vez que o usuário pressionar uma tecla direcional e, posteriormente, apliquem a transformada de *viewport* aos pontos do mundo considerando a nova posição da *window*. Como resultado, o movimento da *window* para determinado sentido irá resultar na movimentação dos objetos na *viewport* para o sentido oposto.

Uma ideia de interface para o seu programa poderá ser como a descrita na figura a seguir:

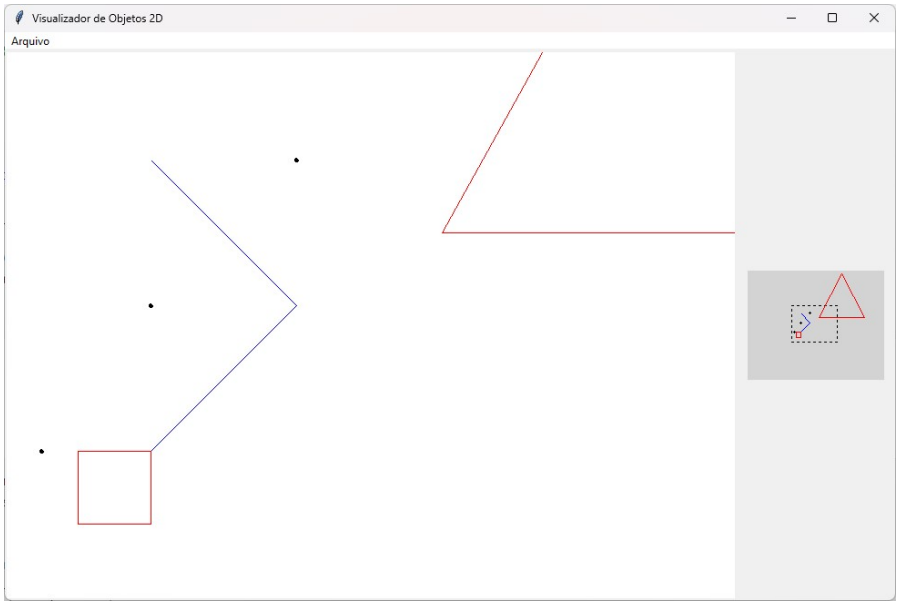

Figura 2: Exemplo de uma interface para o trabalho.

Para a implementação da interface gráfica, utilizem ***python*** com objetos do tipo *canvas* da biblioteca ***tkinter*** para representar as viewports.

Para a leitura do arquivo XML vocês poderão utilizar a biblioteca *xml.etree.ElementTree* .

Para agilizar o processo de aprendizado a respeito do uso da biblioteca *tkinter*, um arquivo será dado com um código base tal.

Sobre os requisitos básicos do trabalho:

- O arquivo de saída também deverá estar no formato XML seguindo o mesmo padrão do arquivo de entrada porém com os dados dos objetos no sistema de coordenadas da viewport escritos após os dados dos objetos no sistema de coordenadas do mundo. A ideia é que se possa ler este arquivo de saída com o mesmo programa escrito para ler a entrada (porém ignorando os dados da viewport, caso necessário).
- No caso da utilização da linguagem Python, utilizem "*notebooks*" para documentar seu código. Dessa forma, ficará mais fácil de explicar o que cada parte do seu código faz.
- Organizem bem código para permitir uma fácil modificação futura. Recomenda-se o uso de dicionários e/ou classes para representar os dados. Por questões de simplicidade, vocês podem usar apenas dicionários e listas neste primeiro trabalho. Mas coloquem o programa principal com a definição da interface e as chamadas para os demais métodos em uma classe.
-
- No arquivo XML que será dado como exemplo, há a definição da window e da viewport como retângulos, ou seja, apenas suas extremidades. Quanto aos objetos gráficos teremos:
  - Ponto2D: 2 coordenadas reais.
  - Reta2D: 2 Pontos não coincidentes.
  - Polígono2D: lista de, ao menos, 3 pontos onde, o último ponto será ligado ao primeiro. Por enquanto, o polígono poderá ser vazado, ou seja, sem informações do seu interior, também chamado de polígono aramado (ou *wireframe*).

O cálculo da transformada de *Viewport* é definido como uma transformação linear simples entre a *window* e a *viewport*. Para encontrar o valor de *x* na *viewport*, x_{vp}, realizamos a transformação linear:

$$x_{vp} = \frac{Xw - Xwmin}{Xwmax - Xwmin} \bullet (Xvpmax - Xvpmin)$$

Para encontrar o valor de *y* na *viewport,* $y_{vp,}$ realizamos uma transformação um pouco diferente, levando em conta a inversão do eixo *y*, uma vez que você deverá considerar que a origem do seu sistema de visualização se encontra no canto superior esquerdo:

$$Y_{vp} = \left(1 - \frac{Yw - Ywmin}{Ywmax - Ywmin}\right) \bullet (Y_{vpmax} - Y_{vpminn})$$

Observações finais:

Os trabalhos poderão ser feitos em DUPLAS (ou individualmente), porém, códigos iguais entre equipes diferentes não serão aceitos.

É necessário entregar um documento para informar como utilizar seu programa corretamente, isto será utilizado como guia para compilação e teste pelo professor. Como dica, utilizem notebooks jupyter seu trabalho prático. Documentem teoricamente e tecnicamente o trabalho de vocês, introduzam o tema, ilustrem o que vocês estão fazendo e informem sobre as decisões de implementação adotadas.

A entrega do trabalho deverá ser feita na sua atividade correspondente no *google classroom*, em um **único arquivo** compactado contendo todo o projeto e DEVIDAMENTE IDENTIFICADOS da seguinte forma:

NomeSobrenome1_NomeSobrenome2.zip