PODER EXECUTIVO
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE RORAIMA
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

DCC703 - COMPUTAÇÃO GRÁFICA

RELATÓRIO DO PROJETO DE RECORTE

ALUNOS:
Marcos Vinícius Tenacol Coêlho - 2021000759

Março de 2025
Boa Vista/Roraima

Resumo

Este relatório descreve a implementação de um programa em Python utilizando o algoritmo de recorte de polígonos de Sutherland-Hodgman, o programa visa demonstrar a aplicação do algoritmo de Sutherland-Hodgman para recortar polígonos conforme uma janela de recorte definida pelo usuário.

Estrutura do Código

desenhar_poligonos: Responsável por desenhar o polígono original em vermelho, o recortado em azul e a janela de recorte em verde.

recortar: Implementa o algoritmo de Sutherland-Hodgman, iterando sobre os lados da janela de recorte para calcular os novos vértices do polígono recortado.

Como funciona o algoritmo de Sutherland-Hodgman ?

O algoritmo funciona cortando o polígono iterativamente contra cada uma das bordas da janela de recorte (esquerda, direita, topo e fundo). Para cada borda:

- Identifica se um vértice está dentro ou fora da região.
- Calcula os pontos de interseção quando uma aresta atravessa a borda.
- Mantém apenas os vértices dentro da região de recorte.

Implementação e testes

Função de recorte:

```
def recortar(poligono, janela):
    x_min, y_min, x_max, y_max = janela
       def dentro(p, borda):
        x, y = p
        return (x >= x_min if borda == "esquerda" else
                x <= x_max if borda == "direita" else
                y >= y_min if borda == "topo" else
                y <= y_max)

    def intersecao(p1, p2, borda):
        x1, y1 = p1
        x2, y2 = p2
        if x1 == x2:
            if borda == "topo":
                x = x1
                y = y_min
            elif borda == "fundo":
                x = x1
                y = y_max
            return (x, y)

        if y1 == y2:
            if borda == "esquerda":
                x = x_min
                y = y1
            elif borda == "direita":
                x = x_max
                y = y1
            return (x, y)

        m = (y2 - y1) / (x2 - x1)
        b = y1 - m * x1
        if borda == "esquerda":
            x = x_min
            y = m * x + b
        elif borda == "direita":
            x = x_max
            y = m * x + b
        elif borda == "topo":
            y = y_min
            x = (y - b) / m
        elif borda == "fundo":
            y = y_max
            x = (y - b) / m
        return (x, y)

    for borda in ["esquerda", "direita", "topo", "fundo"]:
        novo_poligono = []
        for i in range(len(poligono)):
            atual, anterior = poligono[i], poligono[i - 1]
            if dentro(atual, borda) != dentro(anterior, borda):
                novo_poligono.append(intersecao(anterior, atual, borda))
            if dentro(atual, borda):
                novo_poligono.append(atual)
        poligono = novo_poligono
    return poligono
```

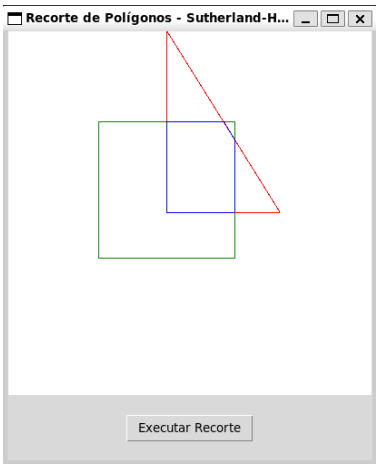
Essa função verifica se os pontos estão dentro ou fora da janela de corte. Caso estejam fora, ela determina em qual setor se encontram e, se necessário, realiza o cálculo da interseção, que pode ser vertical, horizontal ou diagonal.

A partir daqui, serão apresentados os testes com as respectivas coordenadas, onde:

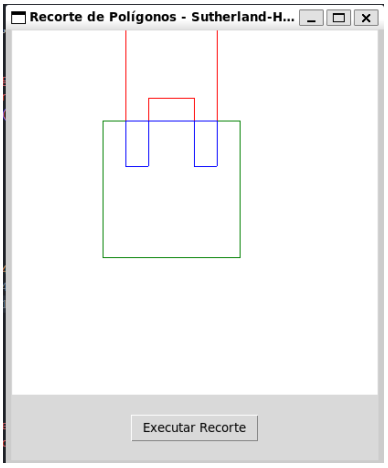
- A área verde representa a janela de corte,
- A vermelha corresponde à figura completa,
- A azul indica a parte recortada.

Sendo as coordenadas:

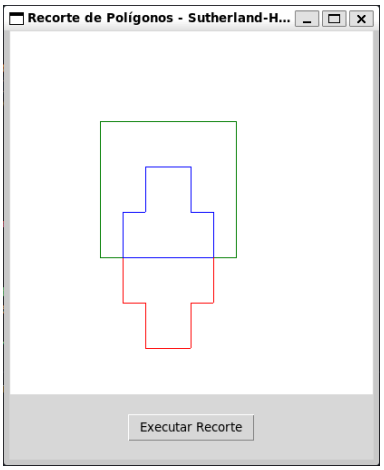
- (3.5,0),(3.5,4),(6,4):



- (2.5, 0), (4.5, 0), (4.5, 3), (4,3), (4, 1.5), (3, 1.5), (3, 3), (2.5, 3):



- (3,3),(4,3), (4,4),(4.5,4),(4.5,6),(4,6),(4,7),(3,7),(3,6),(2.5,6),(2.5,4),(3,4):



- (2.5,1.5),(3.5,2.5),(3,4),(1.75,4),(1.25,2.5):