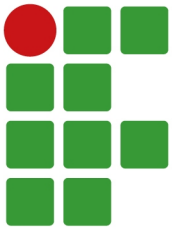
OpenGL/GLFW: Primitivas de Desenho OpenGL e *Clipping*



INSTITUTO FEDERAL
São Paulo

Câmpus
Presidente Epitácio

Prof. Dr. Bruno César Vani

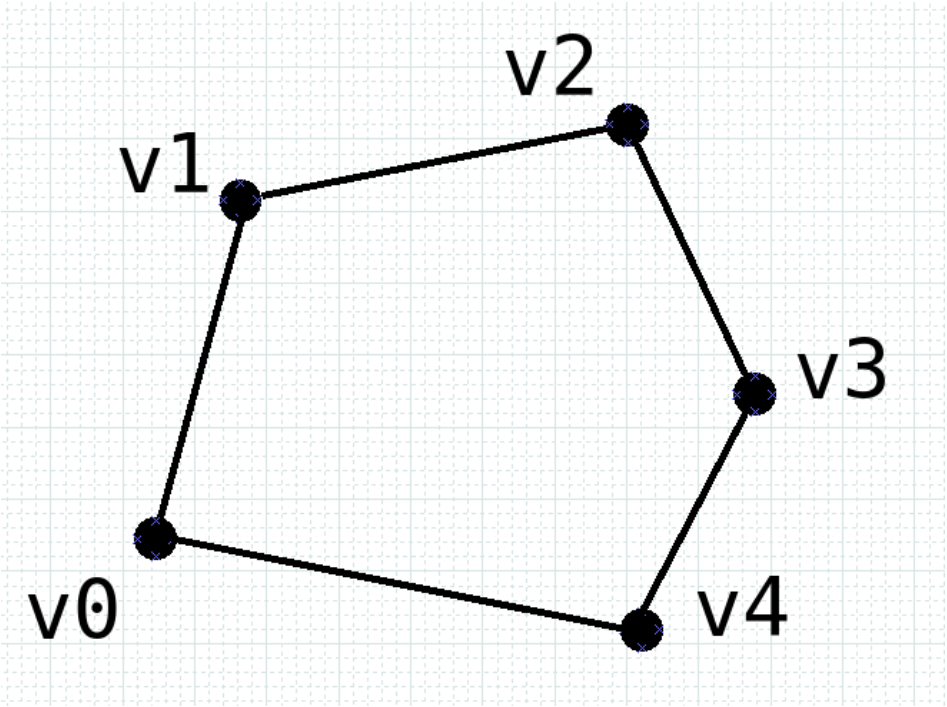
Notas de Aula de Computação Gráfica

BCC / 2020

Conteúdo e Objetivos

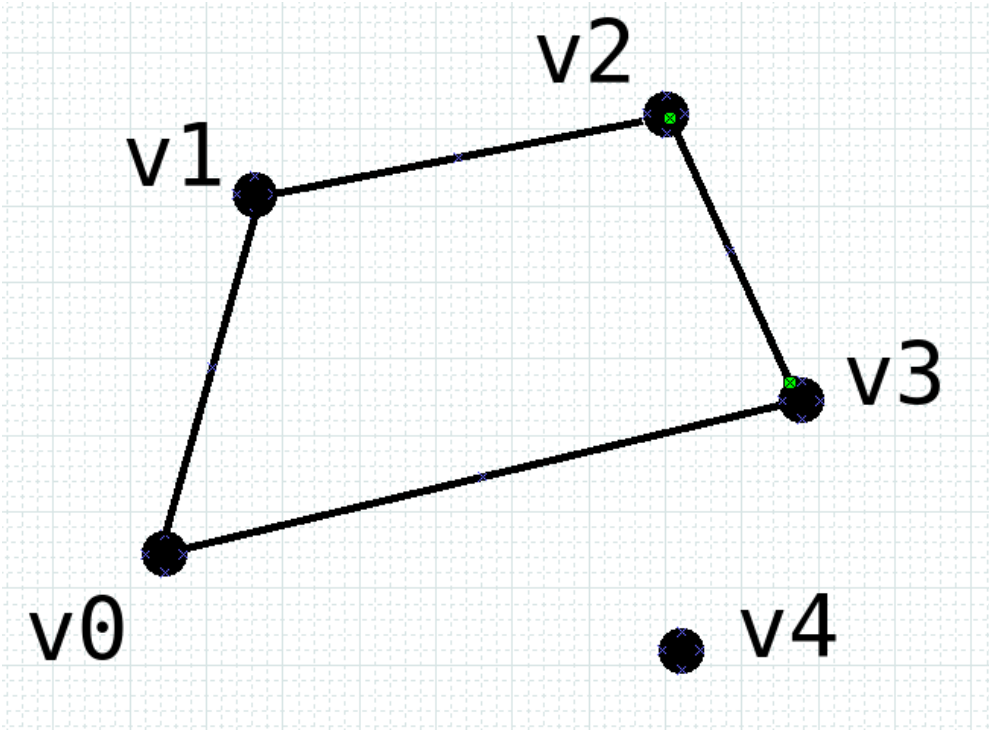
- Compreender os diferentes tipos de primitivas;
- Utilizar configurações de modos de desenho (largura de linhas, configurações de traçado);
- Compor figuras geométricas mais elaboradas com primitivas de triângulos;
- Realizar experimentos e resolver exercícios de fixação.

Polígono: GL_POLYGON



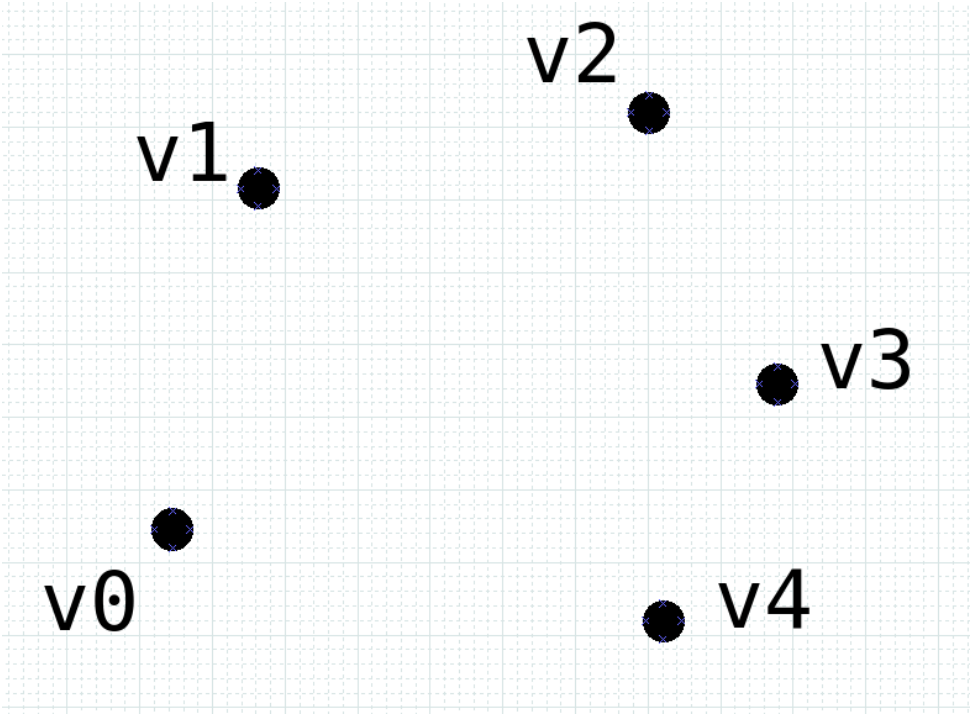
- Esta primitiva não aparece no OpenGL moderno (*core*)
  - O uso de primitivas de triângulos é encorajado em substituição à primitiva de polígono
- Ainda é possível utilizar no modo OpenGL antigo (*legacy* ou modo de compatibilidade)

Quadriláteros: GL_QUADS



- Esta primitiva também não aparece no OpenGL moderno (*core*)
- Ainda é possível utilizar no modo de compatibilidade
- Obs: vértices remanescentes são descartados (quantidade de vértice múltipla de 4)
- Dica: uso de *glPolygonMode* para depuração

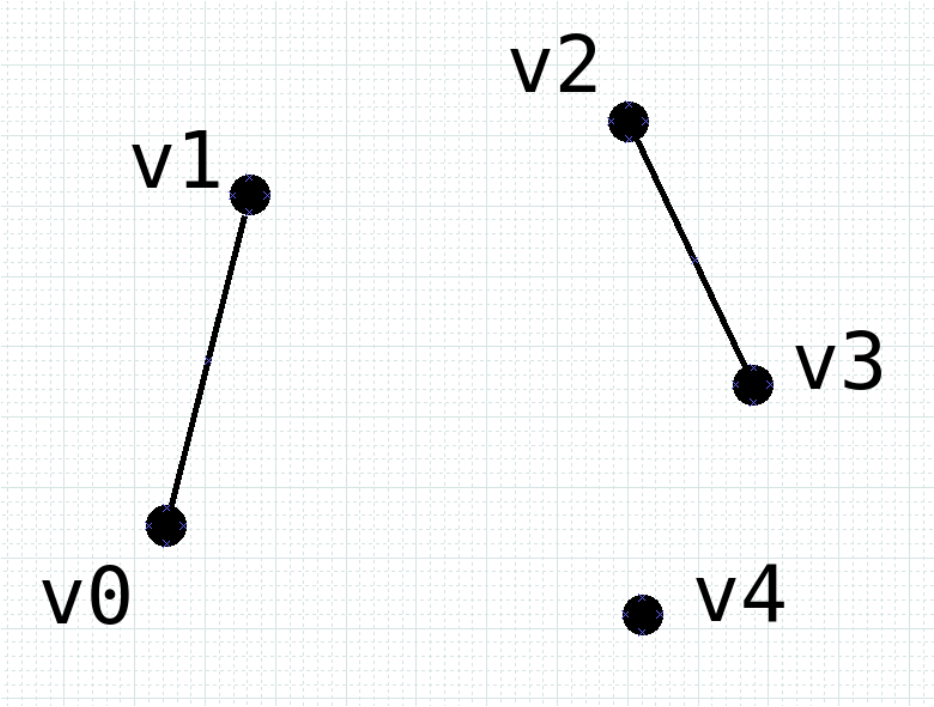
Pontos: GL_POINTS



- Cada vértice representa um ponto em um espaço de coordenadas homogêneas(*);
- Utiliza-se *glPointSize(1.0)* para alterar o tamanho inicial dos pontos

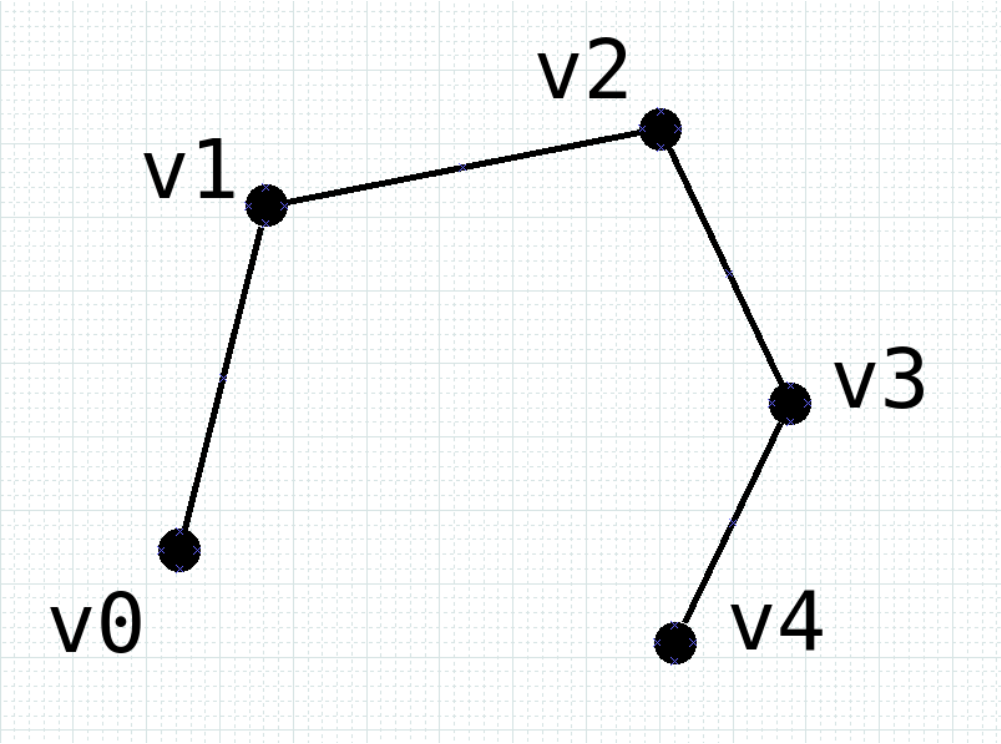
(*) Assunto de aulas posteriores.

Linhas: GL_LINES



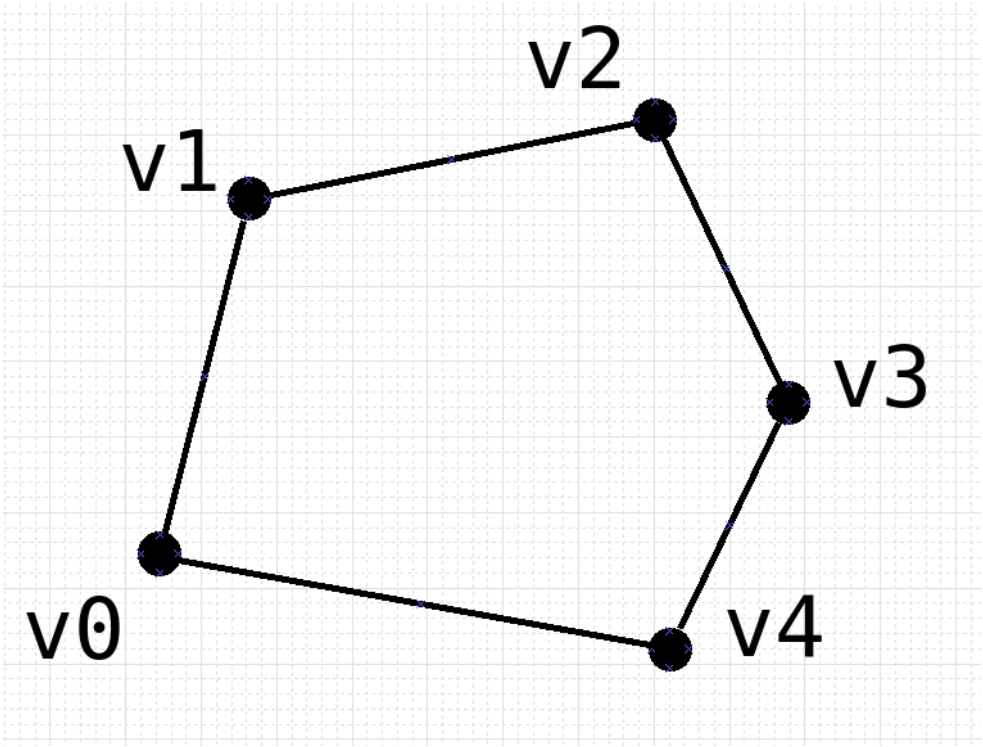
- Vértices são interligados:
  - v0 e v1
  - v2 e v3
- Se a quantidade de vértices é ímpar, o último é descartado
- Configuração de largura da linha: *glLineWidth()*

Linhas: GL_LINE_STRIP



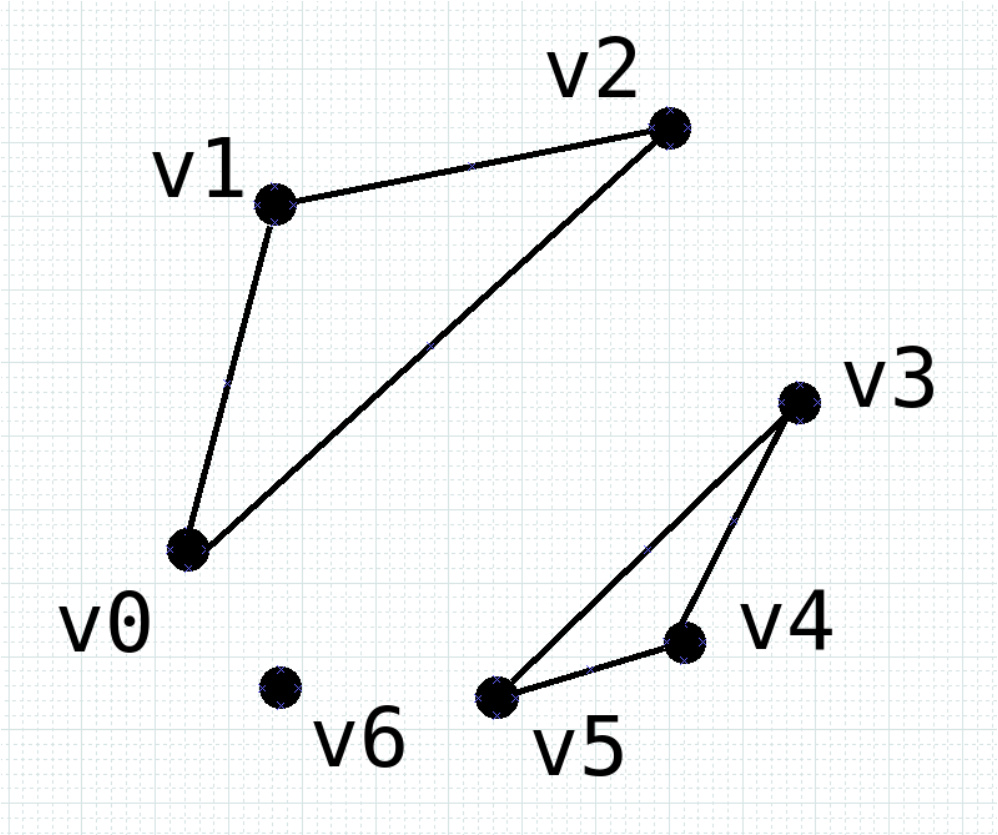
- Vértices são interligados sucessivamente, até o último vértice
- Configuração de largura da linha: *glLineWidth()*

Linhas: GL_LINE_LOOP



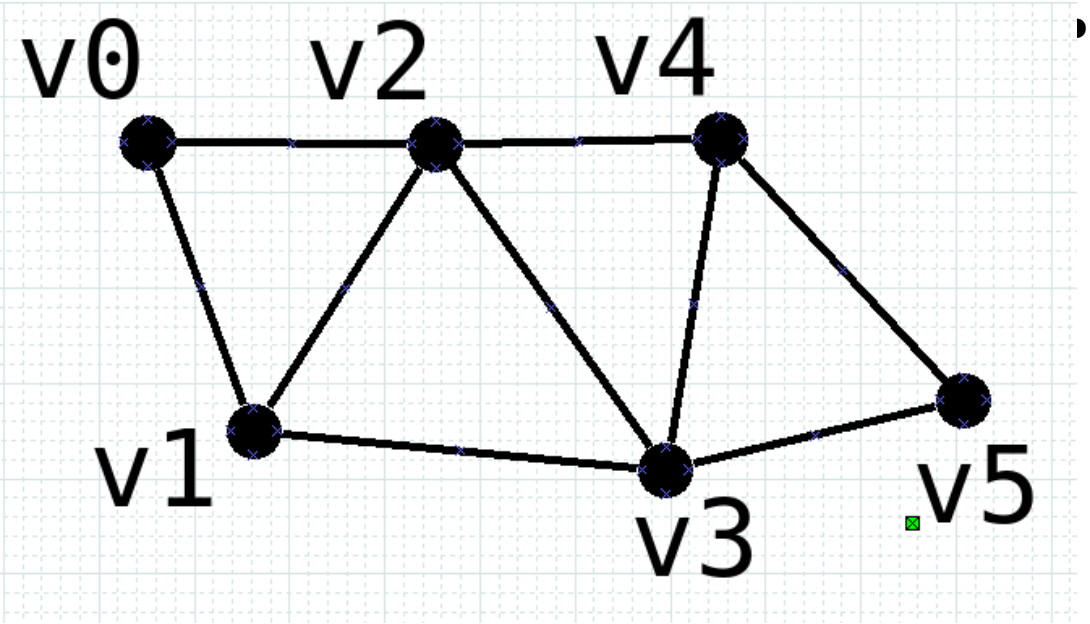
- Vértices são interligados sucessivamente e o último é ligado ao primeiro
- Configuração de largura da linha: *glLineWidth()*

Triângulos: GL_TRIANGLES



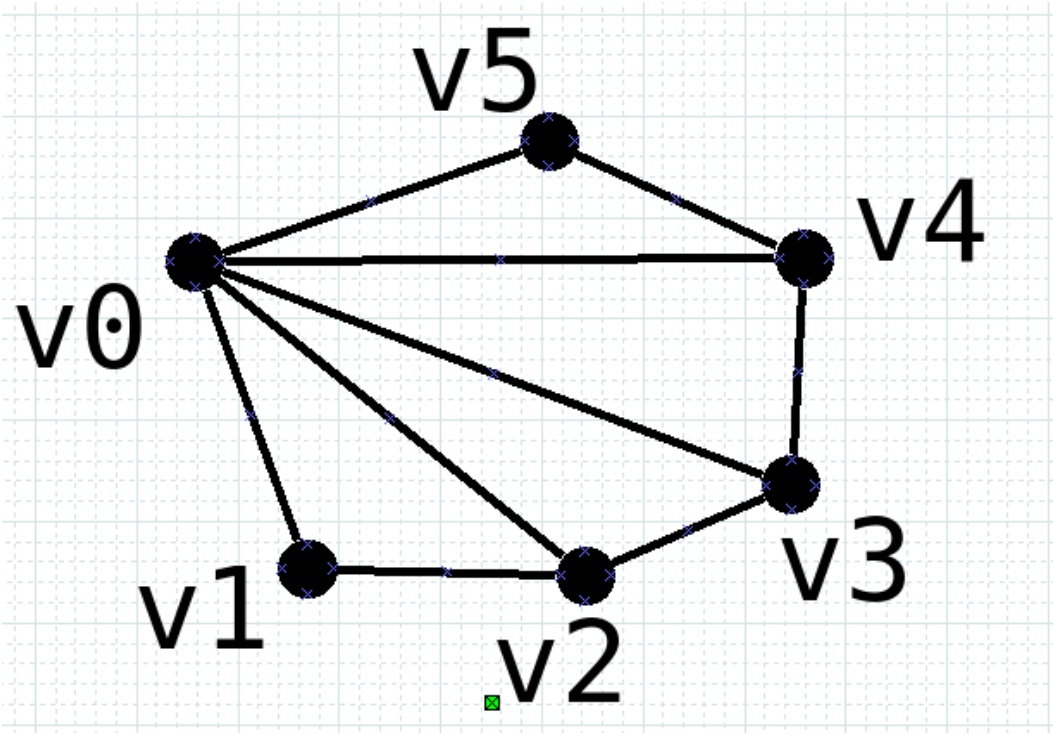
- Três vértices são interligados sucessivamente
- Se a quantidade de vértices não é múltipla de 3, os vértices que sobram são descartados

Triângulos: GL_TRIANGLE_STRIP



- Vértices formam triângulos sucessivamente
  - v0, v1, v2
  - v1, v2, v3
  - ....

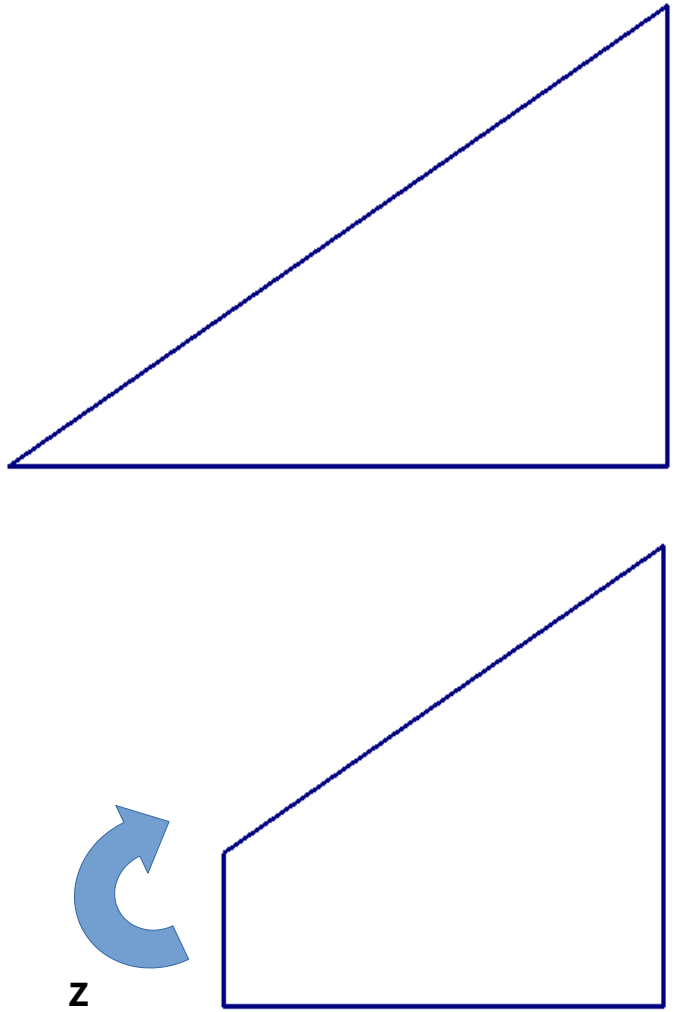
Triângulos: GL_TRIANGLE_FAN



- Vértices formam triângulos sucessivamente;
- Todos incluem o v0:
  - v0, v1, v2
  - v0, v2, v3
  - ....

Clipping

- Durante os estágios de processamento gráfico, eventuais vértices fora de cena são descartados:
  - No caso da projeção ortográfica, os pontos fora do *viewing box* entram neste critério.
- Experimento: “nariz” do triângulo



Considerações Finais

- As primitivas de pontos, linhas e triângulos normalmente possuem suporte nativo em diferentes tipos de hardware de processamento gráfico;
- A configuração glPolygonMode é útil no suporte ao desenvolvimento (depuração visual);
- As primitivas GL_QUADS e GL_POLYGON foram descontinuadas no OpenGL, mas ainda funcionam em modos de compatibilidade.

Experimentos

1) Utilize um papel quadriculado para fazer rascunhos à mão de vértices de objetos no *viewing box*;

2) Alterne a especificação de pontos, linhas e triângulos para observar as propriedades das principais primitivas de desenho;

3) Observe o efeito do *clipping* (vértices fora de cena são cortados pelo OpenGL);

4) Explore a documentação de *glPolygonMode()*.

Exercícios

1) Elabore uma função para desenhar uma esfera 2D, tendo como argumentos a origem (x,y), o raio e o passo de incremento, que definirá o quão redonda ficará a esfera.

2) Crie uma função desenhar retângulo 2D similar à de exercício anterior (4 argumentos delimitadores min x, min y, max x, max y). No entanto, agora, utilize obrigatoriamente uma primitiva de triângulo.