2D Visualization

Joaquim Madeira

Version 0.4 – October 2020

Overview

- Motivation
- CG APIs
- Geometric Primitives
- 2D Visualization

MOTIVATION

CG Main Tasks

- Modeling
  - Construct individual models / objects
  - Assemble them into a 2D or 3D scene
- Animation
  - Static vs. dynamic scenes
  - Movement and / or deformation
- Rendering
  - Generate final images
  - Where is the observer?
  - How is he / she looking at the scene?

Why learn 2D first ?

- A good stepping stone towards 3D
- Many issues easier to understand in 2D

- No need to simulate cameras / light sources / light interaction with objects / …

- 2D is still important and the most common use of CG
  - User interfaces / browsers / documents / …

[techarx.com]

COMPUTER GRAPHICS APIS

Computer Graphics APIs

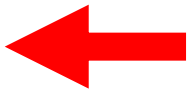
- Create 2D / 3D scenes from simple primitives
- OpenGL / WebGL
  - Rendering
  - No modeling or interaction facilities
- Direct 3D – Microsoft
- VTK
  - 3D CG + Image processing + Visualization
- …

Why OpenGL / WebGL ?

- Industry standard for real-time 2D / 3D CG
- Available on most platforms
  - Desktop / laptop operating systems
  - Mobile devices – OpenGL ES
  - Browsers – WebGL
- Immediate-mode graphics API
  - Applications direct OpenGL to draw the primitives

Why OpenGL / WebGL ?

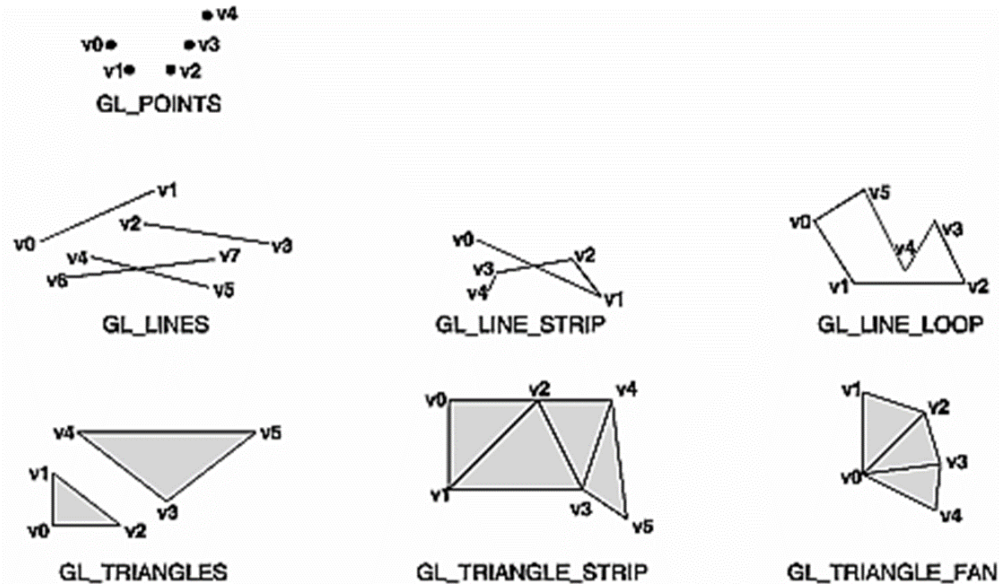
- Older API (OpenGL 1.0) provides features for rapid prototyping
- Newer API provides more flexibility and control



- Supports programmable hardware
  - Modern graphics cards are miniature, highly parallel computers themselves, with many-core GPUs, on-board RAM, etc.

GPUs

- GPUs are a large collection of highly parallel high-speed arithmetic units; several thousand cores!
- GPUs run simple programs ("shaders"):
  - Take in vertices and other data
  - Output a color value for an individual pixel
- GLSL, (O)GL Shader Language, is a C-like language; controls arithmetic pipelines



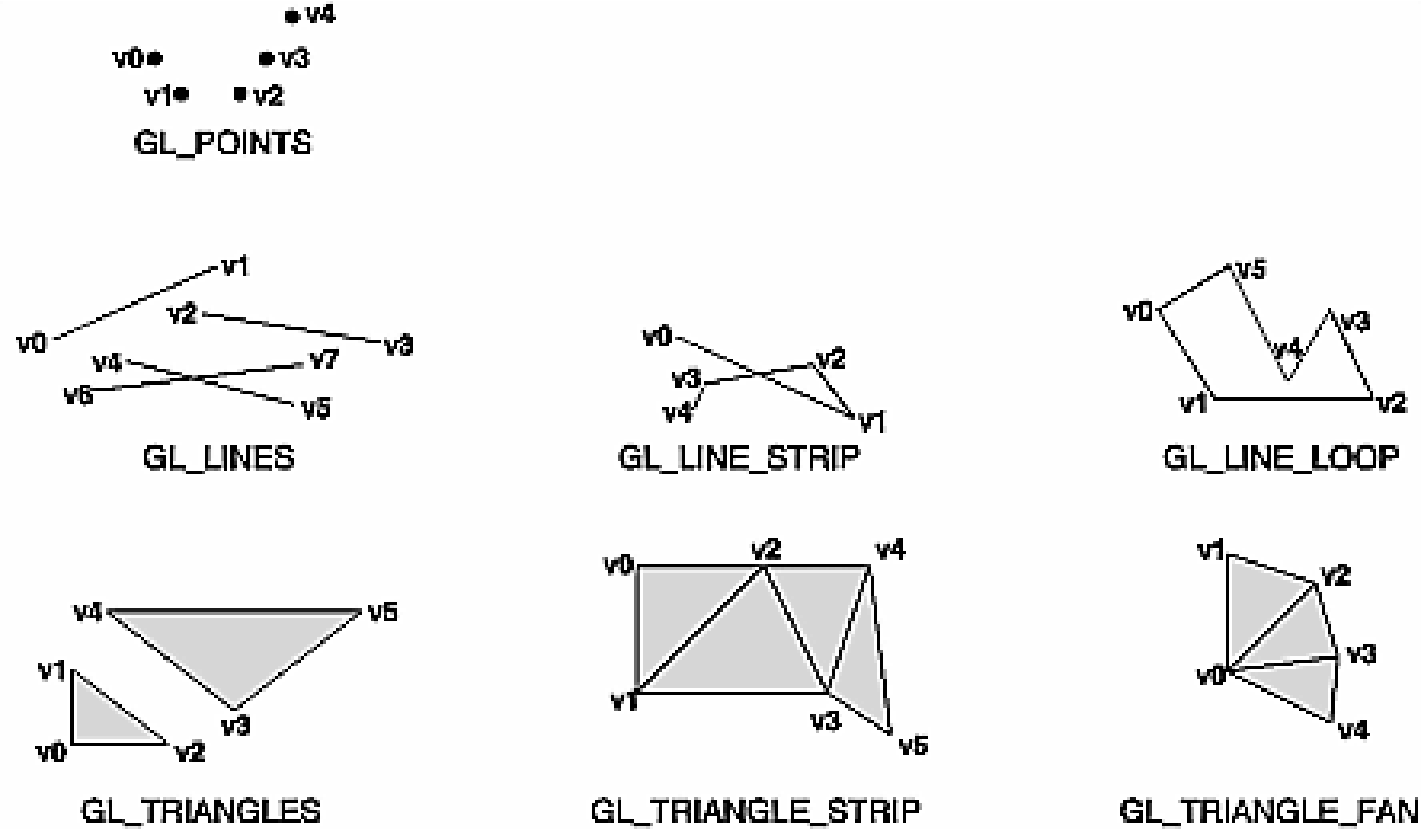
[OpenGL – The Red Book]

GEOMETRIC PRIMITIVES

Geometric Primitives

- Simple primitives
  - Points
  - Line segments
  - Polygons
- Geometric primitives
  - Parametric curves / surfaces
  - Cubes, spheres, cylinders, etc.

OpenGL / WebGL Primitives



[OpenGL – The Red Book]

Attributes

- Determine the appearance of primitives
  - Color
  - Size and width
  - Stipple pattern
  - Polygon mode
    - Display as filled: solid color or stipple pattern
    - Display edges
    - Display vertices
- WebGL supports only a few (gl_PointSize) !

OpenGL Primitives

- Set of points
  - GL_POINTS
  - Point coordinates
  - Color
  - Marker size
  - Marker shape

OpenGL Primitives

- Line segments
  - GL_LINES
  - GL_LINE_STRIP and GL_LINE_LOOP
  - Vertex coordinates
  - Color
  - Width
  - Stipple

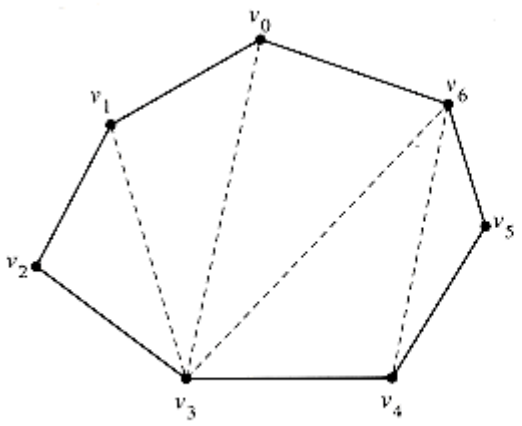
OpenGL Primitives

- Triangles
  - GL_TRIANGLES
  - GL_TRIANGLE_STRIP + GL_TRIANGLE_FAN
  - Fill color and edge color
  - Drawing mode
- General Polygons
  - Define as sets of triangles

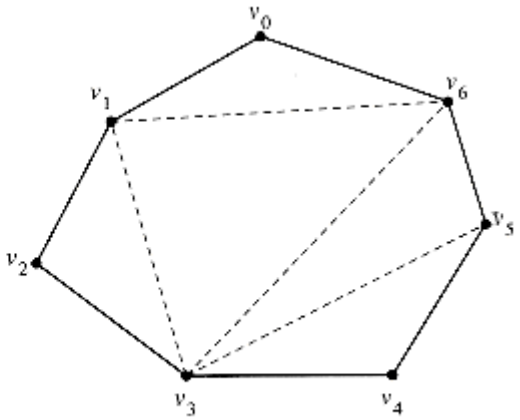
Why triangles?

- Triangles are
  - Simple
    - Edges cannot cross
  - Convex
    - All points on a line segment between two points on a triangle also belong to the triangle
  - Flat
    - All triangle points belong to the same plane
- Application program must tesselate a polygon into triangles

Triangulations

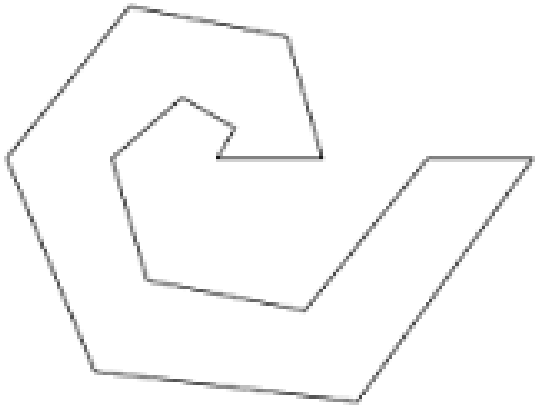


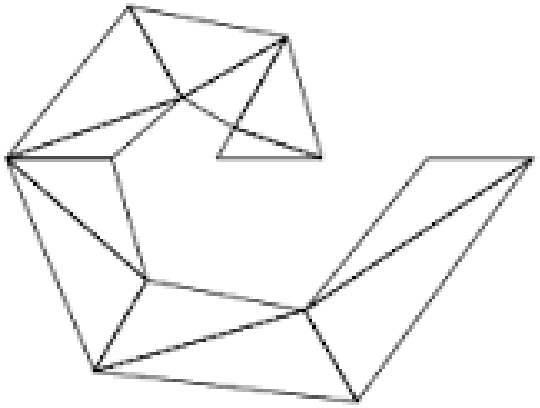
(a)



(b)

[ustc.edu.cn]





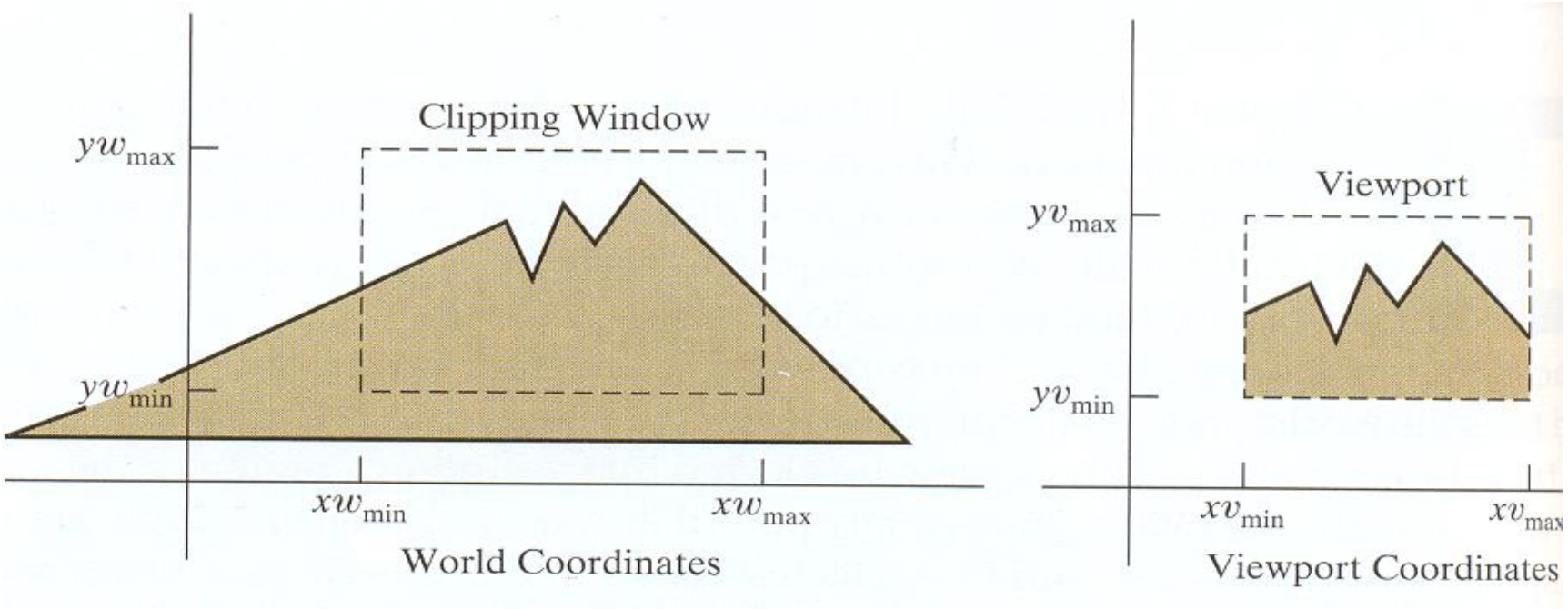
[Mathworld]

2D VISUALIZATION

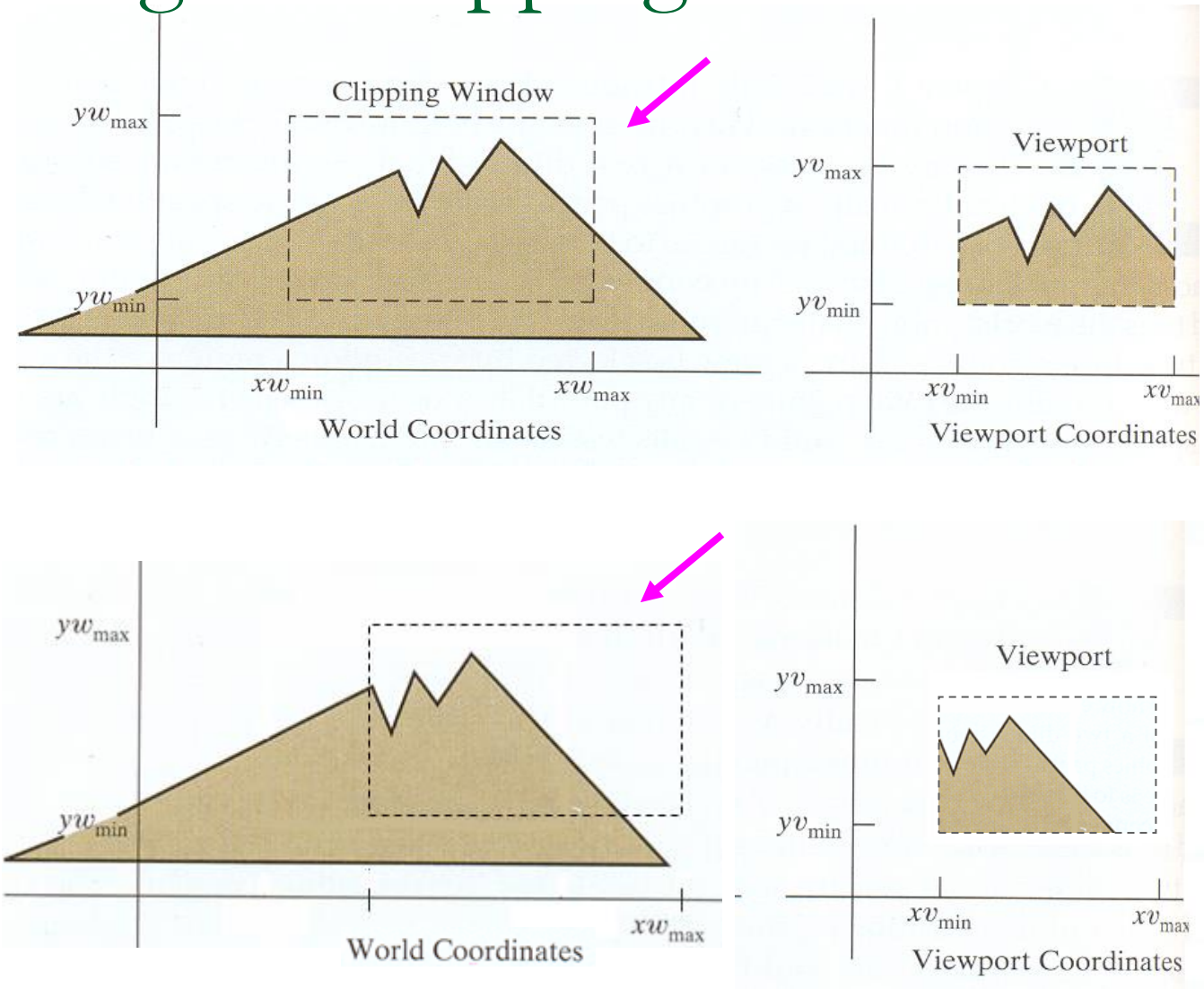
2D Visualization

- Define a 2D scene in the world coordinate system
- Select a clipping window in the XOY plane
  - The window contents will be displayed
- Select a viewport in the display
  - The viewport displays the contents of the clipping window

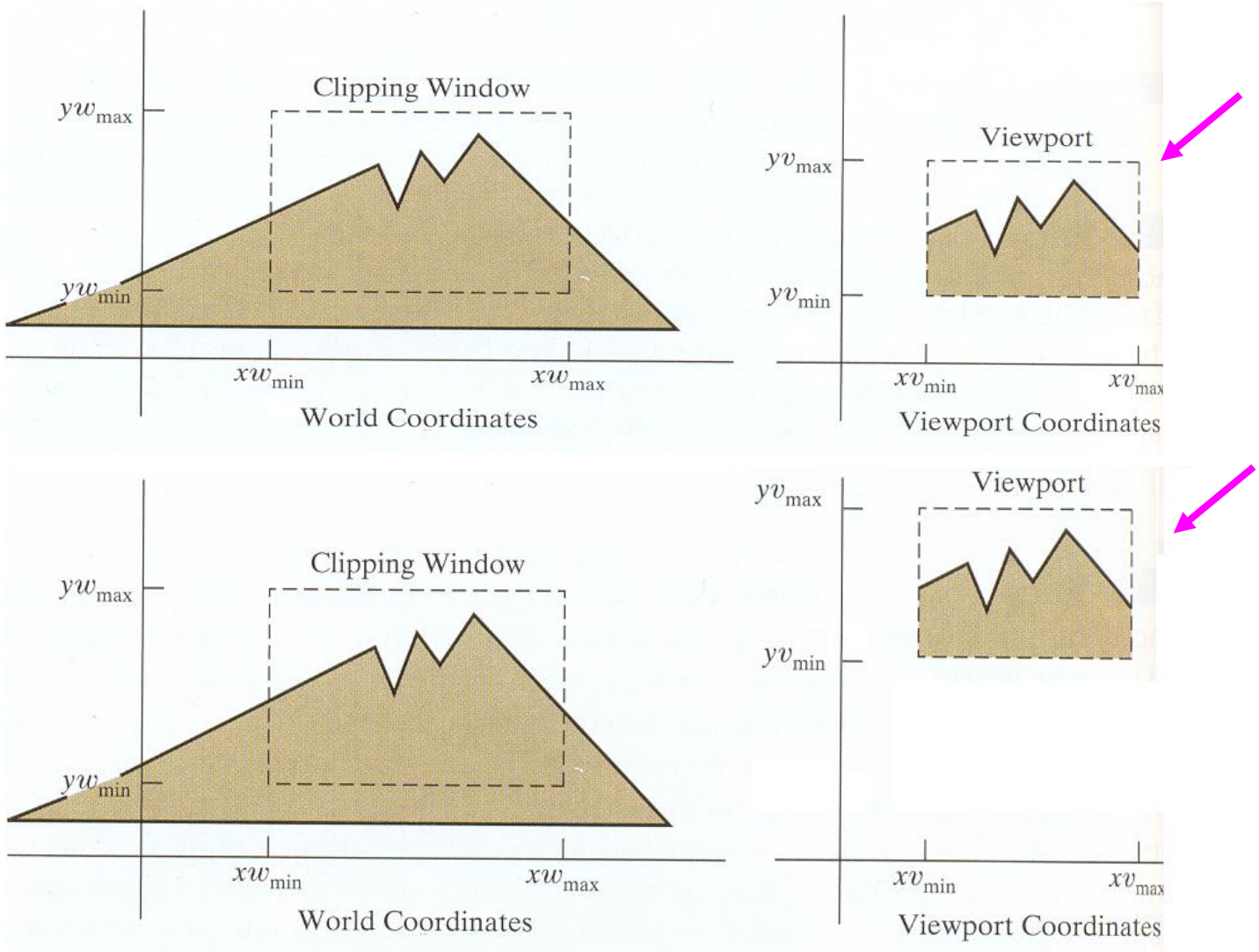
Clipping Window to Viewport Mapping



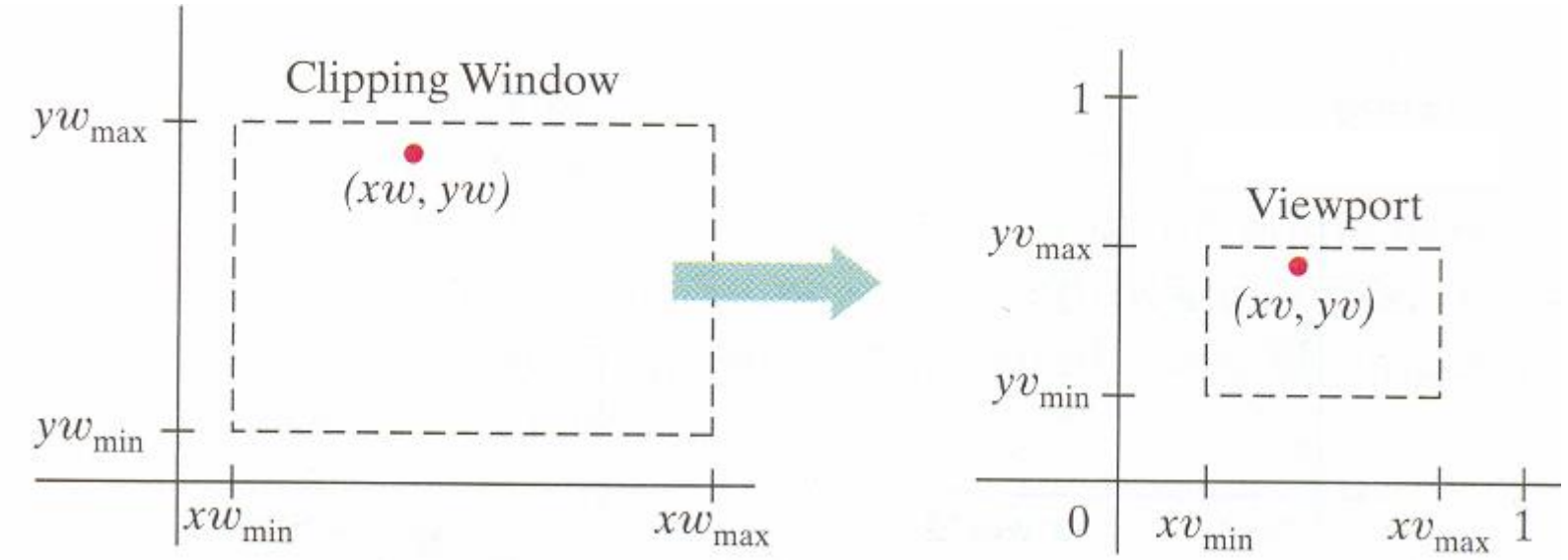
Moving the Clipping Window



Choosing a different Viewport



Coordinate Mapping



Coordinate Mapping

- Keep relative lengths !

$$\frac{xv - xv_{\min}}{xv_{\max} - xv_{\min}} = \frac{xw - xw_{\min}}{xw_{\max} - xw_{\min}}$$

$$\frac{yv - yv_{\min}}{yv_{\max} - yv_{\min}} = \frac{yw - yw_{\min}}{yw_{\max} - yw_{\min}}$$

$$xv = s_x\, xw + t_x$$

$$yv = s_y\, yw + t_y$$

Coordinate Mapping

- Scaling factors and displacements

$$sx = \frac{xv_{\max} - xv_{\min}}{xw_{\max} - xw_{\min}} \qquad sy = \frac{yv_{\max} - yv_{\min}}{yw_{\max} - yw_{\min}}$$

$$t_x = \frac{xw_{\max}xv_{\min} - xw_{\min}xv_{\max}}{xw_{\max} - xw_{\min}}$$

$$t_y = \frac{yw_{\max}yv_{\min} - yw_{\min}yv_{\max}}{yw_{\max} - yw_{\min}}$$

OpenGL

- Default definitions are mostly used!
- Square clipping window
  - Corners: (-1, -1) and (+1, +1)
- Viewport occupies the whole display window
- Changes can be made:
  - The viewport might occupy only part of the display window
  - The clipping window can be larger or smaller
- Pay attention to the aspect ratios !

References

- E. Angel and D. Shreiner, *Interactive Computer Graphics,* 7th Ed., Addison-Wesley, 2015
- D. Hearn and M. P. Baker, *Computer Graphics with OpenGL*, 3rd Ed., Addison-Wesley, 2004
- J. Foley et al., *Introduction to Computer Graphics*, Addison-Wesley, 1993