2D Viewing
2D Viewing
2D Viewing

- How do we specify a VIEW of a scene
  (1) the part of a scene (world) to display
     ("window")
  (2) the place to display that part on the screen
     ("viewport")

- Typically, the scene (world) is defined in
  and convenient coordinate system
  — "world coordinate system"

- The viewport is generally specified in
  ([0, 1], [0, 1]) —
  "normalized device coordinate system"

- These coordinates are mapped to integer pixel
  coordinates — "device coordinate system"
2D-Viewing Pipeline

- **MC** (model coordinate system), models ⇒
- **WC** (world coordinate system), world ⇒
- **VC** (view coordinate system) ⇒
- **NDC** (normalized device coordinate system) ⇒
- **DC** (device coordinate system)
From WC to VC

\[(x_{wc}, y_{wc})\]

\[(x_{vc}, x_{vc})\]
From WC to VC

- From model coordinates to world coordinates
  \[ Obj_{wc} = M_{mc,wc}Obj_{mc} \]

- From world coordinates to viewing coordinates
  \[ M_{wc,vc} = R(\theta) \ast T(\delta x, \delta y) \]
  \[ Obj_{vc} = M_{wc,vc}Obj_{wc} \]
Window and Viewport

- How do we specify a window?

\[ x_{w,\text{min}}, \]

\[ x_{w,\text{max}}, \]

\[ y_{w,\text{min}}, \]

\[ y_{w,\text{max}} \]

- How do we specify a viewport

\[ x_{v,\text{min}}, \]

\[ x_{v,\text{max}}, \]

\[ y_{v,\text{min}}, \]

\[ y_{v,\text{max}} \]

- Oftentimes, we want to keep the same relative placement of \((x_w, y_w)\) in the window to
\((x_v, y_v)\) in the viewport then we must have

\[
\frac{x_v - x_{v,\min}}{x_{v,\max} - x_{v,\min}} = \frac{x_w - x_{w,\min}}{x_{w,\max} - x_{w,\min}}
\]

\[
\frac{y_v - y_{v,\min}}{y_{v,\max} - y_{v,\min}} = \frac{y_w - y_{w,\min}}{y_{w,\max} - y_{w,\min}}
\]

- We shall solve for \((x_v, y_v)\)

\[
x_v = x_{v,\min} + (x_w - x_{w,\min}) s_x
\]

\[
y_v = y_{v,\min} + (y_w - y_{w,\min}) s_y
\]

where scaling factors are

\[
s_x = \frac{x_{v,\max} - x_{v,\min}}{x_{w,\max} - x_{w,\min}}
\]

\[
s_y = \frac{y_{v,\max} - y_{v,\min}}{y_{w,\max} - y_{w,\min}}
\]

- To maintain relative proportions of objects in this window-to-viewport transformation, we need

\[
s_x = s_y
\]
Viewport

viewport
Device Coordinate System

- From normalized device coordinate system to device coordinate system

- We need another
  window-to-viewport transformation

- Why?
  separate transformations from device-dependent requirements
  graphics packages become “device-independent”
  different devices can be used by providing appropriate device drivers
Device Viewport

device viewport

monitor
Clipping

- Which part(s) of an object should be on or off the screen?