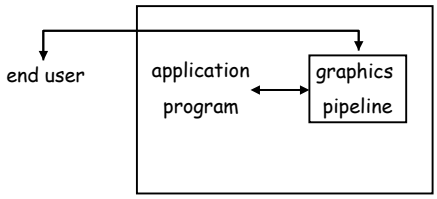
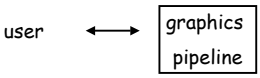
cs155 – z sweedyk

graphics
pipeline
systems

who's who



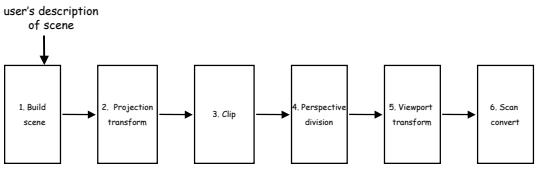
who's who for today



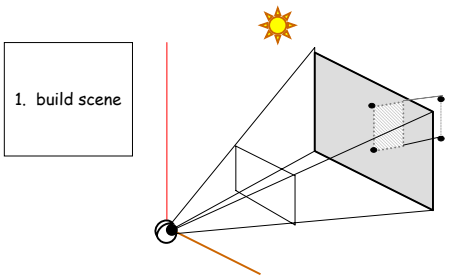
user defined scene description

- models
- lights
- view (eye/camera)

graphics pipeline



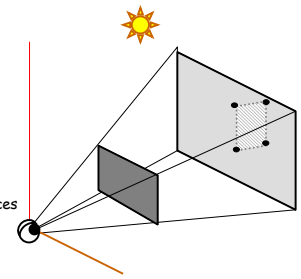
graphics pipeline 1



graphics pipeline 2

2. projection transform

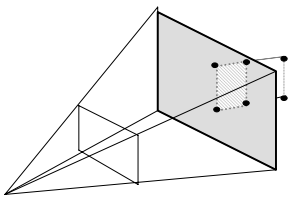
transform vertices to (almost) projected vertices



10/27/2003

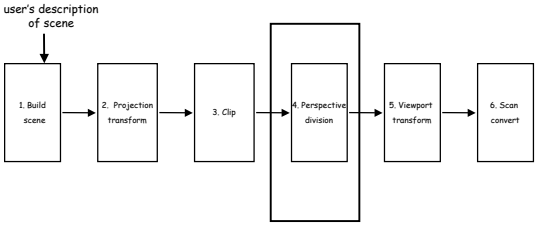
graphics pipeline 3

3. clip



10/27/2003

graphics pipeline

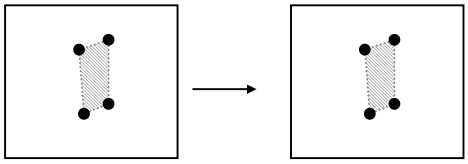


hard to summarize

10/27/2003

graphics pipeline 5

5. viewport transform



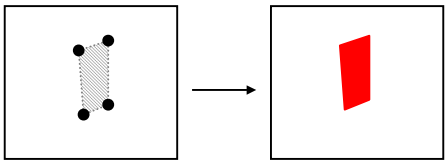
vertices in modified world coordinates

vertices in image coordinates

10/27/2003

graphics pipeline 6

6. scan convert

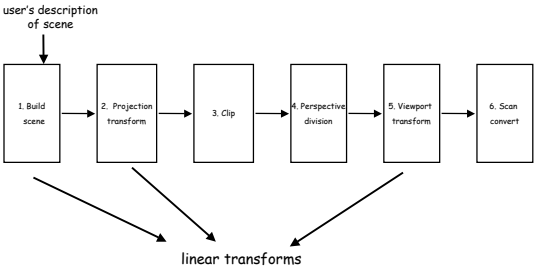


vertices in image coordinates

drawn image in frame buffer

10/27/2003

graphics pipeline

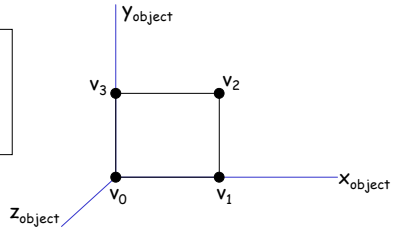


10/27/2003

build scene

User described primitives

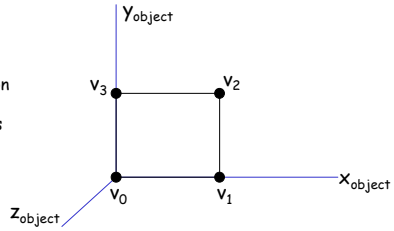
Object coordinates



v_0= (0,0,0), v_1= (1,0,0), v_2= (1,1,0), v_3= (0,1,0)

build scene

pipeline representation is in homogeneous coordinates



v_0= (0,0,0,1), v_1= (1,0,0,1), v_2= (1,1,0,1), v_3= (0,1,0,1)

primitives

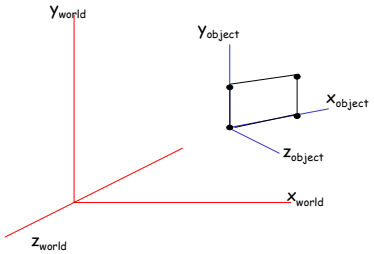
- points
- line segments
- polygons

build scene

User described modeling transforms

World coordinates

Vertex in world coordinates: M_{WV}



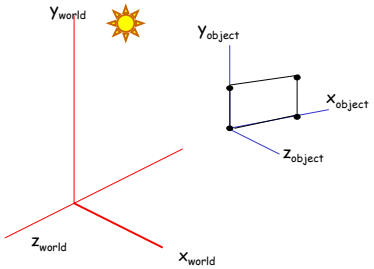
modeling transforms

- scale
- rotate
- translate

build scene

User described lights

(Usually) World coordinates



Lights

- Ambient
- Directional
- Point
- Spot

Exercise (teams of 2)

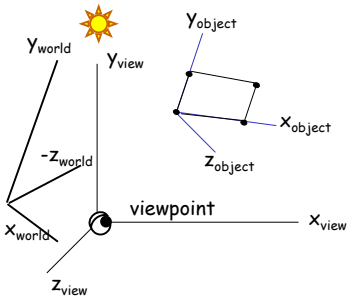
- Vertex in object coordinates: (1,1,2)
- Scale by 2 in x
- Translate by 3 in x and -4 in z
- What is result?
- Write sequence of modeling transform
- Multiply to get composite transform
- Multiply to get vertex in world coordinates
- Put your matrix on the board

build scene

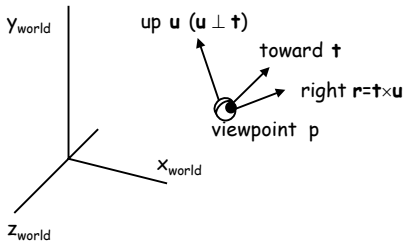
User defined viewpoint
View coordinates

vertex in view coordinates: $M_V M_W v$

lights in view coordinates: $M_V p$, $M_V d$

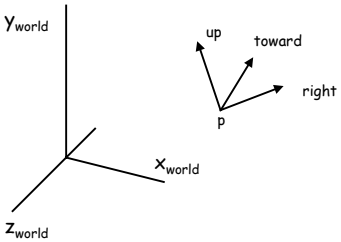


view in world coordinates



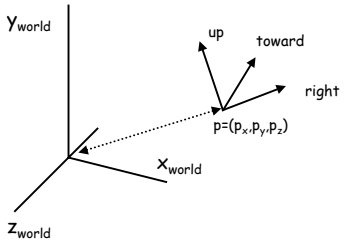
world ↔ view coordinates

translate & rotate: $M_V = M_R M_T$

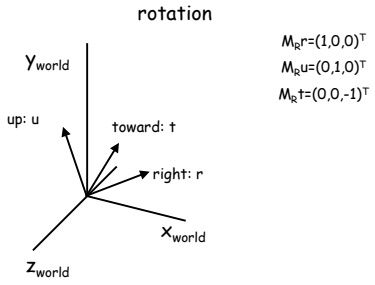


world ↔ view coordinates

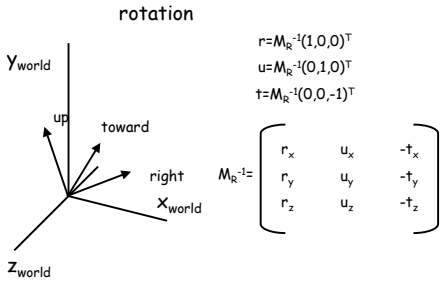
M_T: translate by $(-p_x,-p_y,-p_z)$



world ↔ view coordinates



world ↔ view coordinates



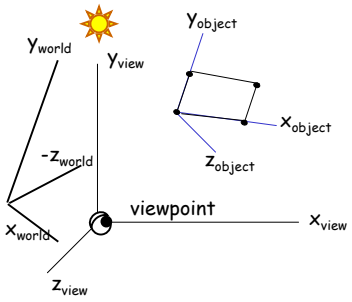
Exercise cont.

- The viewer is at (4,0,-2) looking in the (1,0,0) direction. Up is (0,1,0).
- What is our vertex in view coordinates?
- Write the translation and rotation matrices needed to convert to viewpoint coordinates. (You should be able to compute the inverse matrix by inspection.)
- Multiply to create the composite view transform.
- Multiply to get point in view coordinates.
- Write your view transform on the board.

build scene

vertex in view coordinates: $M_V M_W v$

lights in view coordinates: $M_V p$, $M_V \mathbf{d}$



geometric primitives

object coordinates: v	description of vertex
world coordinates: $M_W v$	description of vertex situated in world
view coordinates: $M_V M_W v$	description of vertex in world as seen from a particular viewpoint

lights

world coordinates: p, **d**	description of light position/direction in world
view coordinates: $M_V p$, $M_V \mathbf{d}$	description of light position/direction in world as seen from a particular viewpoint

note: $M_V \mathbf{d}$ is shorthand for the "multiply vector" operation we've used before!

graphics pipeline