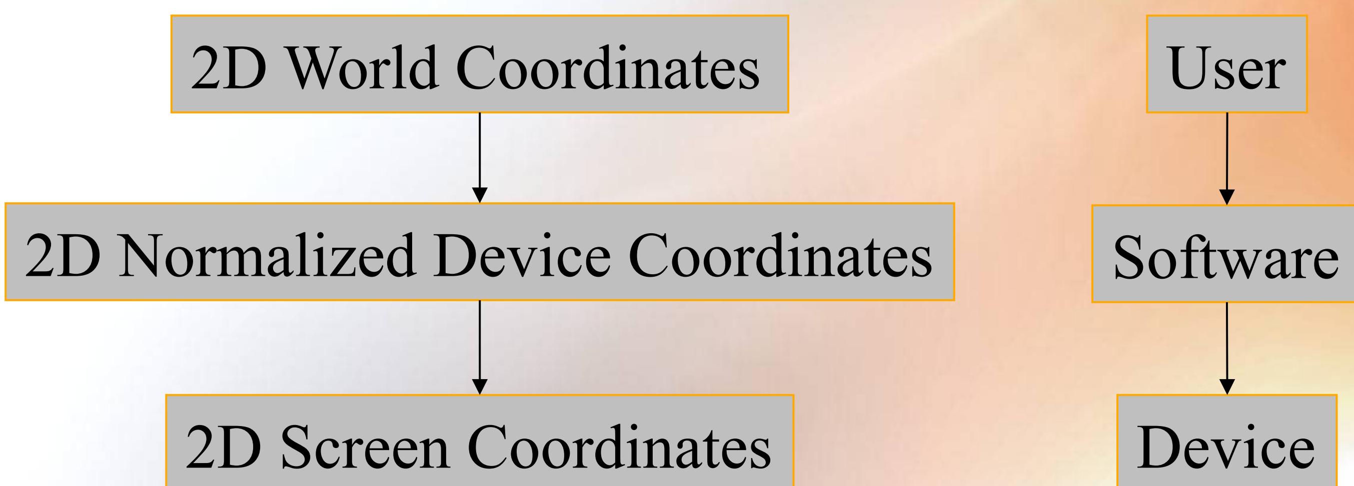
Computer Graphics

2D and 3D Viewing Transformations

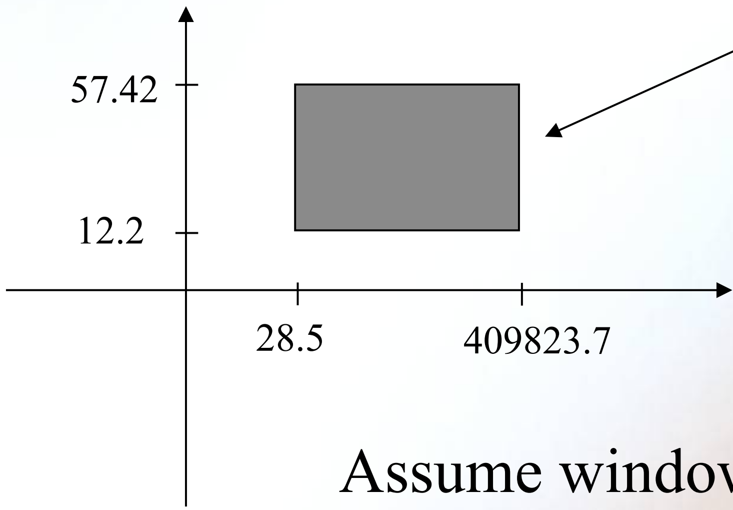
Based on slides by Dianna Xu, Bryn Mawr College

2D Viewing Transformation

- **Converting 2D model coordinates to a physical display device**
  - **2D coordinate world**
  - **2D screen space**
  - **Allow for different device resolutions**



Window: Portion of World Viewed

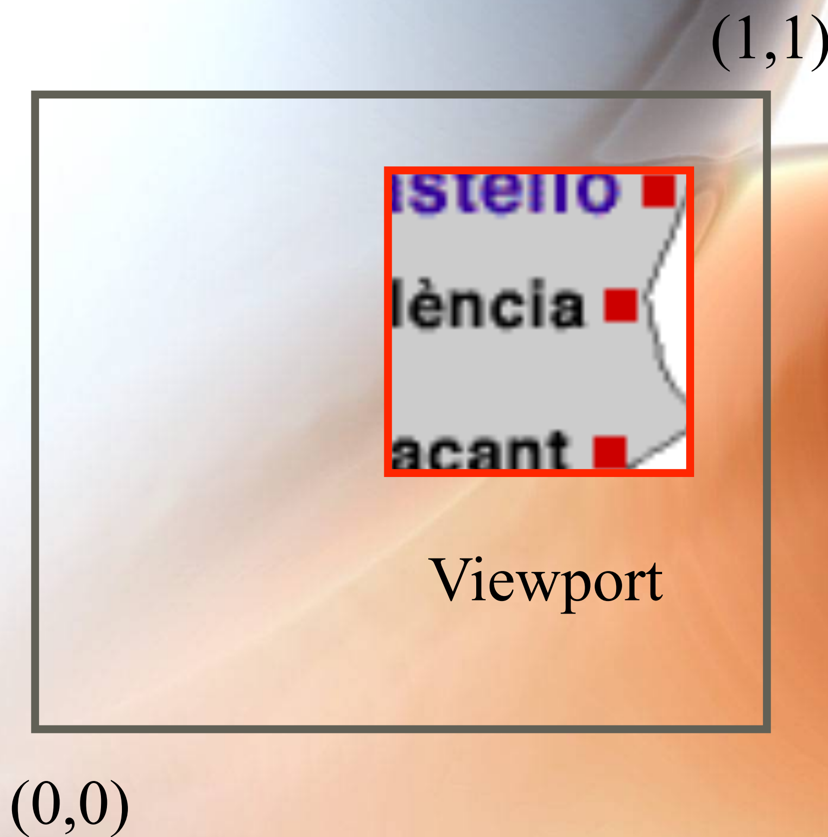


Window = area of interest within world

Assume window is rectangular

World coordinates are chosen at the convenience of the application or user

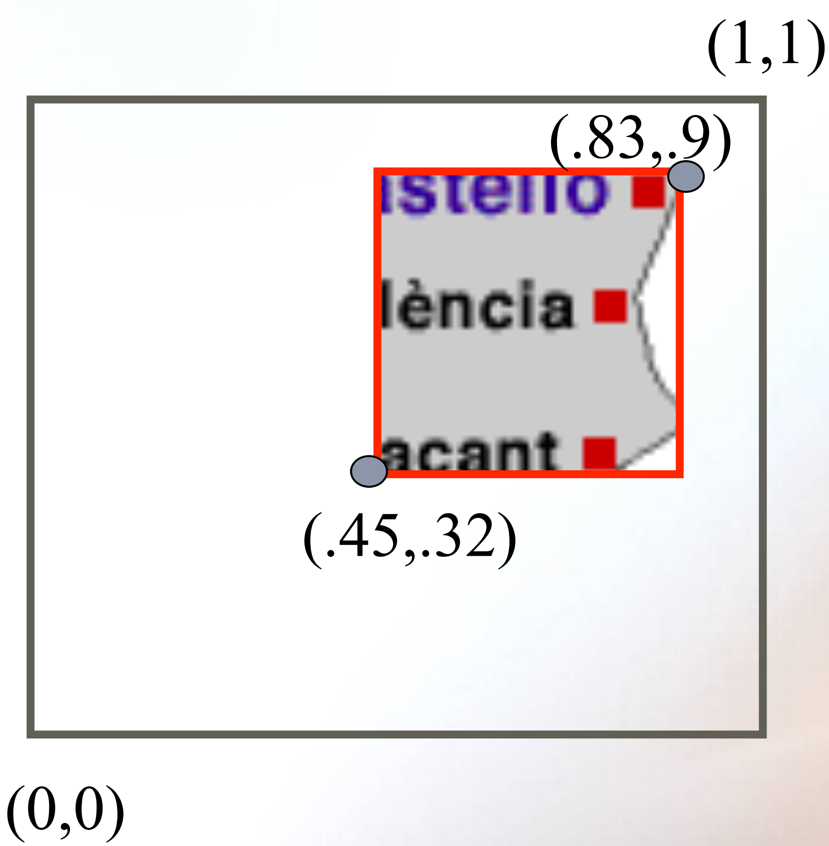
Viewing Transformation (World to NDC)



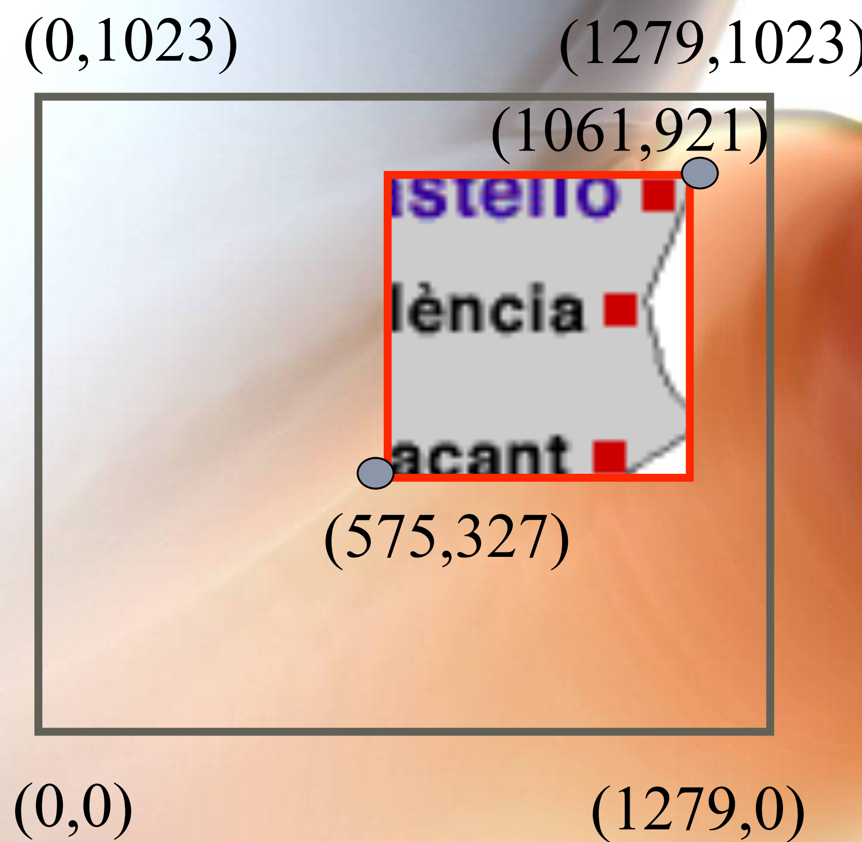
World Coordinates

Normalized Device Coordinates

NDC to Screen



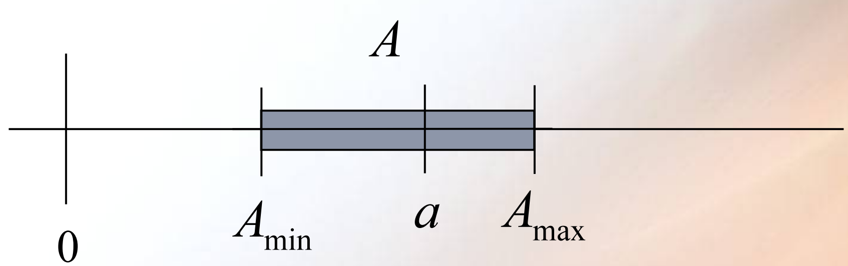
Normalized Device Coordinates



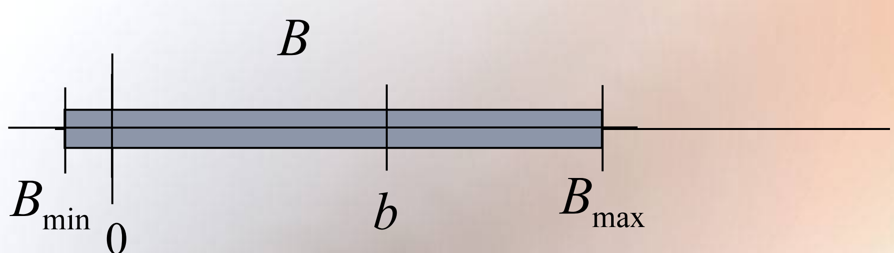
Screen Coordinates

Range Mapping

- Given values in a range A, map them linearly into a (different) range of values B.
- Consider some arbitrary point *a* in A



- Find the image *b* of *a* in B



Solving for the Range Mapping

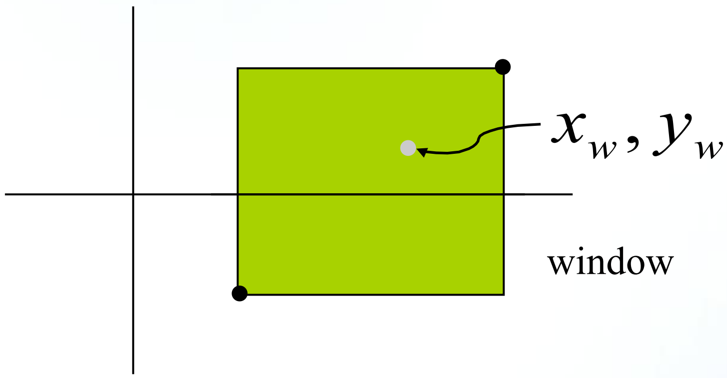
- Using simple proportions: $$\frac{b - B_{\min}}{a - A_{\min}} = \frac{B_{\max} - B_{\min}}{A_{\max} - A_{\min}}$$

- Solving for b: $$b = \frac{B_{\max} - B_{\min}}{A_{\max} - A_{\min}}(a - A_{\min}) + B_{\min}$$

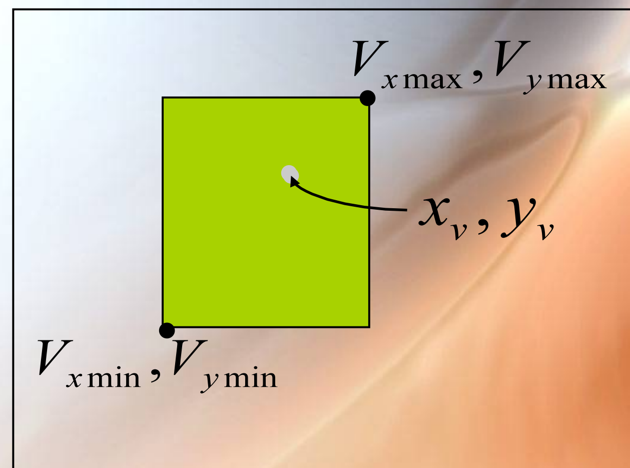
- In terms of transformations, the distance from a to $A_{\min}$ is scaled by the ratio of the two ranges B and A: $$\left(\frac{B_{\max} - B_{\min}}{A_{\max} - A_{\min}}\right)$$

  then translated from the end $B_{\min}$ of B.

The Window to Viewport Transformation



WORLD
COORDINATES

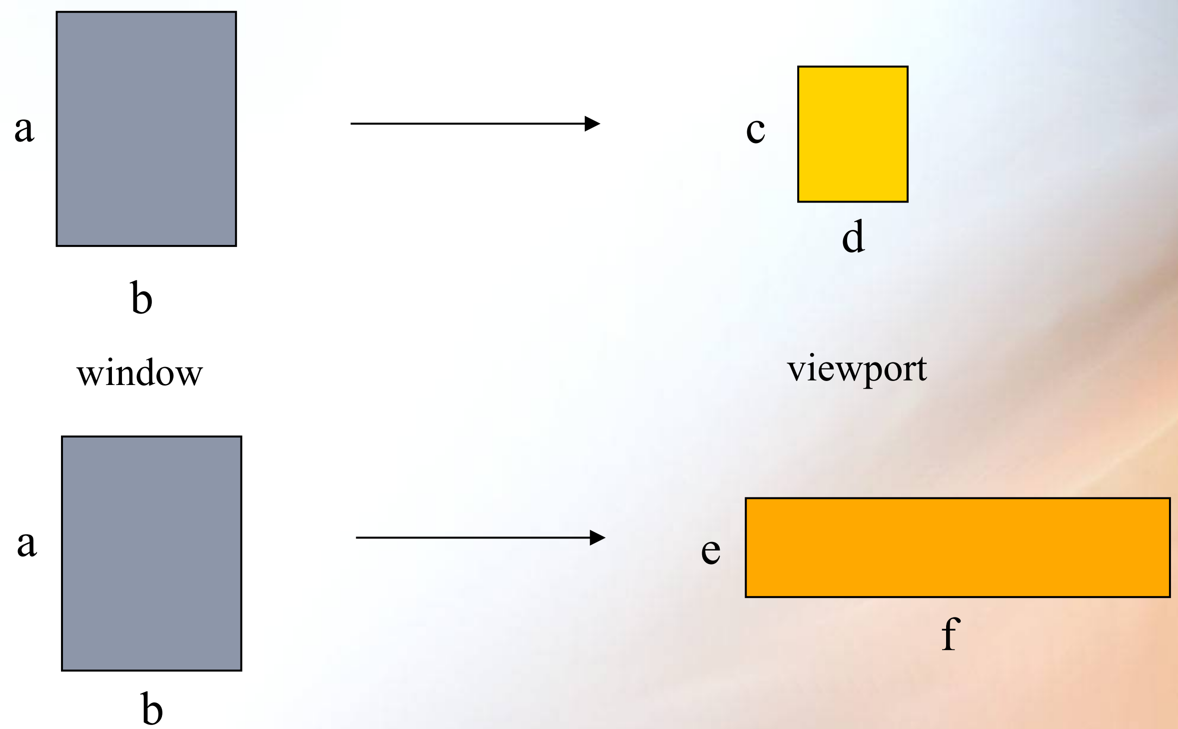


NORMALIZED DEVICE
COORDINATES

$$x_v = \frac{V_{x\max} - V_{x\min}}{W_{x\max} - W_{x\min}}\left(x_w - W_{x\min}\right) + V_{x\min}$$

$$y_v = \frac{V_{y\max} - V_{y\min}}{W_{y\max} - W_{y\min}}\left(y_w - W_{y\min}\right) + V_{y\min}$$

Different Window and Viewport Aspect Ratios



- If $\frac{a}{b} = \frac{c}{d}$ then map causes no distortion
- If $\frac{a}{b} \neq \frac{e}{f}$ then distortion occurs
- To avoid distortion, use $\min\left(\frac{e}{a}, \frac{f}{b}\right)$ as single scale factor in both x, y mapping

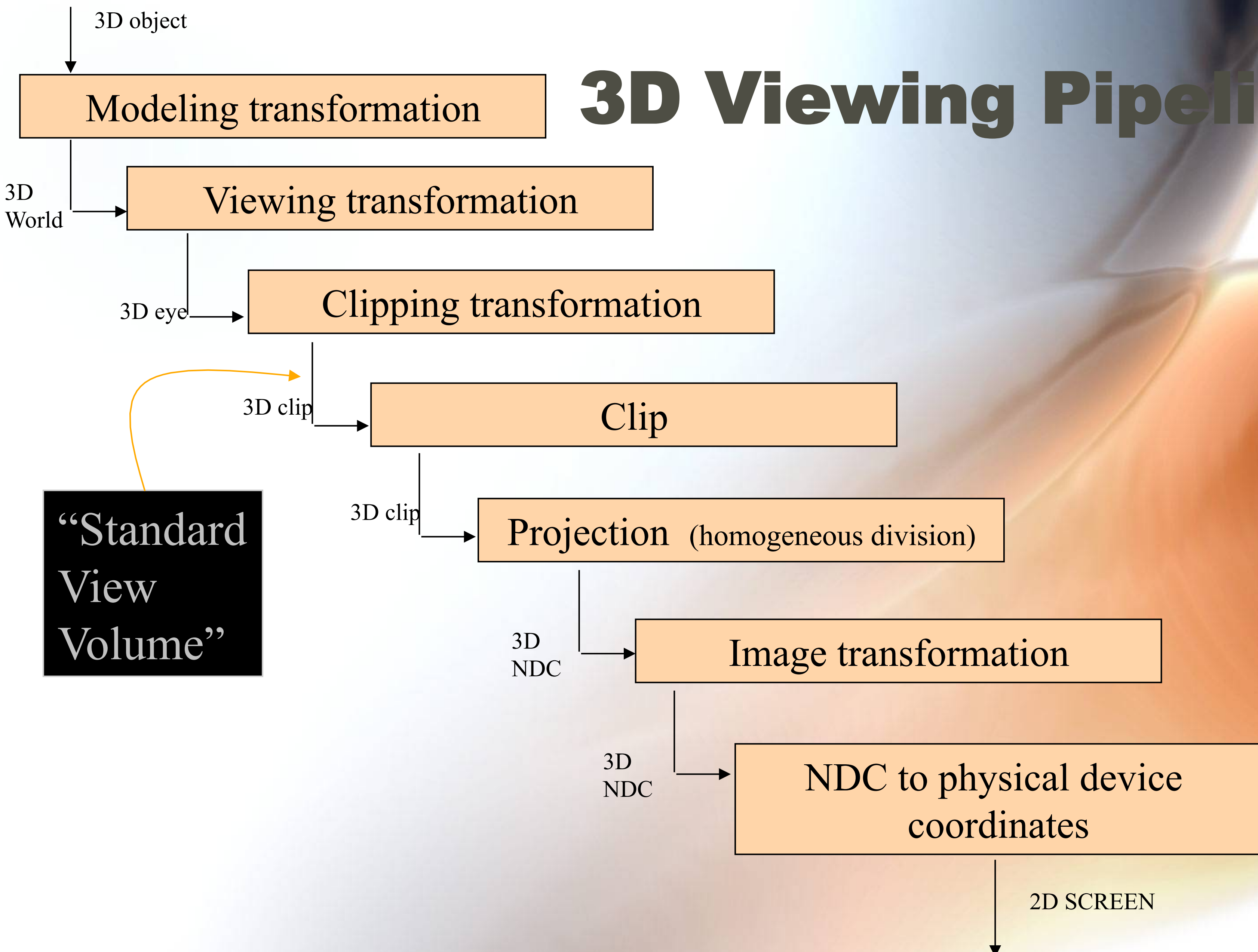
Mapping the Viewport Back into the Window

- **Note that the window-to-viewport transformation can be <u>inverted</u>**

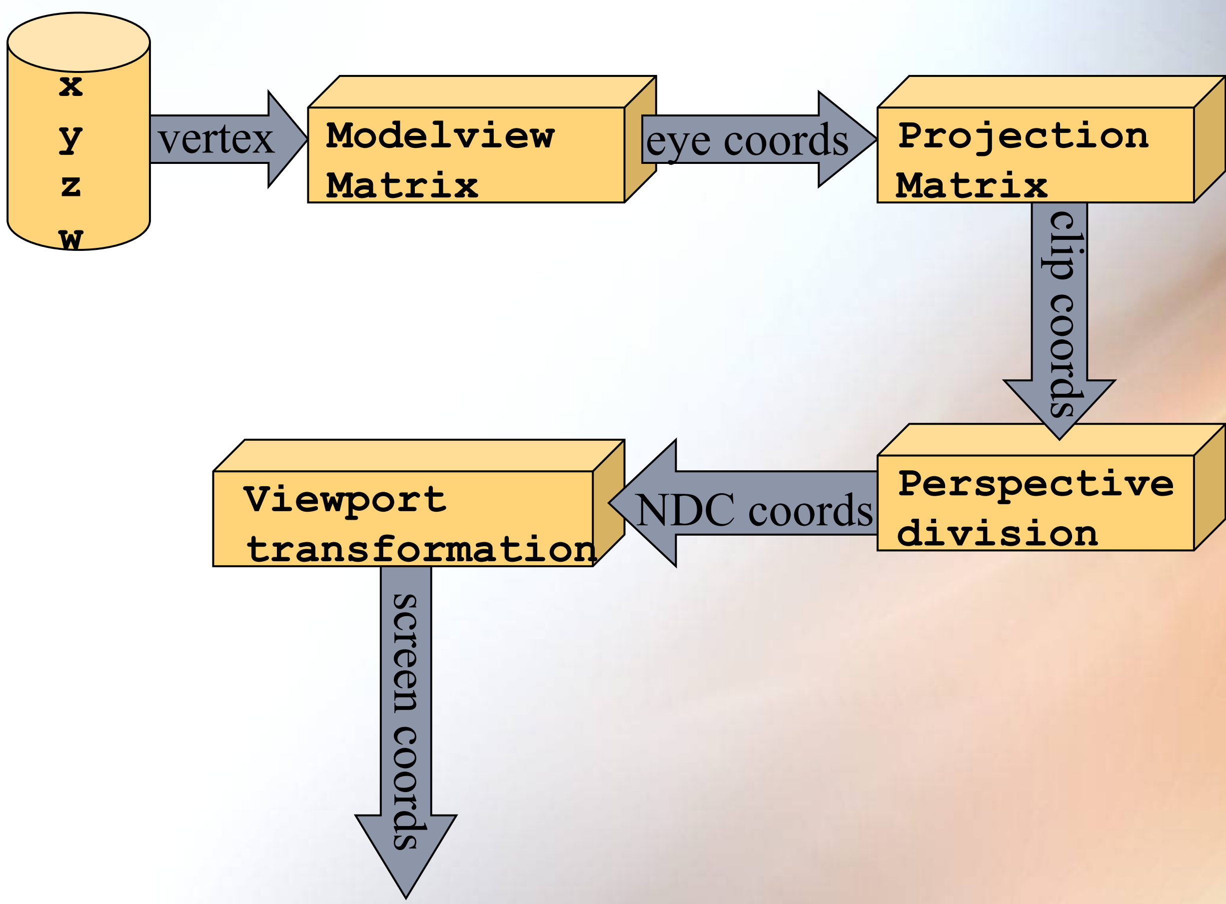
$$x_w = \frac{W_{x\max} - W_{x\min}}{V_{x\max} - V_{x\min}}\left(x_v - V_{x\min}\right) + W_{x\min}$$

$$y_w = \frac{W_{y\max} - W_{y\min}}{V_{y\max} - V_{y\min}}\left(y_v - V_{y\min}\right) + W_{y\min}$$

3D Viewing Pipeline



OpenGL Pipeline

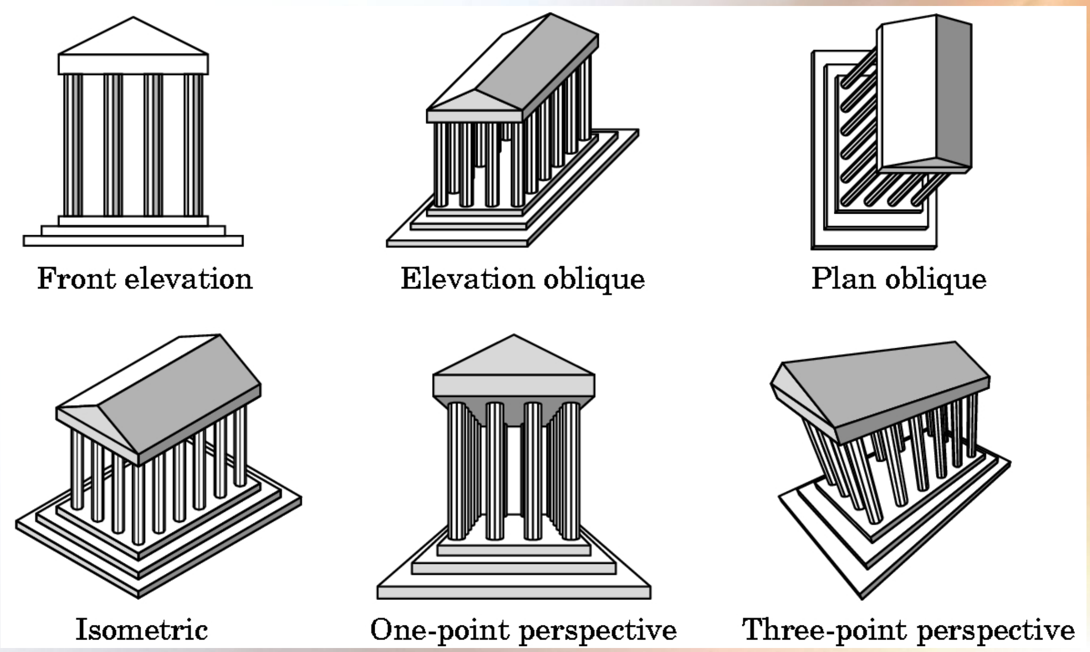


The Camera Analogy

•Modeling transformation →	•Position the object you are photographing
•Viewing transformation →	•Position the viewing volume on the world/ Setting up the tripod
•Projection →	•Lens/zooming
•Viewport transformation →	•Photograph

Classical Viewing

- When an architect draws a building
  - they know which sides they wish to display,
  - and thus where they should place the viewer
  - Each classical view is determined by a specific relationship between the objects and the viewer.

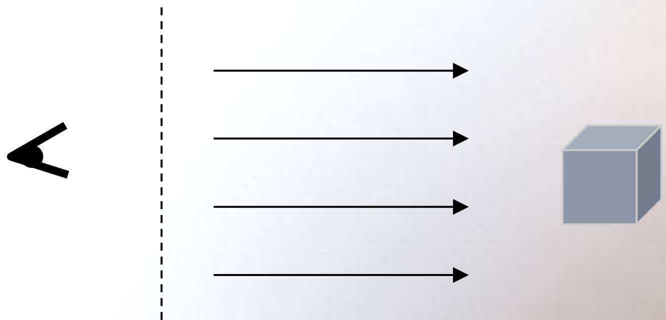


Planar Geometric Projections

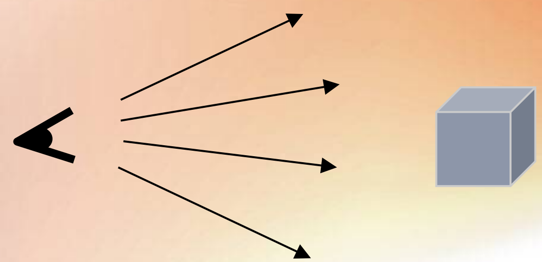
- Standard projections project onto a plane
- Projectors are lines that either
  - converge at a center of projection
  - are parallel
- Such projections preserve lines
  - but not necessarily angles
- Nonplanar projections are needed for applications such as map construction

3D Viewing Transformations

- <u>Parallel</u> or <u>Orthographic</u> projection
  - –Eye at infinity
  - –Need direction of projection
  - –“Projectors” are parallel

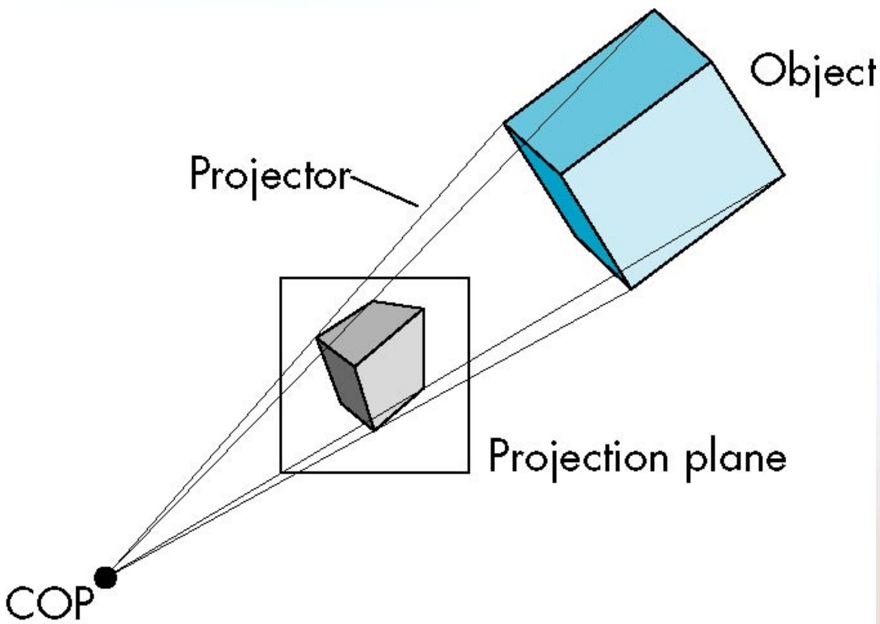


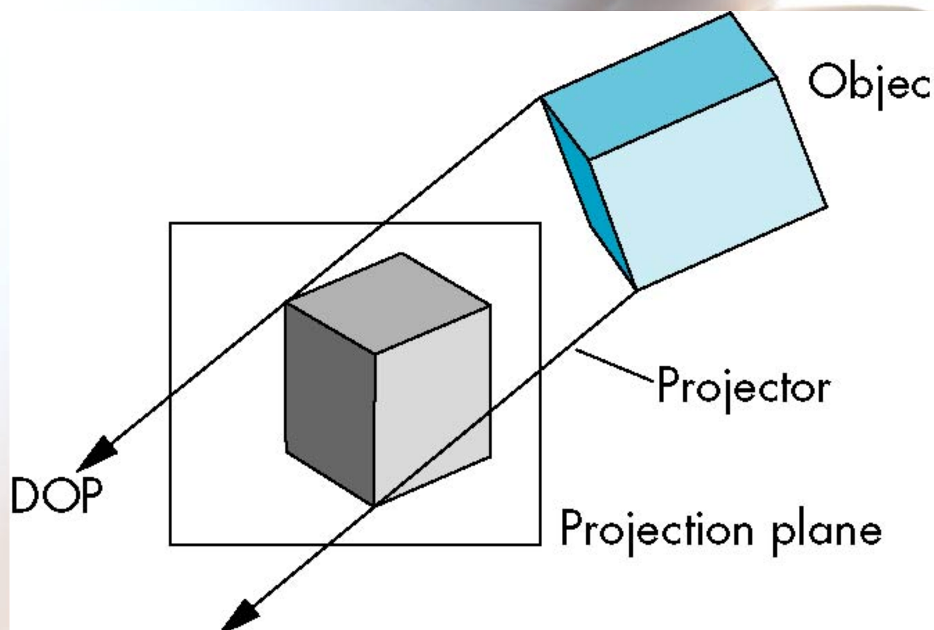
- <u>Perspective</u> or <u>Central</u> projection
  - –Eye at point (x,y,z) in world coordinates
  - –“Projectors” emanate from eye position



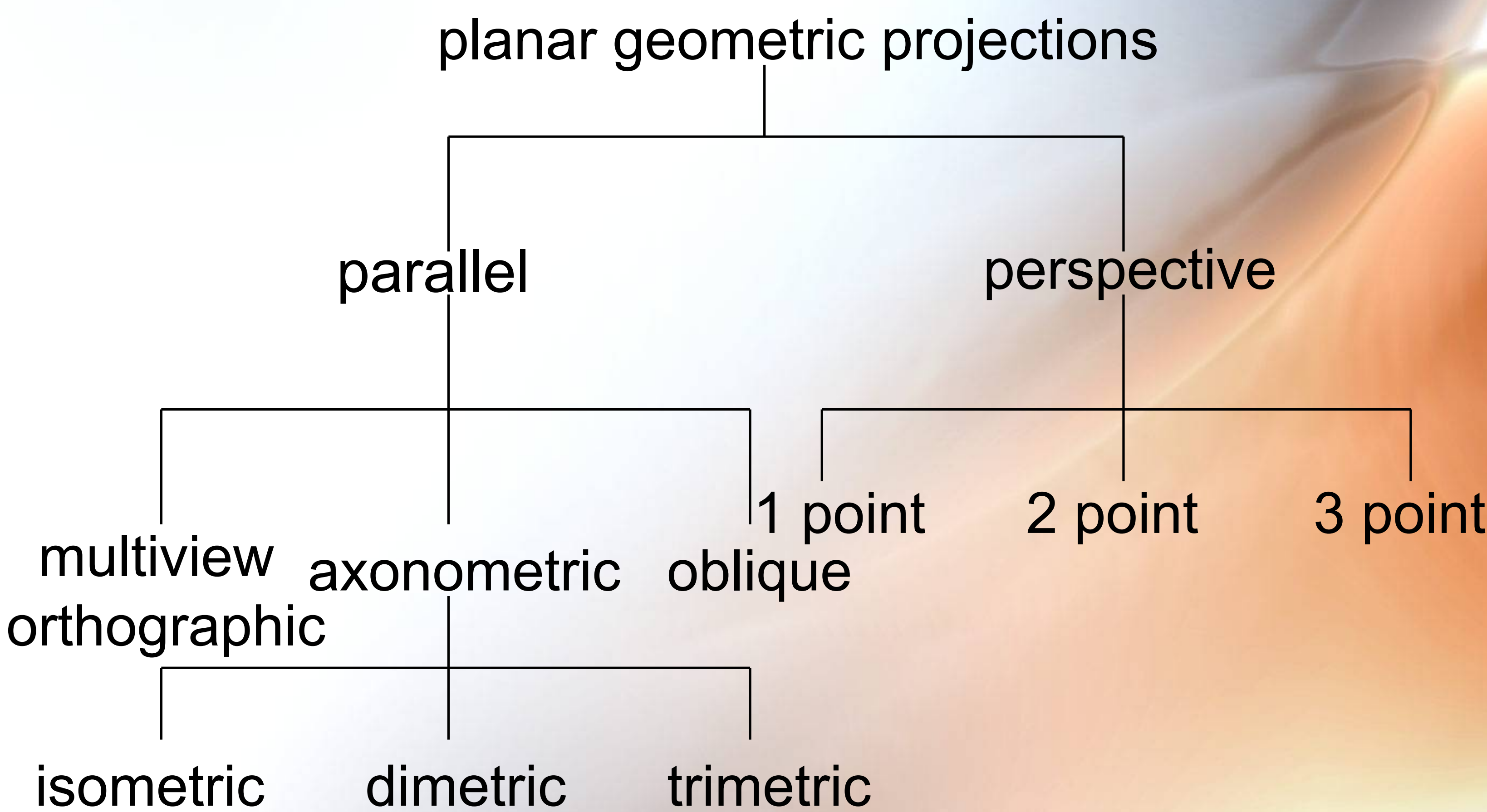
Focal Length/Field-of-View

- **eye near object** **eye at infinity**

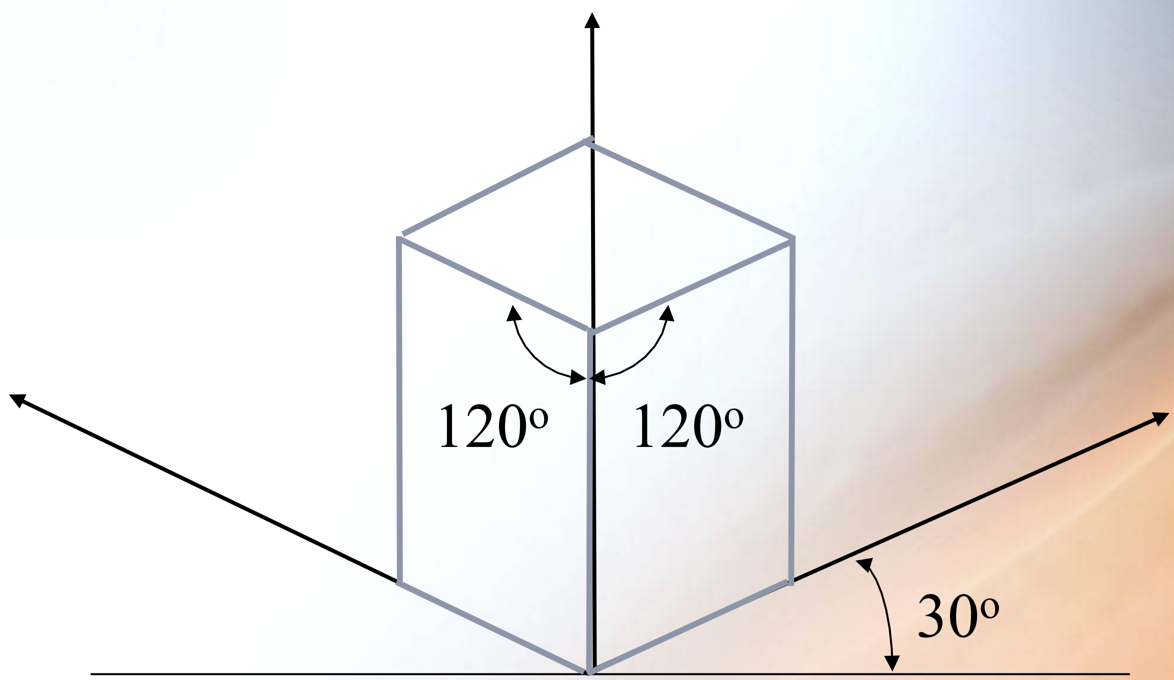




Taxonomy of Planar Geometric Projections



Parallel Projection of Cube



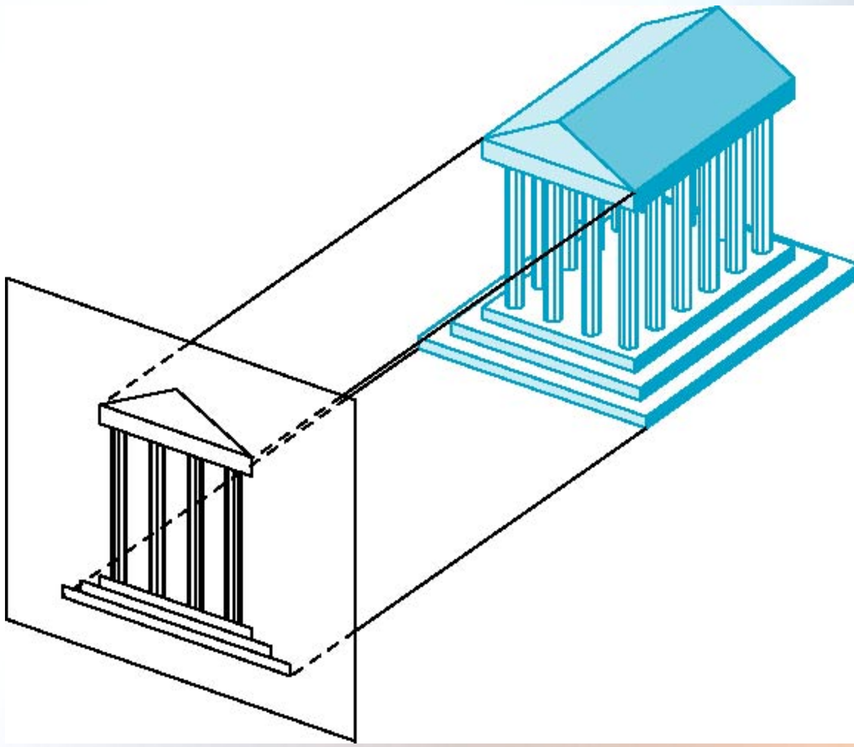
Notice how all parallel line families map into parallel lines in the projection.

Special Parallel Projections

- Orthographic
  - Projection plane is usually parallel to one principal face of the object.
  - Projectors are perpendicular to the projection plane
- Axonometric
  - Also known as the chinese perspective
  - Used in long scroll paintings

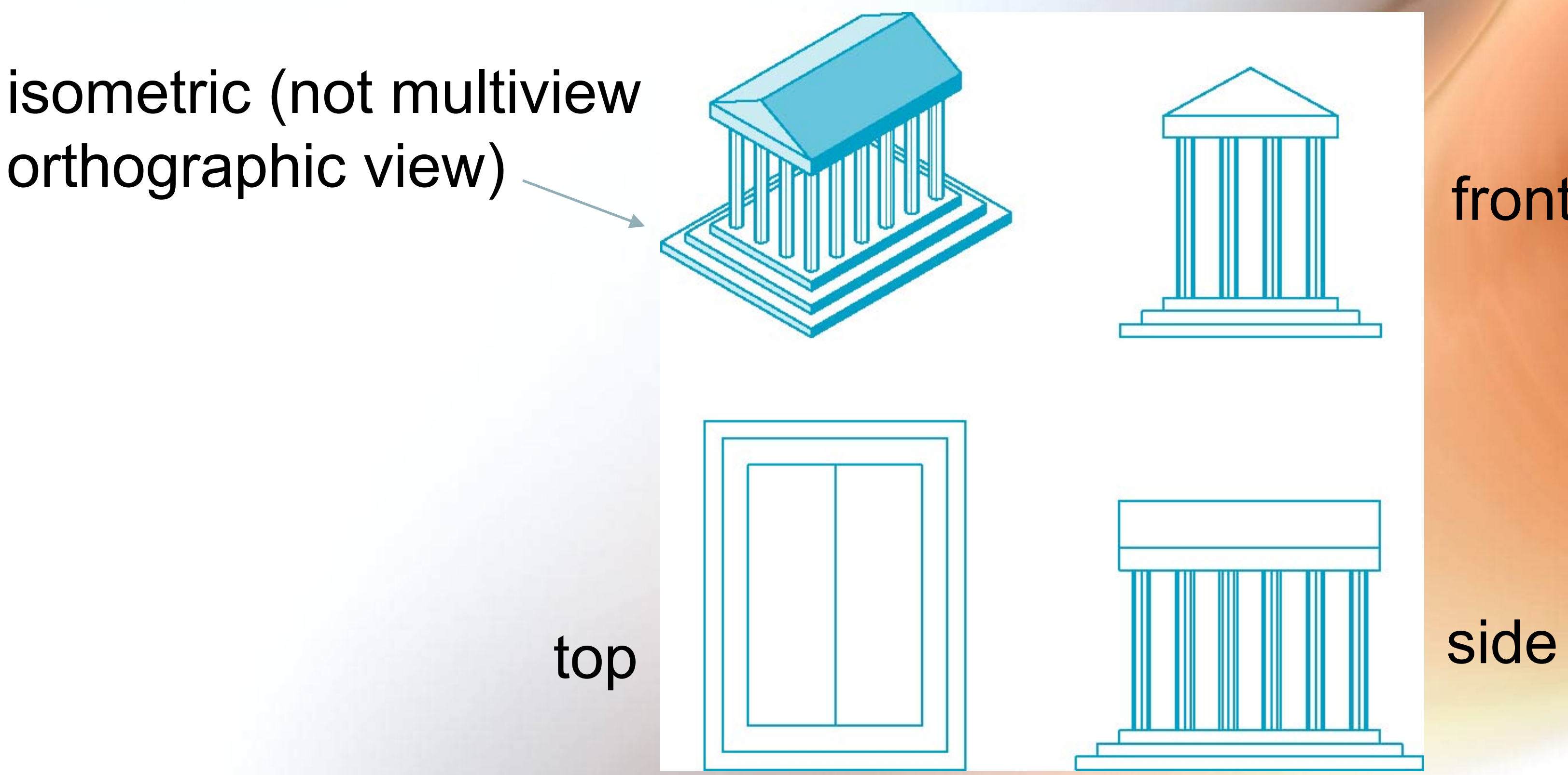
Orthographic Projection

Projectors are orthogonal to projection surface



Multiview Orthographic Projection

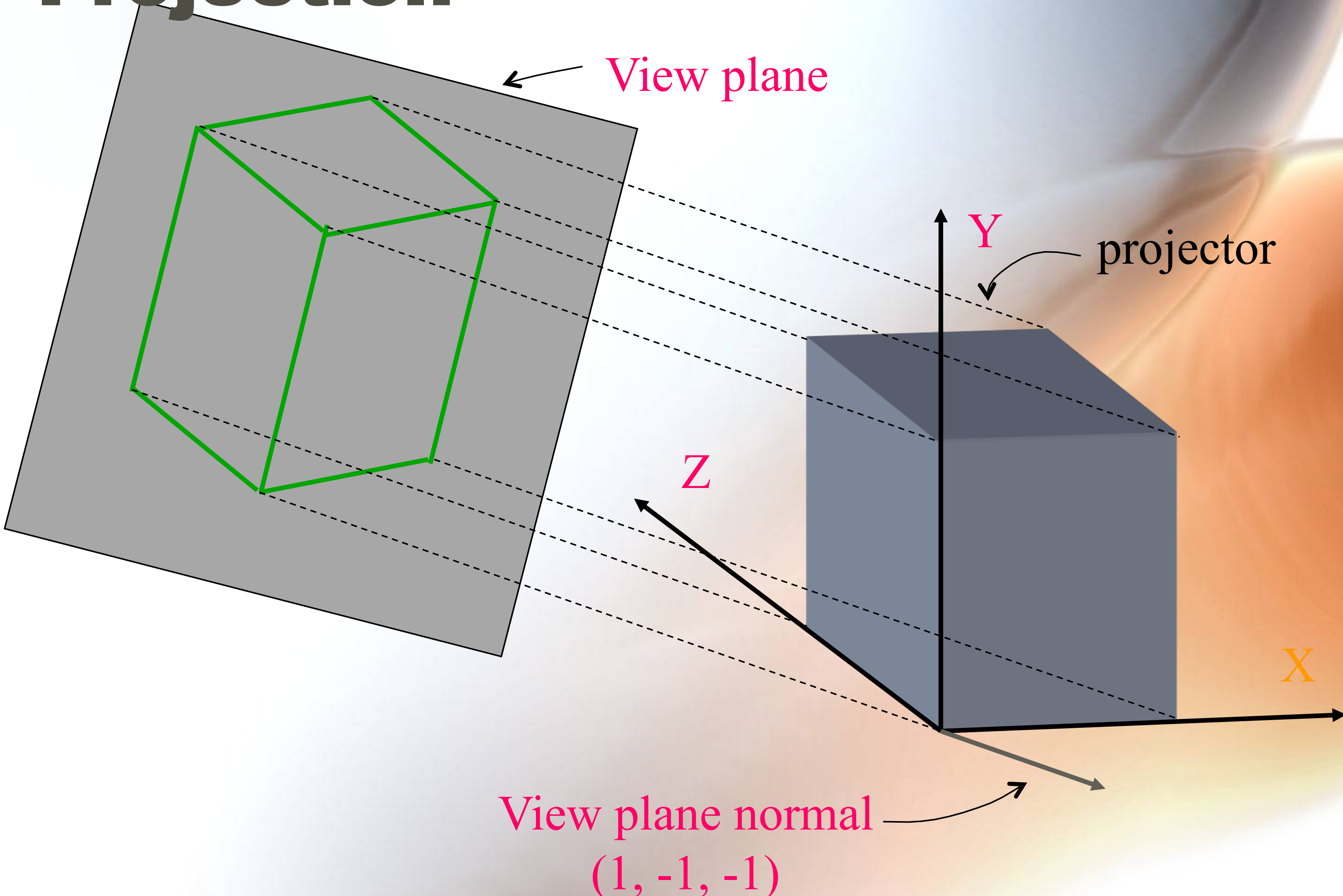
- Projection plane parallel to principal face
- Usually form front, top, side views
- We often display three multiviews plus isometric



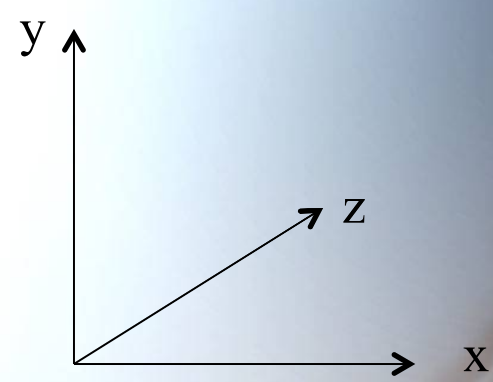
Advantages and Disadvantages

- Preserves both distances and angles
  - Shapes preserved
  - Can be used for measurements
    - Building plans
    - Manuals
- Cannot see what object really looks like because many surfaces hidden from view
  - Often we add the isometric

Construction of Parallel Projection



Axonometry

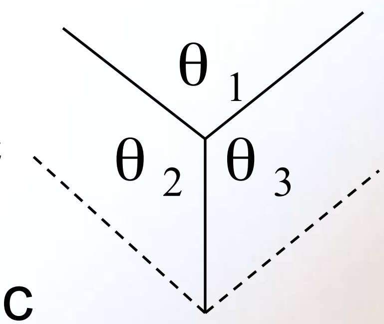


- A drawing technique where the three axes are projected to non-orthographic axes in 2D
- Y usually remains vertical, z is skewed, x is either horizontal or also skewed
- No vanishing point
- Objects remain the same size regardless of distance

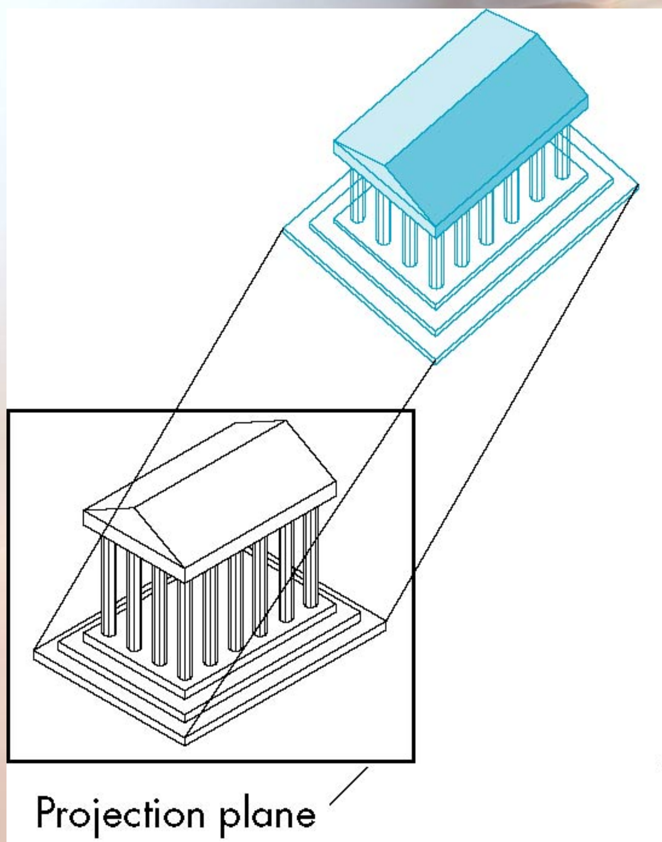
Axonometric Projections

- If we allow the projection plane to be at any angle (not just parallel to a face of the object)

classify by how many angles of a corner of a projected cube are the same

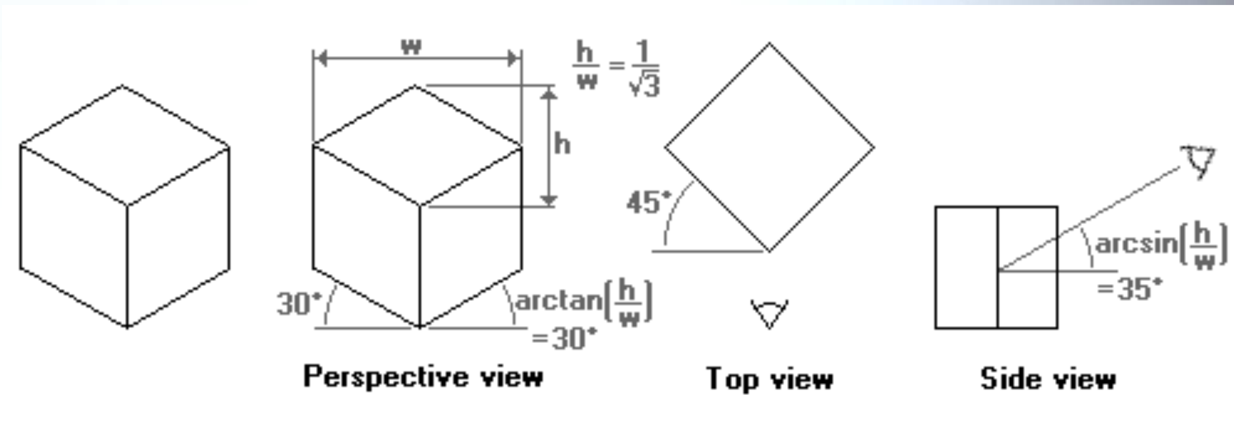


none: trimetric
two: dimetric
three: isometric

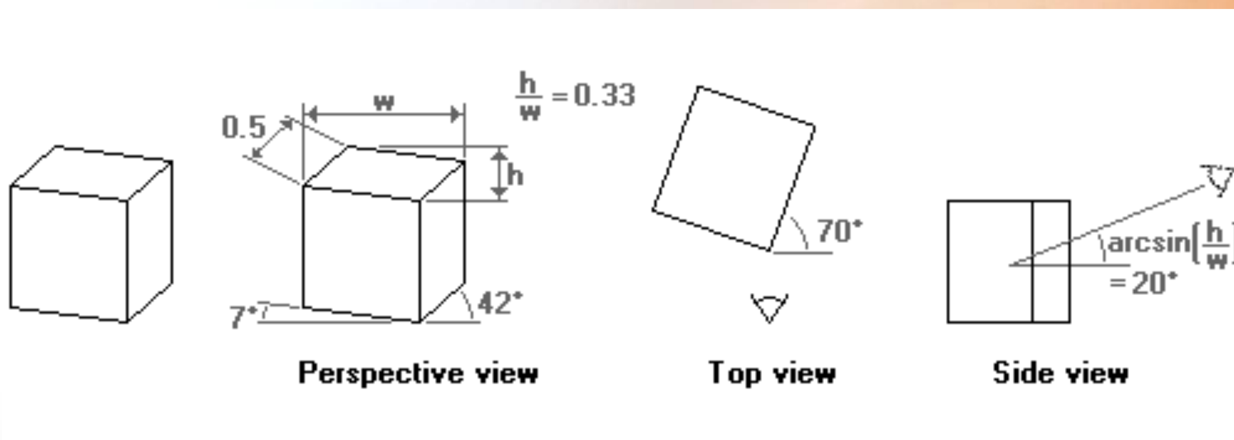


Isometric and Dimetric

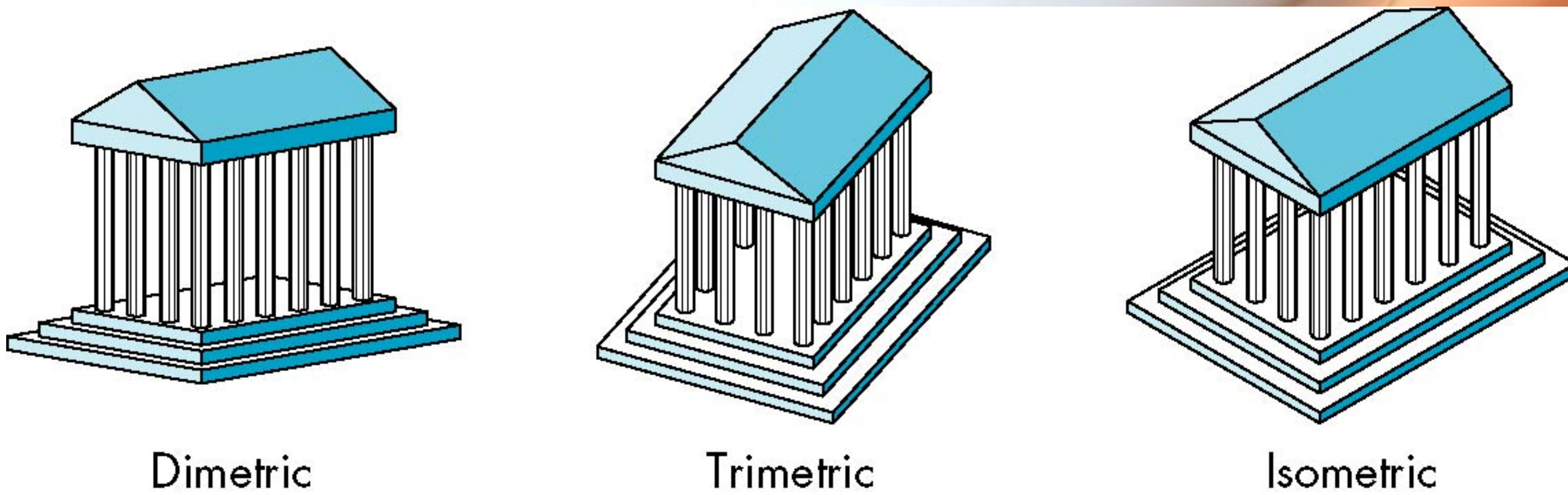
- Isometric: axes have the same metric



- Dimetric: one axis has a different metric



Types of Axonometric Projections

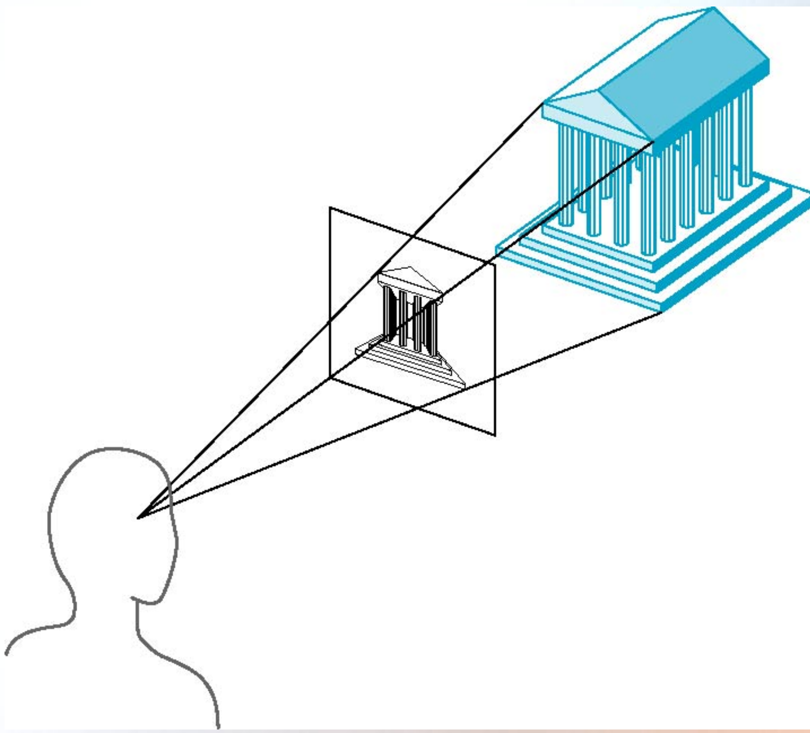


Advantages and Disadvantages

- Lines are scaled (*foreshortened*) but can find scaling factors
- Lines preserved but angles are not
  - Projection of a circle in a plane not parallel to the projection plane is an ellipse
- Can see three principal faces of a box-like object
- Some optical illusions possible
  - Parallel lines appear to diverge
- Does not look real because far objects are scaled the same as near objects
- Used in CAD applications

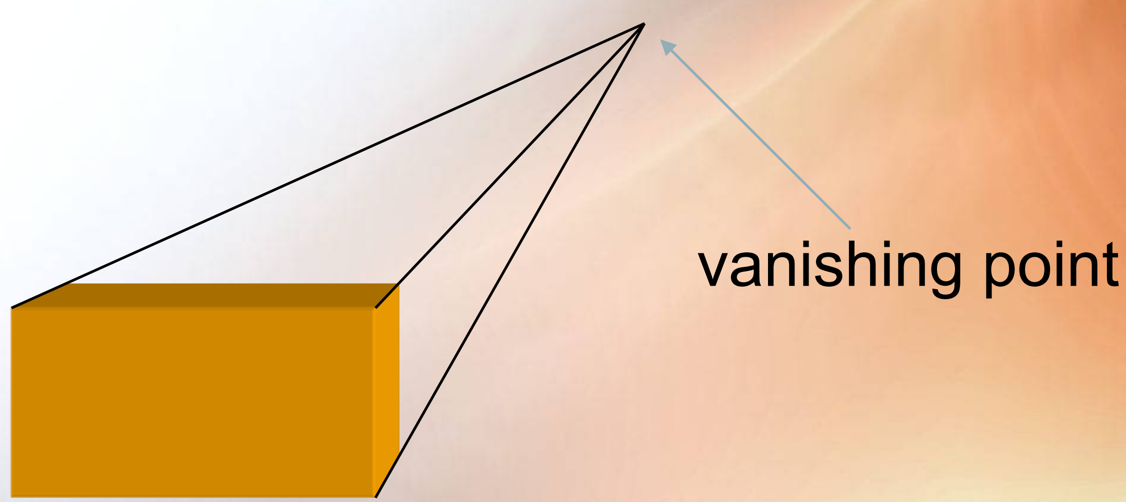
Perspective Projection

Projectors converge at center of projection



Vanishing Points

- **Parallel lines (not parallel to the projection plan) on the object converge at a single point in the projection (the *vanishing point*)**
- **Drawing simple perspectives by hand uses these vanishing point(s)**



One-Point Perspective

- One principal face parallel to projection plane
- One vanishing point for cube

Two-Point Perspective

- On principal direction parallel to projection plane
- Two vanishing points for cube

Three-Point Perspective

- No principal face parallel to projection plane
- Three vanishing points for cube

One, Two and Three Points

Fig 1

Fig 2

Fig 3

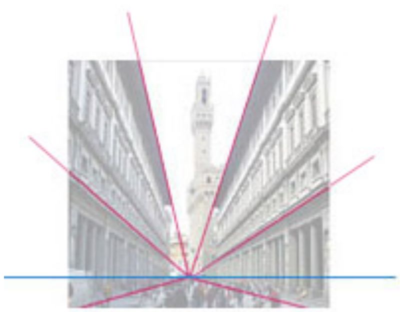


Fig 4

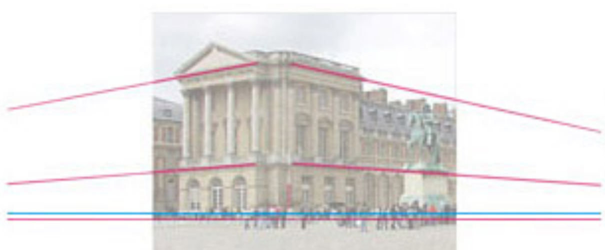


Fig 5

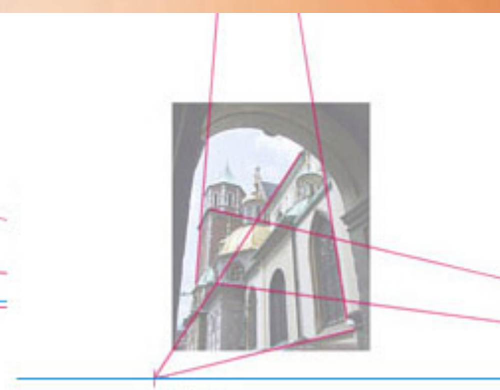
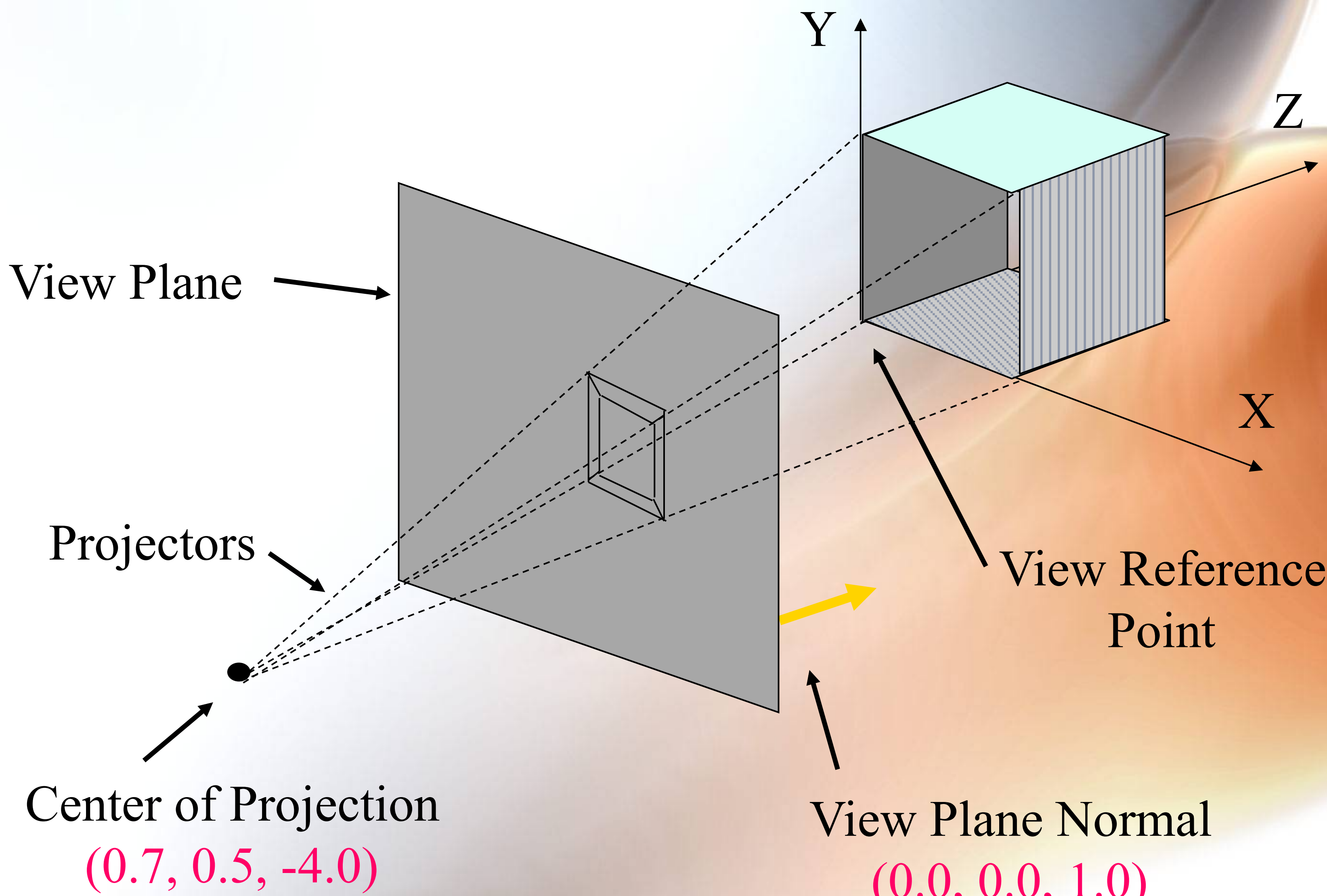
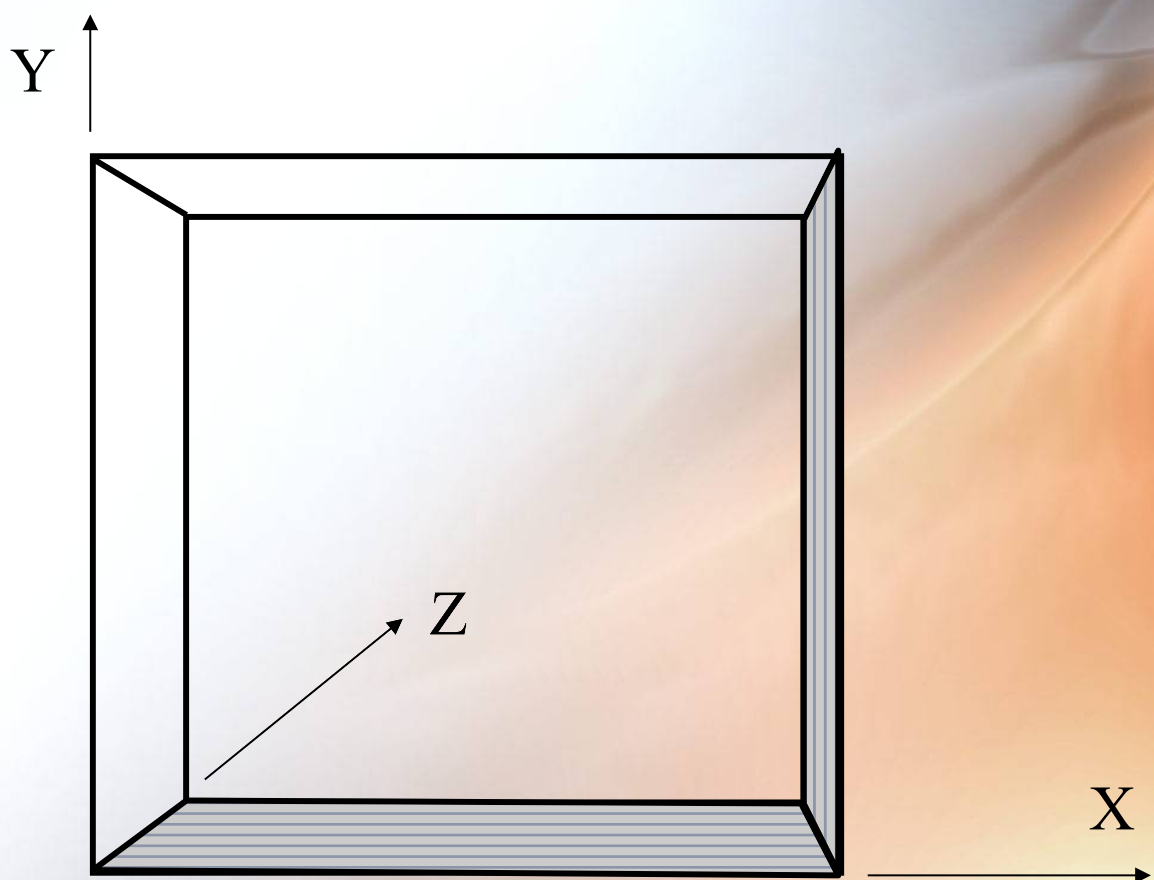


Fig 6

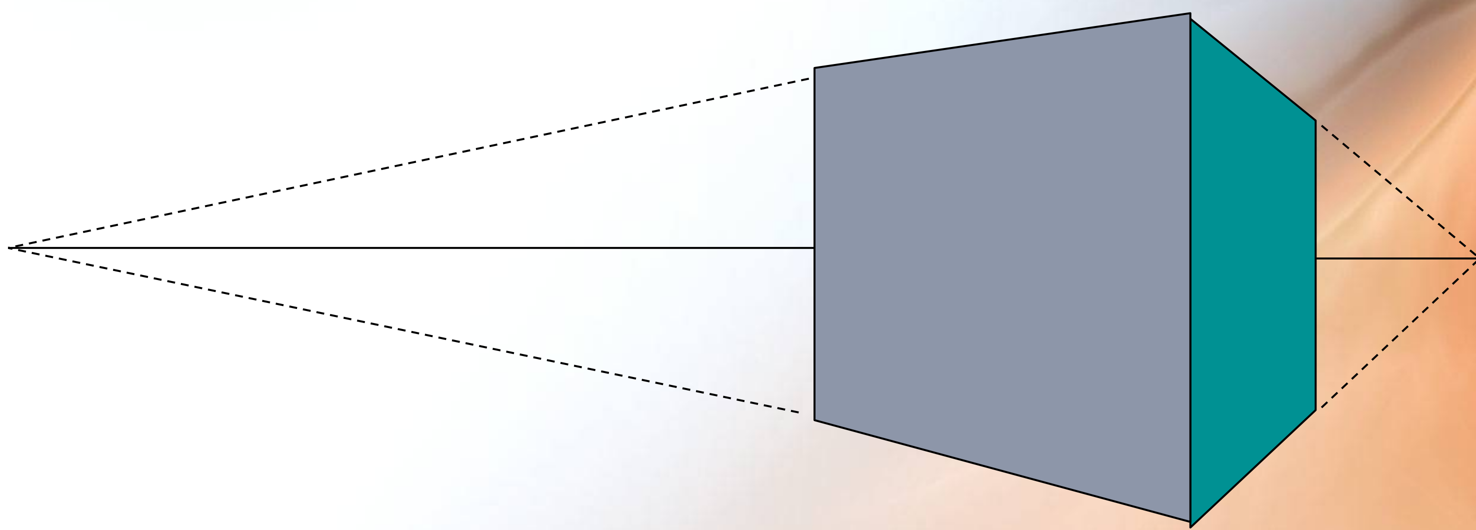
One-Point Perspective Construction



1-Point Perspective Projection of Cube

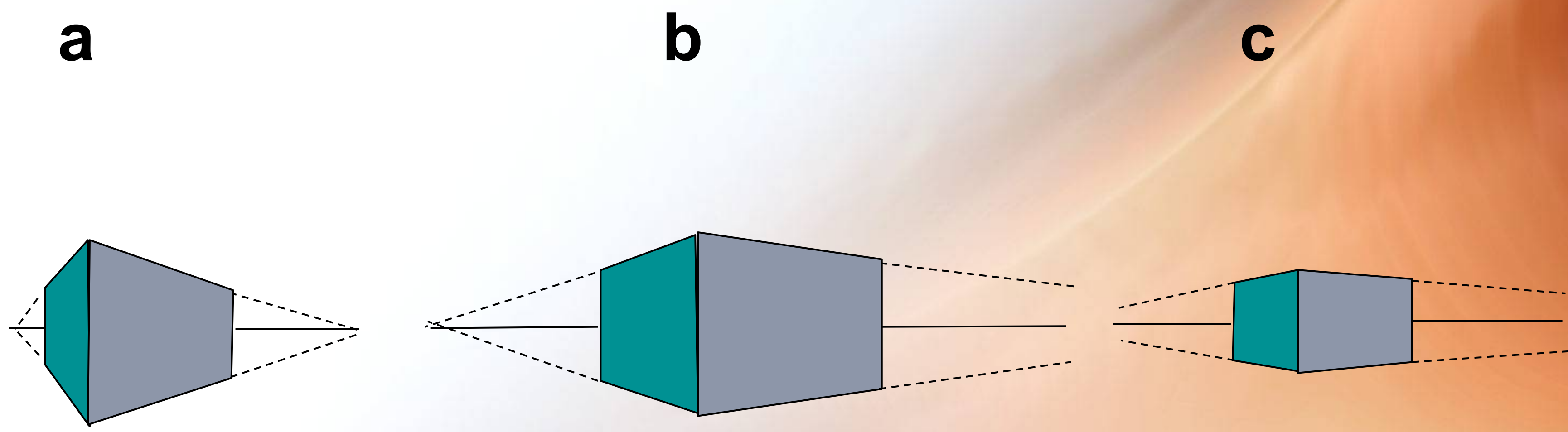


2-point Perspective of a Cube



Varying the 2-Point Perspective Center of Projection

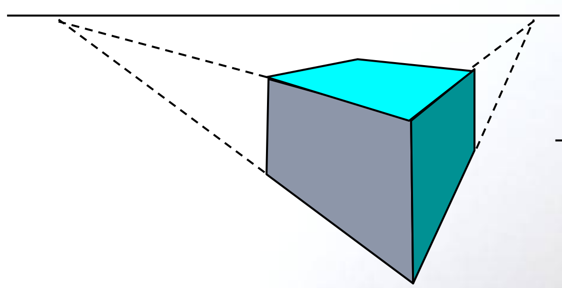
- **Variations achieved by moving the center of projections closer (a) or farther (c) and from the view plane.**



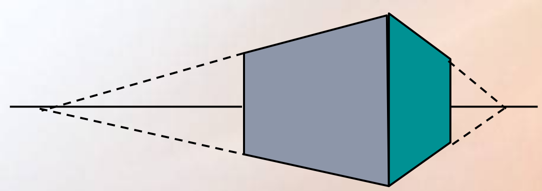
Varying the 2-Point Perspective Center of Projection

- Variations achieved by moving the center of projection so as to show the top of the object (a) or the bottom of the object (c) .

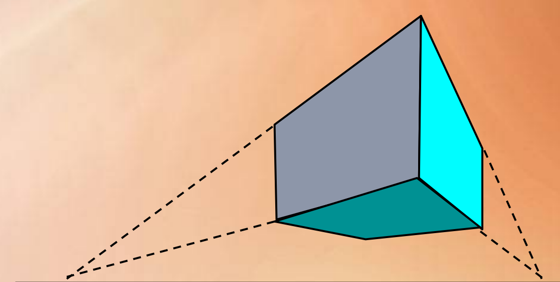
a



b



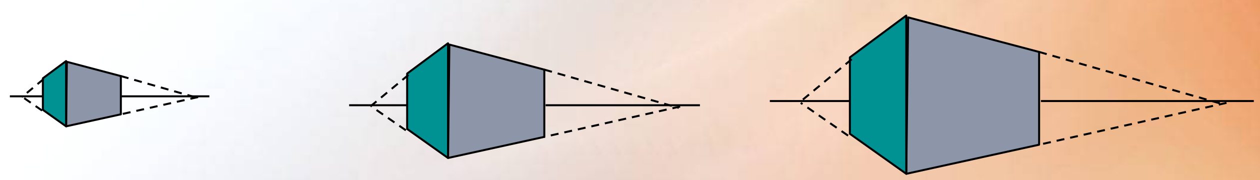
c



Varying the 2-Point Perspective View Plane Distance

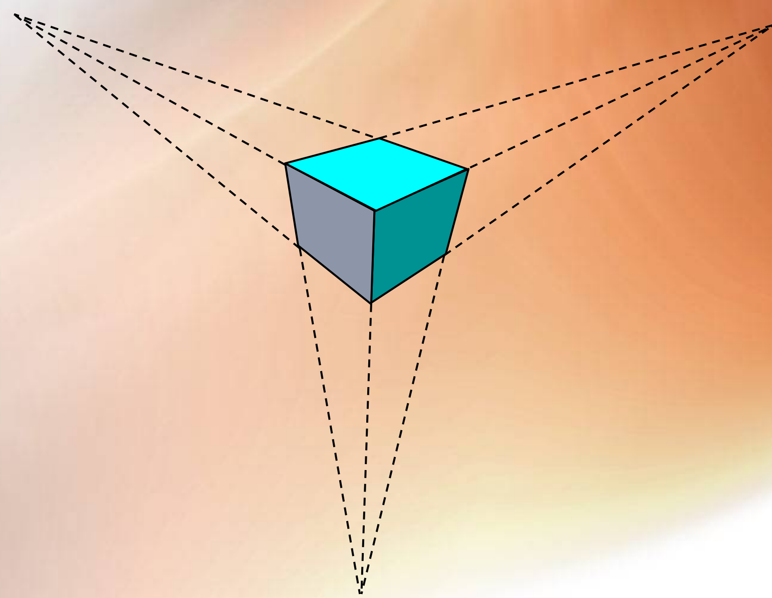
- Effects achieved by varying the view plane distance. (a) has a large view plane distance, (c) a small view plane distance. The effect is the same as a scale change.

a b c



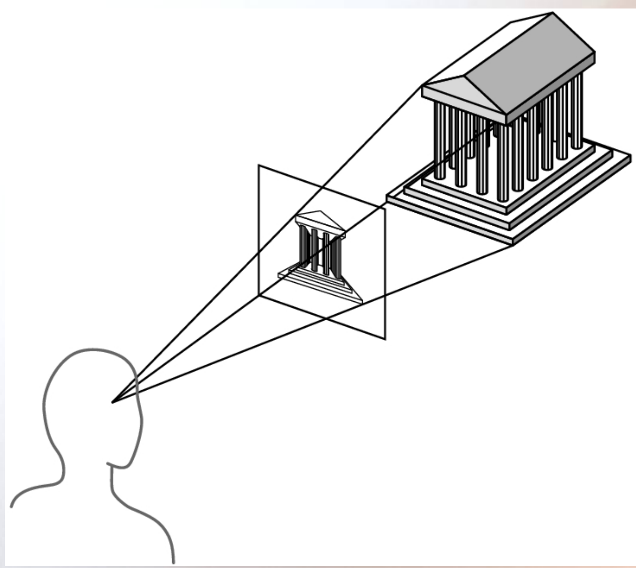
3-Point Perspective Projection

- Only difference from 2-point is that the projection plane intersects all three major axes.



Size

- **All perspective views are characterized by diminution of size (the farther away, the smaller they are)**



Advantages and Disadvantages

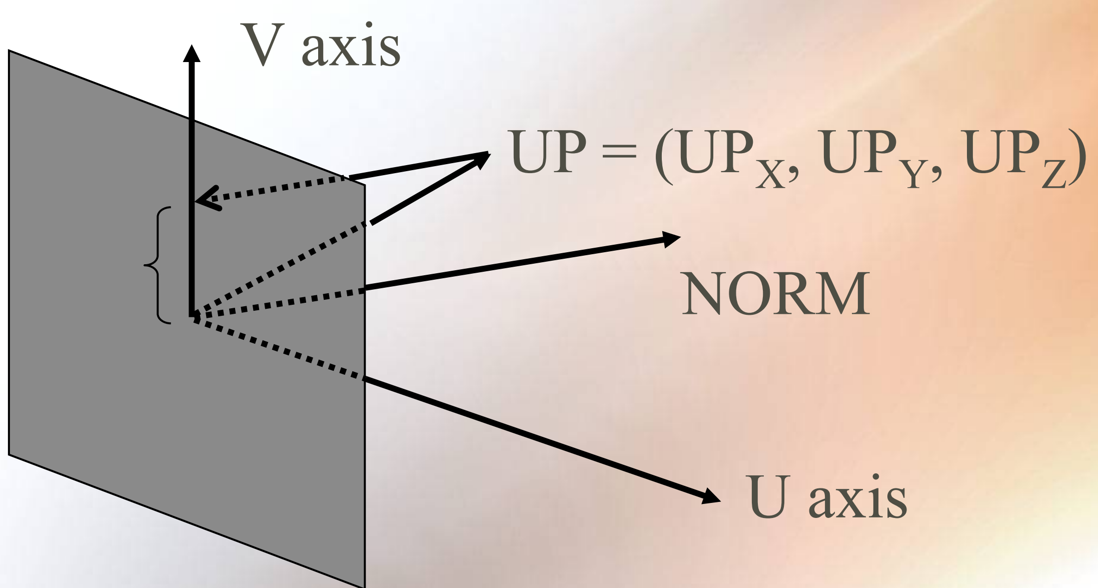
- Objects further from viewer are projected smaller than the same sized objects closer to the viewer (*diminuition*)
  - Looks realistic
- Equal distances along a line are not projected into equal distances (*nonuniform foreshortening*)
- Angles preserved only in planes parallel to the projection plane
- More difficult to construct by hand than parallel projections (but not more difficult by computer)

Defining a Perspective Projection

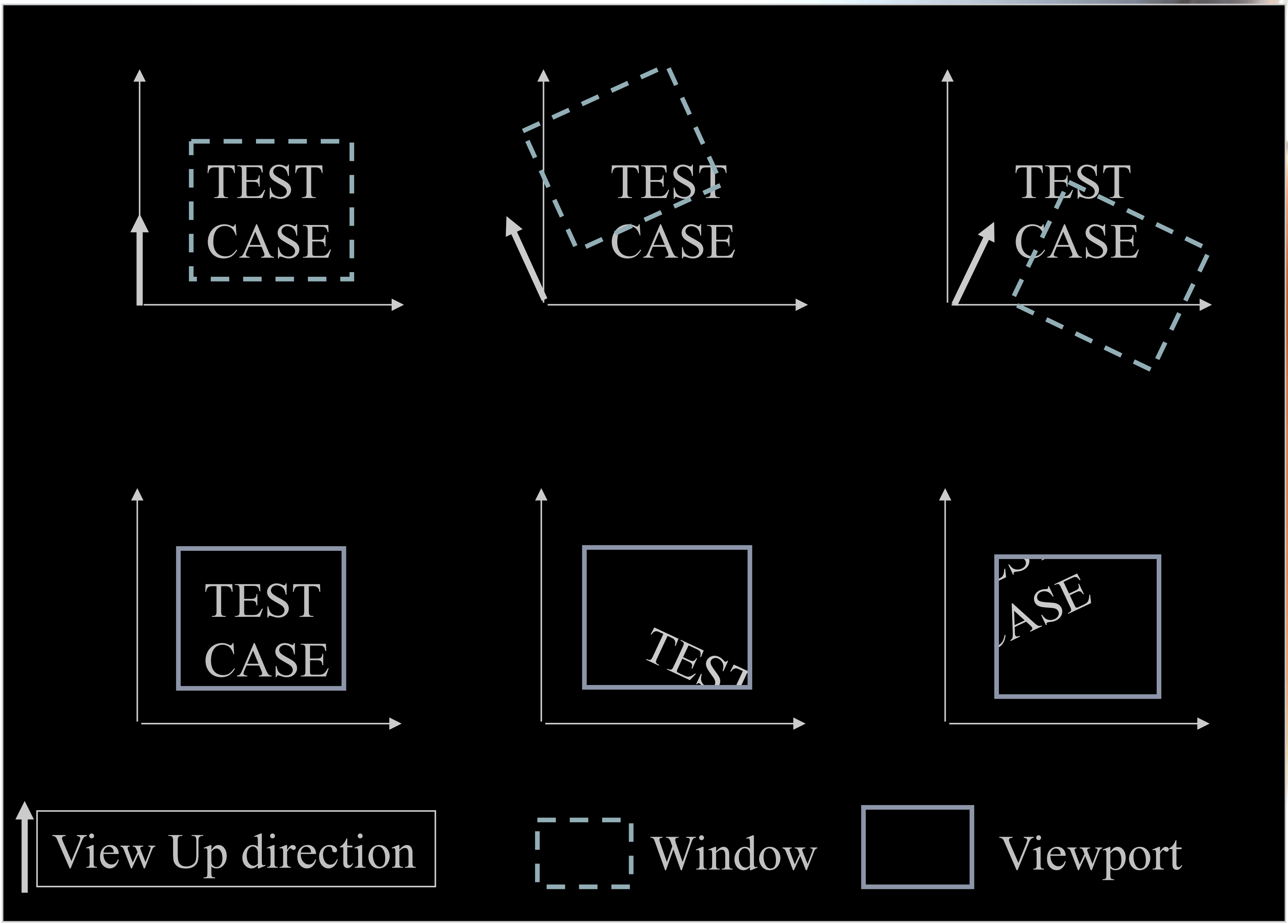
- **Define projection data and center of projection:**
  - View reference point: **a point of interest**
  - View plane normal: **a direction vector**
  - Center of projection: **a point defined relative to the view reference point, where the eye is**
  - View plane distance: **defines a distance along the view plane normal from the view reference point.**
  - View up vector: **The direction vector that will become "up" on the final image.**

The View Up Vector

- Specify direction which will become vertical in the final image: View Up.
- Project View Up vector onto view plane (given by view plane normal)

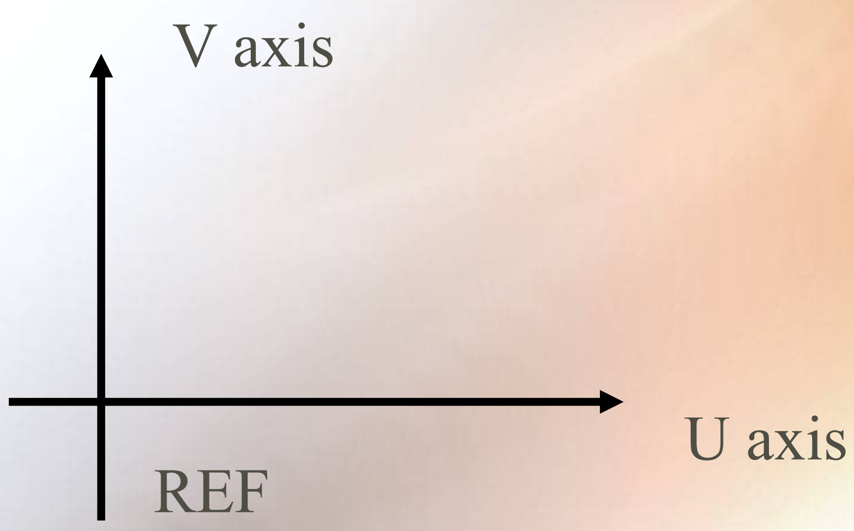


Effect of View Up Vector



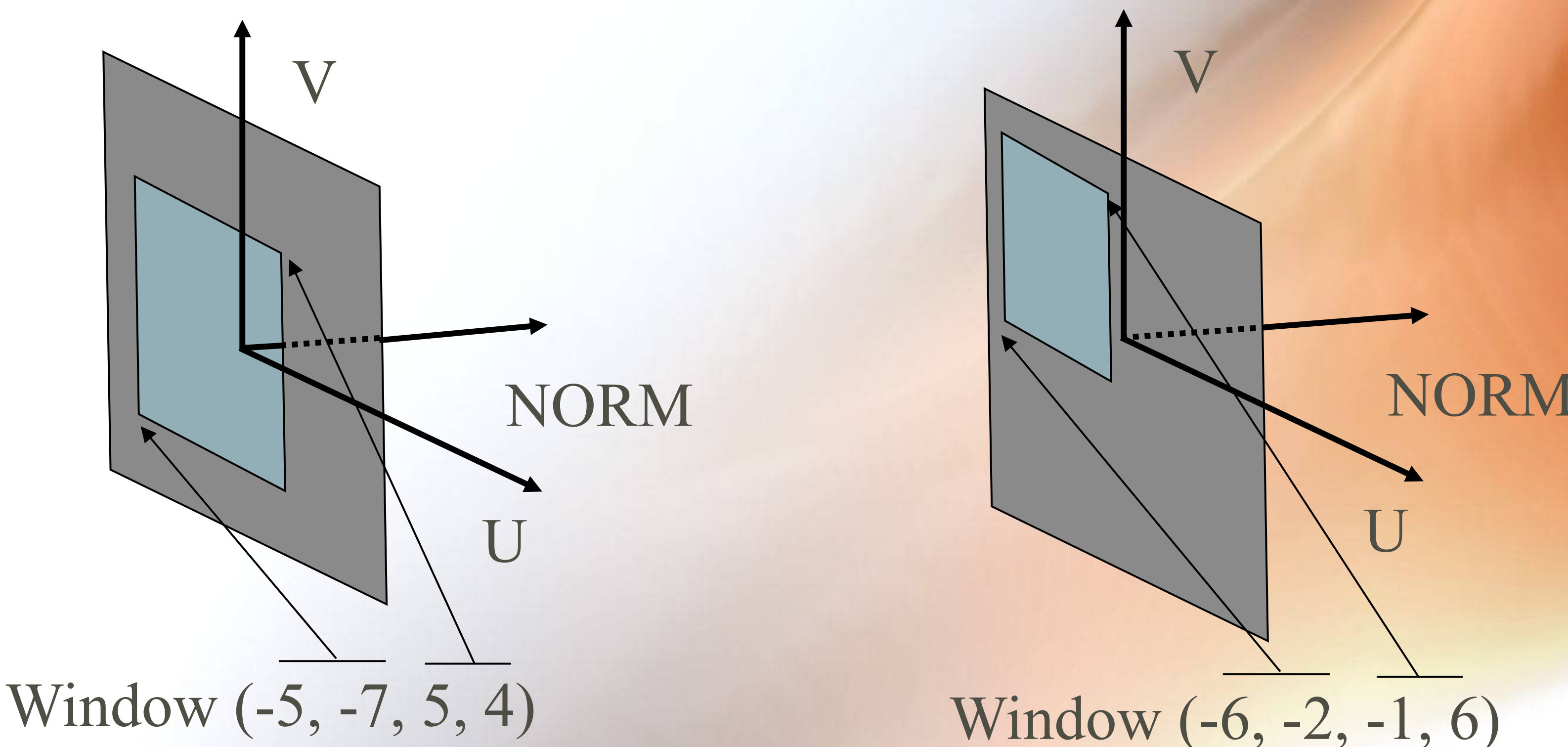
View Plane U - V Coordinate System

- **Origin is the view reference point REF.**
- **U and V axes computed from view up vector and view plane normal.**
- **Used to specify the 2D coordinates of the window.**



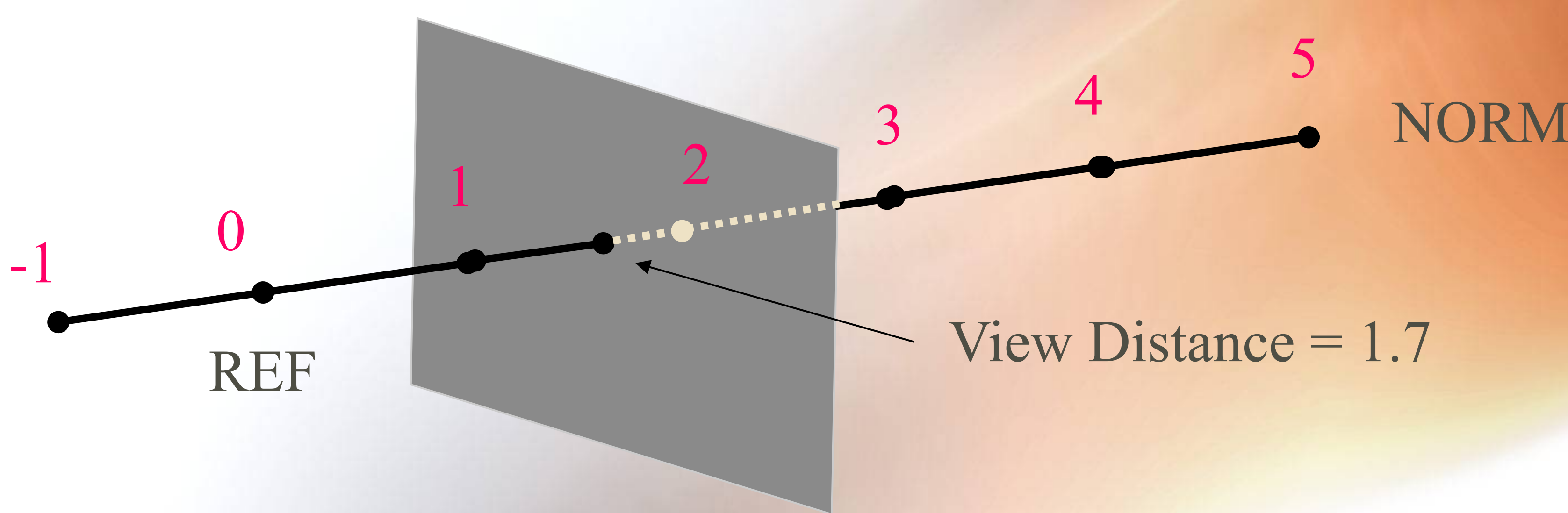
Specifying the Window

- Needed for window to viewport mapping
- Creates top, bottom, left, and right clipping limits.



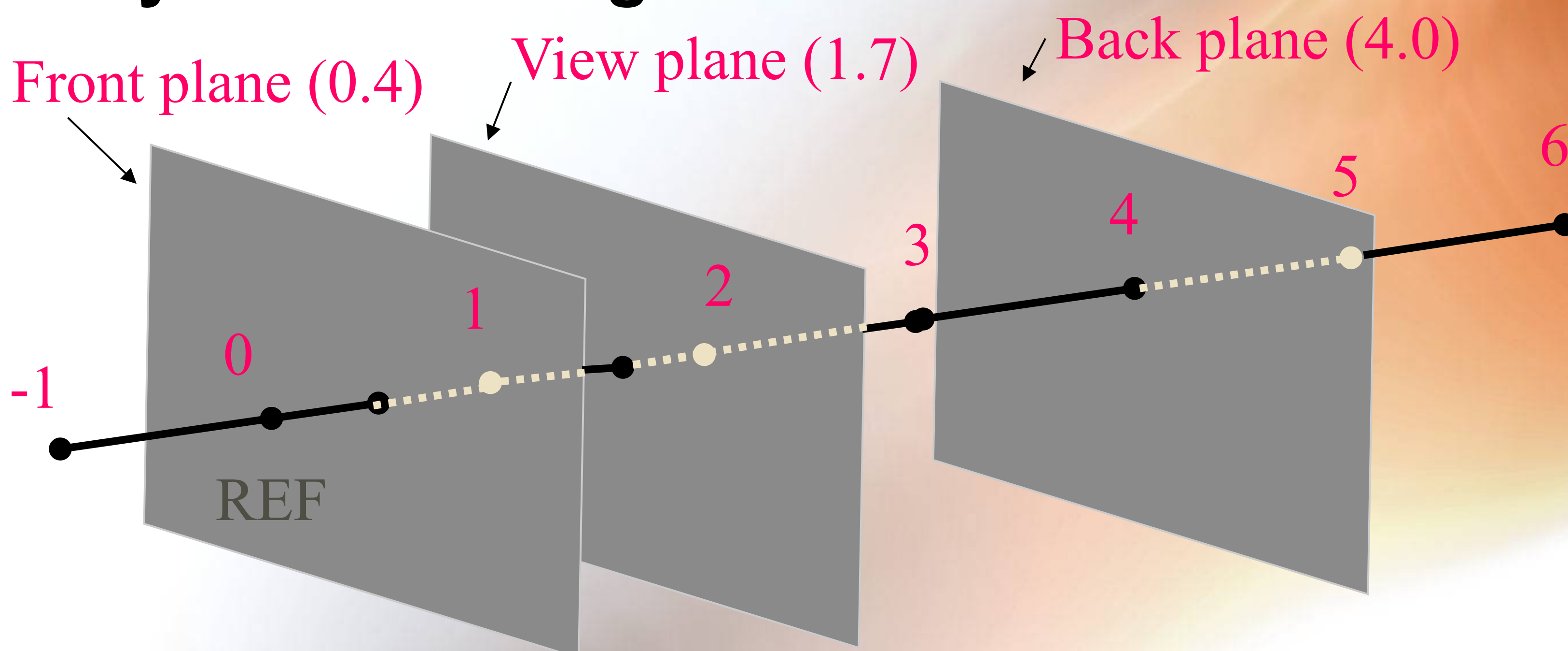
Setting the View Plane Distance

- Distance from view reference point to view plane along view plane normal (NORM).
- Used to zoom in and out.



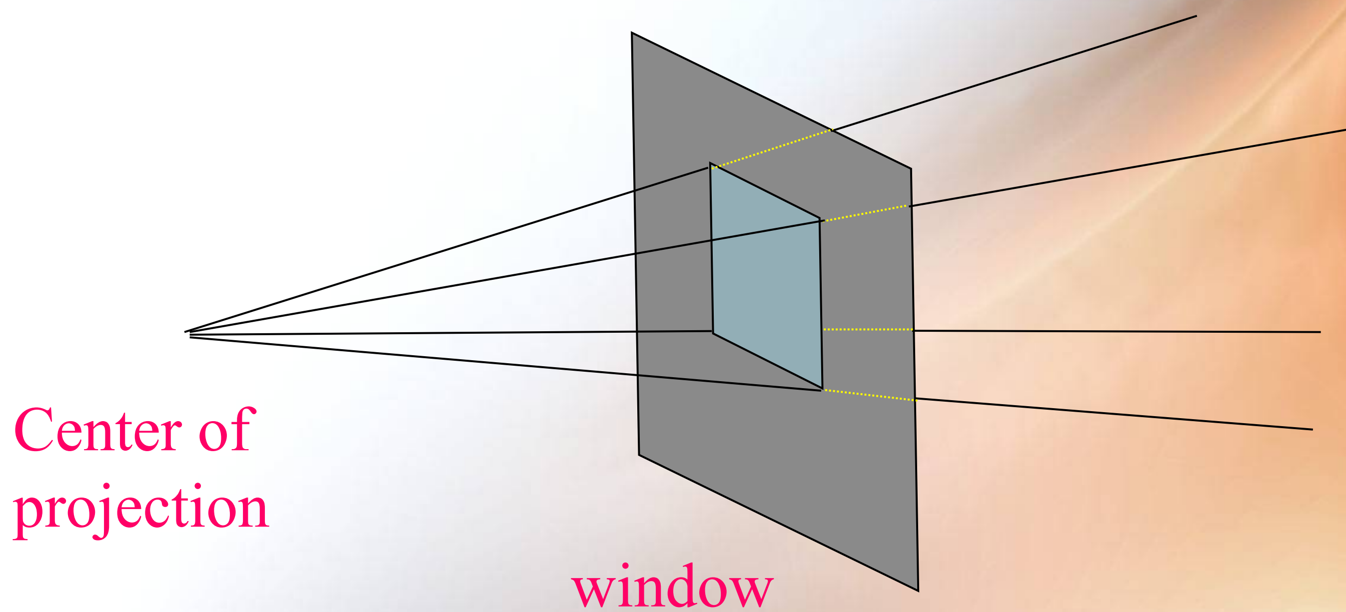
Setting the View Depth -- Front and Back Planes

- Forms front and back of view volume.
- Used for both perspective and parallel projections.
- Any order as long as front < back.



3D Clipping and the View Volume

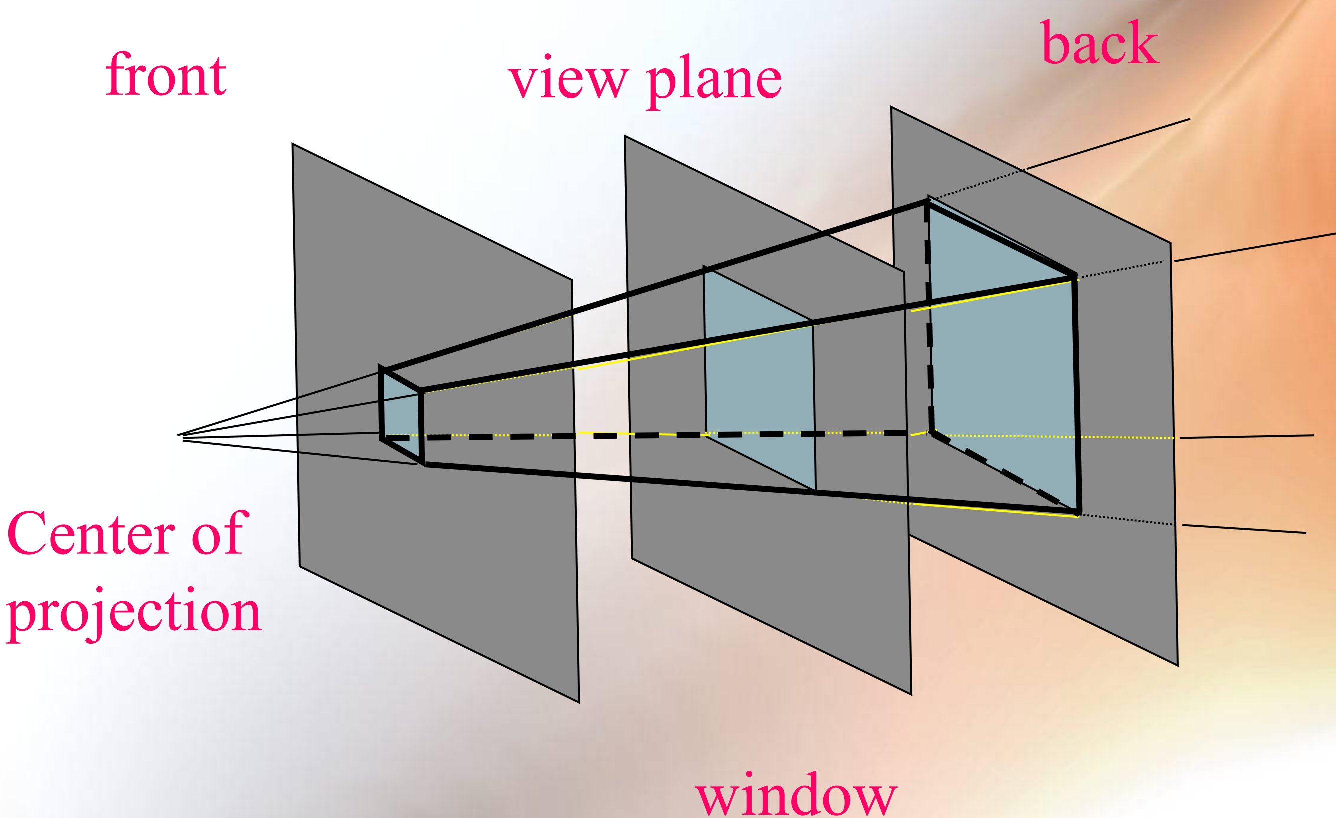
Graphic primitives are clipped to the view volume



The contents of the view volume are projected onto the window.

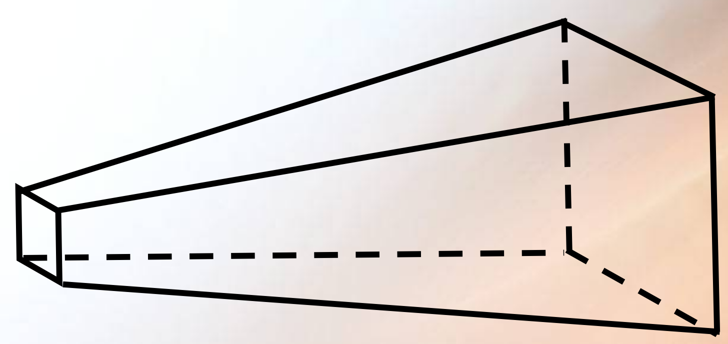
3D Clipping uses the Front and Back Planes, too

- Front and back planes truncate the view volume pyramid.



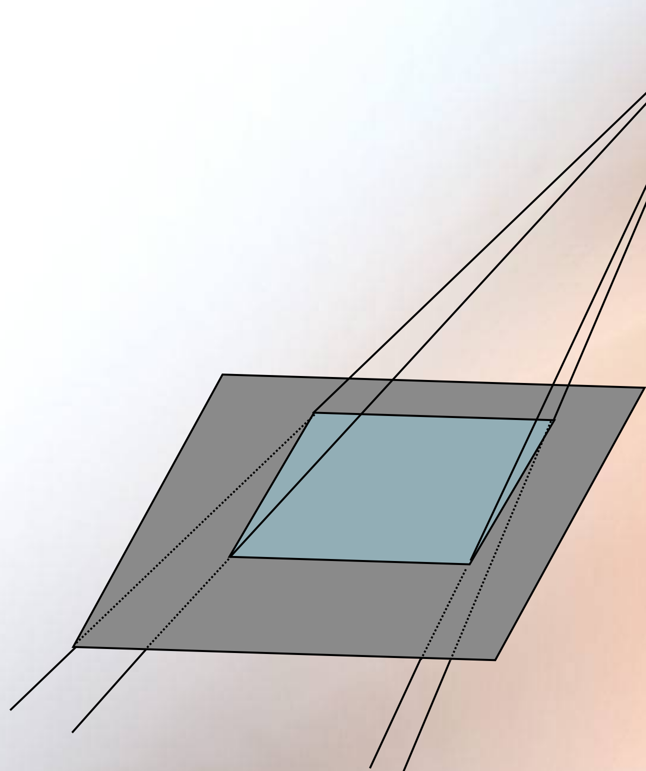
The Perspective Projection View Volume

Voilà! The truncated pyramid **view volume.**

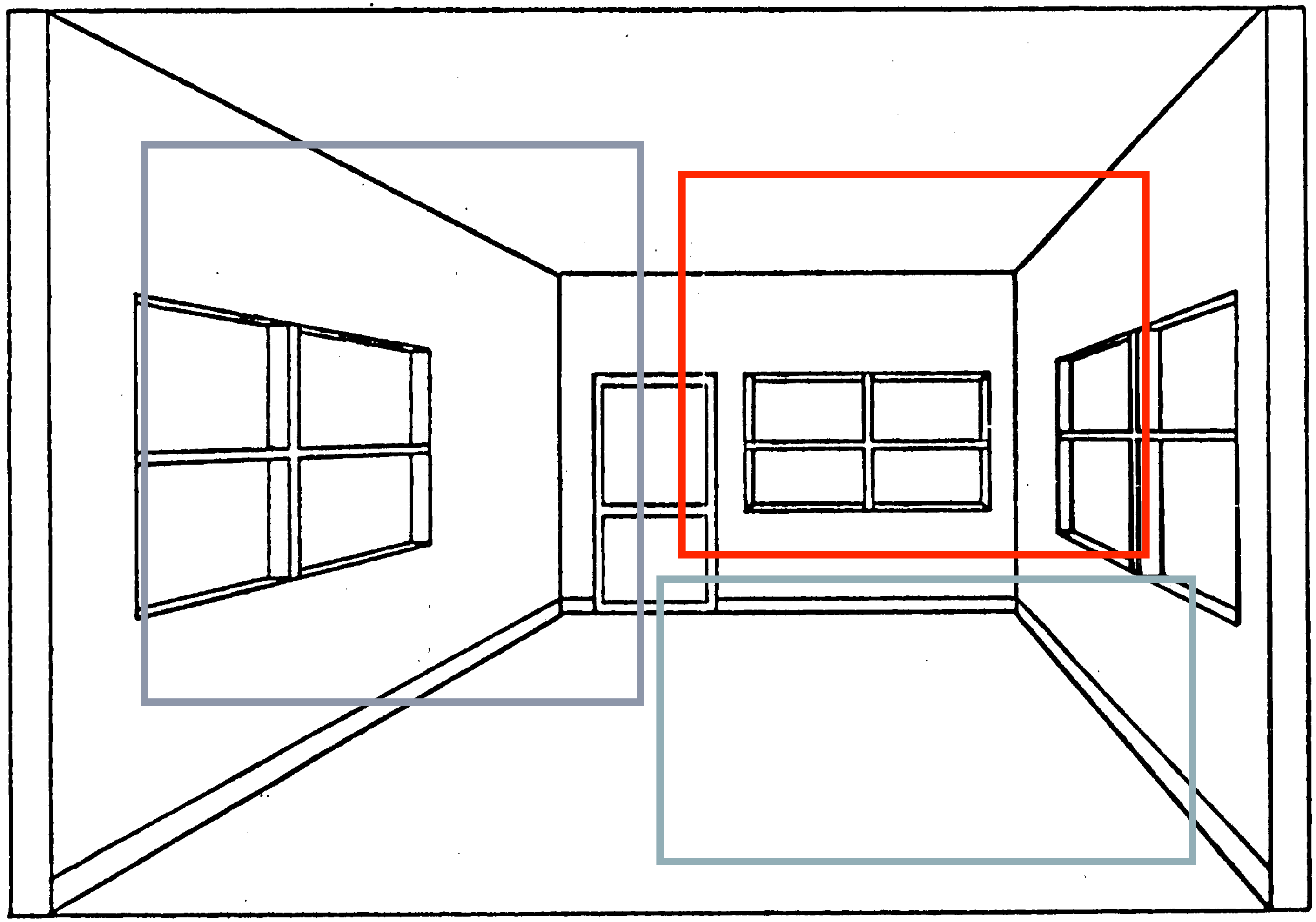


Window Changes Create Cropping Effects

Window need not be centered in UV coordinate system.
Like cropping a photograph.

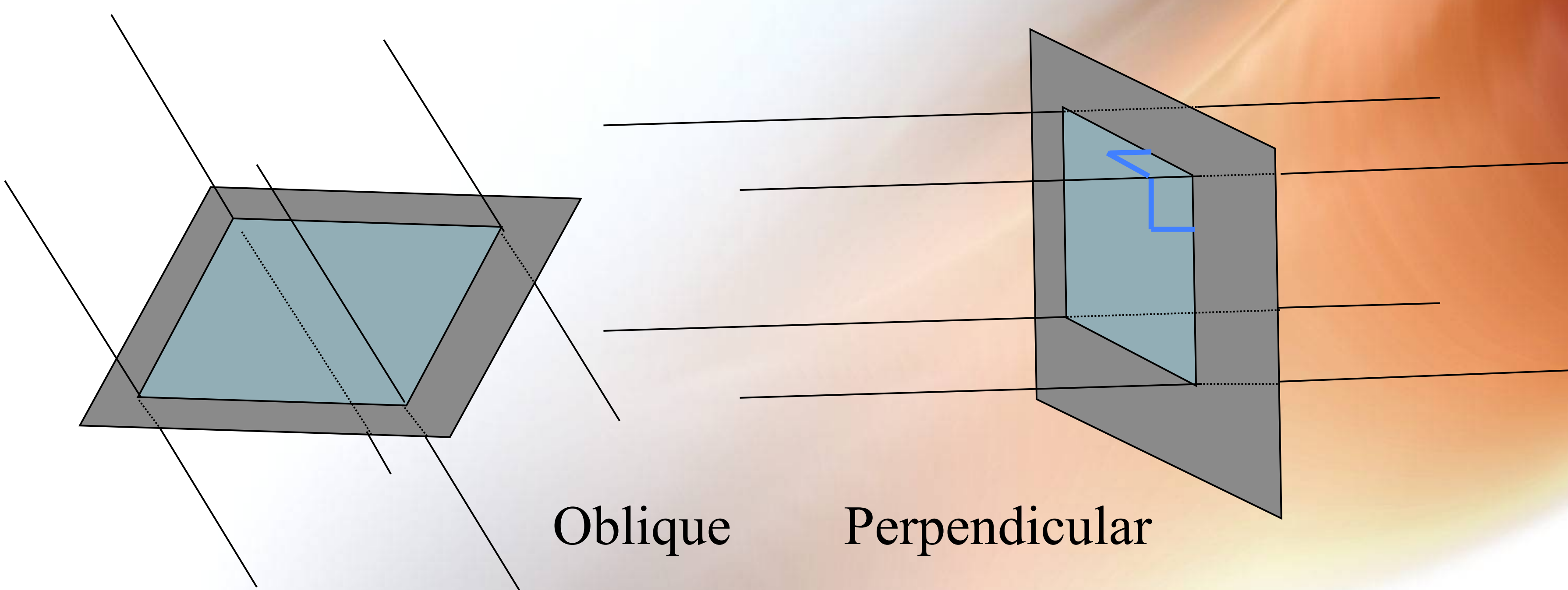


Cropping Example



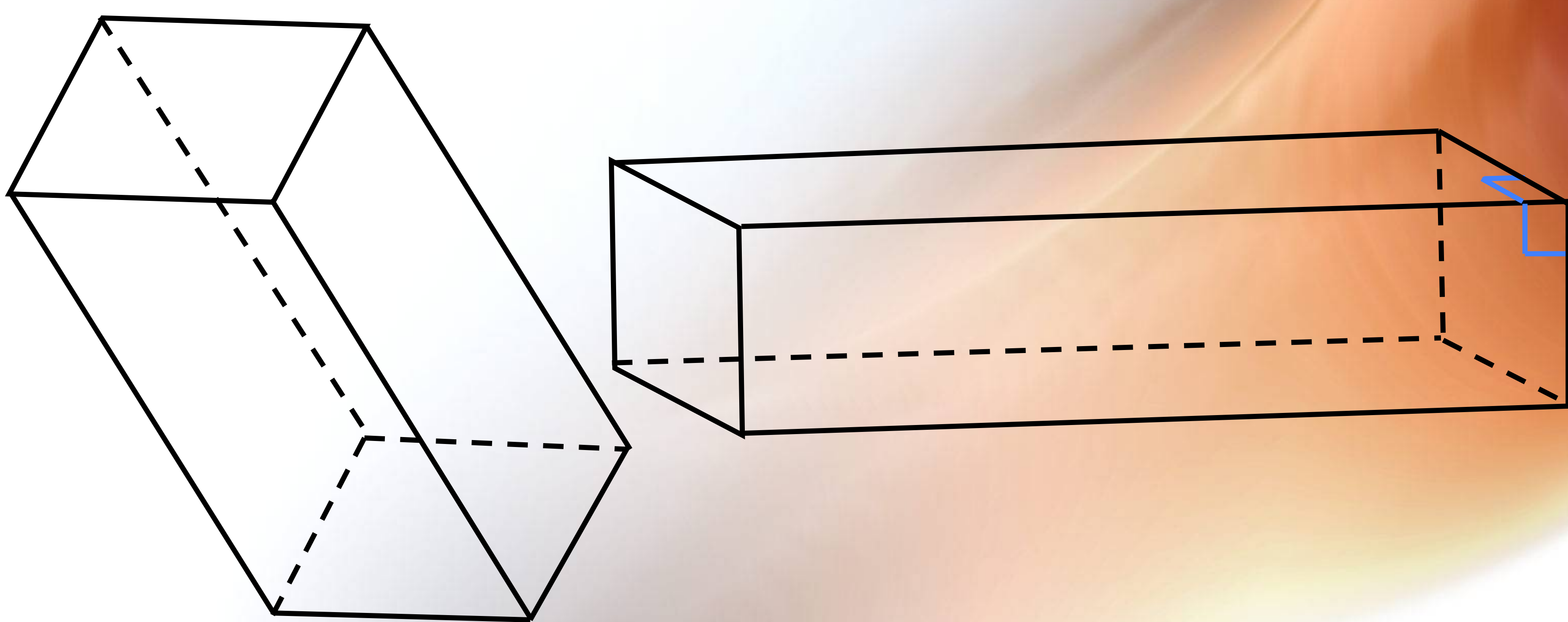
Parallel Projection Clipping View Volume

- View Volume determined by the direction of projection and the window



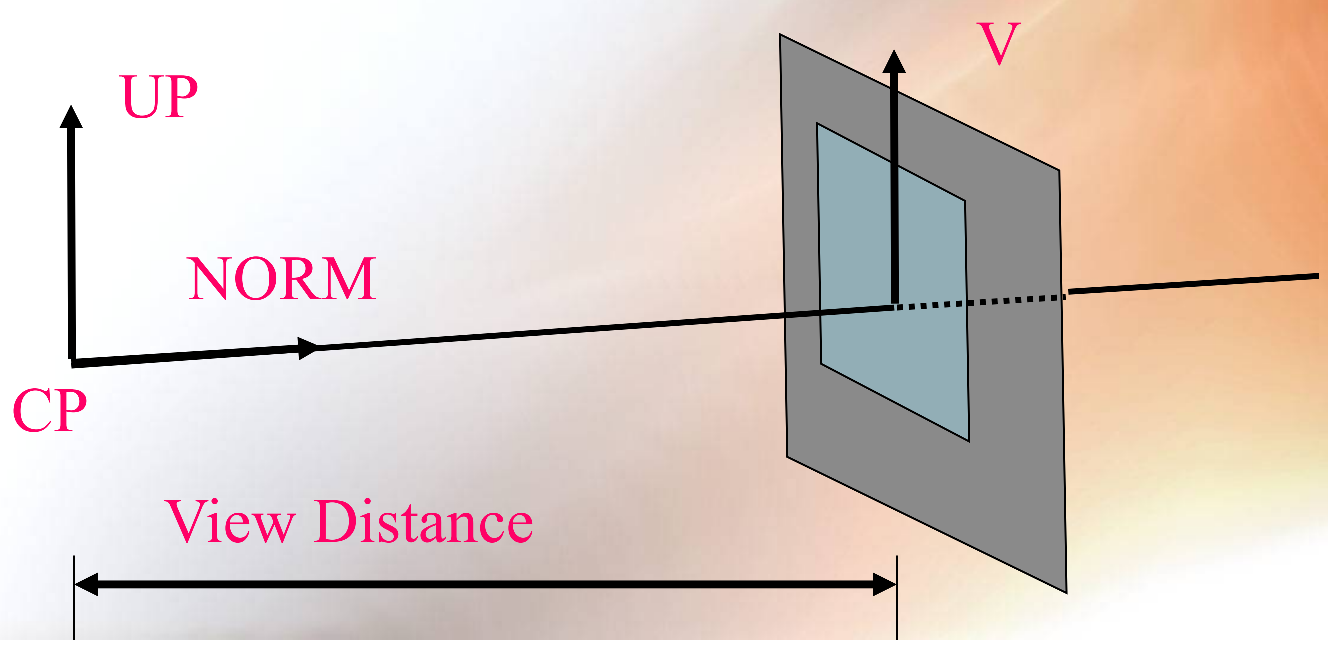
Parallel Projection View Volume

- **View Volume is now a** parallelopiped**.**



The Synthetic Camera

- Translated via CP changes.
- Rotated via UP changes.
- Redirected via View Plane Normal changes (e.g. panning).
- Zoom via changes in View Distance



3D Viewing Pipeline

3D WORLD

Modeling transformation

3D World

Viewing transformation

3D eye

Clipping transformation

3D clip

Clip

3D clip

Projection (homogeneous division)

3D NDC

Image transformation

3D NDC

NDC to physical device coordinates

2D SCREEN

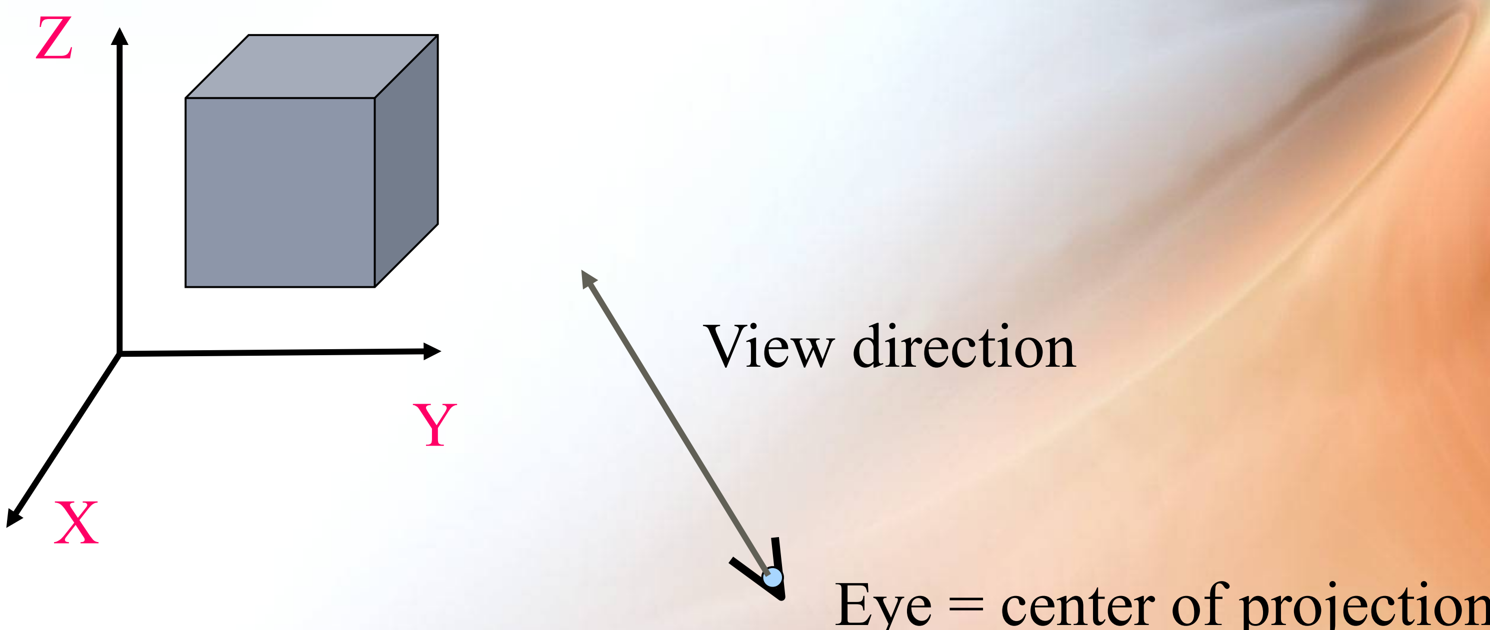
“Standard View Volume”

Transform World Coordinates to Eye Coordinates

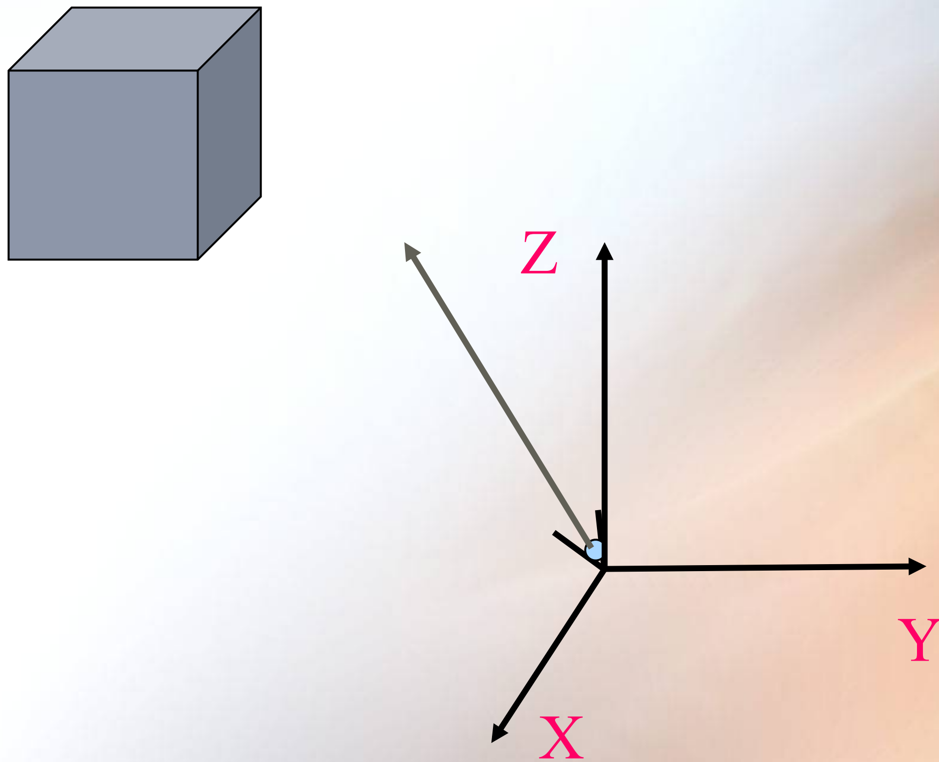
Approximate steps:

- Put eye (center of projection) at (0, 0, 0).
- Make X point to right.
- Make Y point up.
- Make Z point forward (away from eye in depth).
- (This is now a *left-handed* coordinate system!)

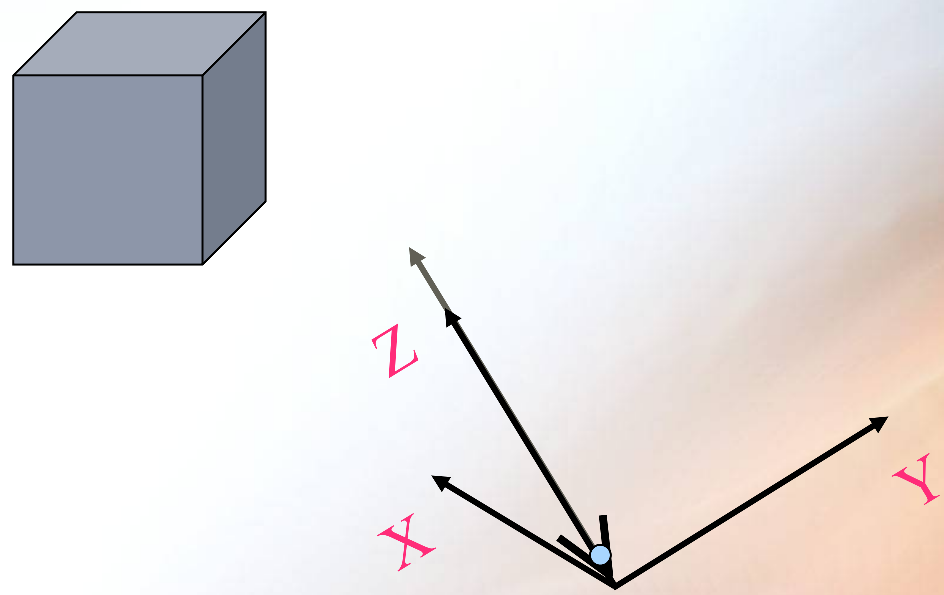
World to Eye Transformation START



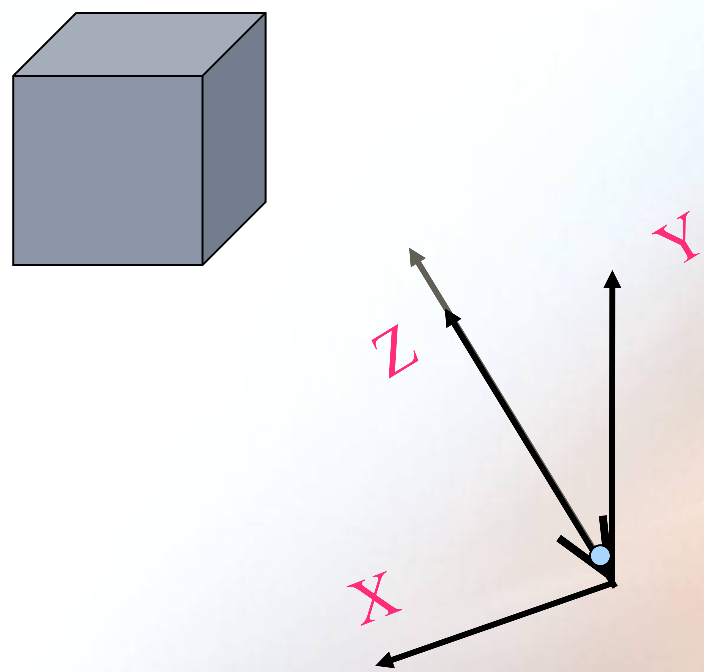
World to Eye Transformation
Translate eye to (0, 0, 0)



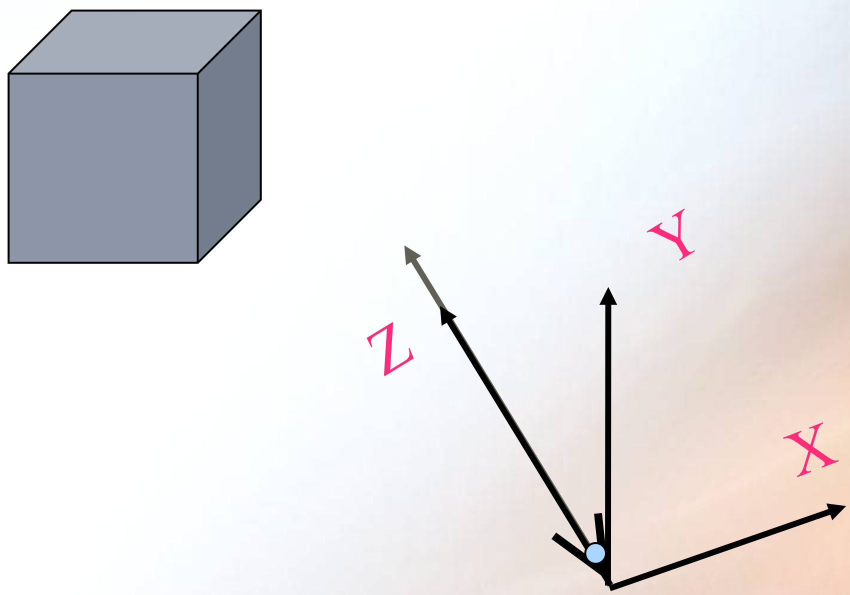
World to Eye Transformation
Align view direction with +Z



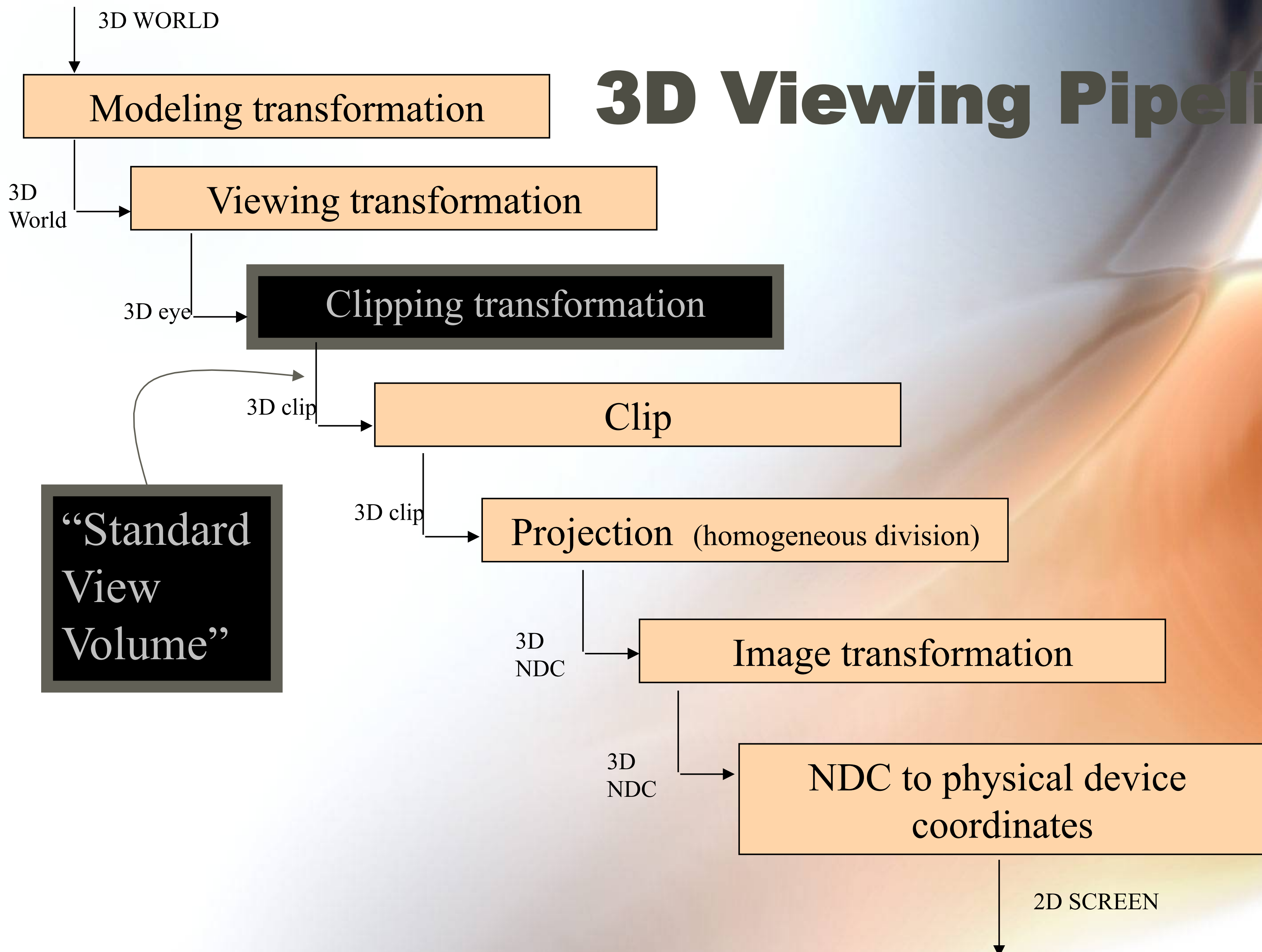
World to Eye Transformation
Align VUP direction with +Y



World to Eye Transformation
Scale to LH coordinate system

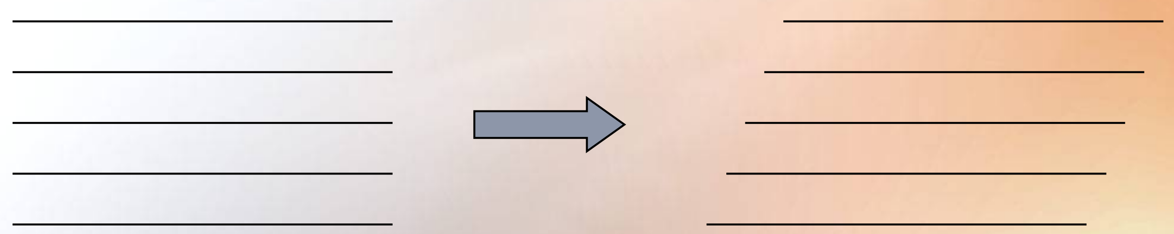


3D Viewing Pipeline

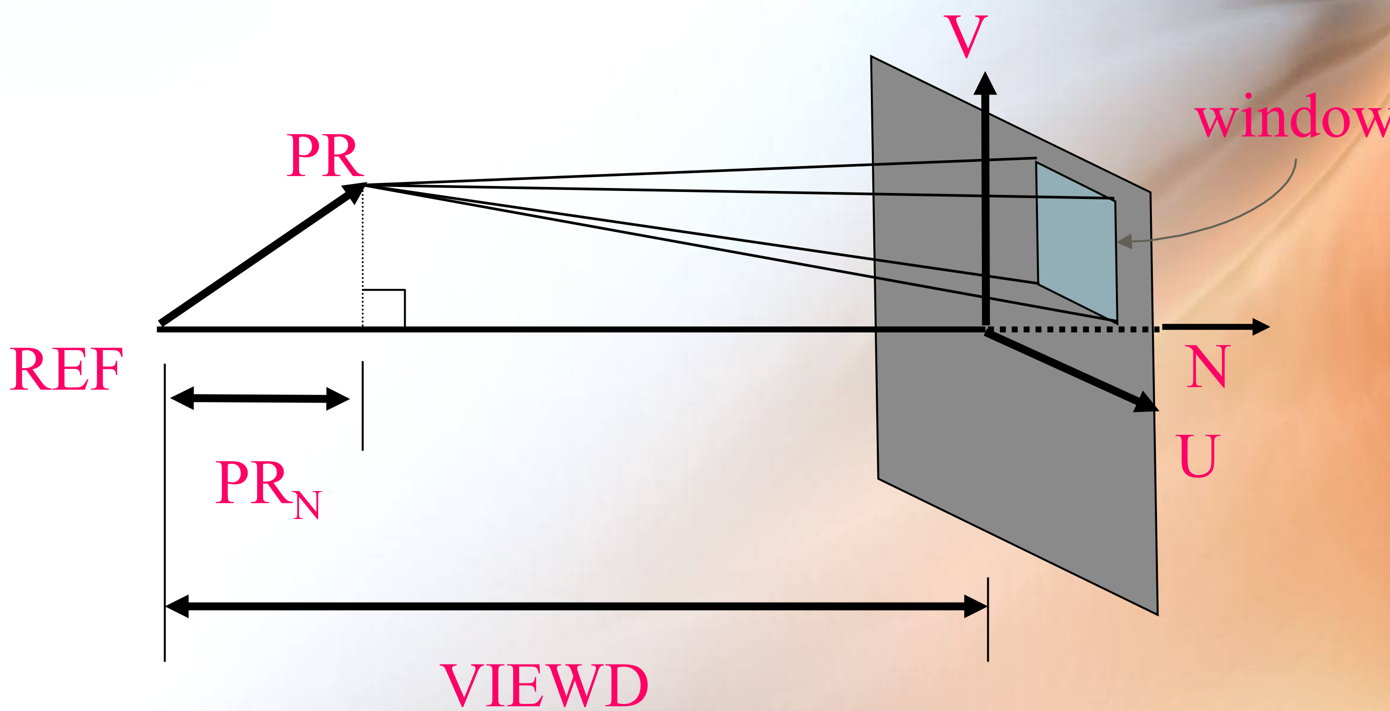


On to the Clipping Transformation

- It remains to do the transformations that put these coordinates into the clipping coordinate system
- We have to shear it to get it upright

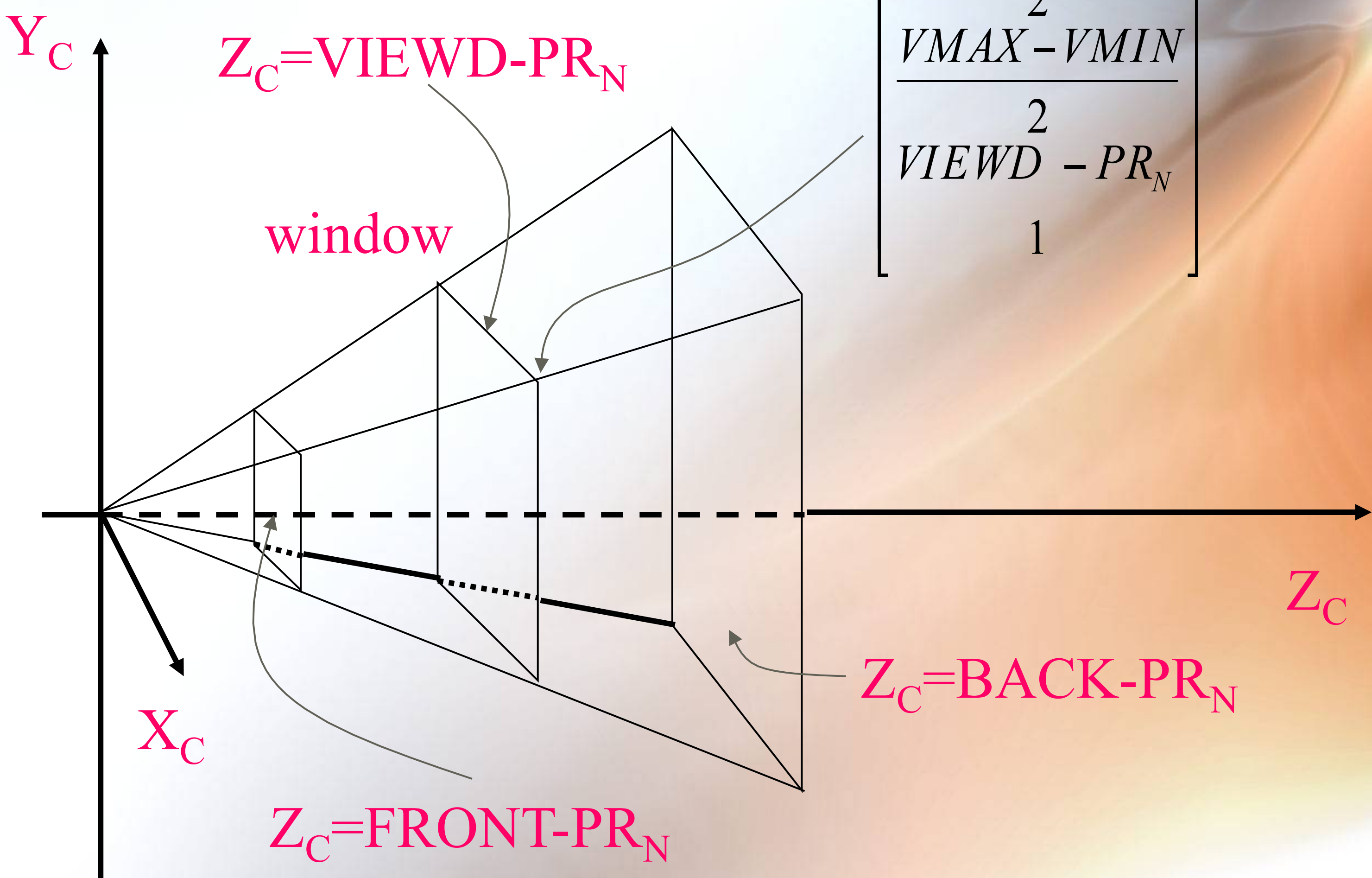


Shear Layout



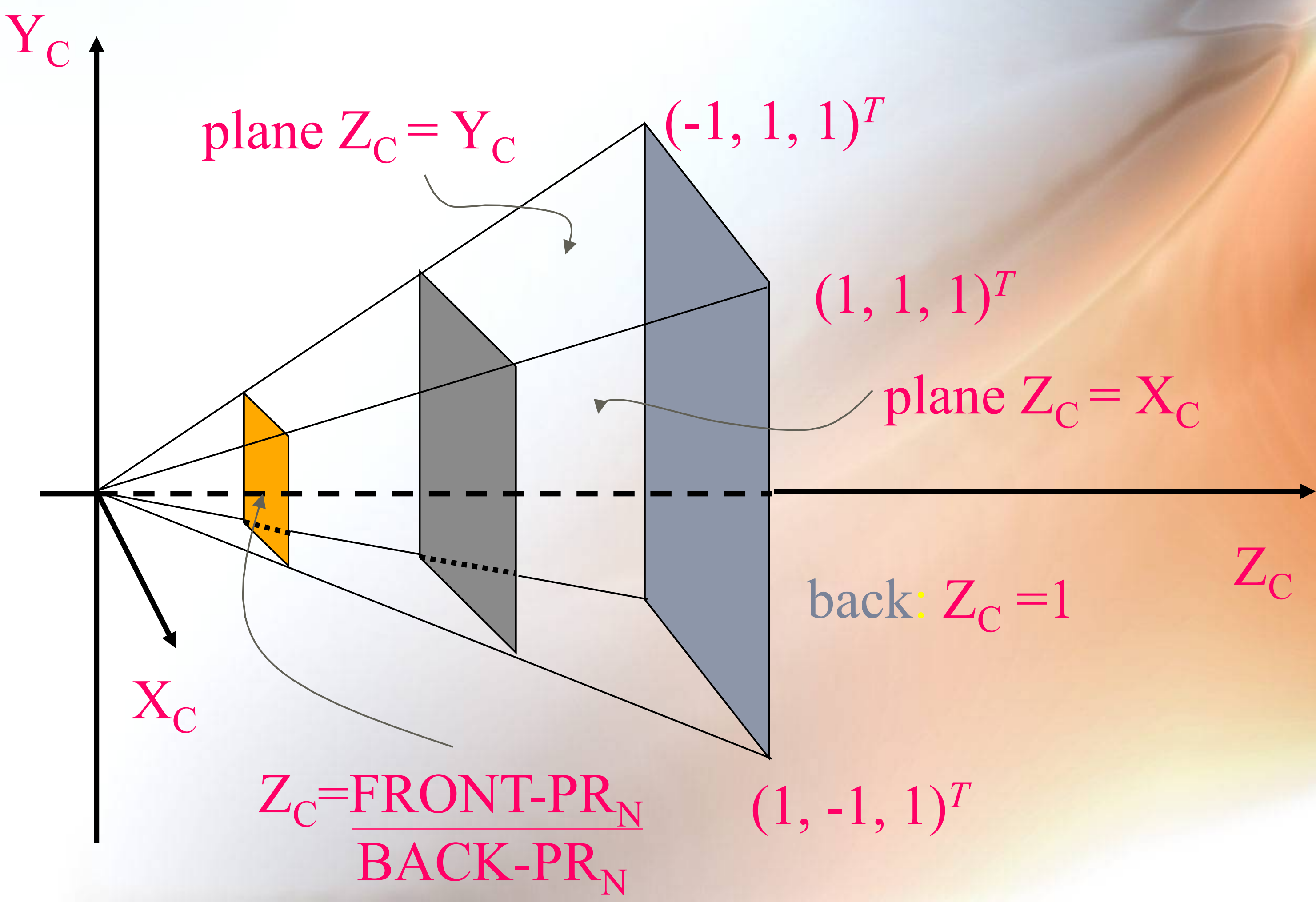
Notice that the view pyramid is not a right pyramid.
We must make it so with the shear transformation

Scaling to Standard View Volume

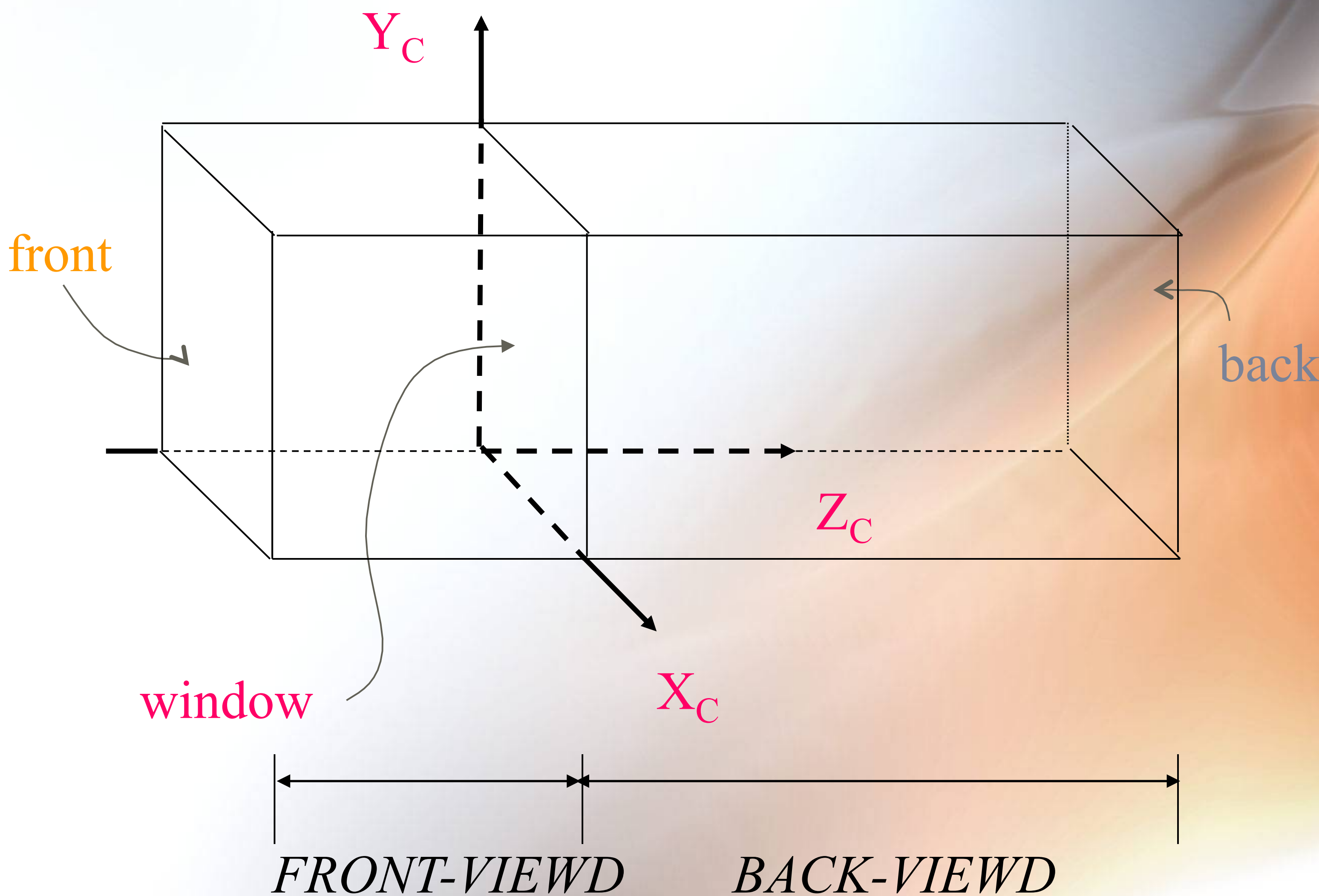


$$\begin{bmatrix} \dfrac{UMAX - UMIN}{2} \\ \dfrac{VMAX - VMIN}{2} \\ VIEWD - PR_N \\ 1 \end{bmatrix}$$

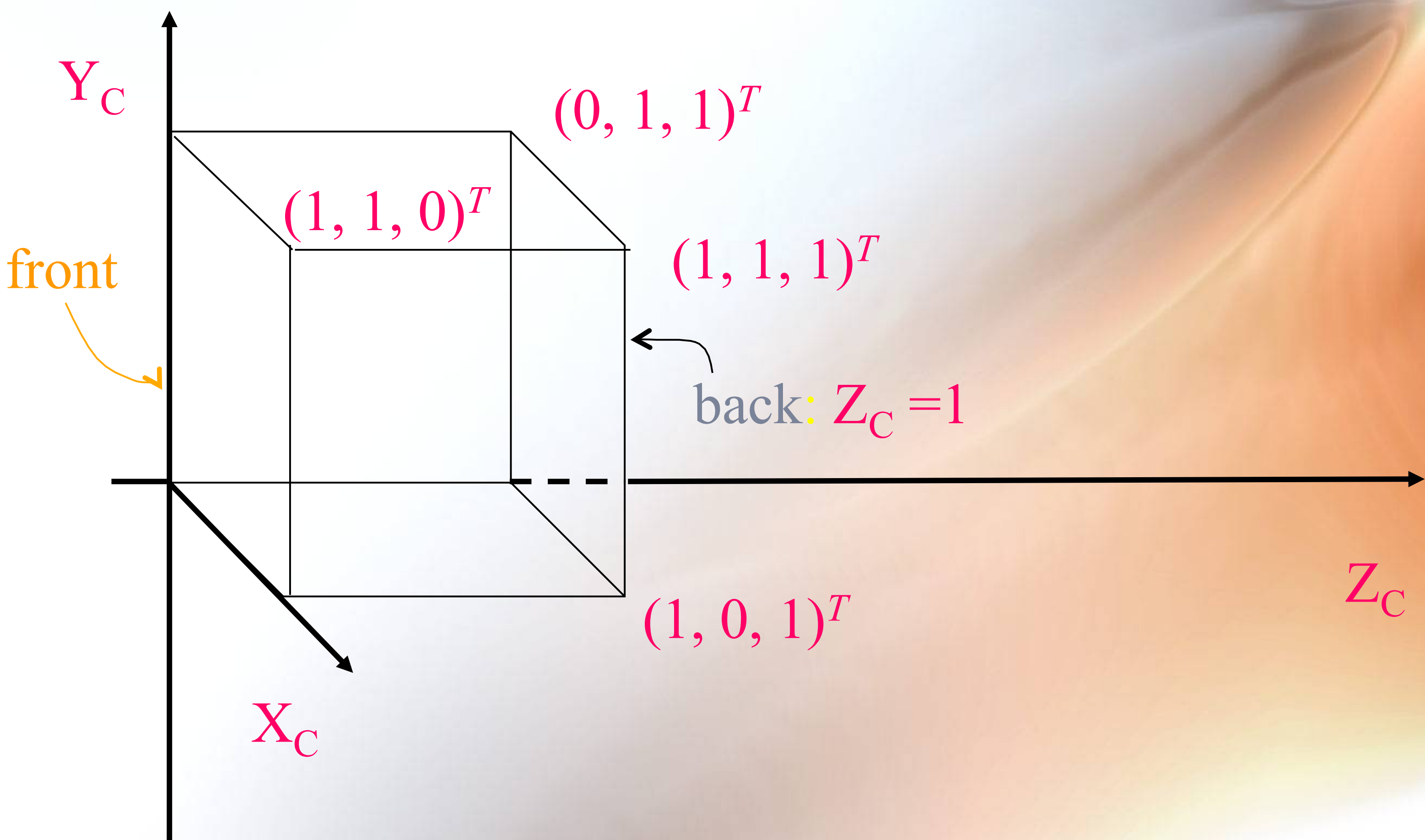
The Standard View Volume for Perspective Case



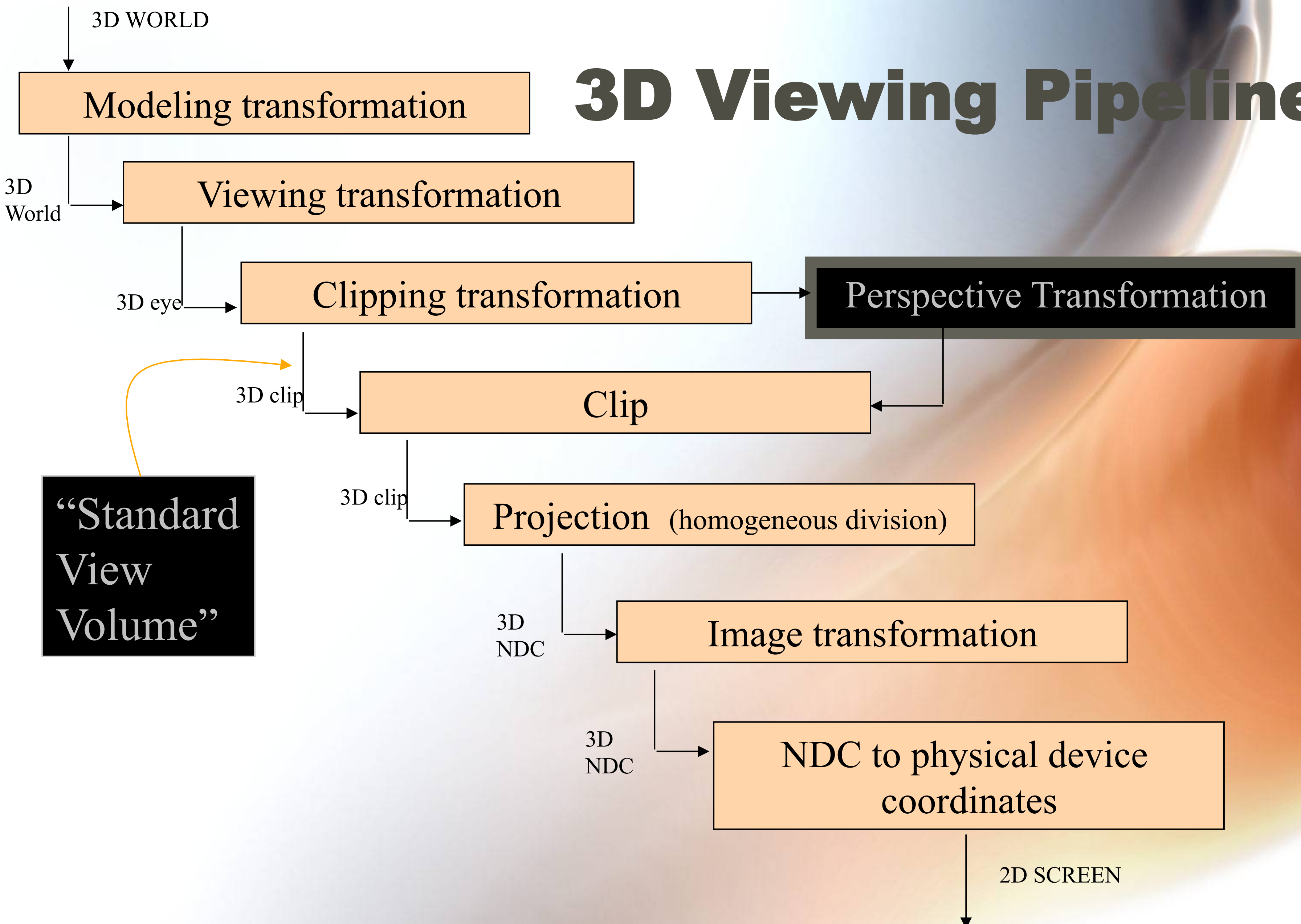
Scaling to Standard View Volume: Parallel



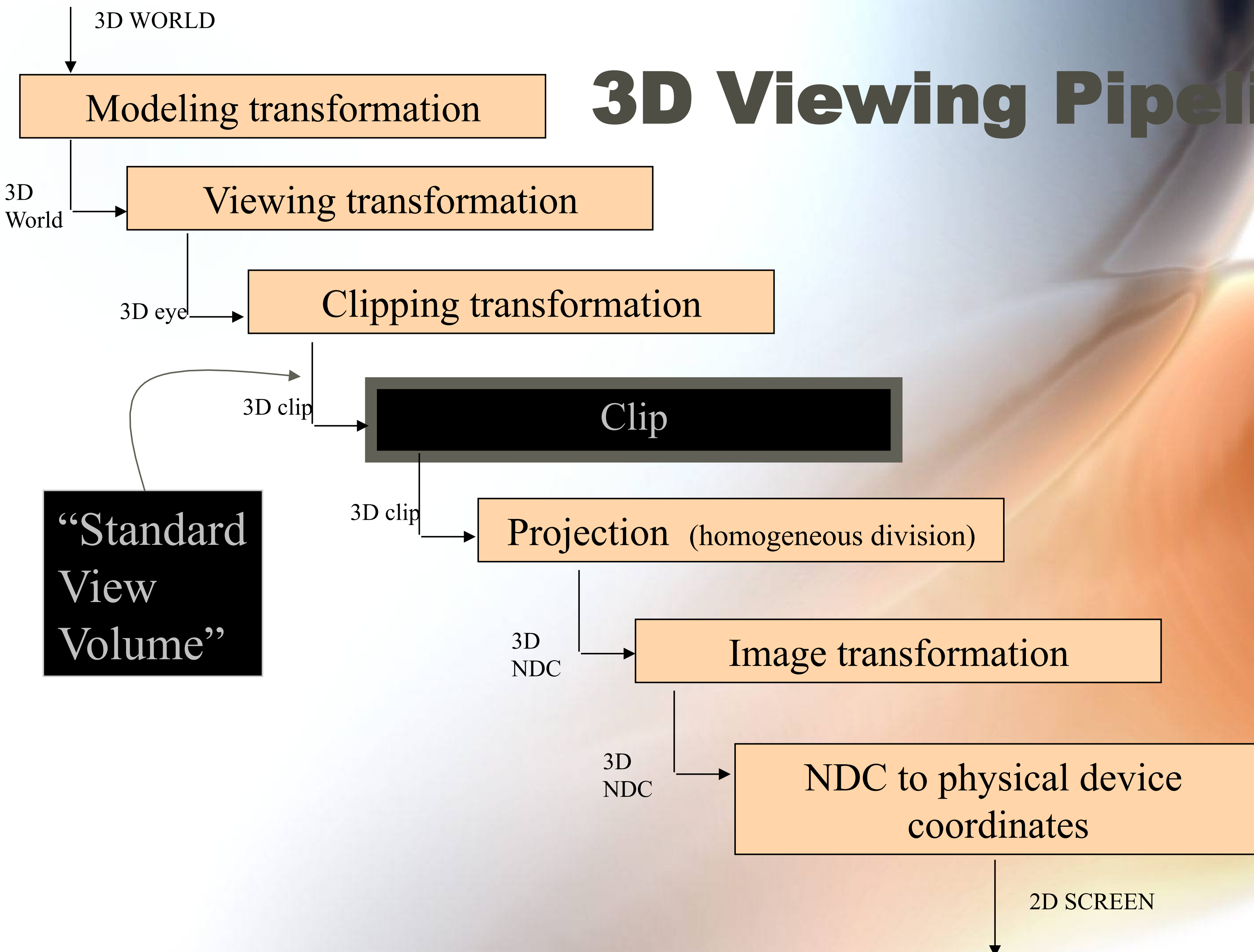
The Standard View Volume for Parallel: The Unit Cube $[0, 1]^3$



3D Viewing Pipeline



3D Viewing Pipeline



Clipping

- Points
- Lines
- Polygons

View Volume Clipping Limits

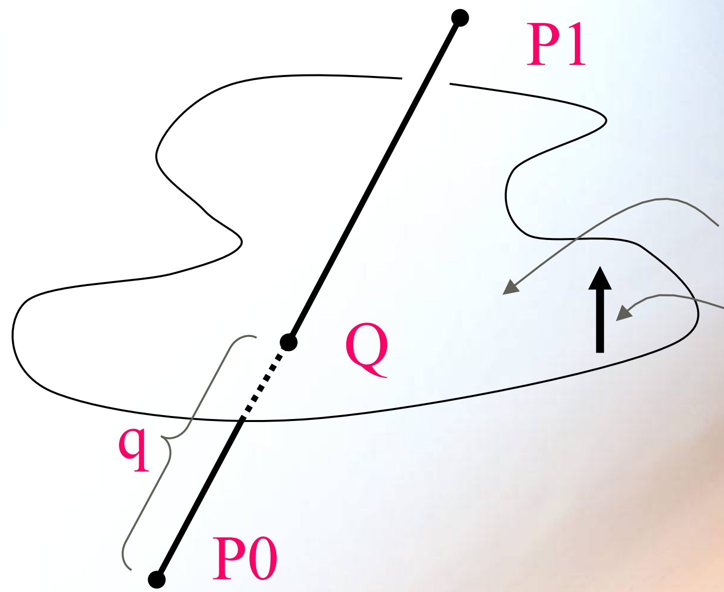
	Parallel	Perspective
Above	$y > 1$	$y > w$
Below	$y < 0$	$y < -w$
Right	$x > 1$	$x > w$
Left	$x < 0$	$x < -w$
Behind (yon)	$z > 1$	$z > w$
In Front (hither)	$z < 0$	$z < 0$

A point (x, y, z) is in the view volume if and only if it lies inside these 6 planes.

Clipping Lines

- Extend 2-D case to 3-D planes.
- Now have 6-bit code rather than 4-bit (above, below, left, right, in-front, behind).
- Only additional work is to find intersection of a line with a clipping plane.
- We might as well do the general case of (non-degenerate) line / plane intersection.

Intersection of Line with Arbitrary Plane



Plane Ax+By+Cz+D=0
normal = (a, b, c)
=(A, B, C)/|(A, B, C)|

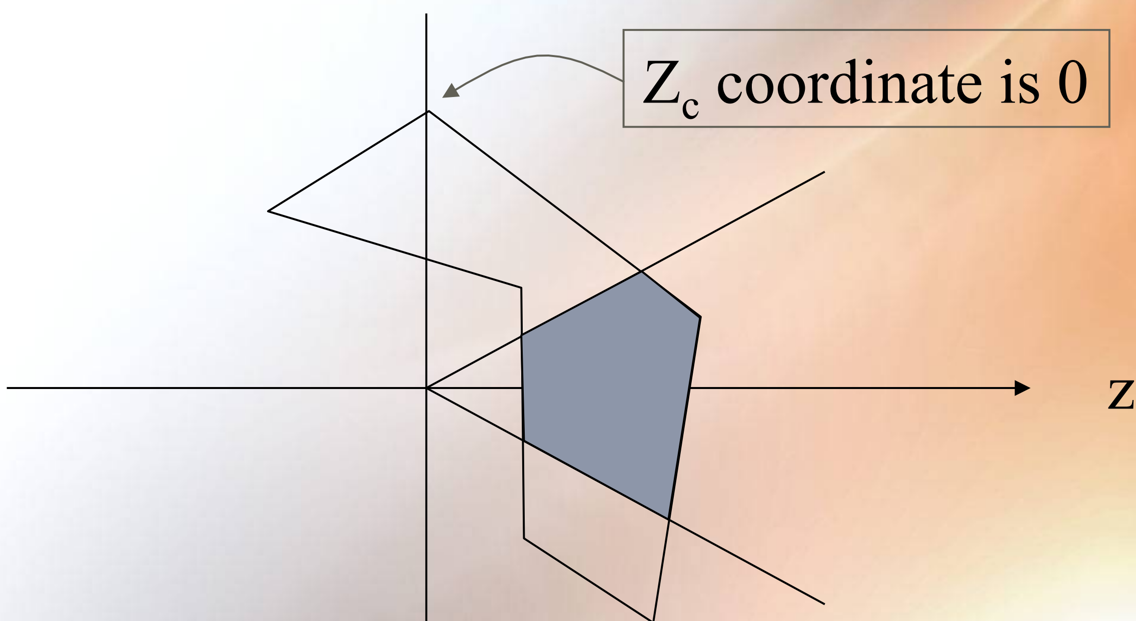
Q = P0 + q (P1 - P0) from parametric form: want Q, thus need q:

q = B0 / (B0 - B1) where

B0 = P0 · (a, b, c) and B1 = P1 · (a, b, c)

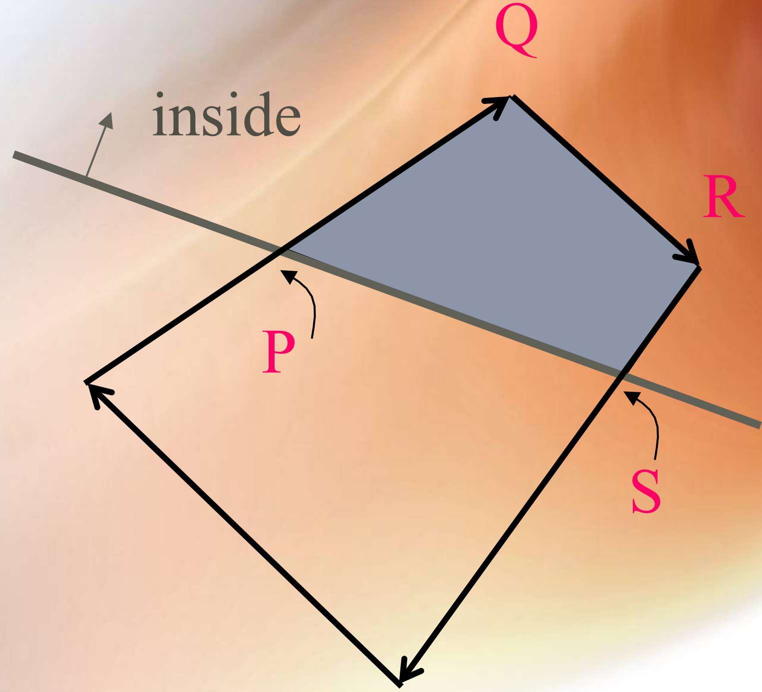
Clipping Polygons

- Clip polygons for visible surface rendering.
- Preserve polygon properties (for rasterization).



Clipping to One Boundary

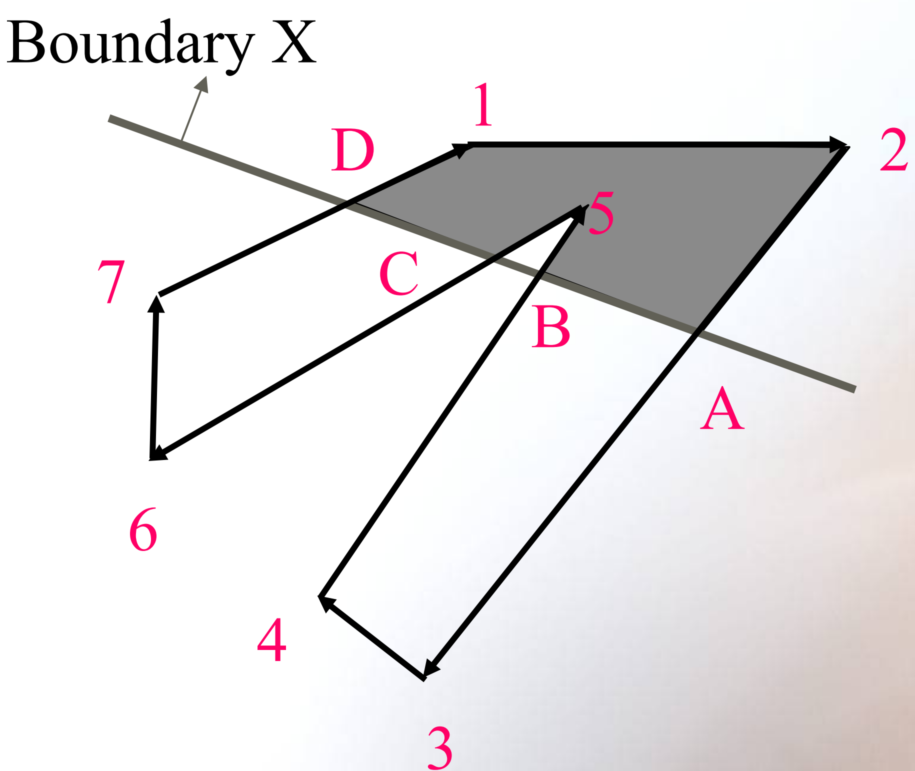
- Consider each polygon edge in turn [O(n)]
- Four cases:
  - ENTER: Output P, Q
  - STAY IN: Output R
  - LEAVE: Output S
  - STAY OUT: (no output)



Therefore clipped polygon is P, Q, R, S.

Clipping Example

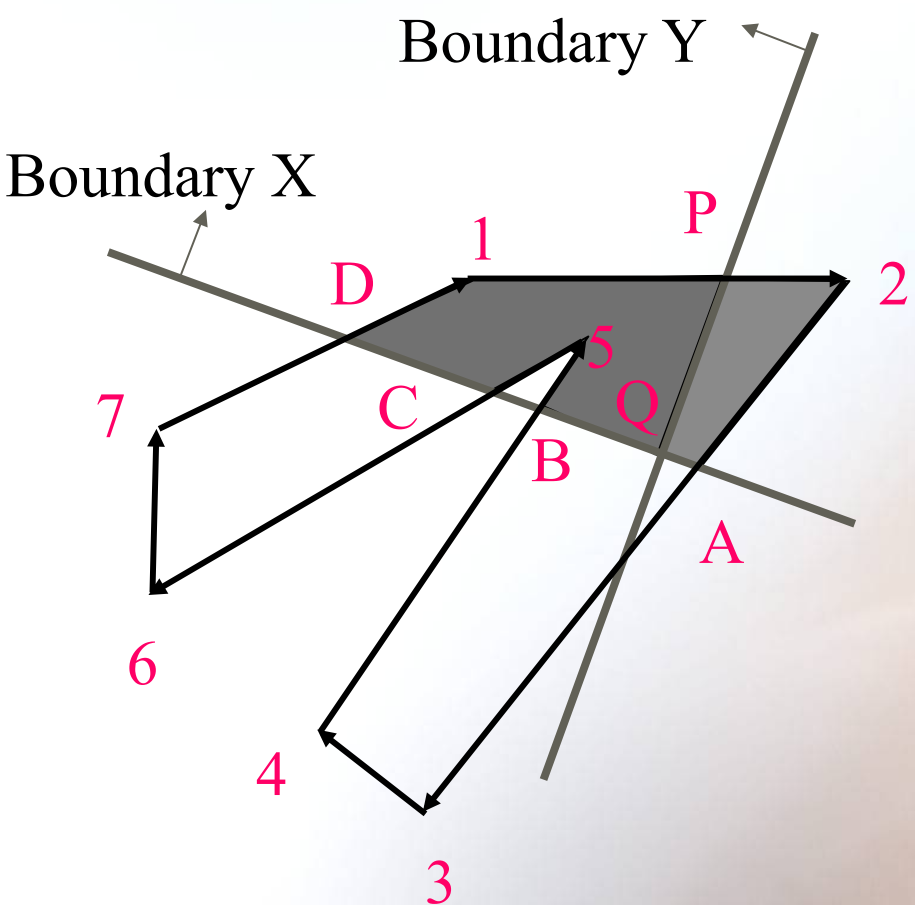
- **Works for more complex shapes.**



Input	Case	Output
1	start	-
2	stay in	2
3	leave	A
4	stay out	-
5	enter	B, 5
6	leave	C
7	stay out	-
1	enter	D, 1

2 A B 5 C D 1

Clipping Against Multiple Boundaries

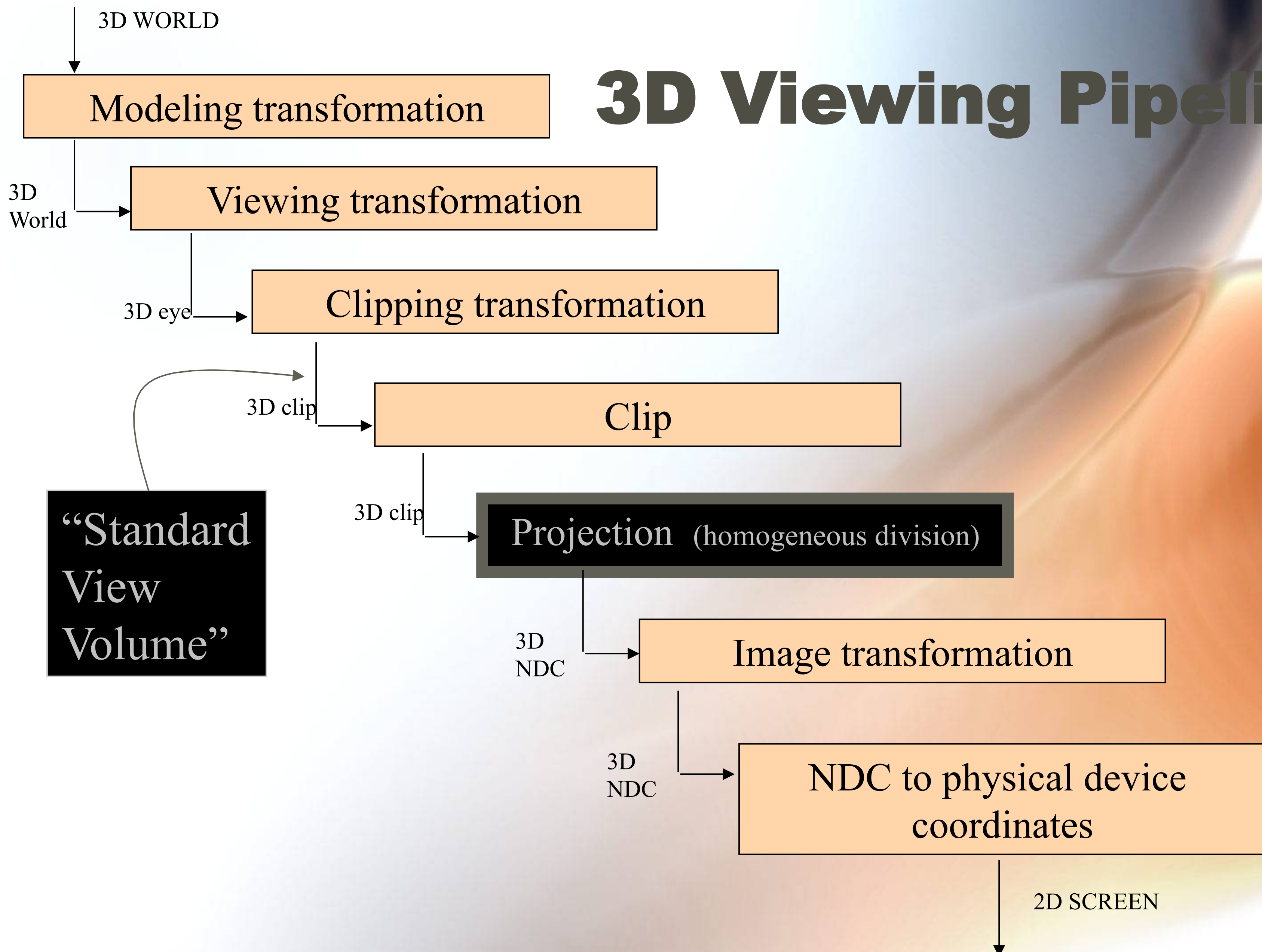


2 A B 5 C D 1

Input	Case	Output
2	start	-
A	stay out	-
B	enter	Q, B
5	stay in	5
C	stay in	C
D	stay in	D
1	stay in	1
2	leave	P

Q B 5 C D 1 P

3D Viewing Pipeline



Normalize Homogeneous Coordinates (Perspective Only)

$$x' = \frac{x}{w}$$

$$y' = \frac{y}{w}$$

$$z' = \frac{z}{w}$$

provided $w \neq 0$

Returns x' and y' in range [-1, 1]
z' in range [0, 1]

3D Window to 3D Viewport in (3D NDC)

- Parallel:
- Standard view volume is unit cube, so nothing to do!

$$X = x_c$$
$$Y = y_c$$
$$Z = z_c$$

- Perspective:
- Must translate view volume by +1 and scale it by 0.5:

$$X = (x_c + 1) / 2$$
$$Y = (y_c + 1) / 2$$
$$Z = z_c$$

3D Viewing Pipeline

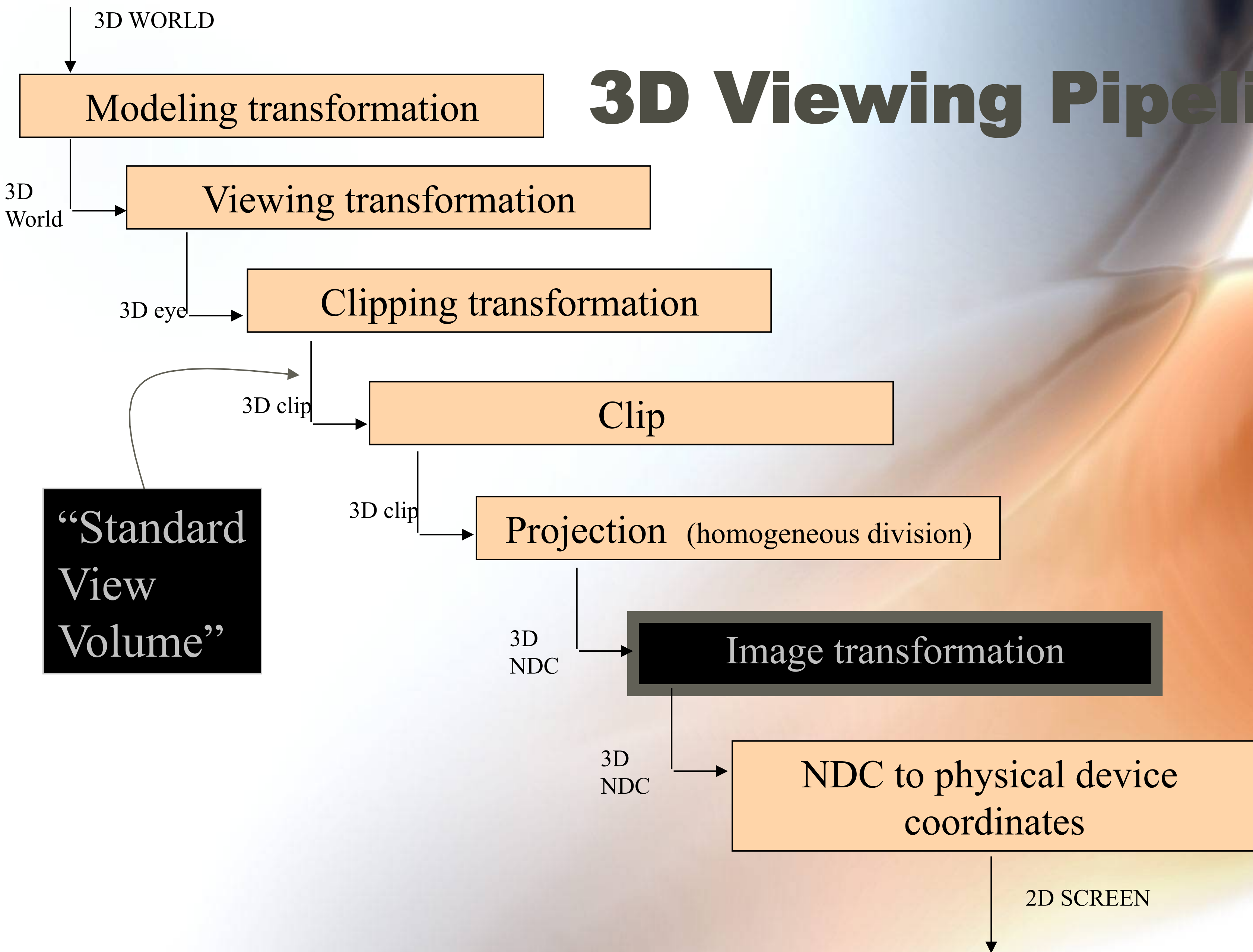
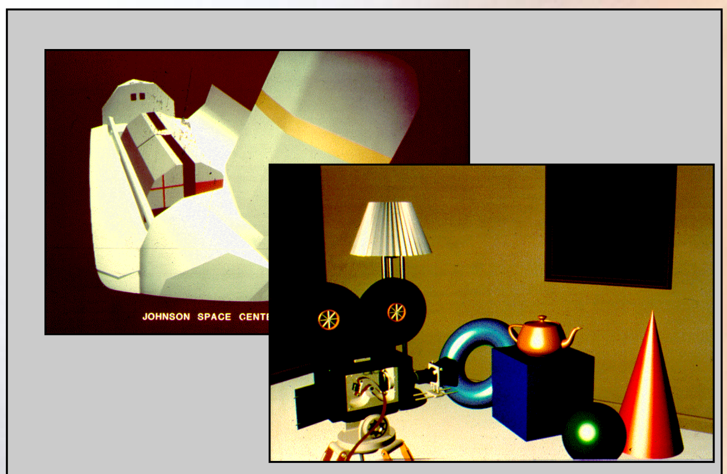
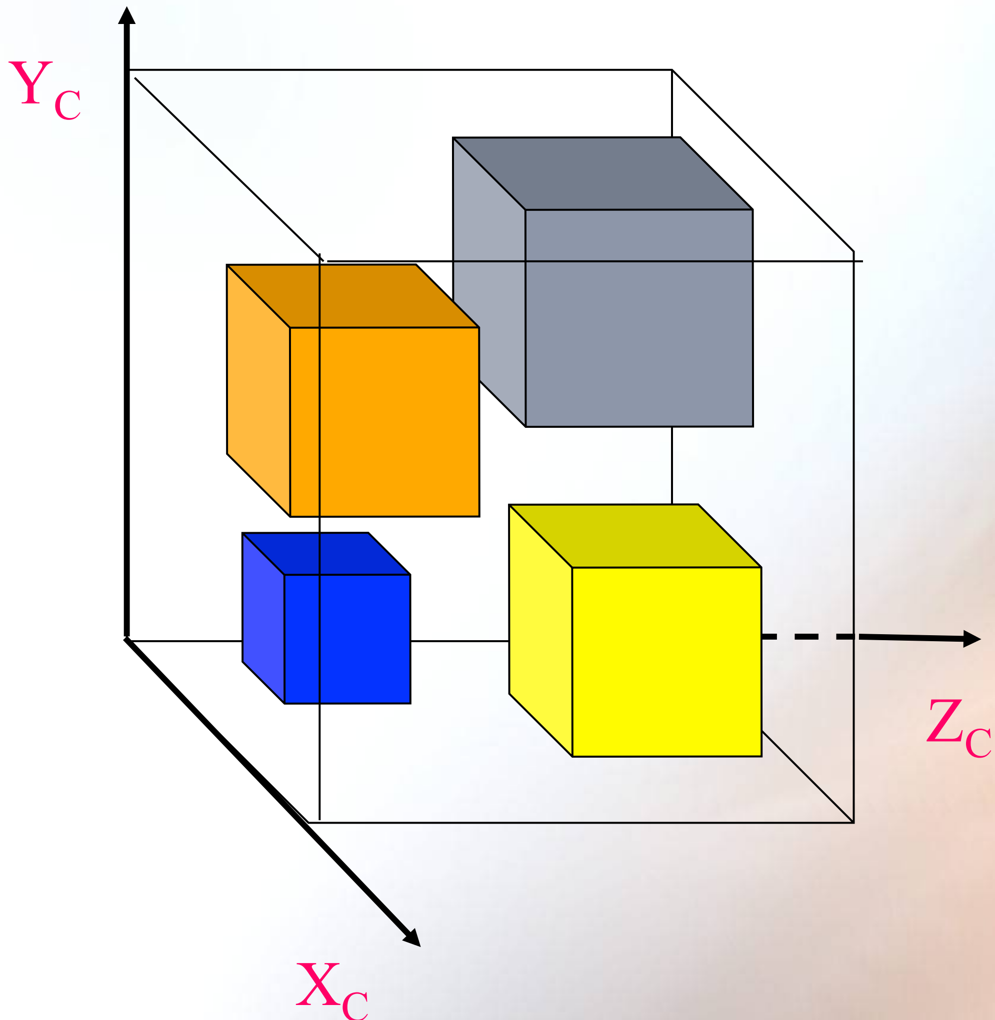


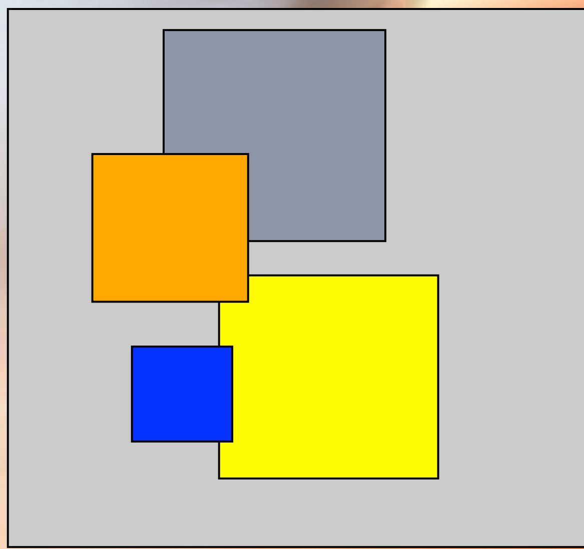
Image Transformations

- Scene transformed into a unit cube $[0,1]^3$.
- We can position this unit cube containing the scene anywhere on the display.
- Obscuration in a layered viewport (e.g. Windows) system.



Viewport Volumes





Screen Appearance (layers displayed back to front)

3D Viewing Pipeline