Lecture 3 of 41

Viewing 2 of 4:
Viewing Transformation

William H. Hsu
Department of Computing and Information Sciences, KSU

Public mirror web site: http://www.kddresearch.org/Courses/CIS636
Instructor home page: http://www.cis.ksu.edu/~bhsu

Readings:

Today: Section 2.3 (esp. 2.3.4), Eberly 2e – see http://bit.ly/ieUq45
Next class: Section 2.3 (esp. 2.3.7), 2.6, 2.7
This week: FVFH slides on Viewing

Lecture Outline

- Reading for Last Class: Sections 2.2.3 – 2.2.4, 2.8 Eberly 2e
- Reading for Today: Section 2.3 (esp. 2.3.4), Foley et al. Slides
- Reading for Next Class: Section 2.3 (esp. 2.3.7), 2.6, 2.7
- Last Time: Basic Viewing Principles
  - Projections: definitions, history
  - Perspective: optical principles, terminology
- Today: View Volume Specification and Viewing Transformation
  - View volumes: ideal vs. approximated
  - Frustum in computer graphics (CG)
  - Specifying view volume in CG: Look and Up vectors
  - Aspect ratio, view angle, front/back clipping planes
  - Focal length
  - Parallel (cuboid) view volume & perspective frustum
  - Normalizing transformation (NT) & viewing transformation (VT)
- Next Time: Fixed-Function Graphics Pipeline
Where We Are

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Topic</th>
<th>Primary Source(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Course Overview</td>
<td>Chapter 1, Eberly 2</td>
</tr>
<tr>
<td>1</td>
<td>CG Basics: Transformation Matrices; Lab 0</td>
<td>Sections (2.1, 2.2)</td>
</tr>
<tr>
<td>2</td>
<td>Viewing 1: Overview, Projections</td>
<td>§ 2.3 esp. 2.3.4, PVM slides</td>
</tr>
<tr>
<td>3</td>
<td>Viewing 2: Viewing Transformation</td>
<td>§ 2.3 esp. 2.3.4, PVM slides</td>
</tr>
<tr>
<td>4</td>
<td>Lab 13: Flush x OpenGL Basics</td>
<td>Ch. 2, 18, Alligant Primer</td>
</tr>
<tr>
<td>5</td>
<td>Viewing 3: Graphics Pipeline</td>
<td>§ 2.3 esp. 2.3.2, 2.6, 2.7</td>
</tr>
<tr>
<td>6</td>
<td>Scan Conversion 1 Lines, Midpoint Algorithm</td>
<td>§ 2.5.1, 3.1, PVM slides</td>
</tr>
<tr>
<td>7</td>
<td>Viewing 4: Clipping &amp; Culling; Lab 1b</td>
<td>§ 2.3, 2.4, 3.1.3</td>
</tr>
<tr>
<td>8</td>
<td>Scan Conversion 2 Polygons, Clipping Intro</td>
<td>§ 2.3, 2.5 esp. 2.3.4, 3.16</td>
</tr>
<tr>
<td>9</td>
<td>Surface Detail 1: Illumination &amp; Shading</td>
<td>§ 2.5, 2.6.1, 2.6.2, 4.3.2, 20.2</td>
</tr>
<tr>
<td>10</td>
<td>Lab 2a: Direct3D / DirectX Intro</td>
<td>§ 2.7, Direct3D handout</td>
</tr>
<tr>
<td>11</td>
<td>Surface Detail 2: Textures, OpenGL Shading</td>
<td>§ 2.6.3, 20.3 – 20.4, Primer</td>
</tr>
<tr>
<td>12</td>
<td>Surface Detail 3: Mappings, OpenGL Textures</td>
<td>§ 20.5 – 20.13</td>
</tr>
<tr>
<td>13</td>
<td>Surface Detail 4: Pixel/Vertex Shading; Lab 2b</td>
<td>§ 3.1</td>
</tr>
<tr>
<td>14</td>
<td>Surface Detail 5: Direct3D Shading, GGLSL</td>
<td>§ 3.2 – 3.4, Direct3D handout</td>
</tr>
<tr>
<td>15</td>
<td>Demos 1: CGA, Fun, Scene Graphs, State</td>
<td>§ 4.1 – 4.3, CGA handout</td>
</tr>
<tr>
<td>16</td>
<td>Lab 3a: Shading &amp; Transparency</td>
<td>§ 2.6, 20.1, Primer</td>
</tr>
<tr>
<td>17</td>
<td>Animation 1: Basics, Keyframes; HW/Exam</td>
<td>§ 5.1 – 5.2</td>
</tr>
<tr>
<td>18</td>
<td>Scene Graphics: Rendering; Lab 3b: Shader</td>
<td>§ 4.1 – 4.7</td>
</tr>
<tr>
<td>19</td>
<td>Demos 2: SFX, Skinning, Morphing</td>
<td>§ 6.3 – 6.6, CGA handout</td>
</tr>
<tr>
<td>20</td>
<td>Demos 3: Surfaces, B-rays/Volume Graphics</td>
<td>§ 10.4, 12.7, Mesh handout</td>
</tr>
</tbody>
</table>

Acknowledgements

Jim Foley  
Professor, College of Computing &  
Stephen Fleming Chair in  
Telecommunications  
Georgia Institute of Technology

Andy van Dam  
T. J. Watson University Professor of  
Technology and Education &  
Professor of Computer Science  
Brown University

Steve Feiner  
Professor of Computer Science &  
Director, Computer Graphics and User Interfaces Laboratory  
Columbia University

John F. Hughes  
Associate Professor of Computer Science  
Brown University

James D. Foley  
Georgia Tech  

Andries van Dam  
Brown University  
http://www.cs.brown.edu/~avd/

Steven K. Feiner  
Professor of Computer Science  
Director, Computer Graphics and User Interfaces Laboratory  
Columbia University  
http://www.cs.columbia.edu/~feiner/

John F. Hughes  
Associate Professor of Computer Science  
Brown University  
http://www.cs.brown.edu/~jf/
Review: Planar Geometric Projection

- Projectors are straight lines, like the string in Durer’s “Artist Drawing a Lute”.
- Projection surface is plane (picture plane, projection plane)

Review: Types of Projections

- Parallel projections used for engineering and architecture because they can be used for measurements
- Perspective imitates eyes or camera and looks more natural
Review: Parallel Projections

1) Multiview Orthographic
   - VPN || a principal coordinate axis
   - DOP || VPN
   - shows single face, exact measurements

2) Axonometric
   - VPN \not\parallel a principal coordinate axis
   - DOP || VPN
   - adjacent faces, none exact, uniformly foreshortened
     (function of angle between face normal and DOP)

3) Oblique
   - VPN || a principal coordinate axis
   - DOP \parallel VPN
   - adjacent faces, one exact, others uniformly foreshortened

Assume object face of interest lies in principal plane, i.e., parallel to xy, yz, or zx planes. (DOP = Direction of Projection, VPN = View Plane Normal)

3-D Viewing: Synthetic Camera

- The synthetic camera is the programmer's model to specify 3D view projection parameters to the computer
- General synthetic camera: each package has its own but they are all (nearly) equivalent. (PHIGS* Camera, Computer Graphics: Principles and Practice, ch. 6 and 7)
  - position of camera
  - orientation
  - field of view (wide angle, normal...)
  - depth of field (near distance, far distance)
  - focal distance
  - tilt of view/film plane (if not normal to view direction, produces oblique projections)
  - perspective or parallel projection? (camera near objects or an infinite distance away)
- CS123 uses a simpler, slightly less powerful model than the book's
  - omit tilt of view/film plane, focal distance

* This package is no longer in use but still has the most general synthetic camera model for perspective and parallel projections.

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
View Volumes

- A view volume contains everything visible from the point of view or direction:
  - what does the camera see?
- Conic view volumes:
  - approximates what eye sees
  - expensive math (simultaneous quadratics) when clipping objects against cone’s surface
- Can approximate with rectangular cone instead (called a frustum)
  - works well with a rectangular viewing window
  - simultaneous linear equations for easy clipping of objects against slises

Conceptual Model of 3-D Viewing Process (for Wireframe)

- Viewport is rectangular area of the screen where a scene is rendered:
  - this may or may not fill Window Manager’s window
  - note: window in computer graphics often means a 2D clip rectangle on a 2D world coordinate drawing, and viewport is the 2D integer coordinate region of screen space to which the clipped window contents are mapped. Window/viewpoint terminology considerably predates Window Manager terminology
- Viewport and 2D cross-section of 3D view volume may have different aspect ratios:
  - viewport mapping (from film plane window to viewpoint’s 2D device coordinates) specifies what to do if aspect ratios differ
View Volume Specification [1]

- We need to know six things about our synthetic camera model in order to take a picture.

1) Position of the camera (from where it’s looking)
2) The Look vector specifies in what direction the camera is pointing
3) The camera’s Orientation is determined by the Look vector and the angle through which the camera is rotated about that vector, i.e., the direction of the Up vector

View Volume Specification [2]

- Aspect ratio of the electronic “film;” ratio of width to height
- Height angle: determines how much of the scene we will fit into our view volume; larger height angles fit more of the scene into the view volume (width angle determined by height angle and aspect ratio)
  - the greater the angle, the greater the amount of perspective distortion
- Front and back clipping planes: limit extent of camera’s view by rendering (parts of) objects lying between them and throwing away everything outside of them
  - Optional parameter — Focal length: often used for photorealistic rendering; objects at distance Focal length from camera are rendered in sharp focus, objects closer or farther away get blurred.
  - Your camera does not need to implement focal length blurring.
**Position**

- Determining the position is analogous to a photographer deciding the vantage point from which to shoot a photo.
- Three degrees of freedom: $x$, $y$, and $z$ coordinates in 3-space.
- This $x$, $y$, $z$ coordinate system is right-handed: if you open your right hand, align your palm and fingers with the $+x$ axis, and curl your middle finger towards the $+y$ axis, your thumb will point along the $+z$ axis.

Adapted from slides © 1997–2010 van Dam et al., Brown University

**Orientation**

- Orientation is specified by a point in 3D space to look at (or a direction to look in) and an angle of rotation about this direction.
- Default (canonical) orientation is looking down the negative $z$-axis and up direction pointing straight up the $y$-axis.
- In general the camera is located at the origin and is looking at an arbitrary point with an arbitrary up direction.

- This is a little abstract. Is there a easier formulation?

Adapted from slides © 1997–2010 van Dam et al., Brown University
Look and Up Vectors

- More concrete way to say the same thing as orientation
  - soon you'll learn how to express orientation in terms of Look and Up vectors
- Look Vector
  - the direction the camera is pointing
  - three degrees of freedom; can be any vector in 3-space
- Up Vector
  - determines how the camera is rotated around the Look vector
  - for example, whether you're holding the camera horizontally or vertically (or in between)
  - Up vector must not be parallel to Look vector (Up vector may be specified at an arbitrary angle to its Look vector)

Note: For ease of specification, the Up vector need not be orthogonal to the Look vector as long as they are not parallel.

Aspect Ratio

- Analogous to the size of film used in a camera
- Determines proportion of width to height of image displayed on screen
- Square viewing window has aspect ratio of 1:1
- Movie theater “letterbox” format has aspect ratio of 2:1
- NTSC television has an aspect ratio of 4:3, and HDTV is 16:9 or 16:10
**View Angle [1]**

- Determines amount of perspective distortion in picture, from none (parallel projection) to a lot (wide-angle lens).
- In a frustum, two viewing angles: width and height angles.
- Choosing View angle analogous to photographer choosing a specific type of lens (e.g., a wide-angle or telephoto lens).

![Diagram](image)


**View Angle [2]**

- Lenses made for distance shots often have a nearly parallel viewing angle and cause little perspective distortion, though they foreshorten depth.
- Wide-angle lenses cause a lot of perspective distortion.

![Diagram](image)

Front and Back Clipping Planes [1]

- Volume of space between Front and Back clipping planes defines what camera can see
- Position of planes defined by distance along Look vector
- Objects appearing outside of view volume don’t get drawn
- Objects intersecting view volume get clipped

Front and Back Clipping Planes [2]

- Reasons for Front (near) clipping plane:
  - Don’t want to draw things too close to the camera
    - would block view of rest of scene
    - objects would be prone to distortion
  - Don’t want to draw things behind camera
    - wouldn’t expect to see things behind the camera
    - in the case of the perspective camera, if we were to draw things behind the camera, they would appear upside-down and inside-out because of perspective transformation
- Reasons for Back (far) clipping plane:
  - Don’t want to draw objects too far away from camera
    - distant objects may appear too small to be visually significant, but still take long time to render
    - by discarding them we lose a small amount of detail but reclaim a lot of rendering time
    - alternately, the scene may be filled with many significant objects; for visual clarity, we may wish to declutter the scene by rendering those nearest the camera and discarding the rest
Front and Back Clipping Planes [3]

- Have you ever played a video game and all of the sudden some object pops up in the background (e.g., a tree in a racing game)? That's the object coming inside the far clip plane.
- The old hack to keep you from noticing the pop-up is to add fog in the distance. A classic example of this is from Turjak: Dinosaur Hunter.
- Now all you notice is fog. This practically defeats the purpose of an outdoor environment! And you can still see pop-up from time to time.
- Thanks to fast hardware and level of detail algorithms, we can push the far plane back now and fog is much less prevalent.

Adapted from slides © 1997 – 2010 van Dam et al., Brown University

Front and Back Clipping Planes [4]

- Putting the near clip plane as far away as possible helps 2 precision. Sometimes in a game you can position the camera in the right spot so that the front of an object gets clipped letting you see inside of it.
- Modern video games uses various techniques to avoid this visual glitch. One technique is to have objects that are very close to the near clip plane fade out before they get cut off, as can be seen from these screenshots of Okami.

This technique gives a clean look while solving the near clipping problem (the wooden fence fades out as the camera follows the running wolf).

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
Focal Length

- Some camera models take a focal length
- Focal Length is a measure of ideal focusing range; approximates behavior of real camera lens
- Objects at distance equal to focal length from camera are rendered in focus; objects closer or farther away than focal length get blurred
- Focal length used in conjunction with clipping planes
- Only objects within view volume are rendered, whether blurred or not. Objects outside of view volume still get discarded

What This Model Can and Cannot Do

- It can create the following view volumes:
  - perspective: positive view angle
  - parallel: zero view angle
- Model cannot create oblique view volume
- Non-oblique vs. oblique view volumes:
  - Non-oblique view volume:
    - Look vector is perpendicular to film plane
  - Oblique view volume:
    - Look vector is at an angle to the film plane
- For example, view cameras with bellows are used to take pictures of (tall) buildings. The film plane is parallel to the façade, while the camera points up. This is an oblique view volume, with the façade undistorted
View volume Specification

- From Position, Look vector, Up vector, Aspect ratio, Height angle, Clipping planes, and (optionally) Focal length together specify a truncated view volume
- Truncated view volume is a specification of bounded space that camera can "see"
- 2D view of 3D scene can be computed from truncated view volume and projected onto film plane
- Truncated view volumes come in two flavors: parallel and perspective

Truncated view volume means we only need to render what the camera can see

Truncated View Volume for Orthographic Parallel Projection

- Limiting view volume useful for eliminating extraneous objects
- Orthographic parallel projection has width and height view angles of zero

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
Truncated View Volume (Frustum) for Perspective Projection

- Projection of up vector
- Width Angle
- Height Angle
- Removes objects too far from Position, which otherwise would merge into "blobs"
- Removes objects too close to Position (would be excessively distorted)

Where’s My Film?

Real cameras have a roll of film that captures pictures
- Synthetic camera “film” is a rectangle on an infinite film plane that contains image of scene
- Why haven’t we talked about the “film” in our synthetic camera, other than mentioning its aspect ratio?
- How is the film plane positioned relative to the other parts of the camera? Does it lie between the near and far clipping planes? Behind them?
- Turns out that fine positioning of Film plane doesn’t matter. Here’s why:
  - for a parallel view volume, as long as the film plane lies in front of the scene, parallel projection onto film plane will look the same no matter how far away film plane is from scene
  - same is true for perspective view volumes, because the last step of compiling the perspective projection is a transformation that stretches the perspective volume into a parallel volume
- To be explained in detail in the next lecture
- In general, it is convenient to think of the film plane as lying at the far clip plane
Viewing Transformation: World \((x, y, z)\) to Camera \((u, v, w)\)

- Placement of view volume (visible part of world) specified by camera's position and orientation
  - Positive (a point)
  - Look and up vectors
- Shape of view volume specified by
  - horizontal and vertical view angles
  - Front and back clipping planes
- Perspective projection: projectors intersect at Position
- Parallel projection: projectors parallel to look vector, but never intersect (or intersect at infinity)
- Coordinate Systems
  - world coordinates – standard right-hand 3-space
  - camera coordinates – camera-space right-handed coordinate system \((u, v, w)\); origin at position and axes related by orientation; used for transforming arbitrary view into canonical view

Adapted from slides © 1997 – 2010 van Dam et al., Brown University
Summary

- Last Time: Taxonomy of Projections
  - Parallel: orthographic (multiview, axonometric), oblique
  - Perspective: one-, two-, three-point
- Today: View Volume Specification and Viewing Transformation
  - View volumes: ideal vs. approximated
  - Frustum in computer graphics (CG)
  - Specifying view volume in CG: Look and Up vectors
  - Aspect ratio, view angle, front/back clipping planes
  - Focal length
  - Parallel (cuboid) view volume & perspective frustum
  - Viewing transformation (VT) preview

Next Time
- Normalizing transformation (NT)
- Fixed-function pipeline

Terminology

- Parallel Projections
  - Orthographic: “dead on”, i.e., \((\text{DOP} \parallel \text{VPN})\)
  - Oblique: “at an angle”, i.e., \((\text{DOP} \not\parallel \text{VPN})\)
- Perspective Projection
  - One-point: one vanishing point, one axis cut by view plane (WOLOG, z axis)
  - Two-point: two vanishing points, two axes cut (WOLOG, x & z axes)
  - Three-point: three vanishing points, three axes cut (or object rotated freely)
- Today: View Volumes, Viewing Transformation
  - Parallel projection (cuboid) view volume vs. perspective projection frustum
  - Look vector (DOP): Eye (COP) to At
  - Up vector: with Look, forms plane perpendicular (orthogonal) to view plane
  - Aspect ratio: width to height (of viewport)
  - View angle: wide angle (fisheye lens) vs. nearly-parallel (telephoto)
  - Front/back (near/far) clipping planes
  - World coordinates (canonical): \((x, y, z)\)
  - User coordinates (arbitrary): \((u, v, w)\), aka \((u, v, n)\) or \((R, U, D)\) in Eberly