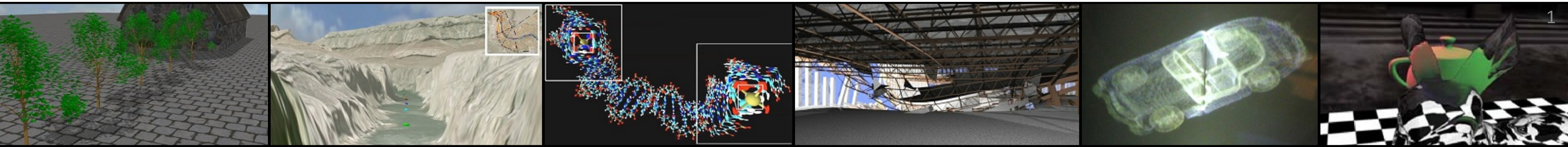
CIS 4930-001: Introduction to Augmented and Virtual Reality

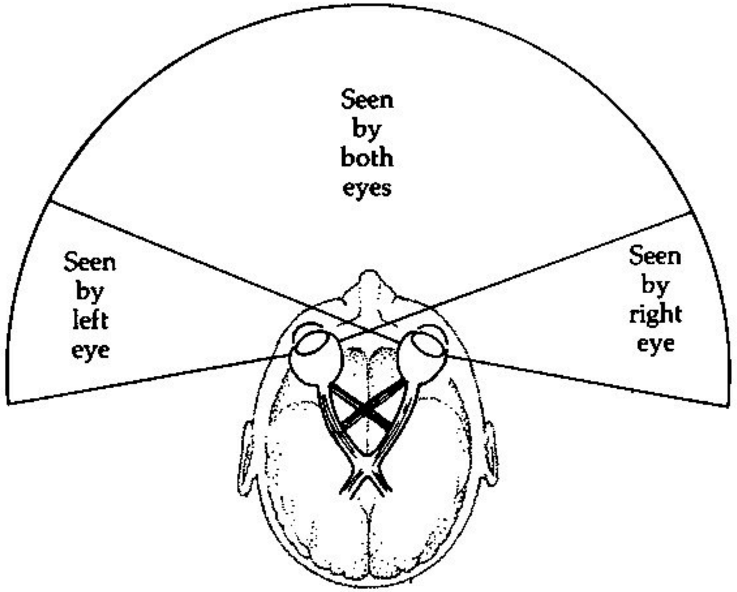
Depth and Projection

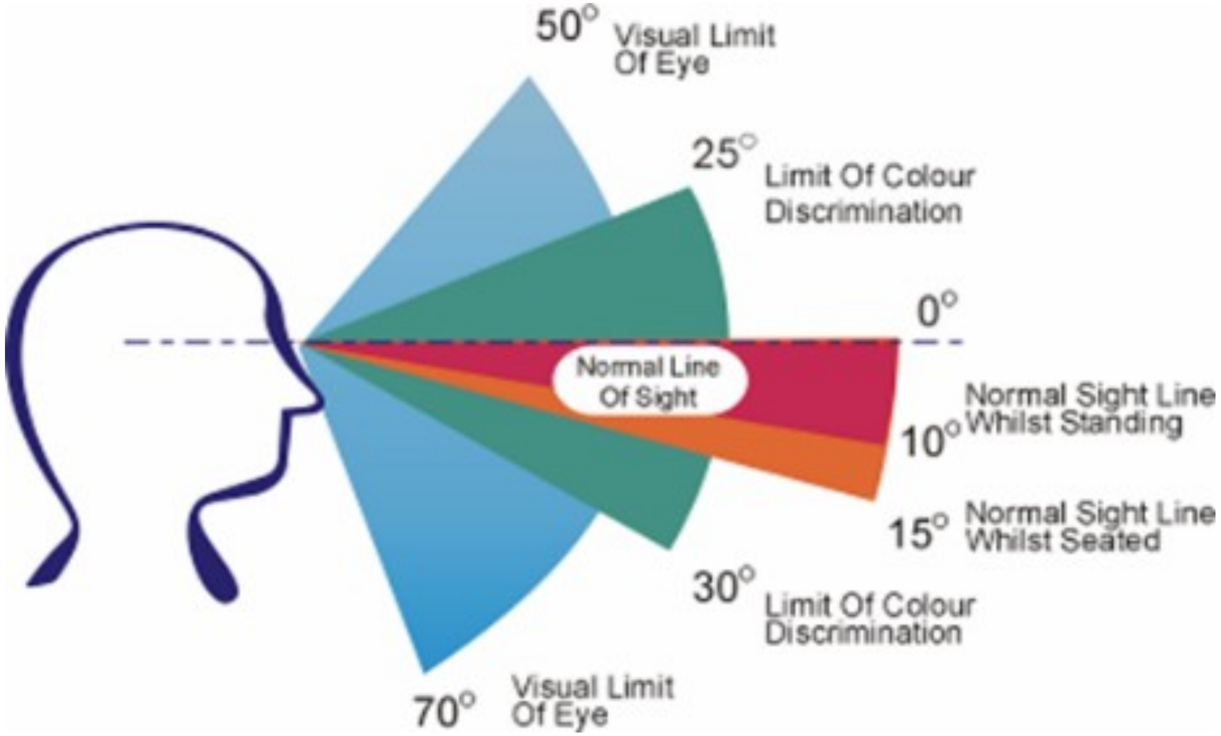
Paul Rosen
Assistant Professor
University of South Florida

Some slides from: Anders Backman, Mark Billinghurst, Doug Bowman, David Johnson, Gun Lee, Ivan Poupyrev, Bruce Thomas, Geb Thomas, Anna Yershova, Stefanie Zollman

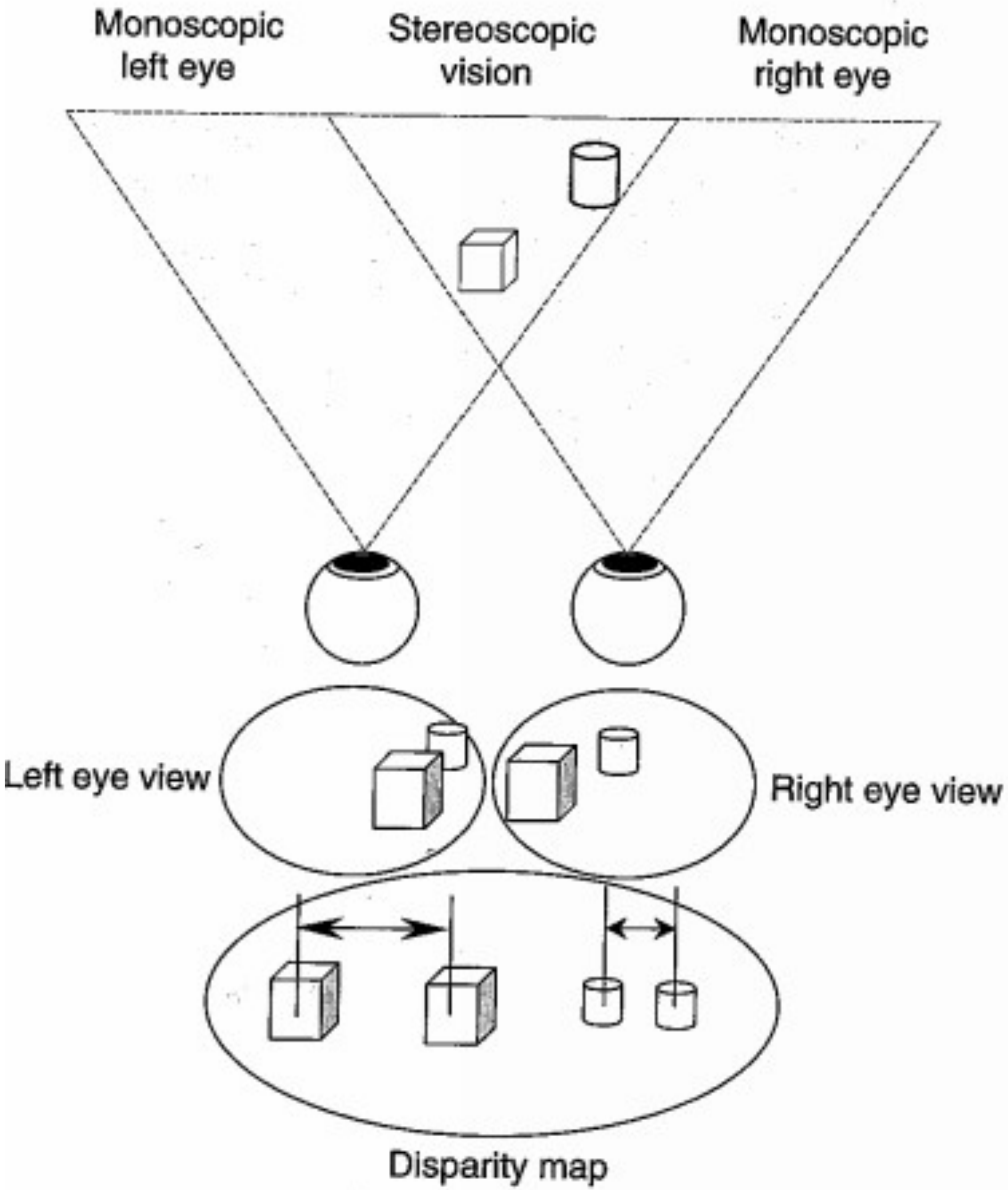


HUMAN HORIZONTAL AND VERTICAL FOV

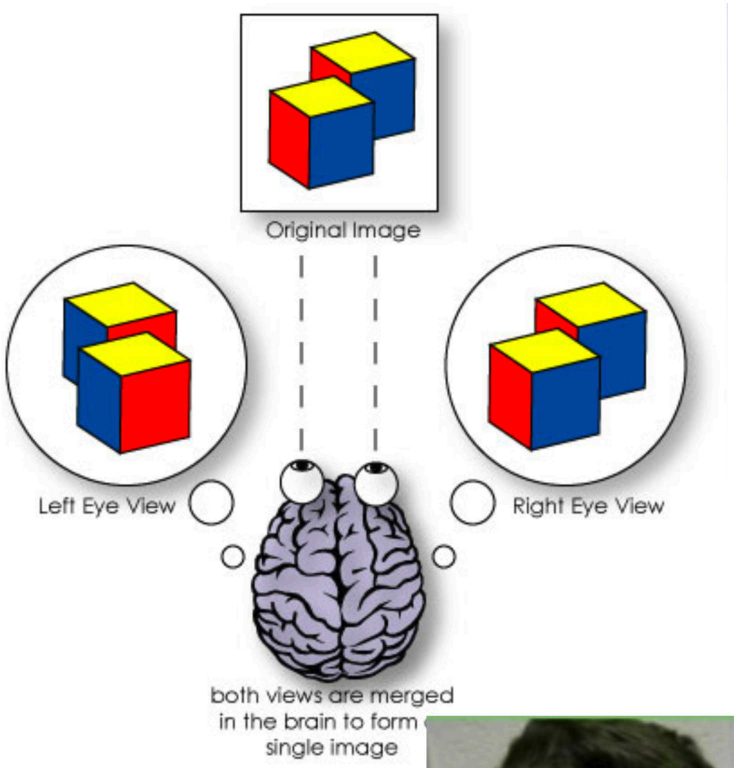


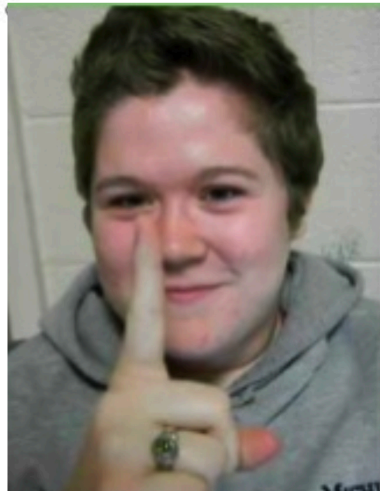


- See up to ~ 210° horizontal FOV, ~ 115° stereo overlap
- Humans can see ~135° vertical (60° above, 75° below)


Monoscopic
left eye
Stereoscopic
vision
Monoscopic
right eye
Left eye view
Right eye view
Disparity map

How do we really see depth?





Depth Perception

The visual system uses a range of different Stereoscopic and Monocular cues for depth perception

Stereoscopic	Monocular
eye convergence angle	eye accommodation
disparity between R/L images	perspective
diplopia	atmospheric artifacts (fog)
	relative sizes
	image blur
	occlusion
	motion parallax
	shadows
	texture

Parallax can be more important for depth perception! Stereoscopy is important for size and distance evaluation

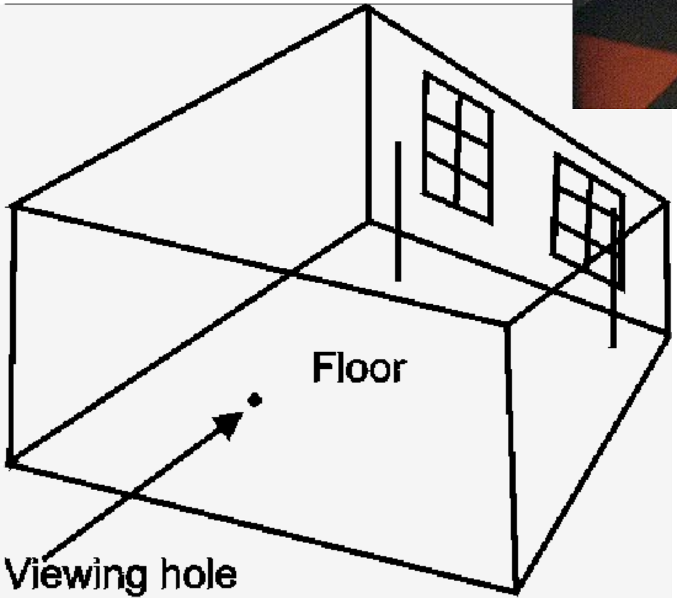
Perspective

Objects get smaller the further away they are and parallel line converge in distance.

Size of known objects

We expect certain object to be smaller than others. If an elephant and a tea cup appear the same size then we expect the elephant to be further away.

Ames Room (Optical Illusion Based on Known Object Size)

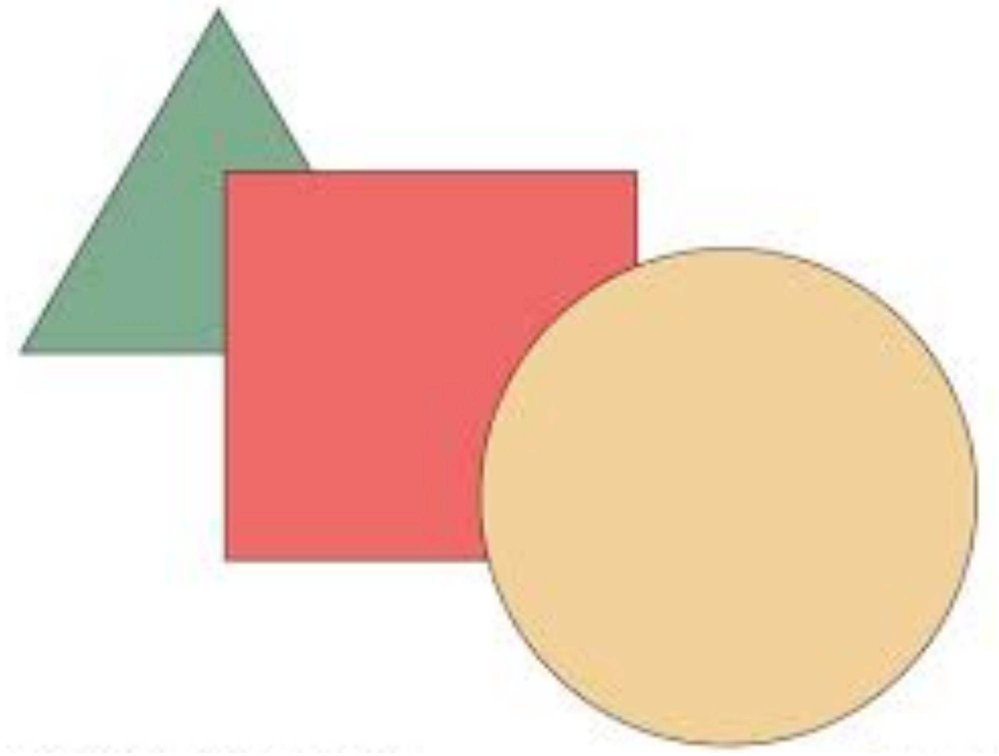


DETAIL

Close objects appear in more detail, distant objects less.

Occlusion

An object that blocks another is assumed to be in the foreground.

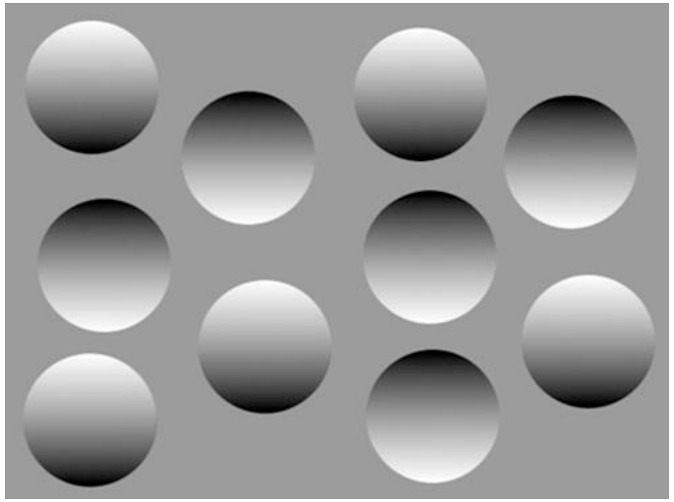


Lighting, shadows

Closer objects are brighter, distant ones dimmer.

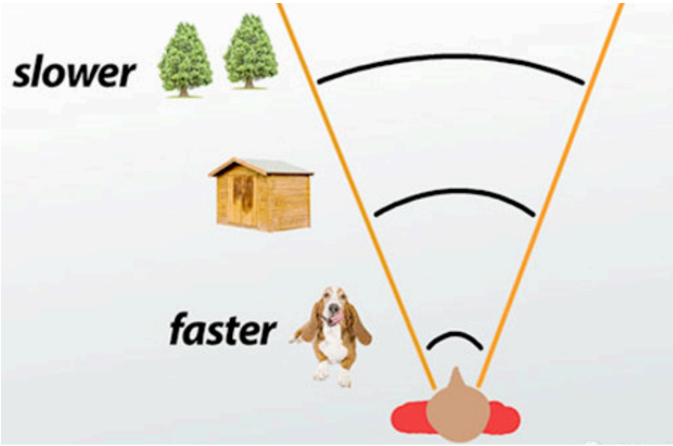
There are a number of other more subtle cues implied by lighting

- The way a curved surface reflects light suggests the rate of curvature
- Shadows are a form of occlusion



Relative motion/Motion parallax

Objects further away seem to move more slowly than objects in the foreground.



https://youtu.be/u9ItKCdDKt0?t=11787

Optical Illusion

https://youtu.be/5cJRe9iDDjs

STEREO VIEWING - DEPTH CUES 3D ONLY

There are other cues that are not present in 2D images

Binocular disparity (Stereopsis)

This is the difference in the images projected onto the back the eye (and then onto the visual cortex) because the eyes are separated horizontally by the intraocular distance (distance between the eyes).

Eyes separated by Inter pupillary distance (IPD)

- 5 – 7.5cm (avg. 6.5cm)

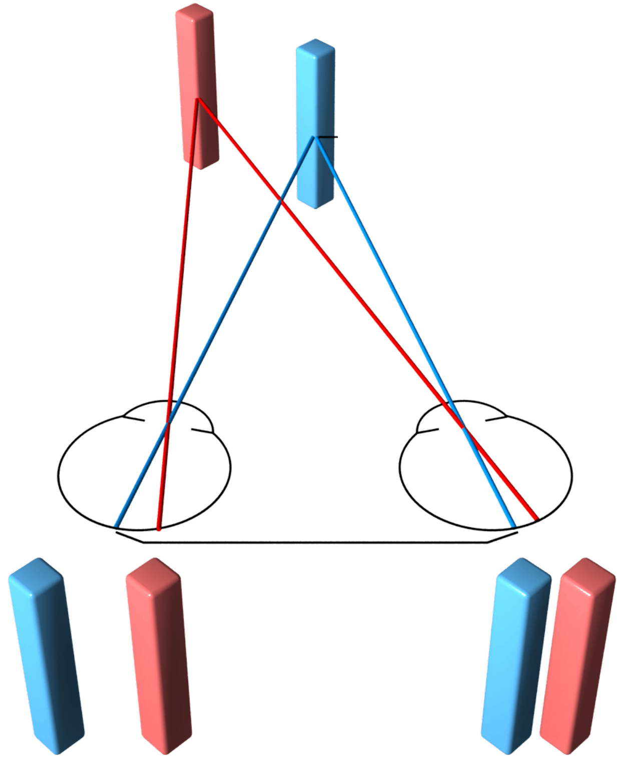
Each eye sees diff. image

- Separated by image parallax

Images fused to create 3D stereo view

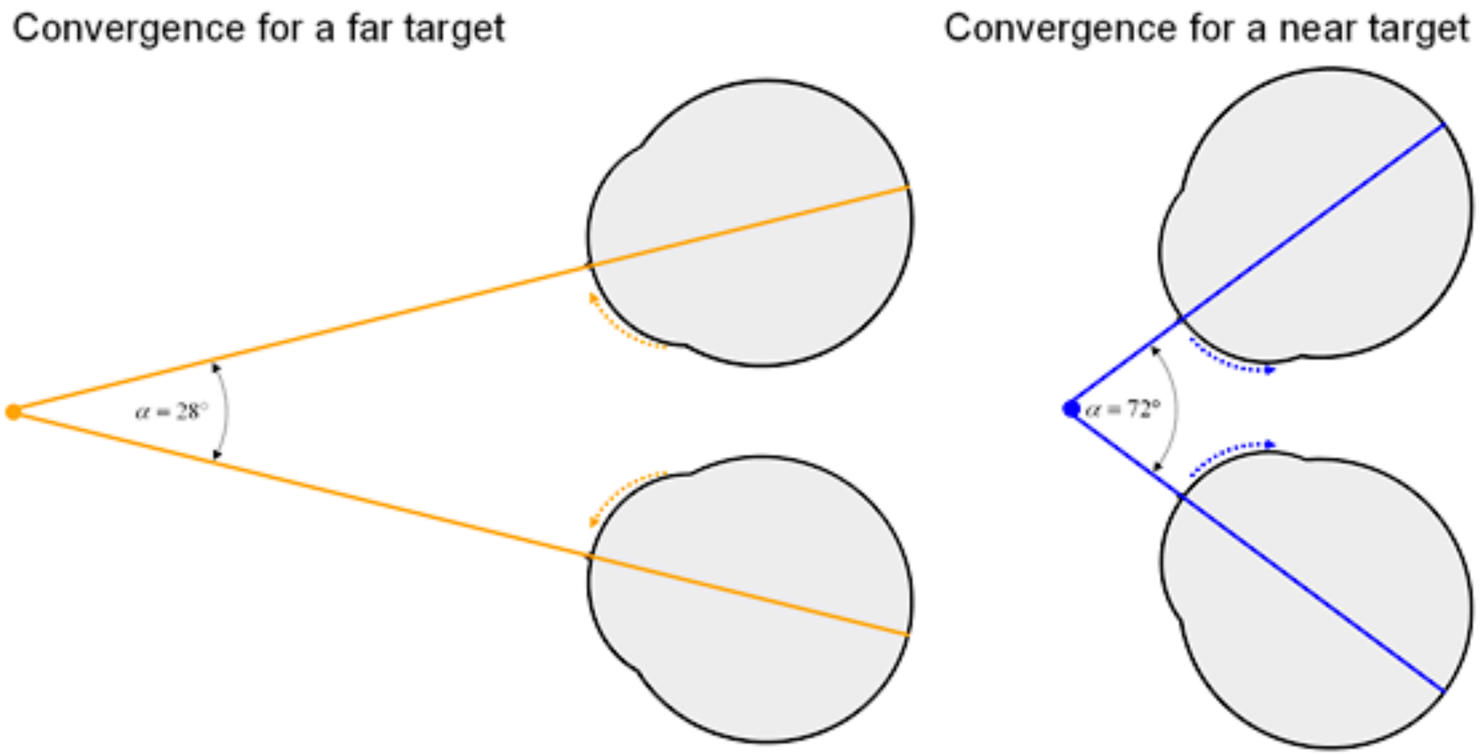
Accuracy

- 0.05mm @ ½m (0.01% error)
- 4mm @ 5m (0.08% error)



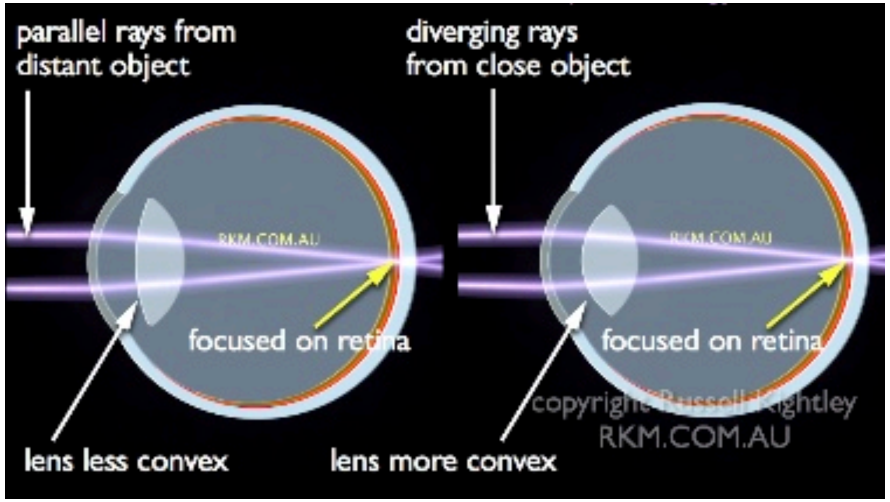
Convergence

This is the muscle tension required to rotate each eye so that it is facing the focal point.

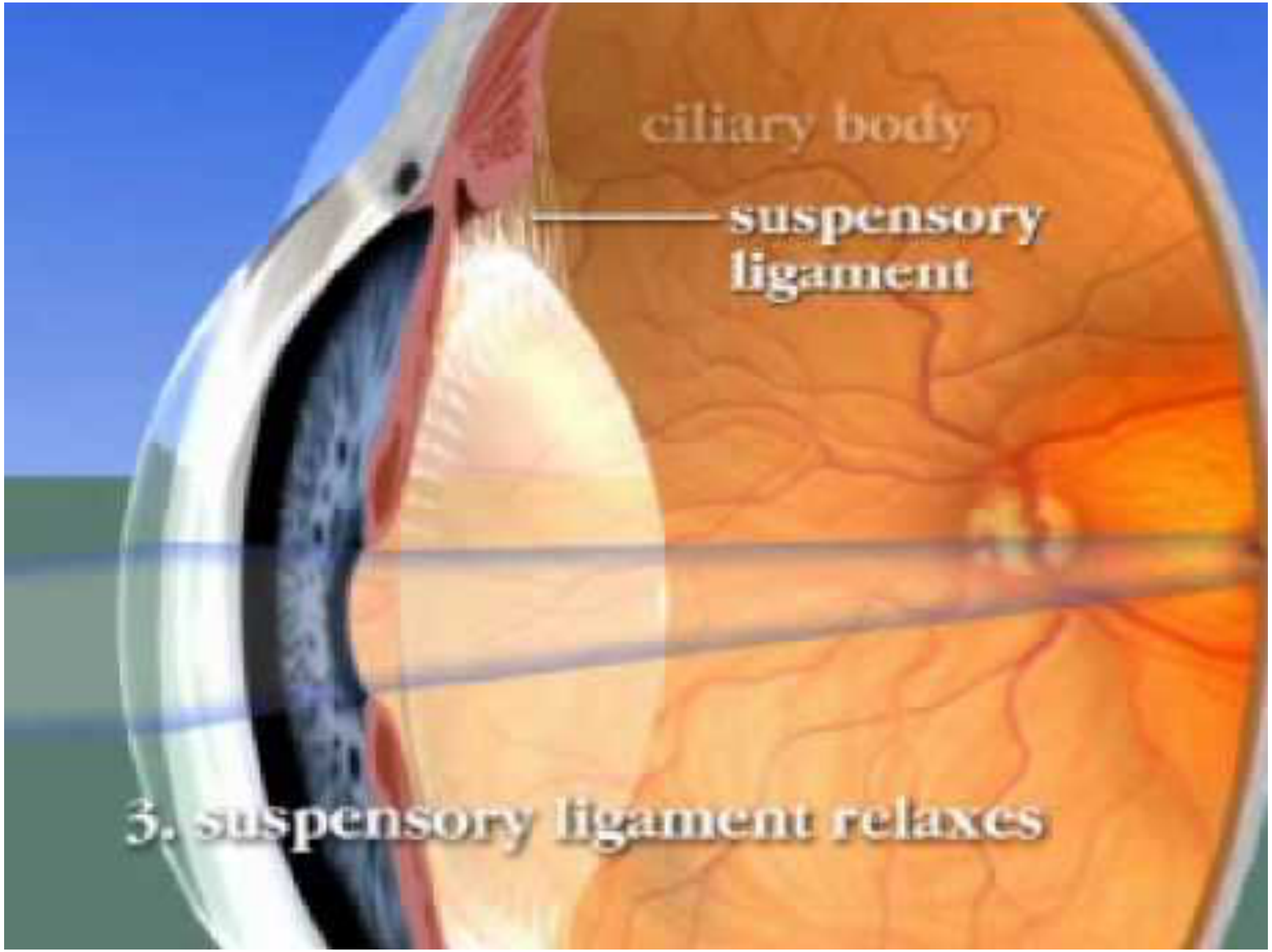


Accommodation

This is the muscle tension needed to change the focal length of the eye lens in order to focus at a particular depth.



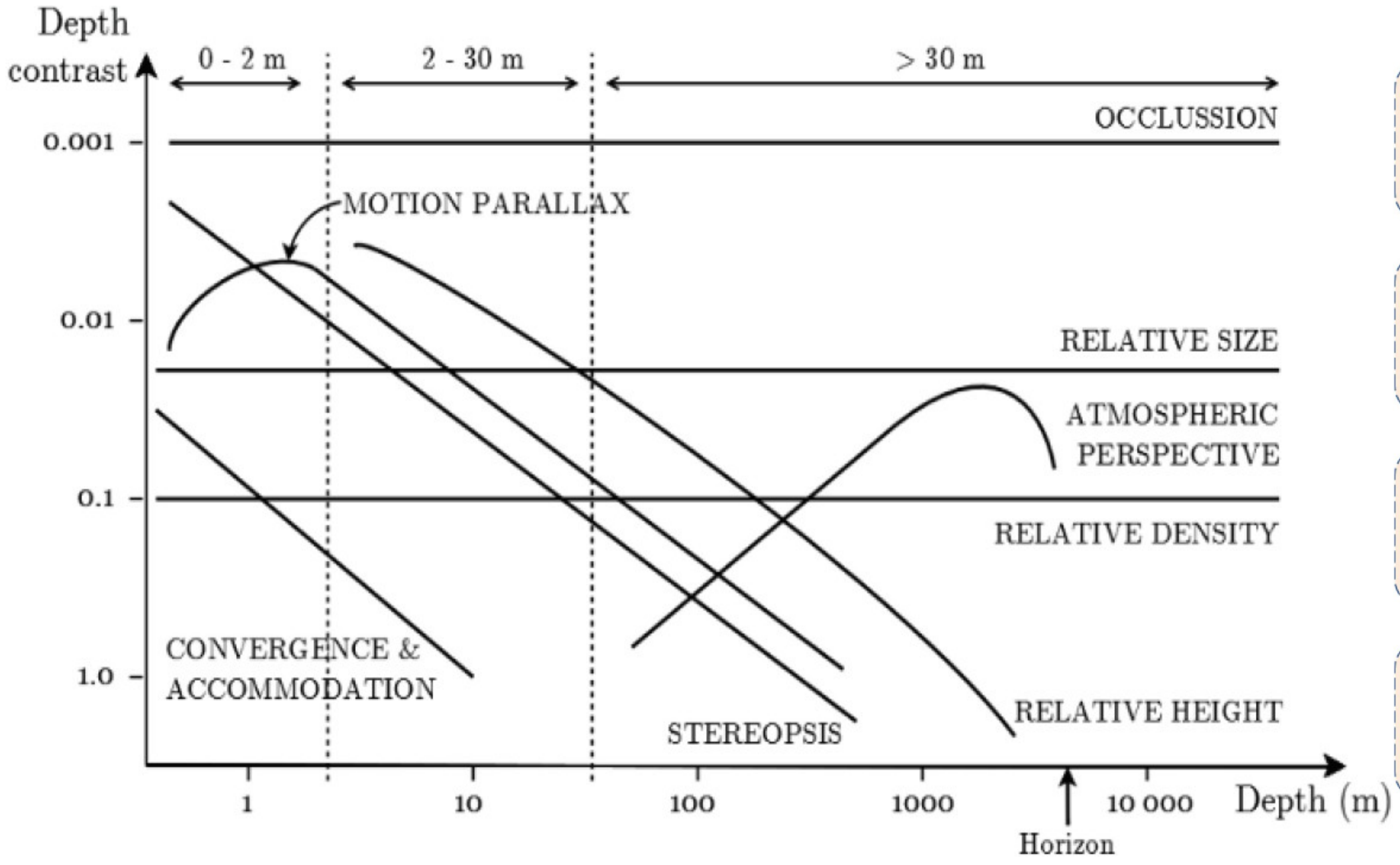
Convergence/Accommodation Demo

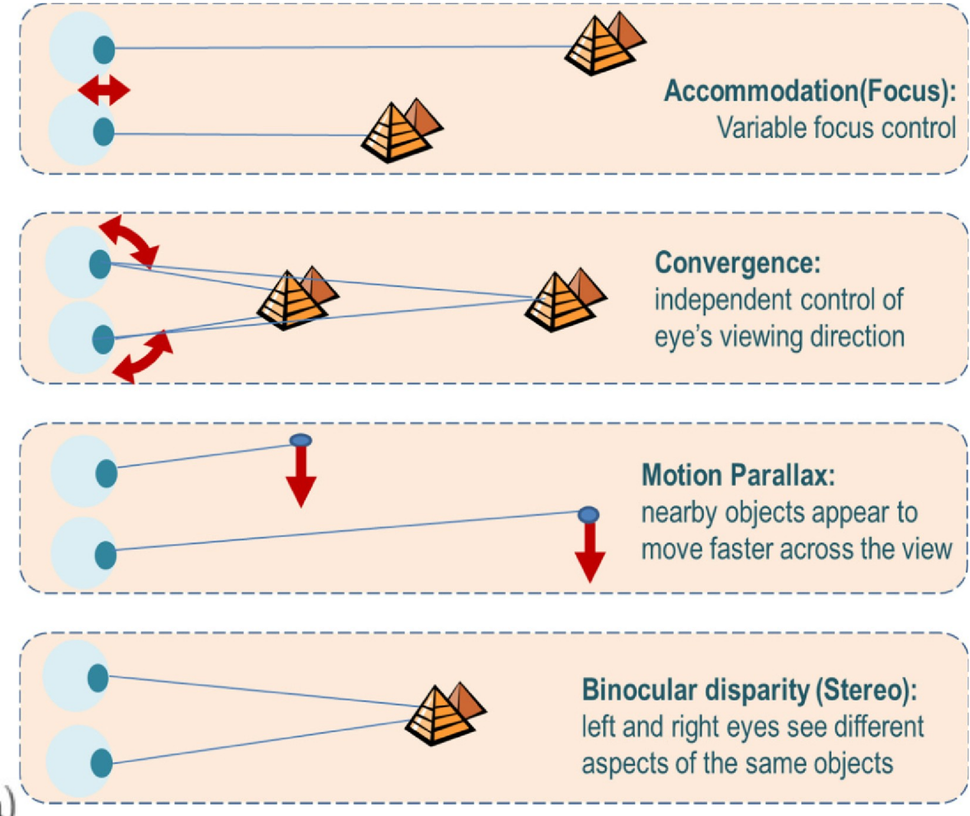


https://youtu.be/p_xLO7yxgOk

Depth Perception Distances

Cutting & Vishton, 1995





Stereo Viewing - Depth cues

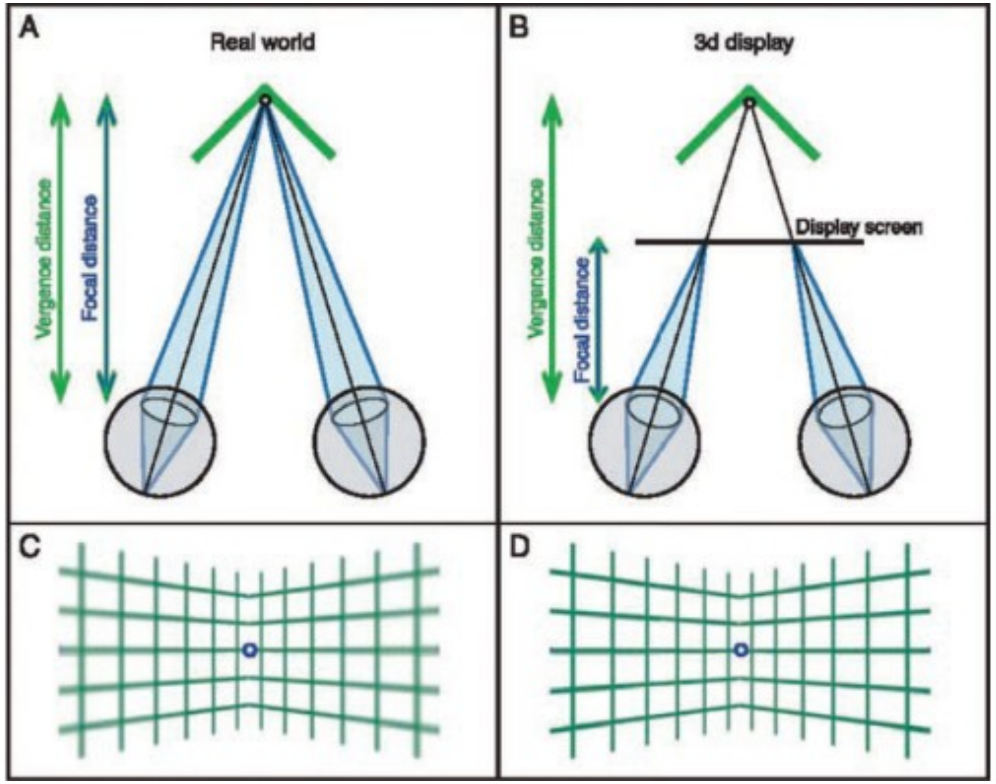
Binocular disparity is considered the dominant depth cue for most people.

Other cues, if presented incorrectly can have a negative impact on the stereo effect

- One cue can become dominant, may not be the correct one
- Depth perception can become exaggerated or reduced, can be uncomfortable to watch
- Or the stereo pairs will not be fused at all, viewer will see two separate images.

Convergence-Accommodation Conflict

Marty Banks, UC Berkeley



effects

- visual discomfort
- visual fatigue
- nausea
- diplopic vision
- eyestrain
- compromised image quality
- pathologies in developing visual system
- …

Looking at real objects, convergence and focal distance match

In VR, convergence and accommodation can miss-match

- Focusing on HMD screen, but accommodating for virtual object behind screen

Problems

Three categories for simulator sickness.

- Proposed by Jason Herald

System factors

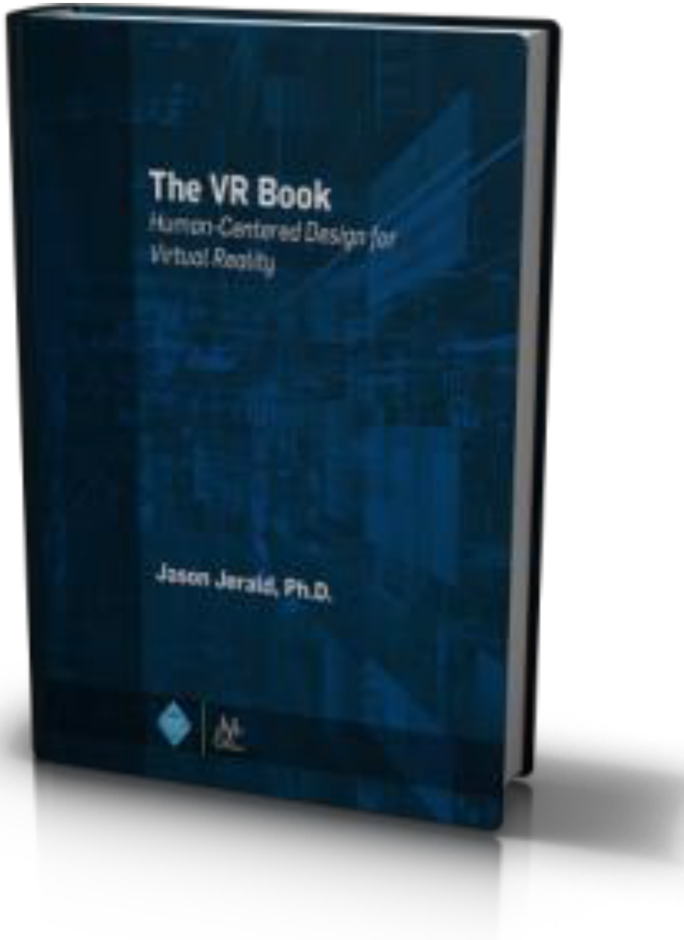
- Latency, resolution, etc

Application Design Factors

- Depth issues, overlapping, large movements

Individual User factors

- Gender (next slide)
- Visual impairments, biological differences (e.g., eye distance), training effects, etc.



IS THERE A GENDER BIAS IN VR?

Apparently, females use different depth queues than males—biological differences or not?

- http://qz.com/192874/is-the-oculus-rift-designed-to-be-sexist/
- Summary: women appear to use lighting and shadow more strongly than men
- Requires more research

In Stanney et al. 2014, females are as much as three times more susceptible to VR sickness than males in some cases.

Generating Computer Images

Specify the location/parameters of camera (this slide deck)

- derivation of projections and transformations

Specify the geometry (next slide deck)

Specify the lights (2 slide decks from now)

We're going to talk about this in the context of generic OpenGL. Other platforms (e.g. DirectX) treat things similarly.

OpenGL

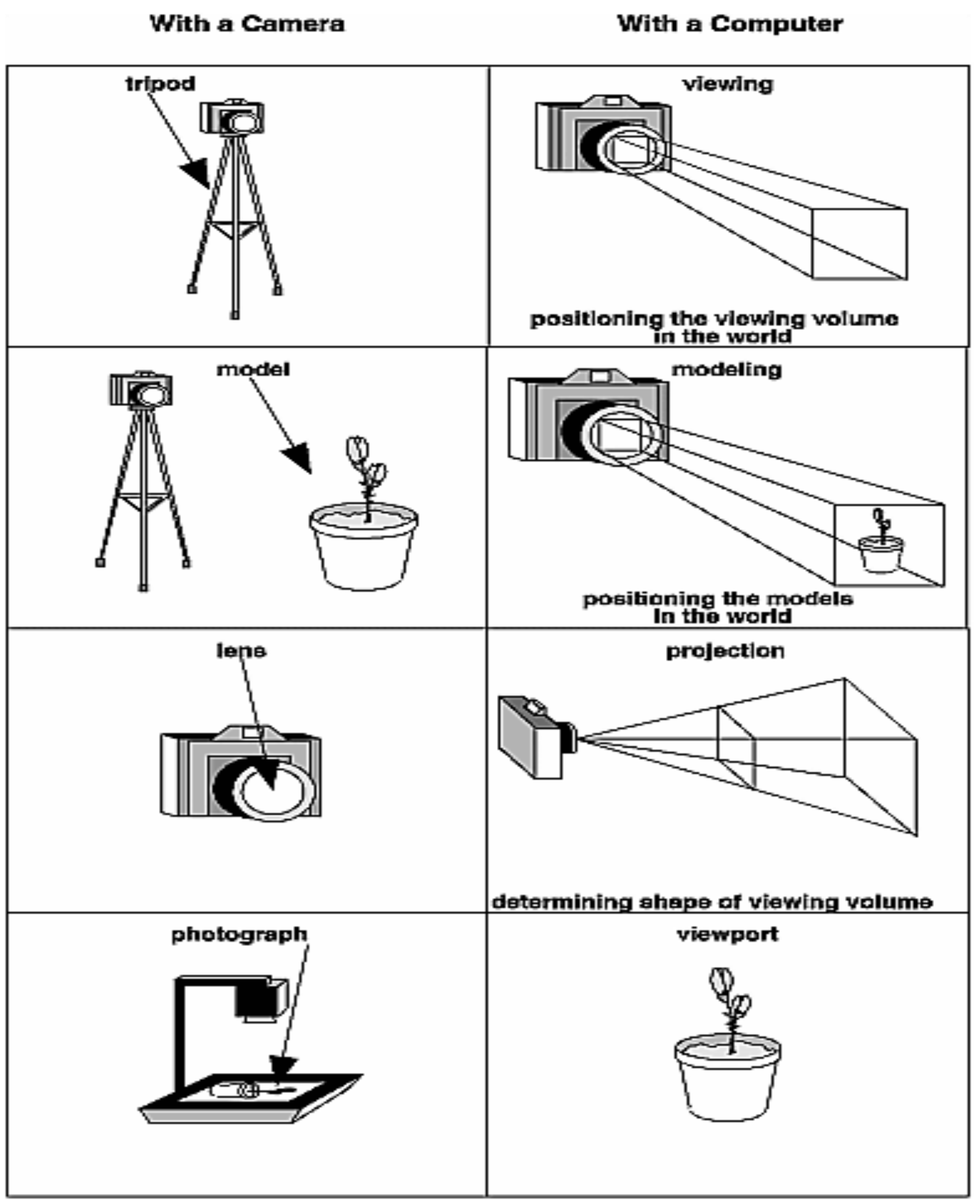
OpenGL is a graphics API, nothing more than a set of functions you call from your program

- Most functions map directly to GPU hardware functionality

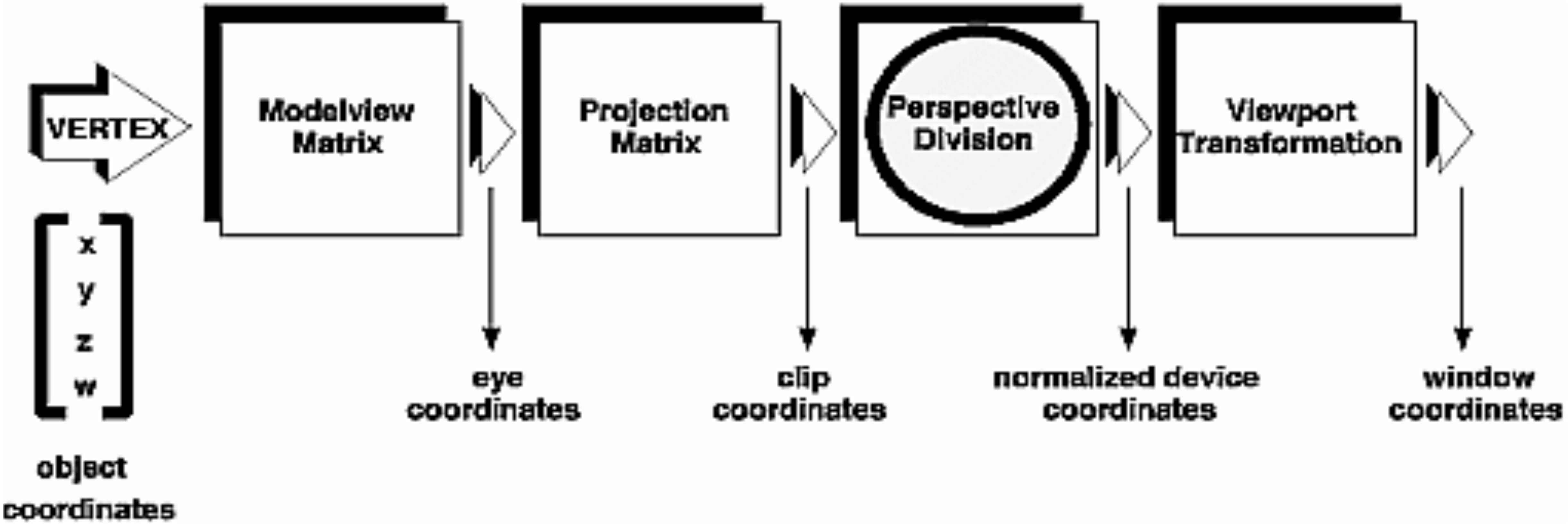
Hides the details of the display adapter, operating system, etc.

Has grown to become a powerful/flexible general purpose compute platform

The Camera Analogy



BASIC VERTEX PIPELINE



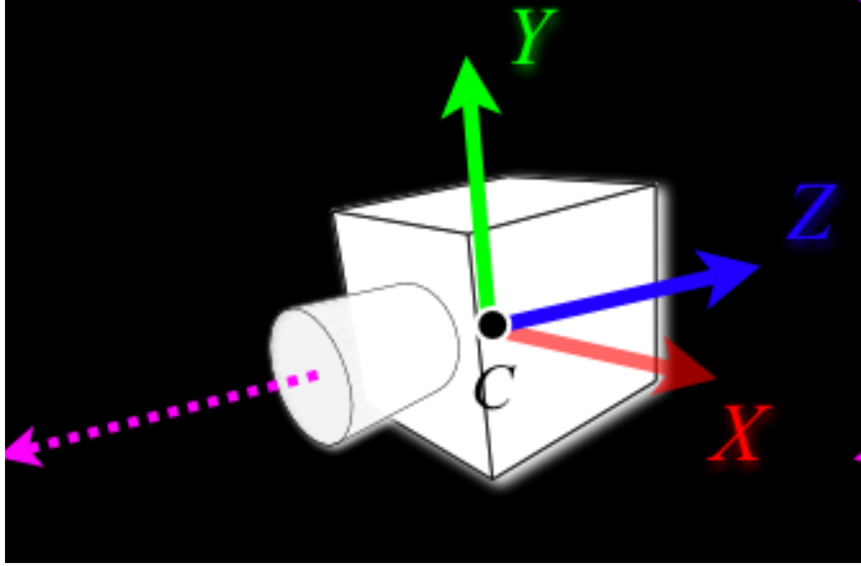
Entire pipeline handled by a series of 4x4 matrix multiplications*

OpenGL: Camera

Three things to specify:

- Projection properties of the camera (Projection)
  - Depth of field?
  - Field of view in the x and y directions?
- Physical location of camera in the scene (ModelView)
  - Where is the camera?
  - Which direction is it pointing?
  - What is the orientation of the camera?
- Physical location of objects in the scene (ModelView)
  - Where are objects?
  - What orientation do they have?

THE OPENGL CAMERA

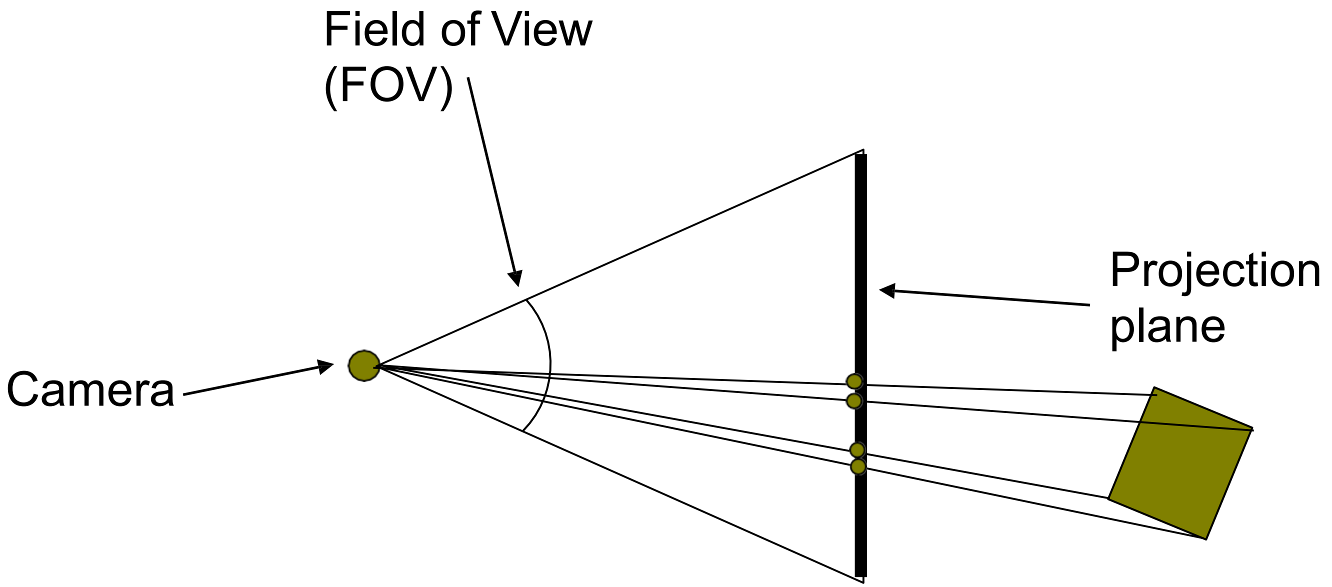


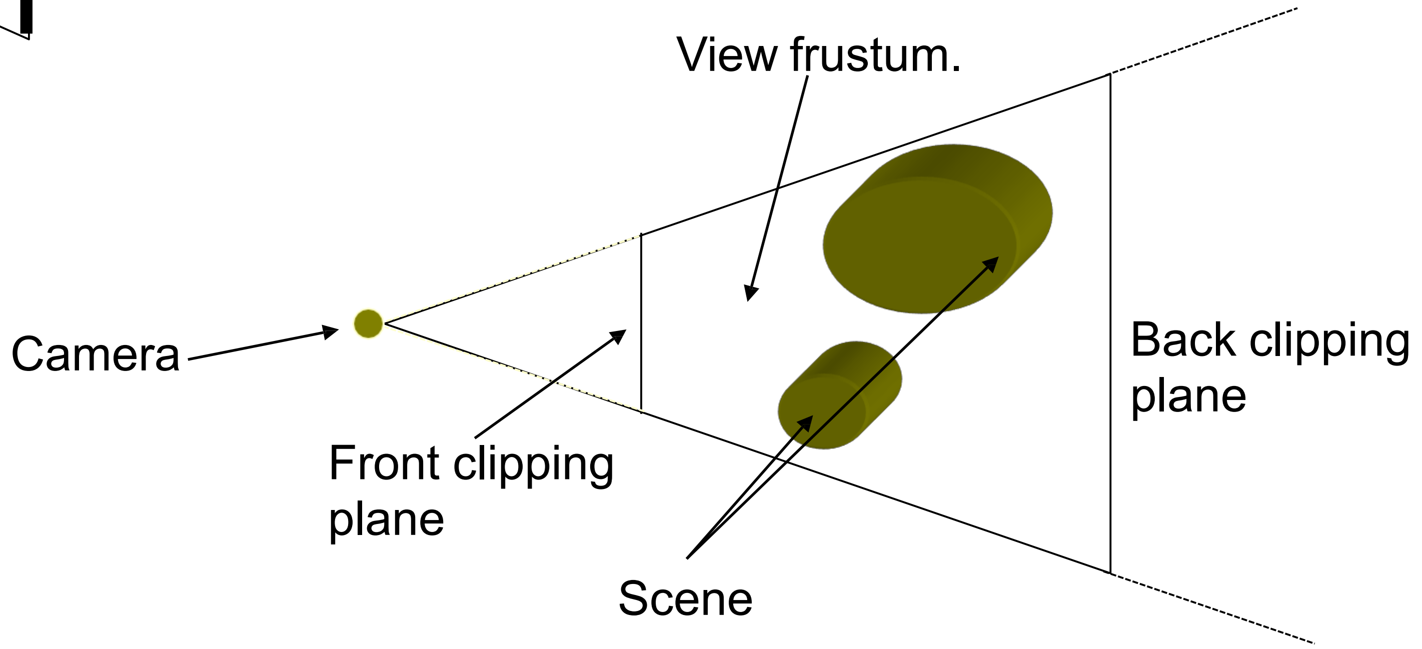
Initially the object and camera frames are the same:

- Default model-view matrix is identity
- Default projection matrix is identity

The camera is located at origin and points in the negative z direction

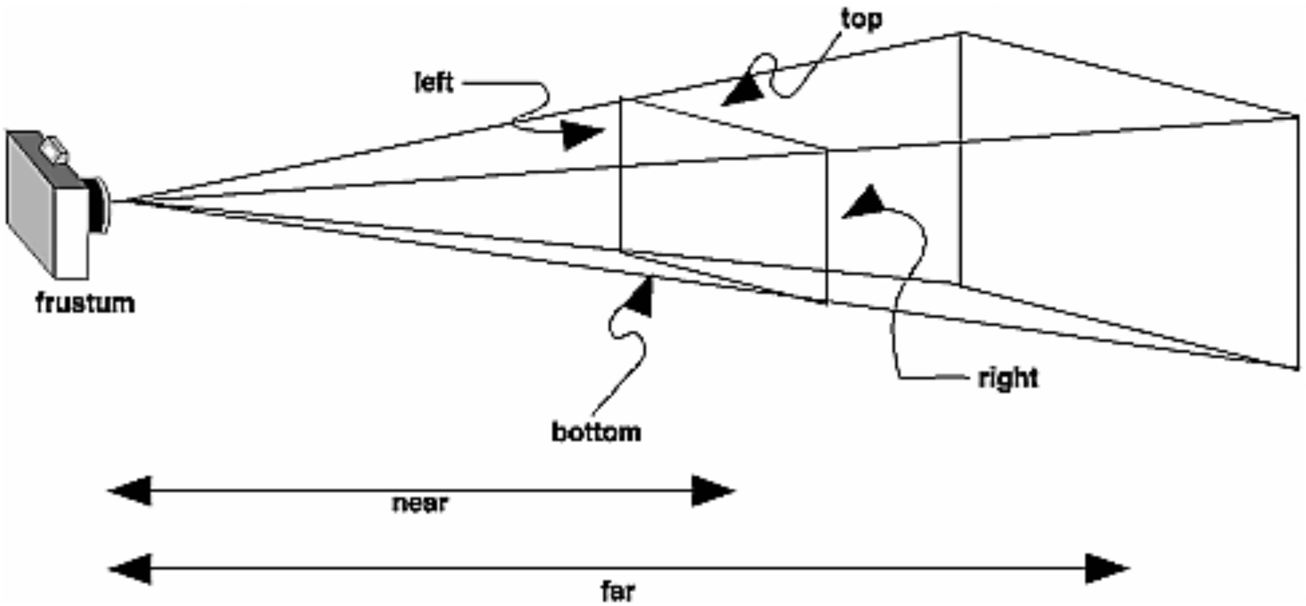
STANDARD 3D GRAPHICS PROJECTION





Perspective Projection

glFrustum(*l, r, b, t, n, f***)**



$$R = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & \frac{-(f+n)}{f-n} & \frac{-2fn}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \text{ and } R^{-1} = \begin{bmatrix} \frac{r-l}{2n} & 0 & 0 & \frac{r+l}{2n} \\ 0 & \frac{t-b}{2n} & 0 & \frac{t+b}{2n} \\ 0 & 0 & 0 & -1 \\ 0 & 0 & \frac{-(f-n)}{2fn} & \frac{f+n}{2fn} \end{bmatrix}$$

Camera Positioning

Handled via the ModelView matrix

View mostly controlled with

- Look at function

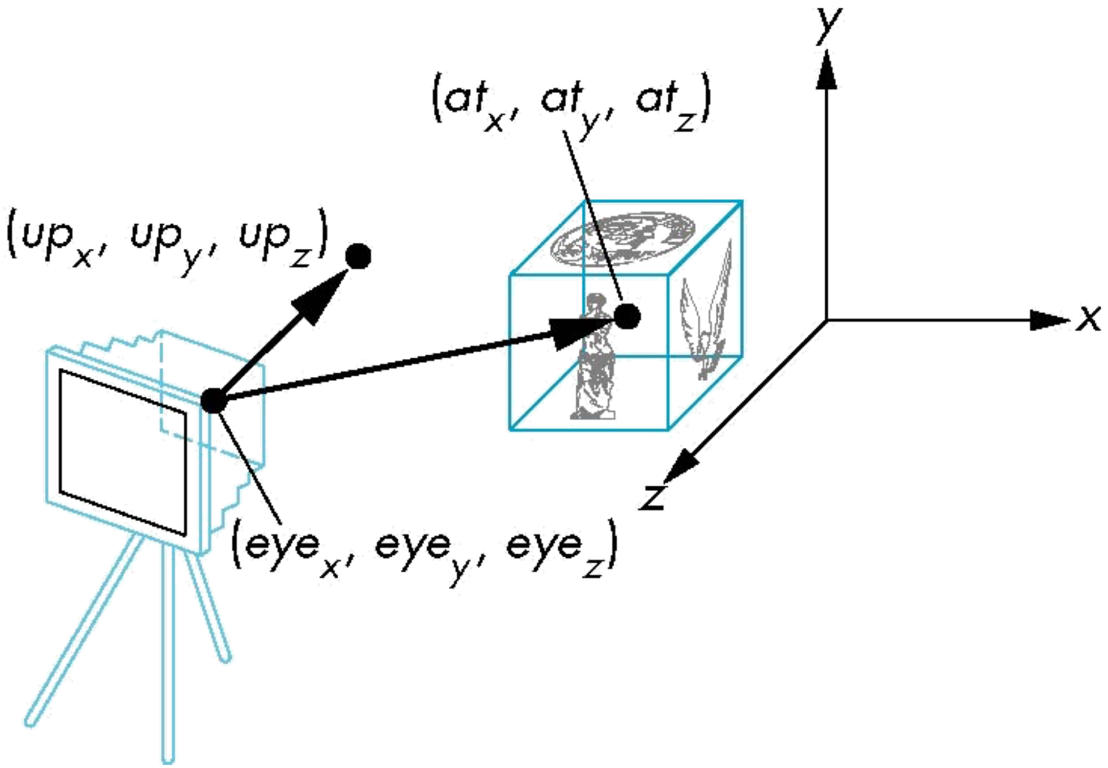
Model mostly controlled using (camera too):

- Translation
- Rotation
- Scaling

LOOK AT FUNCTIONS

Establish a coordinate system by specifying an eye position (eyex,eyey,eyez); a target location (atx,aty,atz); and an up vector (upx,upy,upz)

For a proper coordinate system, up vector should be orthogonal to the view vector (at-eye).



gluLookAt(eyex, eyey, eyez, atx, aty, atz, upx, upy, upz)

OpenGL uses a column-major Matrices

glTranslate*(*x*, *y*, *z*)

$$T = \begin{bmatrix} 1 & 0 & 0 & x \\ 0 & 1 & 0 & y \\ 0 & 0 & 1 & z \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } T^{-1} = \begin{bmatrix} 1 & 0 & 0 & -x \\ 0 & 1 & 0 & -y \\ 0 & 0 & 1 & -z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

glScale*(*x*, *y*, *z*)

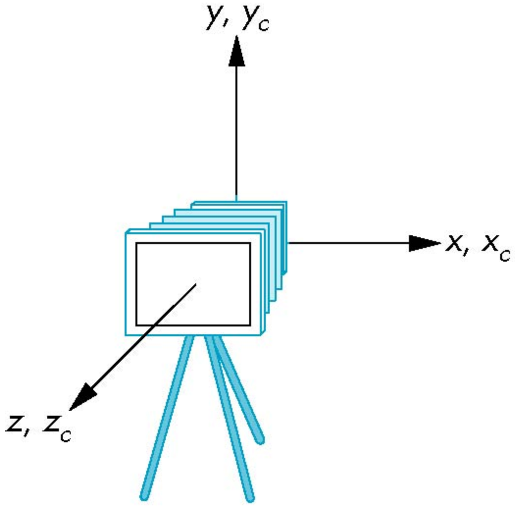
$$S = \begin{bmatrix} x & 0 & 0 & 0 \\ 0 & y & 0 & 0 \\ 0 & 0 & z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \text{ and } S^{-1} = \begin{bmatrix} \frac{1}{x} & 0 & 0 & 0 \\ 0 & \frac{1}{y} & 0 & 0 \\ 0 & 0 & \frac{1}{z} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Moving Camera Back

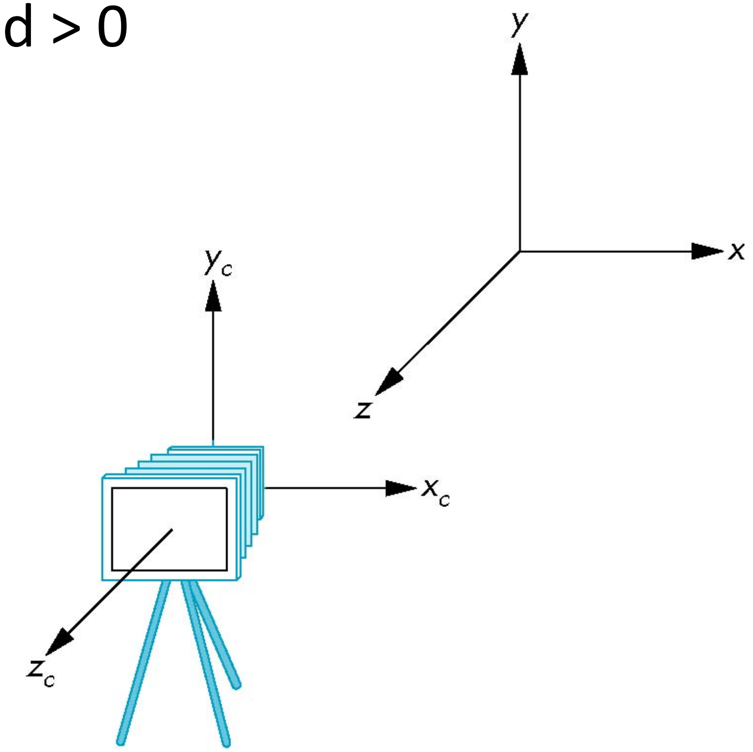
frames after translation by –d

d > 0

DEFAULT FRAMES



(a)



(b)

ROTATIONS

glRotate*$(a, 1, 0, 0)$: $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos a & -\sin a & 0 \\ 0 & \sin a & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

glRotate*$(a, 0, 1, 0)$: $\begin{bmatrix} \cos a & 0 & \sin a & 0 \\ 0 & 1 & 0 & 0 \\ -\sin a & 0 & \cos a & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

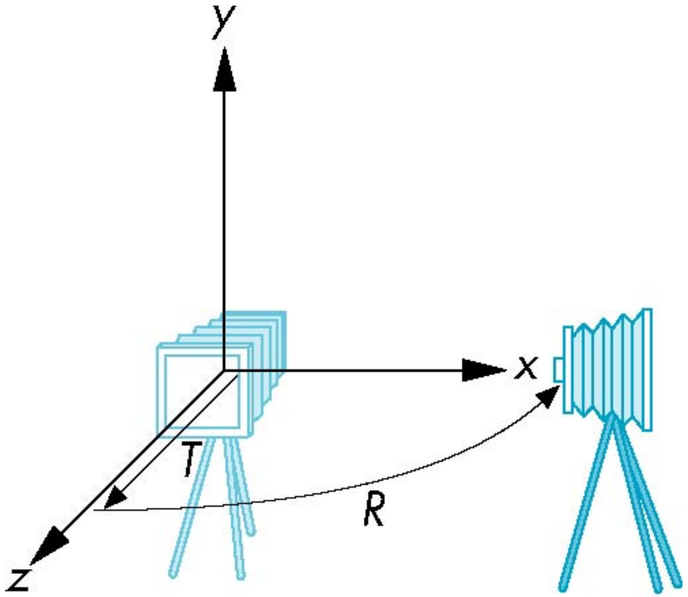
glRotate*$(a, 0, 0, 1)$: $\begin{bmatrix} \cos a & -\sin a & 0 & 0 \\ \sin a & \cos a & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Moving the Camera

We can move the camera to any desired position by a sequence of rotations and translations

Example: side view

- Rotate the camera
- Move it away from origin
- Model-view matrix C = RT



Quaternions

Extension of imaginary numbers to three dimensions

Requires one real and three imaginary components i, j, k

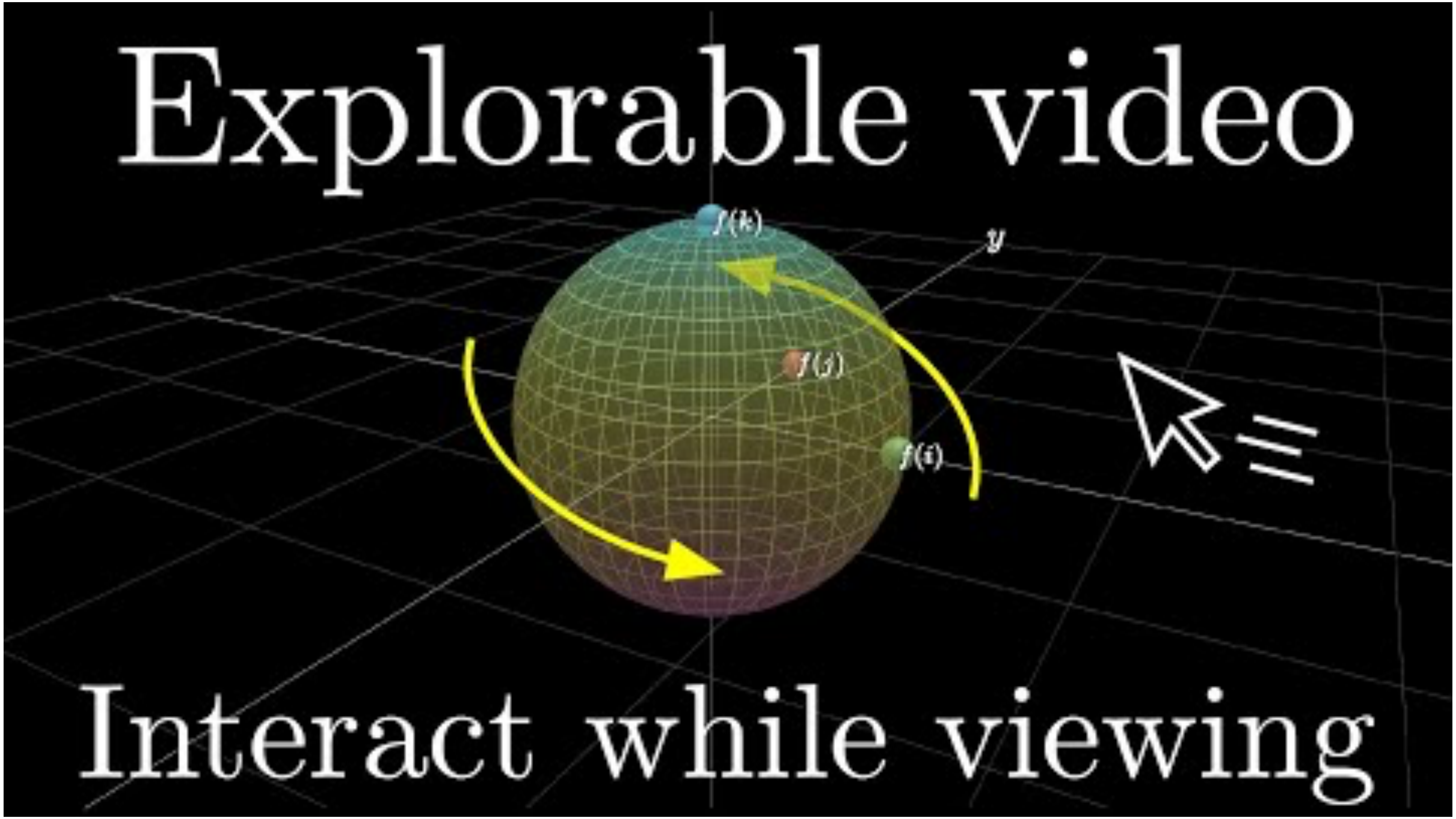
$$q = q_0 + q_1\mathbf{i} + q_2\mathbf{j} + q_3\mathbf{k}$$

Quaternions can express rotations on sphere smoothly and efficiently.

Process:

- Model-view matrix $\rightarrow$ quaternion
- Carry out operations with quaternions
- Quaternion $\rightarrow$ Model-view matrix

Why Quaternoins



https://youtu.be/zjMuIxRvygQ

PARALLAX

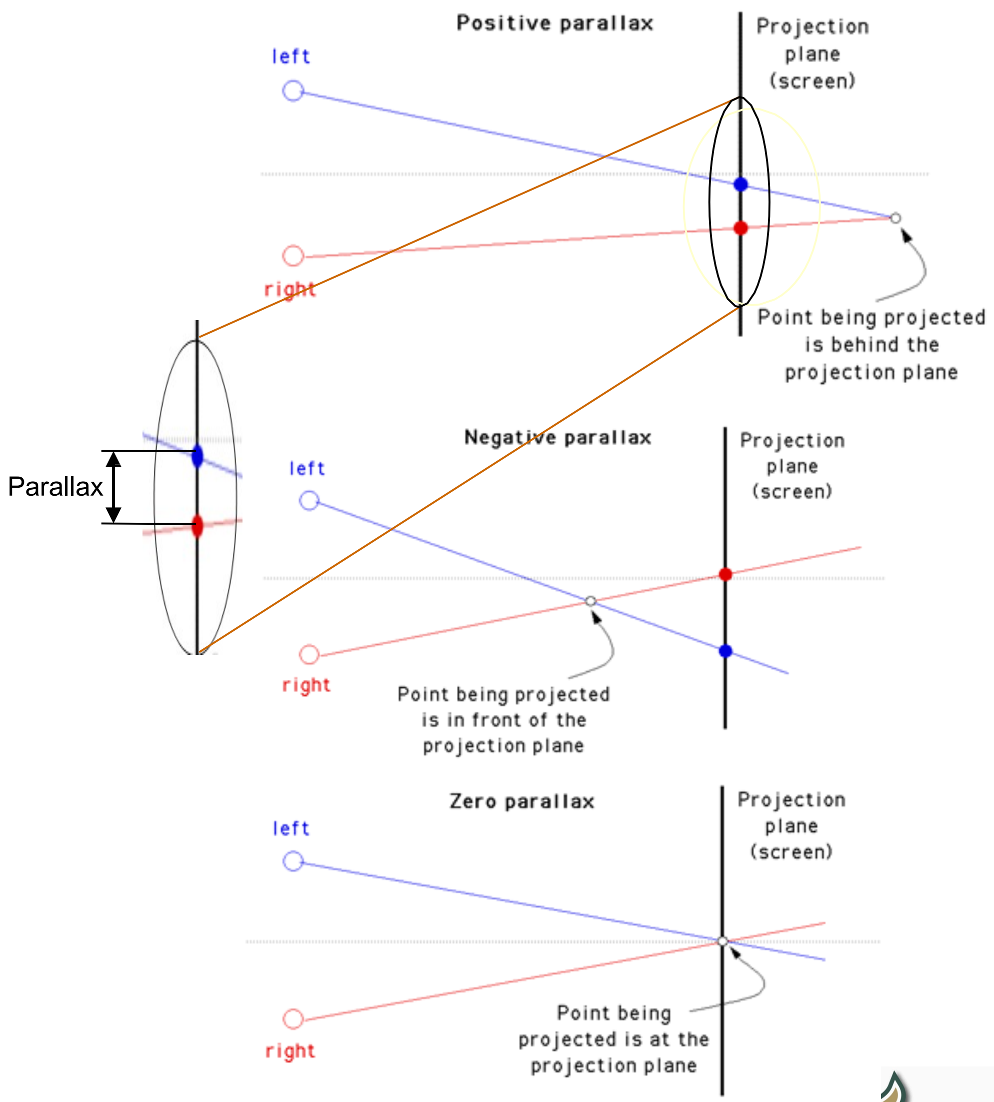
Parallax, the distance between right and left eye projections.

Negative Parallax—Objects infront of projection plane

Positive Parallax—Objects behind projection plane

As objects moves closer to the viewer, the horizontal Parallax increases to infinity.

The maximum Parallax is reached when objects are at infinity, Parallax == interocular distance



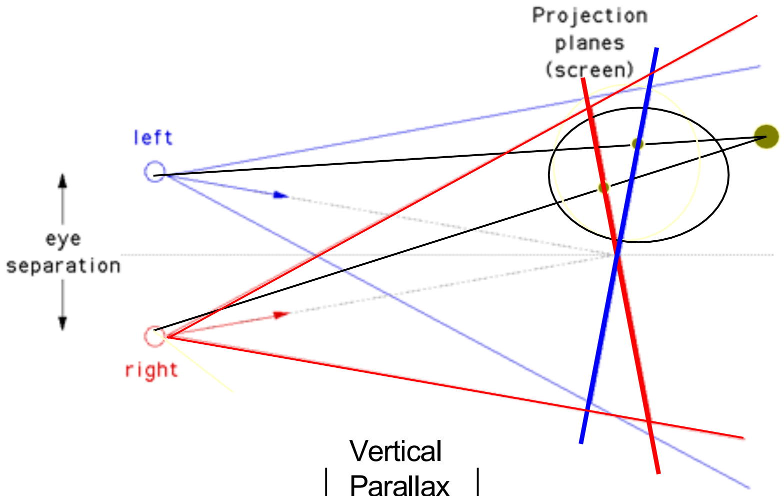
Rendering Stereo - Toe-in (Incorrect)

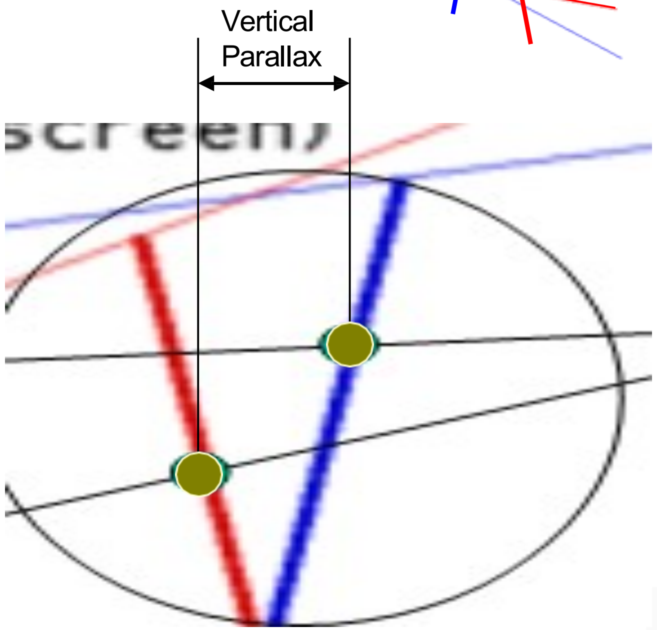
The camera has a fixed and symmetric aperture (frustum)

Each camera is pointed at a single focal point.

Images will still appear stereoscopic but introduces a vertical parallax that will cause increased discomfort levels.

The introduced vertical parallax increases out from the center of the projection plane and is more important as the camera aperture increases (FOV).



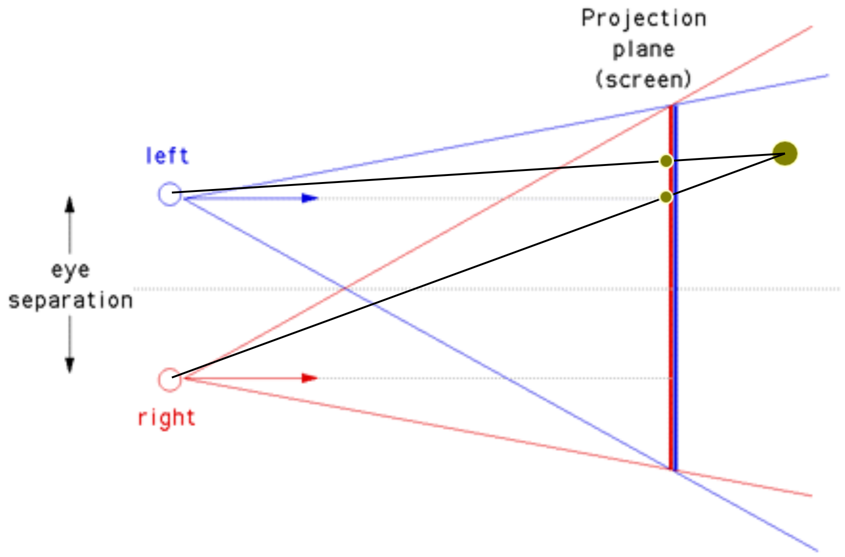


Rendering Stereo - Off-axis (Correct)

This is the correct way to create stereo pairs.

It introduces no vertical parallax and is therefore creates the less stressful stereo pairs.

Note that it requires a non symmetric camera frustum, which is supported by most rendering packages, including OpenGL



Projections of the point are both on a single vertical line == No vertical parallax inVectorize Linear Algebra WebGL

inVectorize

Visual Linear Algebra

Learn the fundamental concepts underlying linear algebra with visual tools in this visual introduction.

Topics Covered:Vector Addition & Scaling , Linear Combination , Span , Basis Vectors , Matrices

View All Topics | Start Learning

Learn WebGL/OpenGL

Does computer graphics seem like black magic? Learn about the linear algebra underlying select topics in computer graphics.

Topics Covered:RGB Colors as Vectors , Why we use 4x4 Matrices , Animations , Projection Matrix

View All Topics | Start Learning

HTTP://WWW.INVECTORIZE.COM