CS 498SL: Virtual Reality

Prof: Steve LaValle, Office: 3318 SC - 2-3pm Tu/Th
TAs: Apollo Ellis - computer graphics
Aldi Sipolins - psychology, VR

About myself:

- Ugrad & PhD at UIUC ECE
- Prof since 2001 in CS; taught 373 a lot
- Robotics researcher: motion plan, sensors, robots, 3D geometry, simulation, physical world

- Oculus VR - 2012-2015
  - started by Palmer Luckey
  - I joined a few days after IC, kickstarter
  - Head tracking, sensor calibration & characterization, perceptual psychology, health & safety
  - Chief Scientist Mar. 2013 - FB acquisition

☆ Slides: Palmer, Finland, CiciaBabcia, Grandma Oculus

Course Administration

http://courses.engr.illinois.edu/cs498sl/

- Two midterm exams, no final
- Machine Problems - 4 or 5] work in pairs
- Final project - Topics coming → faculty pairing possible

VR Lab: 4240 Siebel Center
12 PCs w TitanBlack graphics cards 6GB, 2 monitors, 27"
Oculus Rift DK2s

VR Lab Windows 8
Visual Studio
Unity Free game engine
Programming: - Unity Scripts in C# or Java
- C++ with raw Oculus SDK

Not in CS? Meet people outside!

Text books: - Mather, Foundations of Sensation & Perception, 2nd ed. 2009
- Shirley et al., Fundamentals of Computer Graphics, 2009

→ Why these? - No modern book on VR
- Want technology invariant coverage
- Pieces exist in other fields

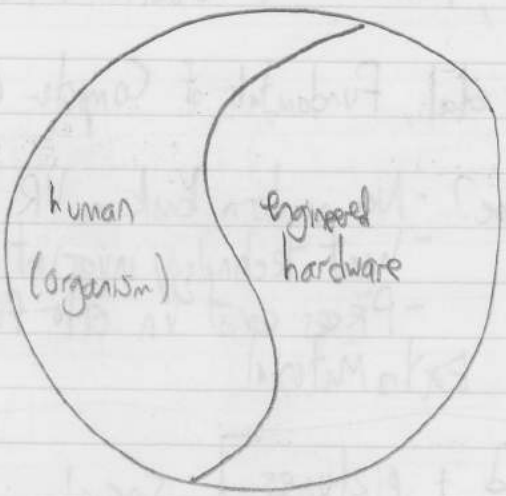
- Check Extra Material

Lectures: Blackboard + pictures
Class forum: Piazza
} Social experiences

Goals:
- Learn how to build a good VR experience
- Understand how VR works (engineering + psychology)
- Learn how to criticize VR
- Learn fundamentals to shape future of VR

good = comfortable + adequate for task
↑ no nausea

Task? Game, write code, maintain relationship, relax, watch film, virtual travel


human
(organism)
engineered
hardware

Definition of VR: (lots of bad defs out there)

> Inducing targeted behavior in an <u>organism</u> by using artificial sensory stimulation, while the organism has little or no awareness of the interference.

Note: Fooling the organism. Not consciously aware.
Note: Does not depend on <u>particular</u> technology, but "artificial" → technology of some kind

Common terms: immersion
presence

* gerbil, treadmill, cockroach, omnitread

<u>Other Concepts</u>

Agmented reality: Like Google Glass

Telepresence: Imagine robot + omnidirectional camera + VR headset

Teleoperation: Remote control

For me, the world inside the headset does not need to be virtual.

Think also about:
- reading a book
- talking on phone
- playing a videogame
- watching a movie
- video conferencing
- listening to music
- Second Life

} sort better

Announcements:

1) Copy of Matthew — scan online
2) Try Unity tutorials
3) Exam dates on Piazza
4) Project list coming in day or two

Some historical perspective:

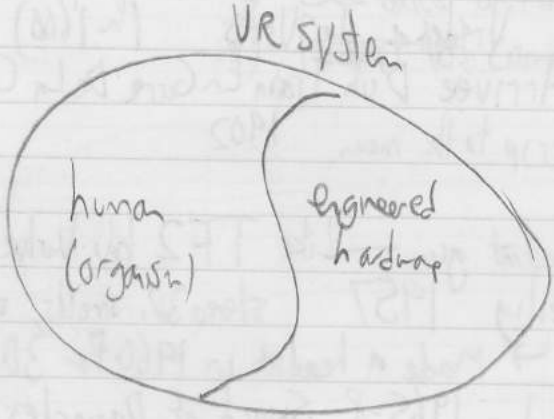
- Oldest cave painting El Castillo hands 40,800 yrs old
- Staring at rectangles: Medieval art, bad perspective (1470)
- Perspective fixed: Vredeman de Vries (~1600)
- Old movie: L'Arrivée D'un Train En Gare De La Ciotat, 1895, Lumière
- old movie: A trip to the moon 1902
- Gravity clip
- Albert Pratt helmet gun — Like TF2 by Valve for Oculus 1916
- Sensa-rama, Heilig 1957 stereo 3D, smells, seat vibrations, stereo sound
  ↳ Made a headset in 1960 for 3D movies - Telesphere
- Ivan Sutherland - 1965-8 Sword of Damacles - Video
- CAVE Projection system U. Illinois-Chicago 1991
- VFX1 by Forte, mid 90s
- Virtuality gaming systems
- Lawn mower Man 1992
- Nintendo Virtual Boy, 1995 - red lines & headaches, death of VR

hype!

Beware of hype!

Assignment: - Read over Unity Interface Overview & Manual "Extra material"

- Read Mather, Ch 1


VR system
human
(organism)
Engineered
hardware

Forgotten From last time:

Books: Mather, Foundations of Sensation and Perception, 2nd Ed., 2009
Shirley et al, Fundamentals of Computer Graphics, 2nd Ed, 2009
Assignment: Read Mather, Ch1 by next Thu class.; read/watch Unity tutorials
- How to make good VR experiences
  - ↳ comfortable & adequate for task

* Show slides and videos, to complete the history

BIRDs-EYE VIEW

Pick a sensor on your body: eye, ear, finger, tongue, nose,
↳ sense? def?
↓
stereo smell with further nostrils??

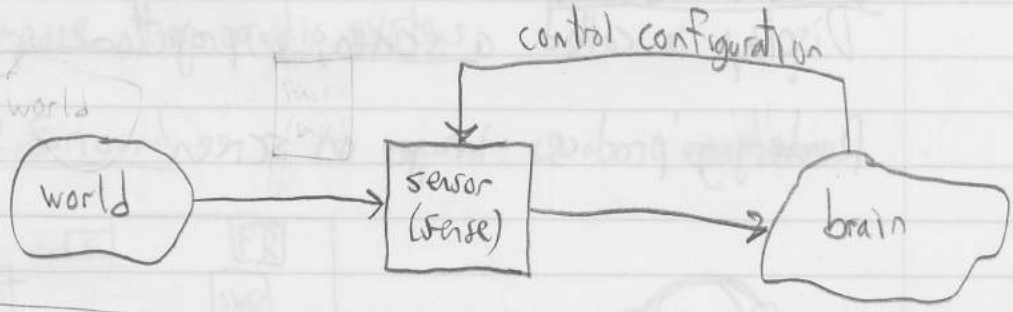
Each sensor moves through space or changes in some way
- Controlled by brain (nervous system?)
- Degrees Of Freedom (DOFs)
- Each sensor has a configuration space: set of all configurations

Ex Left ear — Position - 3DOFs, Orientation - 3DOFs → 6DOFs total

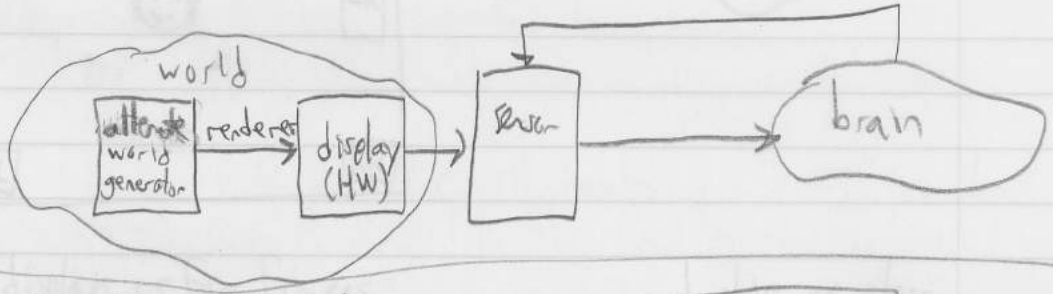
Right eye — More complicated: - 6DOF
- Rotates with head fixed
- Refocuses (accommodation)

BIRDS EYE VIEW: HARDWARE

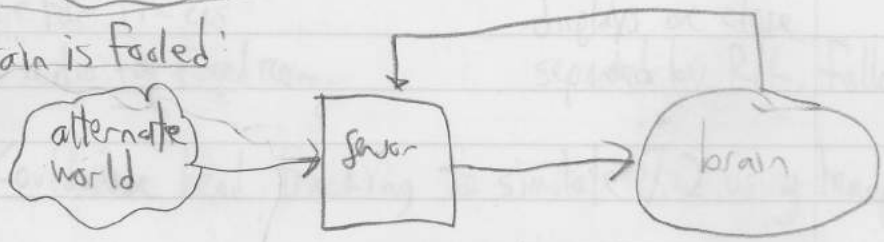
An abstract view:



Alter the world:



The brain is fooled:



Displays and rendering

Audio and hearing

Display is called a speaker 

Rendering produces sounds for the ear(s)

Video and seeing

Display is called a screen, or projector

Rendering produces images on screen

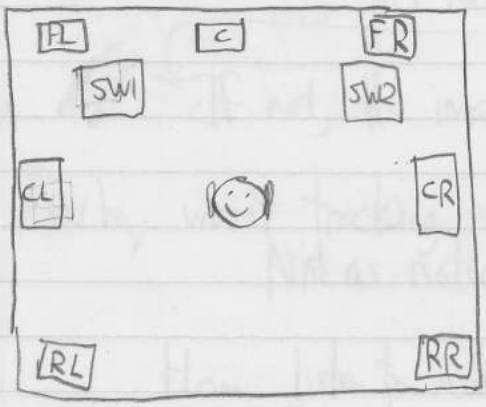
No tracking needed if:

1) rotation only

2) monoscopic

Compare two audio systems

7.2 Surround



- displays are far from eyes
- localized audio for fixed room.

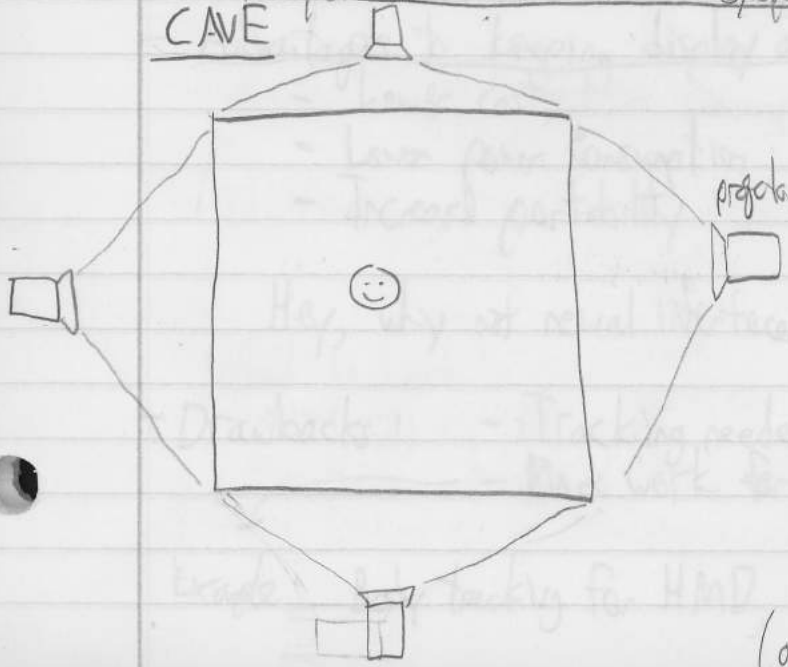
earphones

displays are close
separated by R/L, follows head rotation

Note: Could use head tracking to simulate 7.2 using headphones

Compare two VR visual systems

CAVE



HMD (Rift, Morpheus)

Like "eyephones"

Need to track head to show proper images
(or else Zeiss Cinemizer OLED glasses)

Make comparisan to audio:

- Head tracking was not needed for 7.2 surrad ad CAVE
- " IS needed for HMDs. Why?

If not, the image will follow your head, unlike the real world

Why wasn't tracking used for headphones?
Not as noticeable for audio, as for video.

Hmm- listen to music with headphones and turn your head.
Does it seem like the band is rotating around you?
They are fixed in 7.2
Same is true for CAVE and HMD

- To fool the brain, we need to undo the sensor transformations caused by head and body motion!

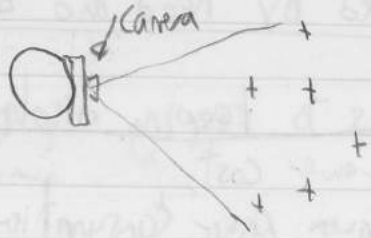
- Advantages to keeping display as close as possible to sensor!
  - Lower cost,
  - Lower power consumption
  - Increased portability

Hey, why not neural interfaces? Cut the optic nerve!

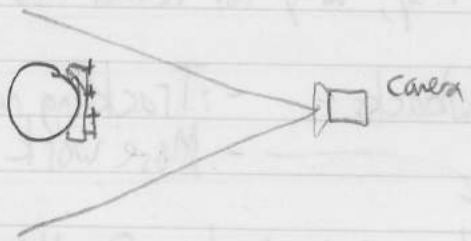
- Drawbacks! - Tracking needed
  - More work for the alternate world generator

Exaple! Body tracking for HMD, not CAVE

Inside-out



Outside-in



Regarding CAVE: Tracking not needed unless want to change viewpoint based on head motion. Includes stereo. Multiple viewers? Time multiplexing fixes

Birds-Eye View: VR Hardware

Tracking hardware components:

- Inertial Meauerut Unit (IMU):
  - Originally designed for aircraft, spacecraft
  - Mainly provide data for orientation estimation
  - Now available in MEMS thanks to smartphones (Inversense IMU-3000; St. Microelectronics)
  - From a few dollars to hundreds of thousands; calibration crucial
  - Measure: 3D angular velocity (rad/sec) - gyroscope
    3D linear acceleration (m/sec^2) - accelerometer

- Magnetometer: 3D magnetic field (Gauss)

- Camera: VGA, 60FPS, etc → 1) Inside-out: On headset
  2) Outside-in: In room, fixed

- Depth camera - like Kinect

What to track?
Head, eyes
Hands, whole body
Entire surrounding environment

Display components:

Visual: - (CRT, LCD, OLED) + lenses (including vector display (stroked plasma))
- Light field display
- Virtual retina display

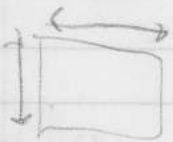
Audio: - Standard speaker / earphones
- Bone conduction

Touch: = Haptic (force feedback) devices
- keyboard, Xbox controller, mouse

Smell? Taste? Vestibular?

Visual rendering: Computer graphics (HW + SW) (GPU)

Visual display: Resolution: # of pixels in each axis
Frame rate: Displayed frames per second (Hz)
Pixel switching speed

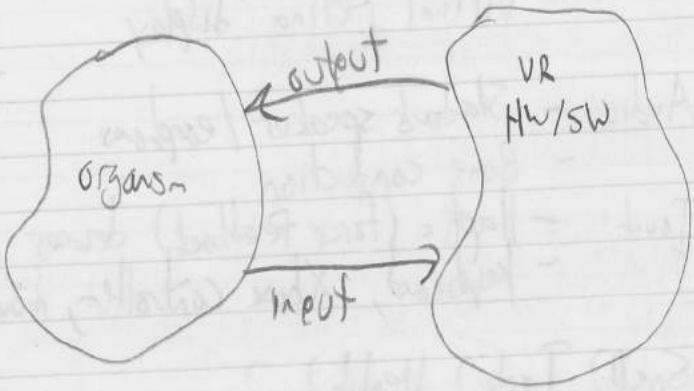


[Audio is similar with sampling rate & quantization levels]

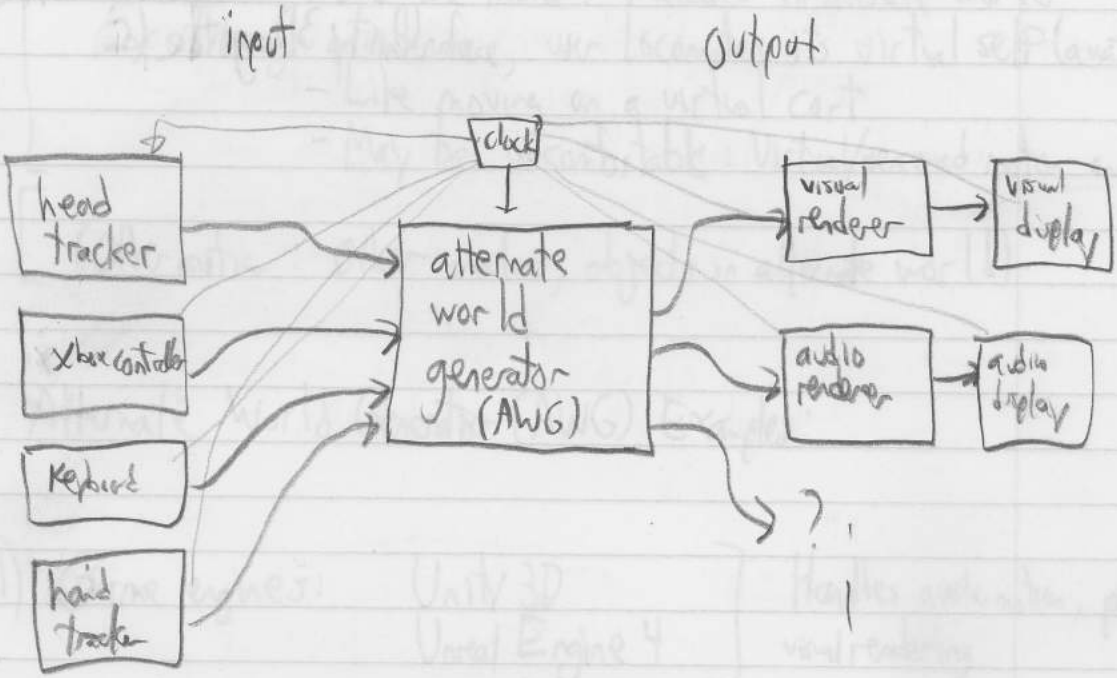
Computer: CPU, GPU

Lens — Palmer's / as fra Fry's

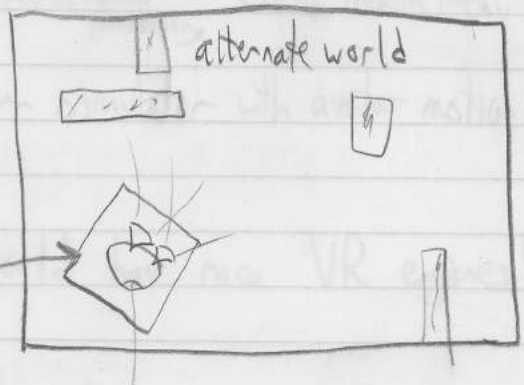
Input: Xbox controller, keyboard, mouse


output
VR
HW/SW
organism
input

Birds-Eye View Software



Some geometry: Superimposing alternate and (real) physical worlds



- Stuff: walls, plants, ...
- Collision detection
- Physical simulation
- Avatar model

mobility region in physical world

head tracker at work here!

Assignment: Read/Review Shirley Ch 2, 5, 6
Lavalle Ch 3, 4
Unity 3D guide

Two main kinds of motions:

self
1) User moves and motion is matched in alternate world
2) Through an interface, user locomotes its virtual self (avatar)
   - Like moving on a virtual cart
   - May be uncomfortable: Virtual/perceived motion only

others
(Other motion: other avatars, objects in alternate world)

Alternate World Generator (AWG) Examples:

1) Game engines: Unity 3D, Unreal Engine 4 — Handles avatar motion, physics, visual rendering

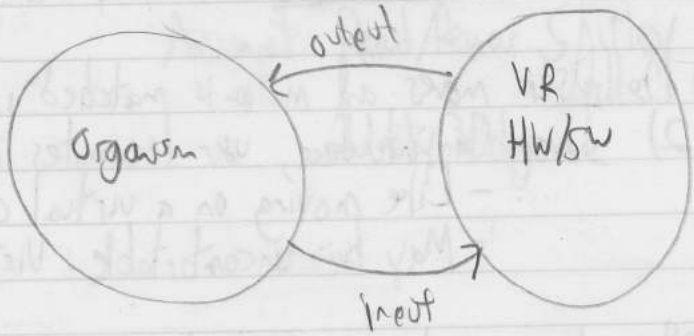
   Powerful, but not optimized for VR.

2) Google Street View server + interface for navigation

3) robot + camera + video processing → drone, mobile robot, humanoid

4) Write your own simulator with avatar motion controls.

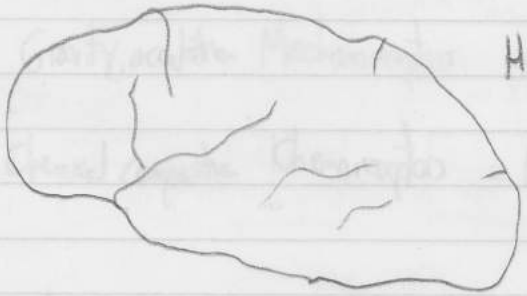
Someday we should have nice VR engines!


output
Organism
VR
HW/SW
input

Birds-Eye View: Sensation & Perception
(Perceptual psychology; vision science)

VR Questions
- How do we perceive depth? (3D)
- What causes nausea, fatigue?
- How do we perceive an audio source?
- How much resolution is enough in a display?
- What is presence?

Our perception process appears to be simple:
Immediate, effortless, direct

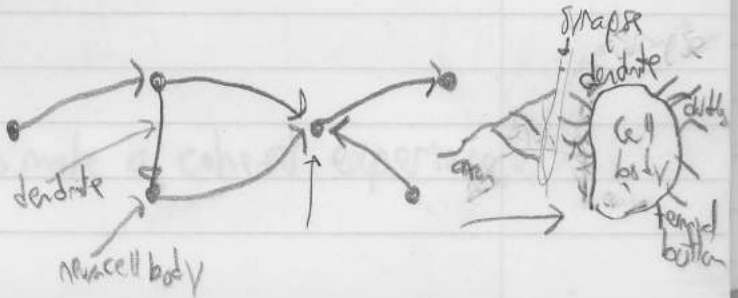


Human brain

Cerebral cortex:
- ~ 10 billion neurons
- outer layer, folded sheet 2.5 mm thick
- most advanced part of brain;
- most neurons devoted to vision perception
  ↳ really?

Computers can't beat cerebral cortex!

Like a big directed graph:



☆ Show optical illusions

Classification of Senses

Sense	Stimulus	Receptor	Sensory structure	(optional) Robot Equivalent
Vision	Electromagnetic energy	Photoreceptors	Eye	Camera
Hearing	Air pressure waves	Mechanoreceptors	Ear	microphone
Touch	Tissue distortion	Mechanoreceptors Thermoreceptors	skin, muscle,	pressure, force sensors thermometer
Balance (Vestibular)	Gravity, acceleration	Mechanoreceptors	Vestibular organs	IMU
Taste/Smell	Chemical composition	Chemoreceptors	Mouth/Nose	pH meter?

Order of importance for our class:

1) VISION!
2) Balance
3) Hearing
4) Touch
5) Taste/Smell

The brain fuses these to make a coherent experience

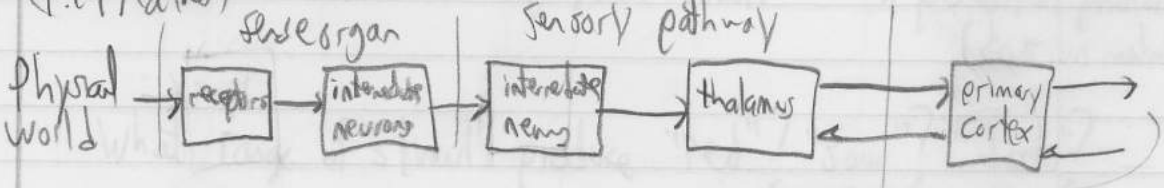
Most important for us: Vision + Balance (Vestibular)

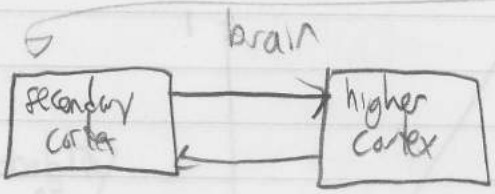
1) Vection → Optical flow disagrees with vestibular
2) Vestibulo Ocular Reflex (VOR)
   - Affects sense of stationarity
   - Eye counter rotates with head rotation to fixate eyes on a target.

Some fundamental concepts in sensation & perception

Receptors: Transducers that convert stimulus energy to neural impulses (Much like an engineered sensor to an analog or digital signal)

(p.14 Mather)





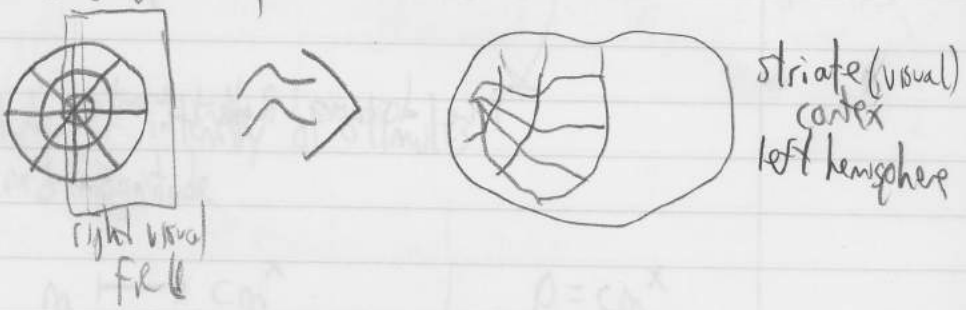
Hierarchical Processing!

Sensory system selectivity
- Each cell has a receptive field
- Limited frequency, color, shape, - - - for each (tiny "activation zone")
- Electrodes to perform single-unit recording (Display particular stimulus, determine whether cell activates)

like: /// not 0

Organization of neurons (network spatial topology!)

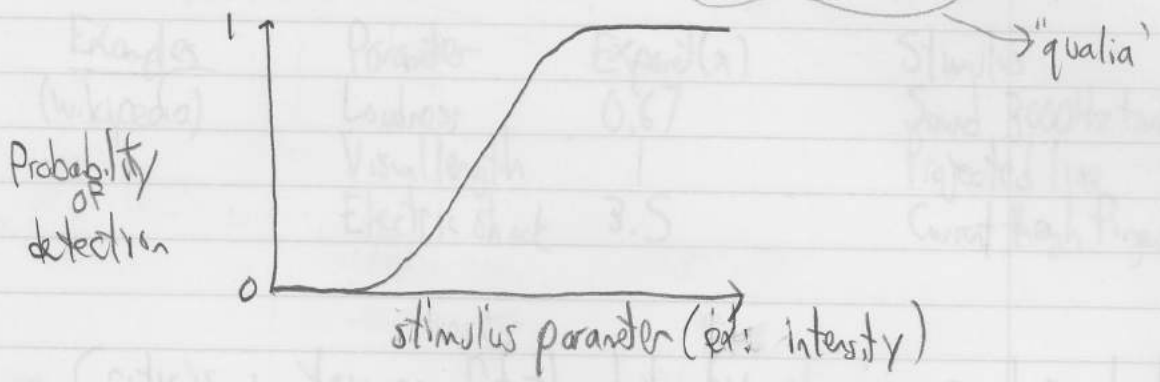
Topographic Map



Psychophysics: Scientific study of:

physical stimuli → perceptual phenomena (conscious mental state)

What range of stimuli produce "red"? "sour"? "dark"? ... ?



Joke! Stigler's Law of Eponymy

No scientific law is named after its original discoverer

True discoverer: Robert Merton

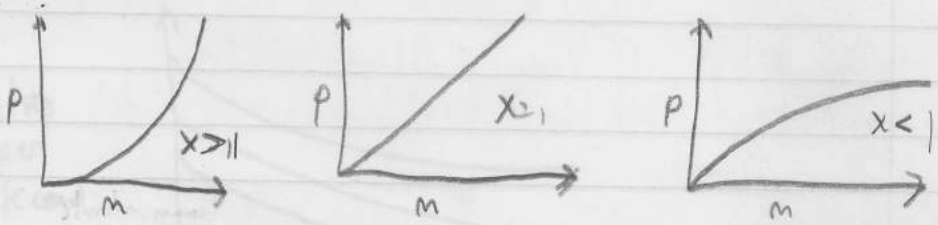
Magnitude estimation:

Stevens power law (1961) - See wikipedia

n = magnitude or intensity of stimulus
p = perceived magnitude

$$n \longmapsto cn^x \qquad p = cn^x$$

x = exponent that depends on particular experiment
(could have $x<1$, $x>1$, $x=1$)
c = constant based on units, etc.



Examples (wikipedia)	Parameter	Exponent(x)	Stimulus
	Loudness	0.67	Sound 3000Hz tone
	Visual length	1	Projected line
	Electric shock	3.5	Current through fingers

Criticism: You can fit the data by choosing c and x, but is the law always "correct"?

Related: Just-Noticeable Difference (JND)

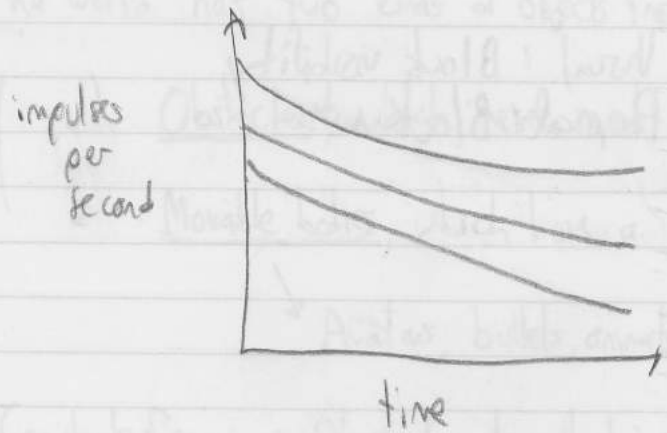
Weber's law:

$$\frac{\Delta m}{m} = c$$

(in other words, only percent change matters)

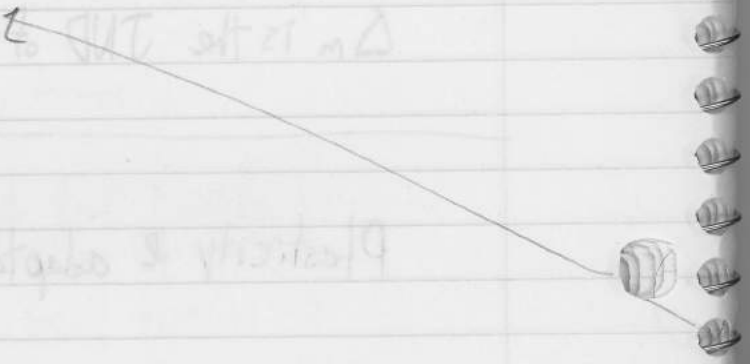
Δm is the JND of stimulus magnitude

Plasticity & adaptability



Maybe better name:

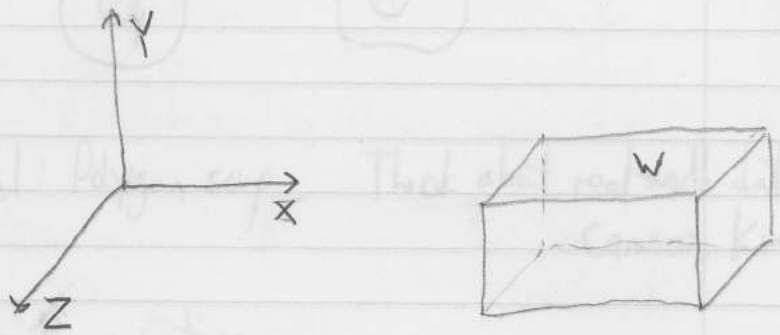
1) Stationary models
2) Movable models



Obstacles: 1) Visual: Block visibility
2) Physical: Block motion

Geometric Modeling

Let W be a 3D world: $W \subseteq \mathbb{R}^3$



Every point in W described by coordinates $(x,y,z) \in W$

The world has two kinds of objects inside:

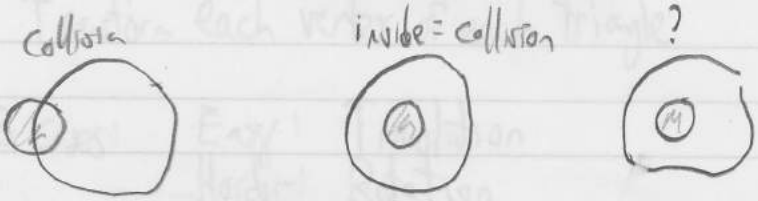
1) Obstacles, which never move: walls, furniture,
2) Movable bodies, which have a space of possible transformations (configuration space)
   - Avatars, bullets, animals, blowing leaves, coffee cup

Coordinate frames:
- Obstacles described in world frame
- Each (rigid) body described in its own body frame

Let's model obstacles, but some principles apply for bodies

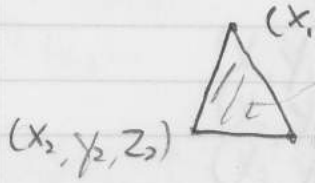
Two choices:
1) Solid representation
2) Boundary representation

Boundary representation is easier, but almost always hard to distinguish "inside" from "outside"



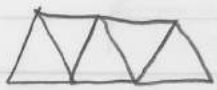
Worst-case model: Polygon soup. Think about real-world data from cameras, Kinect,

Lets use triangle primitives



→ A planar patch

Graphics problem: How to render the surface?
- Compute normal — ambiguity problem



Strip
- approximate any surface
- saves vertices
- might have numerical holes
- GPUs like these!

Mesh

Transforming Rigid Bodies

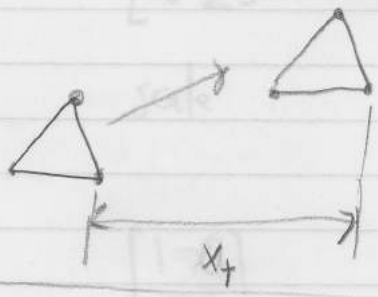
Transform each vertex of each triangle

		DOFs 2D	DOFs 3D
3 cases: Easy!	Translation	2	3
Harder!	Rotation	1	3
Hardest!	Rotation + Translation	3	6

Translation

Shift triangle by (x_t, y_t, z_t)

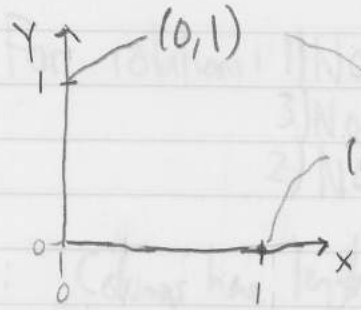
$$(x_1, y_1, z_1) \rightarrow (x_1 + x_t, y_1 + y_t, z_1 + z_t)$$
$$(x_2, y_2, z_2) \rightarrow (x_2 + x_t, y_2 + y_t, z_2 + z_t)$$
$$(x_3, y_3, z_3) \rightarrow (x_3 + x_t, y_3 + y_t, z_3 + z_t)$$



We need to get to 3D rotation. Start with 2D linear transformations:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

22)



basis for Cartesian coordinate

$\hat{x}, \hat{y}, \hat{z}$; $\hat{i}, \hat{j}, \hat{k}$

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} m_{11} \\ m_{21} \end{bmatrix}$$

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{12} \\ m_{22} \end{bmatrix}$$

Examples

$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ no effect

$\begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$ scale

$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$ mirror

$\begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$ rotate 180

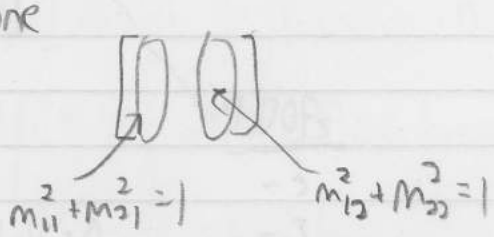
$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$ x-shear

$\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ y-shear

For rotations: 1) No stretching of axes - preserve scale
3) No mirror images - preserve orientation
2) No shearing - orthogonal axes must result

4 DOFs $\begin{bmatrix} : & : \\ : & : \end{bmatrix}$

-1 | 1: Columns have length one



-1 | 2: Columns are "perpendicular": Inner product = 0

$$m_{11} m_{12} + m_{21} \cdot m_{22} = 0$$

-0 | 3: $\det\left(\begin{bmatrix} : & : \\ : & : \end{bmatrix}\right) = 1$ (not -1)
↳ mirror image

DOFs = 1

Which matrices remain as rotations?

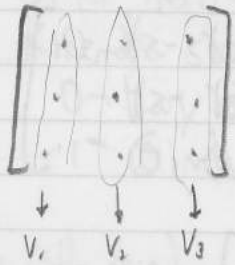
$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \quad \text{for all } \theta \in [0, 2\pi)$$

To rotate (x, y):

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

A circle of rotations $m_{11} = \cos\theta$
$m_{21} = \sin\theta$

Now try 3D rotations. Which 3×3 matrices are rotations?



1) $\|v_1\| = \|v_2\| = \|v_3\| = 1$
2) $v_1 \cdot v_2 = 0 \quad v_2 \cdot v_3 = 0 \quad v_1 \cdot v_3 = 0$
3) $\det([\,\vdots\ \vdots\ \vdots\,]) = 1$

9 DOFs
−3
−3
−0
3 DOFs

A 3D "surface" (manifold)

3 Canonical Rotations: Yaw, Pitch, Roll

Yaw:
$$R(\alpha) = \begin{bmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{bmatrix}$$
leave y coordinate untouched

Pitch:
$$R(\beta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & -\sin\beta \\ 0 & \sin\beta & \cos\beta \end{bmatrix}$$
leave x untouched
exactly 2D rotation

Roll:
$$R(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$
leave z untouched

matrix mult

Every 3D rotation R can be achieved by some α, β, γ with $R = R(\alpha)R(\beta)R(\gamma)$
with $\alpha \in [-\pi, \pi]$, $\beta \in [-\pi, \pi]$, $\gamma \in [-\pi/2, \pi/2]$?? Show ??

Example

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & c\alpha & -s\alpha \\ 0 & s\alpha & c\alpha \end{bmatrix}\begin{bmatrix} c\beta & 0 & s\beta \\ 0 & 1 & 0 \\ -s\beta & 0 & c\beta \end{bmatrix}\begin{bmatrix} c\gamma & -s\gamma & 0 \\ s\gamma & c\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Pitch $\beta = \pi/2 \rightarrow \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$

Result:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & s\alpha & c\alpha \\ 0 & -c\alpha & s\alpha \end{bmatrix}\begin{bmatrix} c\gamma & -s\gamma & 0 \\ s\gamma & c\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ s(\alpha+\gamma) & c(\alpha+\gamma) & 0 \\ -c(\alpha+\gamma) & s(\alpha+\gamma) & 0 \end{bmatrix}$$

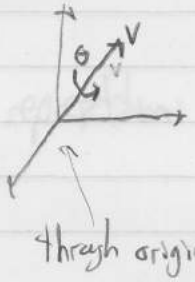
apply trig identity

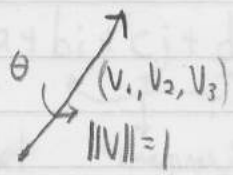
(Roll+Yaw) yields same orientation, regardless of Roll and Yaw angles.

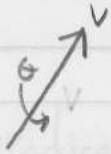
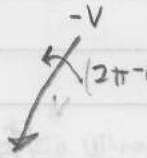
Problems
- Order matters: 3D rotations not commutative!!!
- Kinematic singularities and nonuniform representation
  - Vary α, β, α at constant speed, and rotation varies wildly
  - Gimbal lock

(Illustrate with prop) - Problem: Rotation axis keeps changing on body

Euler's rotation theorem (1776): All 3D rotations (or orientations) have an axis-angle representation





Note  and  are the same rotation!

It would be nice to have some algebra

$$(v, \theta) \circ (v', \theta') = ?$$

so that we always know the resulting axis and angle. → keeps changing! (Not in 2D case)

(In OpenCV, exponential coordinates used: $(\theta v_1, \theta v_2, \theta v_3)$ — scale is the angle!)

Nice! 3 parameters for 3 DOFs
$(0,0,0)$ = identity

Note: 3 DOFs, as needed for 3D rotation

Unit quaternions as a better representation of 3D rotation.

$q = (a, b, c, d) \in \mathbb{R}^4$ and $a^2 + b^2 + c^2 + d^2 = 1$ ⇒ "normalized"

Set of all q is a hypersphere (S^3)

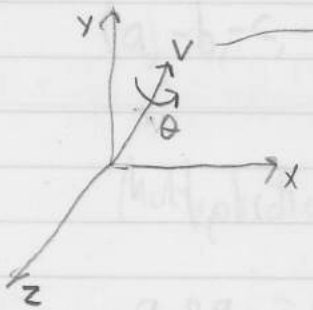
Unity3D/games representation: $(x, y, z, w) = (b, c, d, a)$

careful!

Most common representation: $a + bi + cj + dk$

"real" part "imaginary" parts like complex numbers $a + bi$

Encoding a 3D rotation using unit quaternions



$v = (v_x, v_y, v_z)$ $\|v\| = 1$

$\longleftrightarrow$ $(\cos\theta/2, v_x \sin\theta/2, v_y \sin\theta/2, v_z \sin\theta/2)$

a b c d

arccos to recover angle

Just renormalize to recover axis

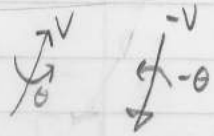
$\frac{b, c, d}{\sqrt{b^2 + c^2 + d^2}}$

Useful examples:

$(1, 0, 0, 0)$	identity rotation
$(0, 1, 0, 0)$	pitch by π X
$(0, 0, 1, 0)$	yaw by π Y
$(0, 0, 0, 1)$	roll by π Z
$(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, 0)$	pitch by $\pi/2$
$(\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}, 0)$	yaw by $\pi/2$
$(\frac{1}{\sqrt{2}}, 0, 0, \frac{1}{\sqrt{2}})$	roll by $\pi/2$

the bigger the axis part the bigger the rotation

Inverse and multiple representations



(a, b, c, d) equivalent $(-a, -b, -c, -d)$

inverse inverse

equivalent

$(a, -b, -c, -d)$ $(-a, b, c, d)$

Multiplication

Let $p = (b, c, d)$

$q_1 \circ q_2 = q_3$

$q_1 \circ q_2 = (a_1 a_2 - p_1 \cdot p_2, \; p_1 \times p_2 + a_1 p_2 + a_2 p_1)$

a_3 b_3, c_3, d_3

Can't be commutative!!!

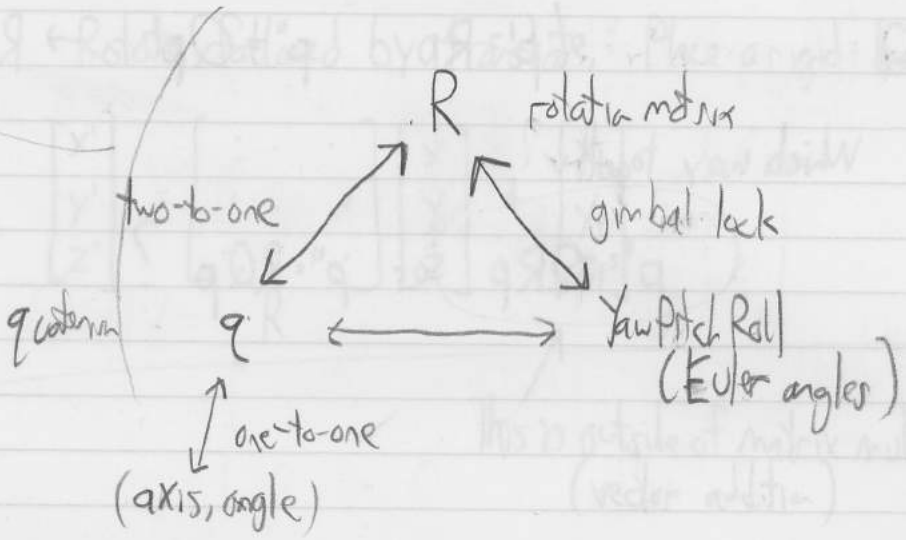
R - useful because "native"; rotate directly

q - no singularities!
- small change in q = small change in orientation (just like θ in 2D rotation)
- normalizable, hemispheres

YPR - intuitive
- bad singularities!

$$R_3^{-1} R_2^{-1} R_1^{-1} R_1 R_2 R_3 = R_3^{-1} R_2^{-1} R_2 R_3 = R_3^{-1} R_3 = I$$

Conversion



$$R_1 \cdot R_2 = R_3$$
$$\downarrow \quad \downarrow \quad \uparrow$$
$$q_1 \circ q_2 = q_3$$

Same result either-way: Find a proof!

Note: Can apply q directly to rotate (x, y, z).

Operation order: Which is applied first? $R_1 R_2$

$R_1(R_2[\begin{smallmatrix} x \\ y \\ z \end{smallmatrix}])$ → R_2 is!

Beware of inverse!

$(R_1 R_2 R_3)^{-1} = R_3^{-1} R_2^{-1} R_1^{-1}$ (linear algebra) group theory)

Which is first? $p = (x, y, z)$

R Q $\quad p' = Rp \quad p'' = Qp' \rightarrow$ R before Q

Which way together?

$p'' = QRp$ or $p'' = RQp$?

First make:

$$\begin{bmatrix} & & & 0 \\ & R & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rotate using quaternion q directly

$p = (x, y, z, 1) \quad p' = qpq^{-1}$

$q^{-1} = (a, -b, -c, -d)$

Homogeneous transformation matrices — A math hack

Rotate followed by translate: Place a rigid body anywhere

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \underbrace{\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}}_{R} \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}$$

This is outside of matrix multiplication (vector addition)

Make a 4×4 matrix (and 4D vector) that is algebraically equivalent

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} & R & & \begin{matrix} x_t \\ y_t \\ z_t \end{matrix} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

Same result, but extra "1" in 4th position

Now you get one 4×4 matrix $T = \begin{bmatrix} R & \begin{matrix} x_t \\ y_t \\ z_t \end{matrix} \\ 0\ 0\ 0 & 1 \end{bmatrix}$ that places rigid bodies

Important question: What is T^{-1}? (How to undo the placement?)

What is R^{-1}? $R^{-1} = R^T$ ← transpose

↑ only for "orthogonal matrices"
↓ orthonormal

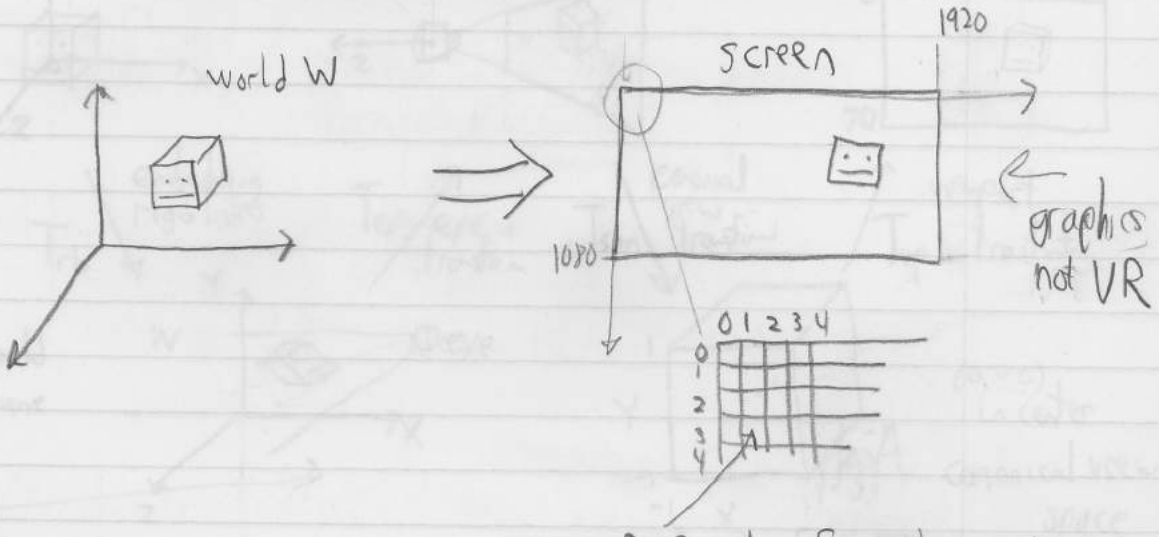
Opposite of x_t, y_t, z_t is $-x_t, -y_t, -z_t$

Is $T^{-1} = \begin{bmatrix} R^T & \begin{matrix} -x_t \\ -y_t \\ -z_t \end{matrix} \\ 0\ 0\ 0 & 1 \end{bmatrix}$? No!!

Wrong order of operations

$$T^{-1} = \begin{bmatrix} R^T & \begin{matrix} 0 \\ 0 \\ 0 \end{matrix} \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -x_t \\ 0 & 1 & 0 & -y_t \\ 0 & 0 & 1 & -z_t \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Viewing Transformations



Questions!

How should artificial light propagate?
How to discretize geometric model onto screen?
How to determine what is visible or blocked?
How to make correct perspective?
Efficiency?? Take a shortcut if flaws are not perceptible

Now let's worry only about transforming points from W to screen
(points, line segments, triangles, ...)
The remainder is the rendering pipeline.

Apply

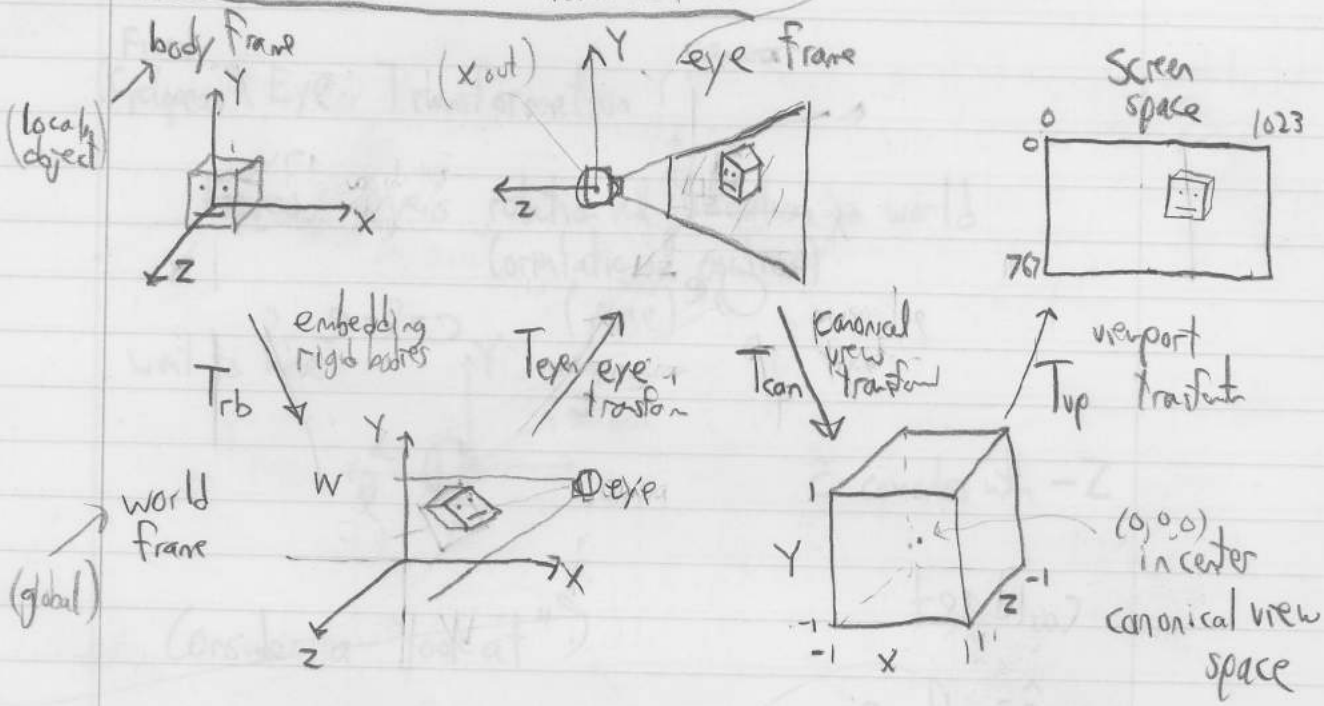
$$T \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} i \\ j \end{bmatrix}$$

Units

Body frame, world frame, eye frame: meters (i.e. feet)
Canonical view frame: none
Screen frame: pixel index

The Chain of Transformations

→ cyclopean!



→ Let T denote a 4x4 homogeneous transformation matrix for viewport from body coordinates

Invert T_l, T_r for stereo rendering

$$T = T_{vp} T_{can} T_{eye} T_{rb}$$

for VR!

Insert T_{dist} to compensate for lens distortion

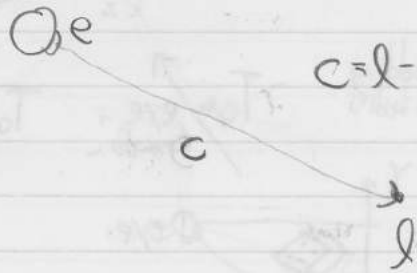
Look at each transformation:

T_{rb} - already covered (rotation + translation)

T_{eye} - becomes an inverse of a rigid body placement

T_{can} - apply perspective transform and rescale, translate

T_{vp} - units conversion and more rescale, translate



could get

$$\hat{c} = \frac{\ell - e}{\|\ell - e\|}$$

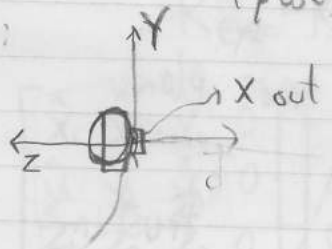
ℓ = look-at point (ℓ_x, ℓ_y, ℓ_z)

Why is $\hat{y}$ not simply $\hat{v}$?
This allows sloppy input! $\hat{v}$ need not be perpendicular to image plane.

Fixed
(Cyclopean) Eye Transformation

Express eye's rotation and translation in world
(orientation and position)
(pose)

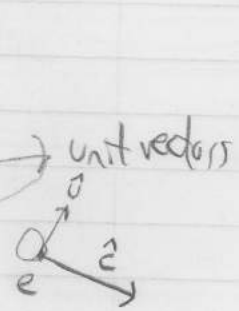
want to have:



$\hat{c}$ coincides with $-Z$

Consider a "look at":

1. Position of eye: e
2. Looking direction in eye coord: $\hat{c}$
3. Up Direction: $\hat{U}$



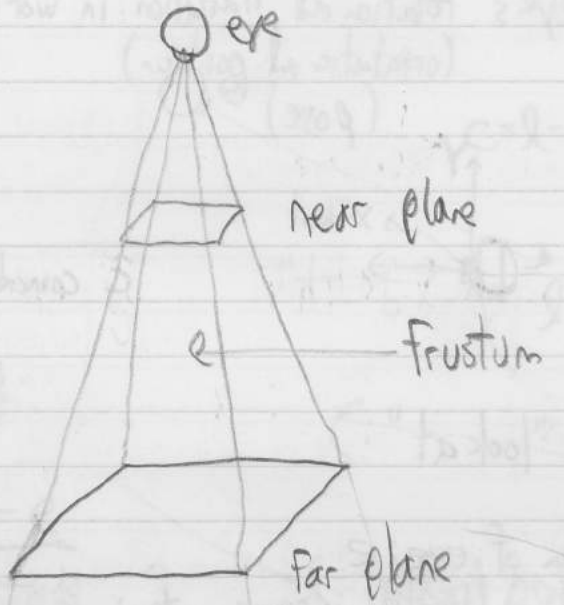
Make a transform to place eye in W

Coordinate axes for eye:

$$\hat{x} = \hat{U} \times \hat{z}$$
$$\hat{y} = \hat{z} \times \hat{x}$$
$$\hat{z} = -\hat{c}$$

$$R_{eye} = \begin{bmatrix} \hat{x}_1 & \hat{y}_1 & \hat{z}_1 \\ \hat{x}_2 & \hat{y}_2 & \hat{z}_2 \\ \hat{x}_3 & \hat{y}_3 & \hat{z}_3 \end{bmatrix}$$

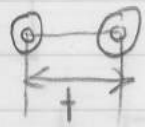
To transform world into eye view, need inverse transform!



$$T_{lefteye} = \begin{bmatrix} 1 & 0 & 0 & +t/2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{eye}$$

$$T_{righteye} = \begin{bmatrix} 1 & 0 & 0 & -t/2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} T_{eye}$$

t = inter eye distance in virtual world



($t = 0.064_m$ on average in real world)

Interpretations: 1) Dragging the world as if it were a rigid body
2) Try eye at origin ($e = (0,0,0)$)
Compare: -Yaw world by θ (without moving eye)
-Yaw eye by $-\theta$ (" " world)

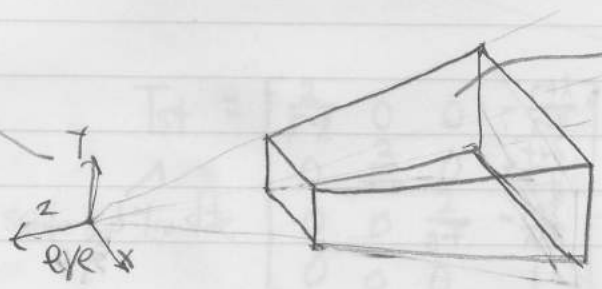
Thus, $R_{eye}^{T} = R_{eye}^{-1}$

$$T_{eye} = \begin{bmatrix} \hat{x}_1 & \hat{x}_2 & \hat{x}_3 & 0 \\ \hat{y}_1 & \hat{y}_2 & \hat{y}_3 & 0 \\ \hat{z}_1 & \hat{z}_2 & \hat{z}_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -e_x \\ 0 & 1 & 0 & -e_y \\ 0 & 0 & 1 & -e_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

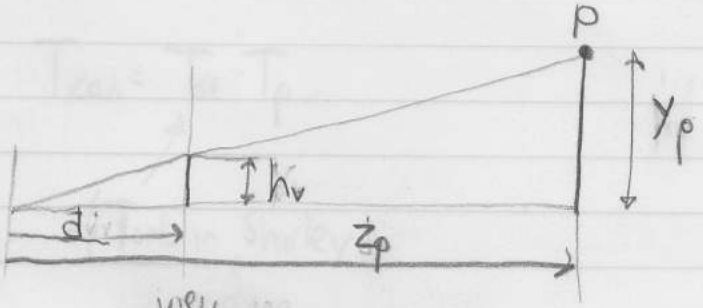
→ Inverse translation

Stereo

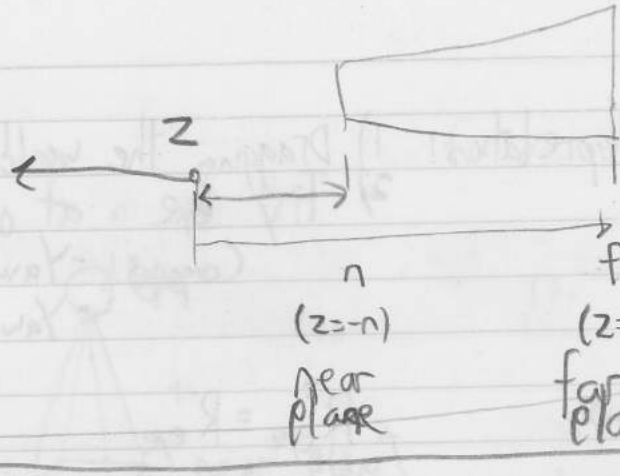
Canonical View Transform



- projective geometry (space of lines through origin)



Perspective transform: $h = \frac{d}{z_p} y_p$ → dividing by z_p is not linear



The transformed corners of viewing frustum:

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

Tricking matrix multiplication to do the algebra

1D example (homogeneous)

$$\begin{bmatrix} y' \\ h' \end{bmatrix} = \begin{bmatrix} d & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} dy_p \\ z_p \end{bmatrix}$$

Define $h := \frac{y'}{h'} = \frac{dy_p}{z_p}$ (same as previous figure)

n, f

$$T_p = \begin{bmatrix} n & 0 & 0 & 0 \\ 0 & n & 0 & 0 \\ 0 & 0 & n+f & -fn \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

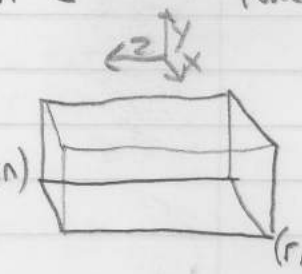
perspective

$x' = nx$ → $\left(\frac{x'}{h'}, \frac{y'}{h'}\right)$

$y' = ny$

$z' = z(n+f) - fn$ → strange; maintains both ordering (used in pipeline)

$h' = z$



$$T_{st} = \begin{bmatrix} \frac{2}{r-l} & 0 & 0 & -\frac{r+l}{r-l} \\ 0 & \frac{2}{t-b} & 0 & -\frac{t+b}{t-b} \\ 0 & 0 & \frac{2}{n-f} & -\frac{n+f}{n-f} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

scale and translate

x: l..r (left-right)
y: b..t (botta-top)
z: n..f (near-far)

$$T_{can} = T_{st} T_p$$

(Torth in Shirley)

One remaining part: T_{vp} converts $-1..1$ range to pixel coordinates

n_x = # of horizontal pixels (eg. 1920)
n_y = # of vertical pixes (eg. 1080)

$$T_{vp} = \begin{bmatrix} n_x/2 & 0 & 0 & \frac{n_x-1}{2} \\ 0 & n_y/2 & 0 & \frac{n_y-1}{2} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} i \\ j \end{bmatrix} \leftarrow \text{pixel indices}$$

Light

Three interpretations:

1) particles

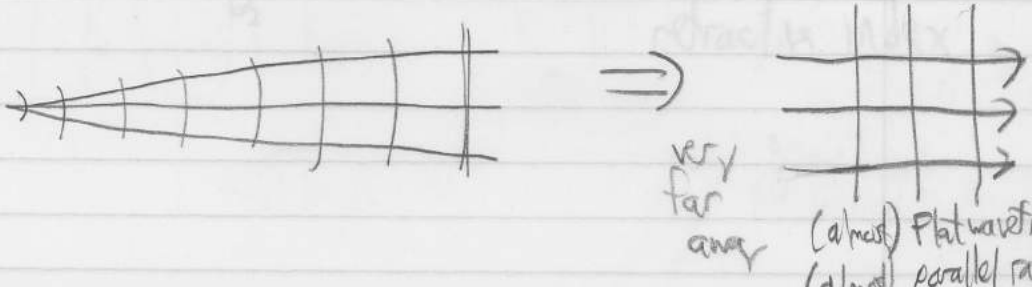
photons — $f = c/\lambda$ (f: frequency, c: speed of light, λ: wavelength)

2) rays ⟹ computer scientists love this! (graphics, computational geometry)

3) waves))))

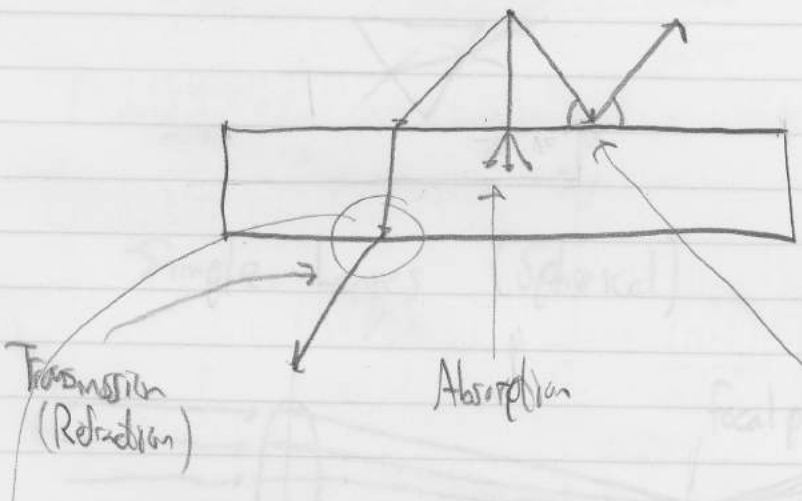
Point source of light:
- Generates wavefronts
- Rays are orthogonal to wavefronts

Without mirrors or lenses, rays <u>always</u> diverge

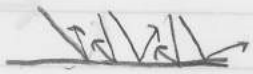


parallel rays — Zero vergence
Collimated rays — rays to or from infinity

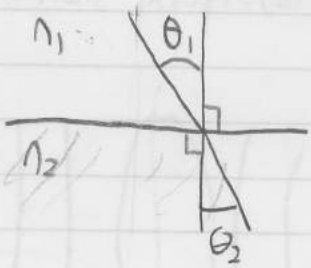
We use materials to bend the rays/waves



Reflection types: Specular 

Diffuse 

→ Snell's Law

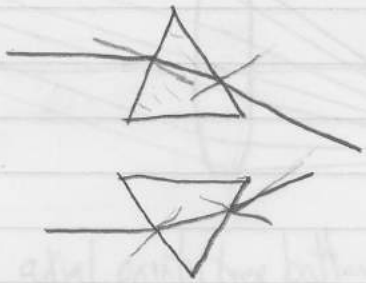


$$\frac{n_2}{n_1} = \frac{\sin\theta_1}{\sin\theta_2}$$

$$n_i = \frac{c}{\text{Speed in medium}}$$

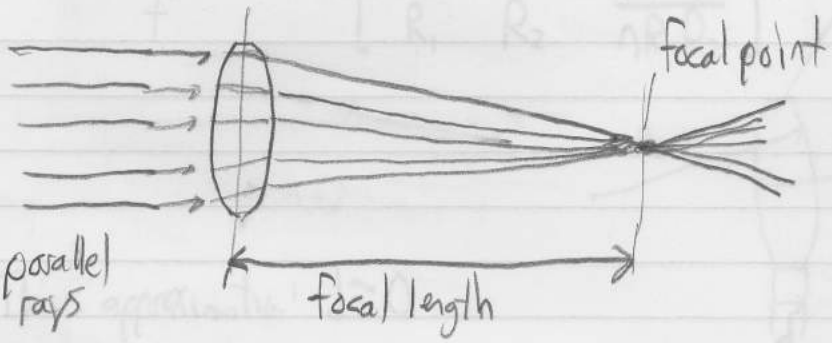
refractive index

Refraction

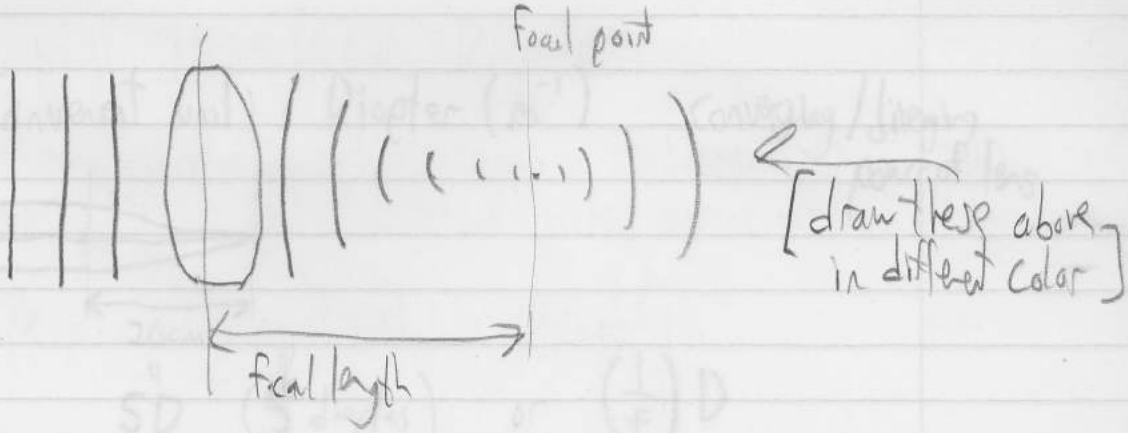


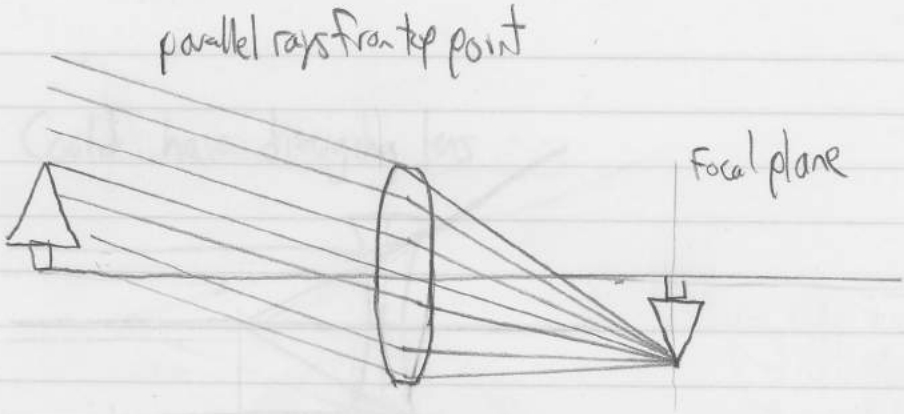
Two interfaces

Simple Lenses (Spherical)



Wave front interpretation

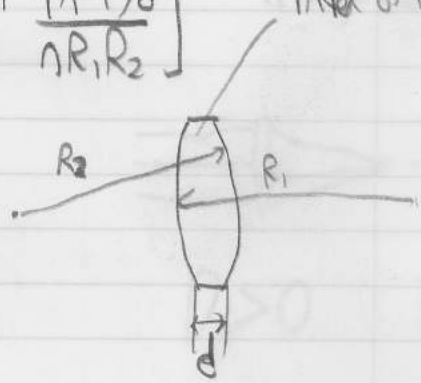




[could do axial point (tree bottom) in another color]

Lensmaker's equation

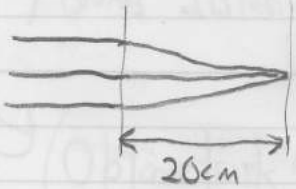
$$\frac{1}{f} = (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2} + \frac{(n-1)d}{nR_1R_2}\right]$$



Thin approximation $d \approx 0$

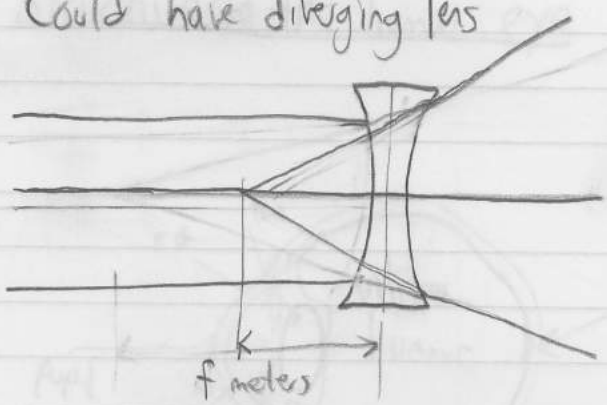
$$\frac{1}{f} = (n-1)\left[\frac{1}{R_1} - \frac{1}{R_2}\right]$$

Convenient unit: Diopter (m^{-1}) converging / diverging power of lens

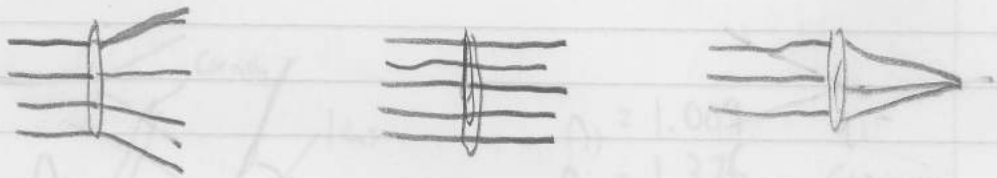


5D (5 diopters) or $\left(\frac{1}{f}\right)D$

Could have diverging lens



Lens power $\left(\frac{-1}{f}\right)D$ → negative because lens causes divergence



$D<0$ $D=0$ $D>0$

Combining lenses or interfaces!

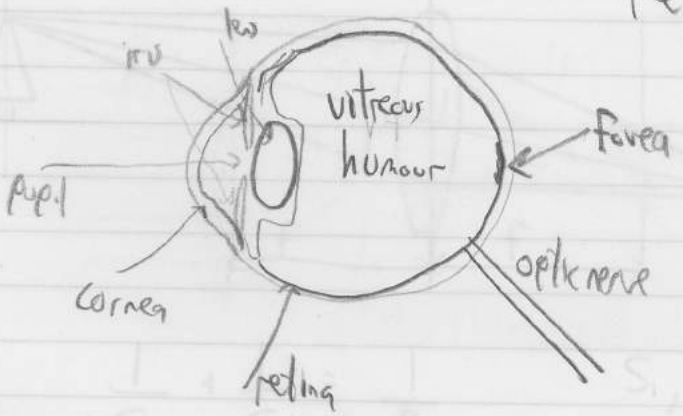
$$D = D_1 + D_2 + D_3 \cdots$$

Assumes distance between lenses or surfaces ≈ 0
(thin lens approximation)

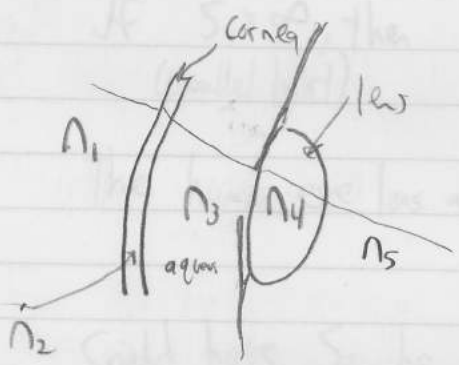
(Optometrists must not like algebra!)

Frm p.159
W.J. Smith
Modern Optical
Engeering

Structure of the human eye



retina contains "focal points"
↳ non-planar!
↳ non-uniform receptors

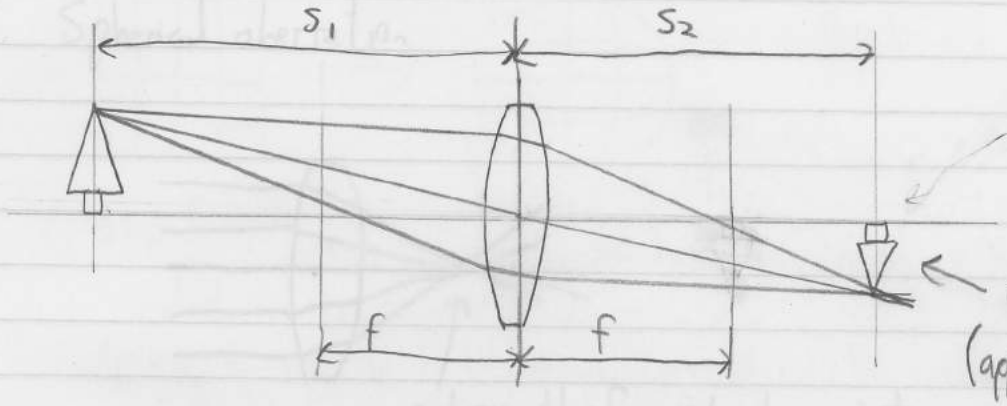


$n_1 = 1.009$ air
$n_2 = 1.376$ cornea
$n_3 = 1.336$ ocular fluid (aqueous)
$n_4 = 1.413$ lens
$n_5 = 1.337$ vitreous

(48 is entry to cornea)

Total power of eye's optical system: 59.52D
(16.8 mm)

Imaging properties of a lens



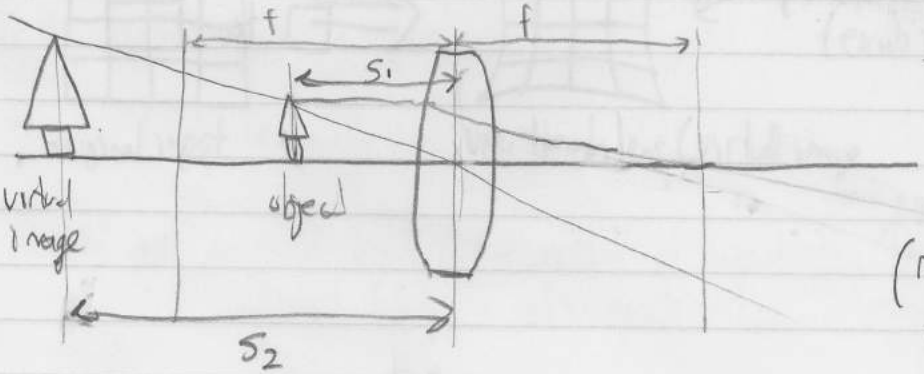
← real image (appears on retina but curved)

$$\frac{1}{s_1} + \frac{1}{s_2} = \frac{1}{f} \qquad s_1, s_2 > f$$

If $s_1 = \infty$, then $s_2 = f$
(parallel light)

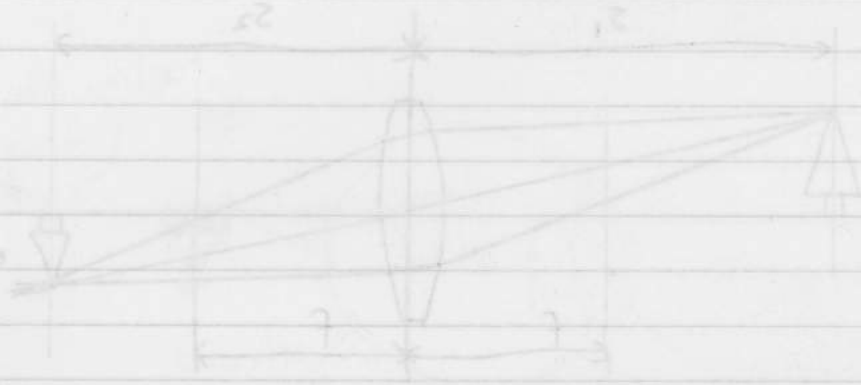
The human eye lens adjusts is f (or D) to bring closer objects into focus

Could have s_2 be negative: magnifying glass

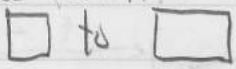


$s_2 < 0$

Think about looking into Rift (real image still produced at retina by eye lenses)

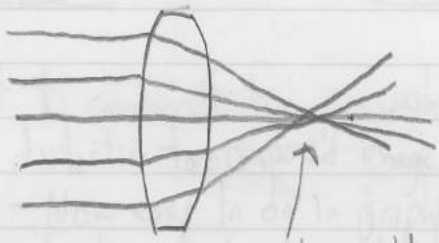


Along horizontal axis, this is like watching a stretched movie from



Lens Aberrations

1. Spherical aberration



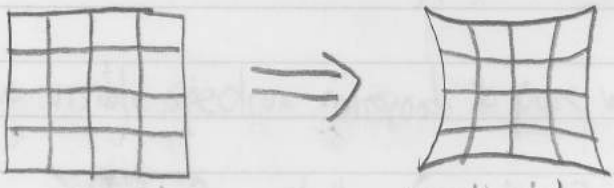
not exactly focused at a point

Solution: elliptical ( ) ← hyperbolic Harder to manufacture (aspheric lens)

Related: Look into edges of Rift lens and image might not be focused

2. Optical distortion

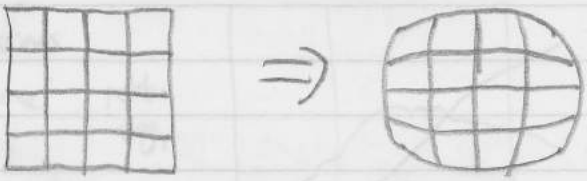
For high field of view lenses, the image is distorted



original input view through lens (virtual image

← pincushion distortion (exists in Rift)

Inverse of pincushion is <u>barrel distortion</u>:



Think old CRT monitors

In Rift to compensate for pincushion distortion, apply a barrel distortion to rendered image.
- Now easy to do in graphics hardware (1990s had warped images)
- Apply in last stage of chain of transforms

$r \longmapsto f(r)$

(only a function of radius, or distance to image center)

3. Chromatic aberration

Light is composed of waves, varying between ~400nm and ~700nm wavelengths
(<400nm is ultraviolet; >700nm is infrared)

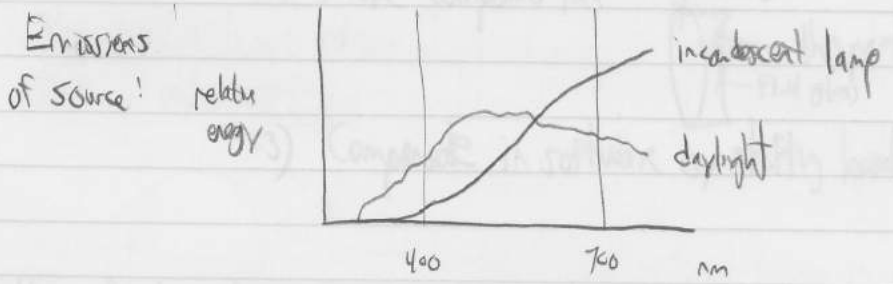
(Each wave has frequency $f = \frac{c}{\lambda}$)

The visible spectrum corresponds to pure waves (single sinusoid ~~~):

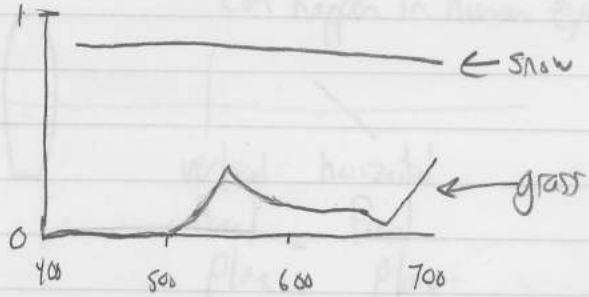
Red, Orange, Yellow, Green, Blue, Indigo, Violet

Nearly all perceived light is a mixture of wavelengths/frequencies
(Note: White is not in spectrum)

Spectral power - Like a histogram of wavelengths

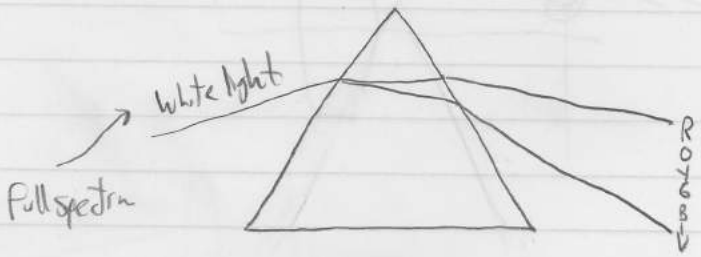


Spectral reflectance



Perceiving color of an object! Source spectrum AND reflectance spectrum

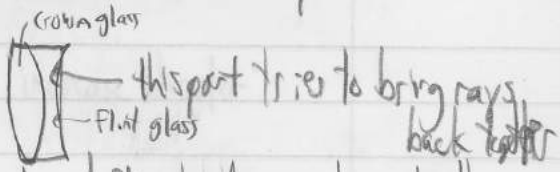
Dispersion: Speed of light in medium depends on its wavelength!



Something happens in a lens

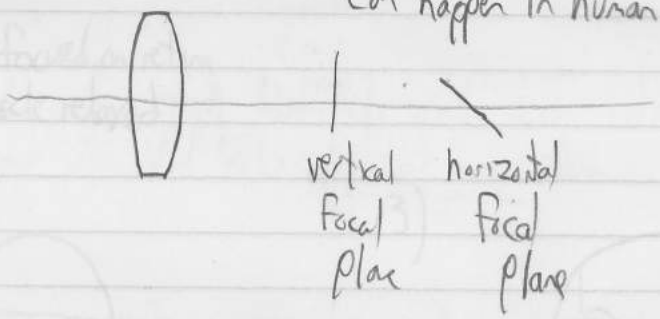
Eg. Blue Focal length < red focal length

Solutions: 1) Find lens with high Abbe number (expensive)
2) Make compound lens



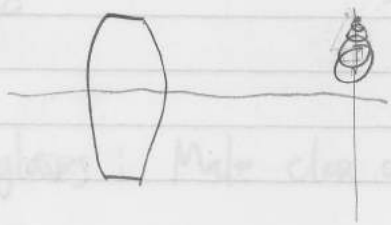
3) Compensate in software by shifting based on pixel wavelengths

4. Astigmatism
- Elliptical eccentricity in lens
- Can happen in human eye



Vertical waves vs horizontal waves! no common focus

5. Coma - off axis distortion of image (like "comet")



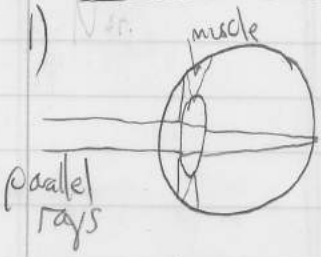
Airy pattern due to varying lens thickness from edge to center

How does the eye's lens/image system work?

Accommodation: Lens shape changes to increase diopter

Normal eye examples:

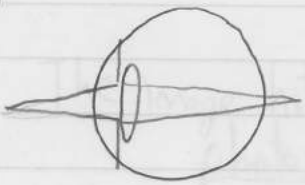
1)



- Note that retinal image is not planar!
- Eye lens is not circular

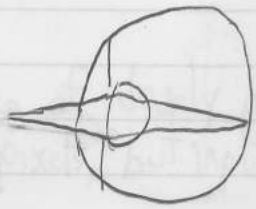
- Sharply focused on retina
- Eye muscle relaxed

2)



- Blurry on retina
- Eye muscle relaxed

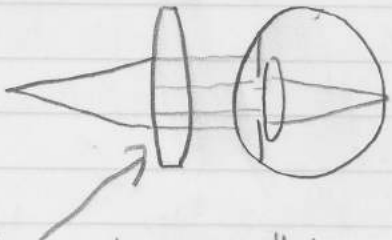
3)



(refocus)

- Eye uses muscles to accommodate
- Diopter increases

4) Reading glasses: Make close objects "seem like" infinite distance



- lens makes rays parallel
- No accommodation required

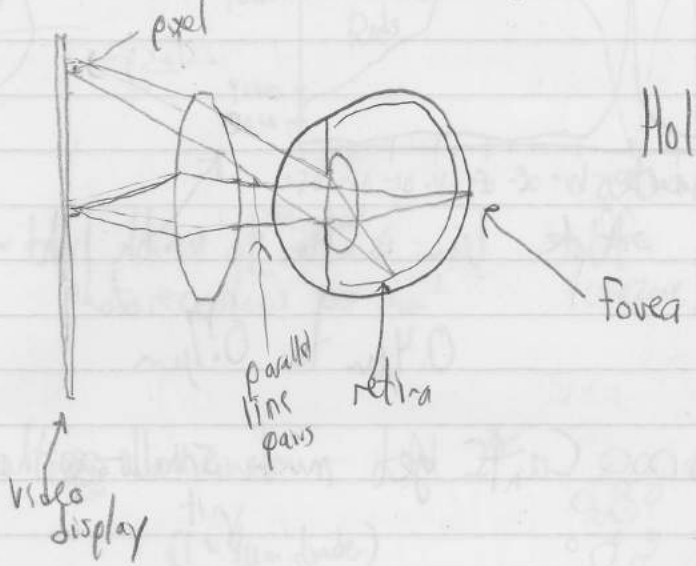
"The Dress"

Blue & Black
White & Gold

Fornel - Red & Yellow

News flash! The dress contained thermochromatic dye!
When you take a picture, the color reacts on temp

A video rendering system and optical system put real image on retina (lens+eye)



Hold eye fixed!

The image hits a spherical surface of densely packed photoreceptors: Like pixels, but input instead of output

Two kinds: Rods and Cones

	Rods	Cones
Number (per eye)	120,000,000	6,000,000
Function	Low light intensity	Color sensitivity

Diameter

Note 1 μm is close to visible light wavelengths

0.4 μm to 0.7 μm

Can't get much smaller without bleed over, interference

Area of retina: 1094 mm^2
Density at fovea: ~200,000 / mm^2
→ closer to 200,000,000 photoreceptors

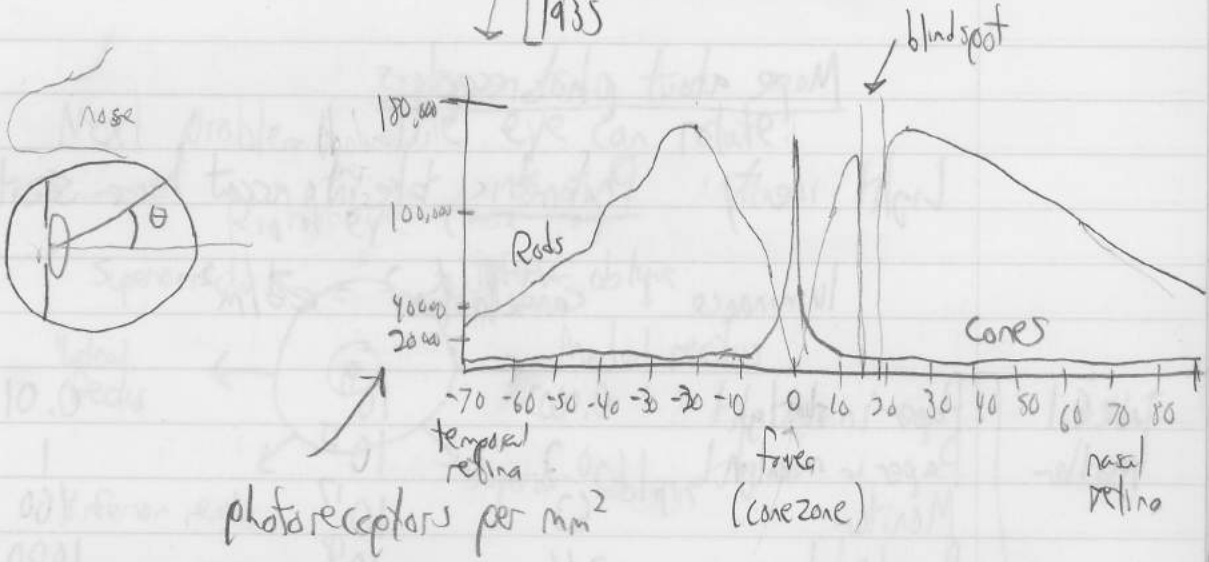
$\sqrt{200,000,000} \approx 14,142$

What about varying density?

If resolution to low ← photoreceptors

Image of one pixel

G. Osterberg
Acta Opthalmol. Suppl. 6:1
1935



At 0°: all cones tiny (1 to 4 μm diameter)

At 2°: cones get bigger (4 μm - 16 μm diameter)

At 50°: mostly rods

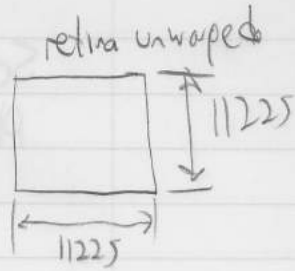
rod ~ 1 μm diameter

Implications: - Sharp color at retina; sharp monochrome at periphery
- Retina/color requires high light intensity

How much resolution is enough for VR?

Rough estimate $\sqrt{126{,}000{,}000} \approx 11225$

total photoreceptors



Perhaps screen should be 16K x 16K
Any more is a waste!

50b)

More about photoreceptors

Light intensity: Photometric take into account human sensitivity

Luminance: candelas/m^2 = cd/m^2

Table 6.1 Matter

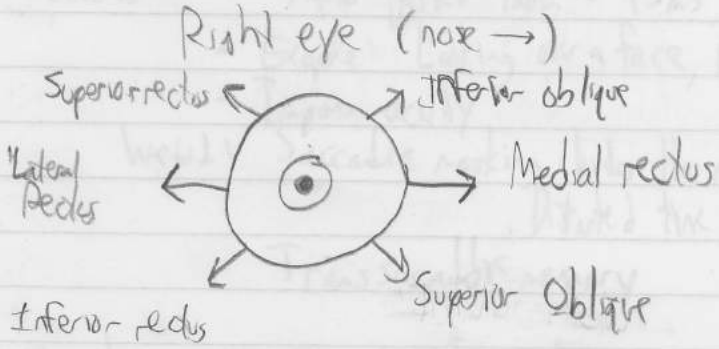
Paper in starlight	0.0003	10^{14}	0.01
Paper in moonlight	0.3	10^{15}	1
Monitor	63	10^{17}	100
Room Light	316	10^{18}	1000
Blue sky	2500	10^{19}	10,000
Paper in sunlight	40,000	10^{20}	100,000
	Luminance cd/m^2	Photons/ $m^2 \cdot sr \cdot s$	Photons/receptor

	Scotopic vision	Photopic vision
Photoreceptors	Rods	Cones
Light levels	$<0.01\ cd/m^2$	$>10\ cd/m^2$
Dark adaption	35 min	10 min
Color	Monochromatic	Trichromatic

Scotopic: Dark light vision (night)
Photopic: Bright light vision (day) <

- 90% of cells devoted to this
- we are "day time" animals

Next problem: The eye can rotate!



LM, MR: Eye yaw (up to 50 degrees), adduction (non-vergence part)

SR, IO, IR, SO: Eye pitch, avoiding roll

Types of eye movements

	Conjugate	Disjunctive
Voluntary	Saccade Pursuit	Convergence Divergence
Involuntary	Vestibulo-Ocular Optokinetic	
	Microsaccades	

If VR tracking/render/distortion is wrong, the world appears "swimmy"

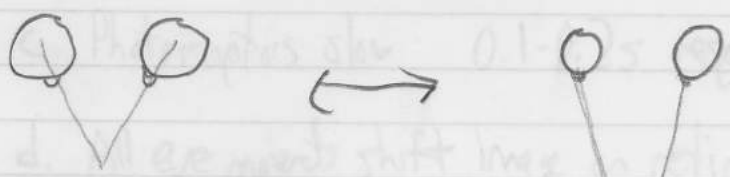
Or real world can appear swimmy — take off glasses

Movents

1. Saccades - rapid "jerks" latin < 45ms, 900°/s
   - Example: Looking over a face, reading
   - Improves acuity

   Weird: Saccadic masking hides the motion intervals from brain
   Distorted time perception - second hands on clock
   Transsaccadic memory

2. Smooth pursuit
   - Track a moving visual signal
   - Example: Moving tennis ball, car
   - <30°/s, otherwise saccades added
   - Reduces motion blur (slow photoreceptors)
3. Vestibulo-ocular reflex
   - Bypasses brain (≈ 10ms, fastest reflex in body)
   - Keep image stability
   - Counter rotations for 6DOF head motion
   - VOR Gain adaptation
4. Optokinetic
   - Alternating between pursuit & saccade
   - Example: Watch a passing train
5. Convergence/Divergence - Stereo Vision



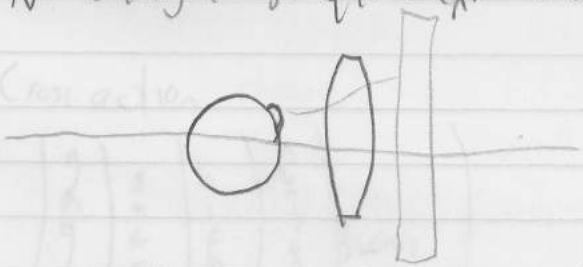
Note: Accomodation happens too, but not in VR at present!!

Result: very hard to design VR system that is "equivalent" to display

6. Microsaccades
   - 1/30 to 2 degrees motion
   - Purpose: largely debated, unknown
   - Might cause illusory motion
     ↳ recall slide earlier...

Regarding VR and eye motion:

1. Not looking through optical axis (even if eyeball centered)



   Problems: focus, distortion, maybe nausea

2. VOR gain adaptation – swimming effect in VR and real world

3. Complicated interaction between display & photoreceptors
   a. Pixels switch color, intensity at some rate (20ms LCD, 80μs OLED)
   b. Frames might be off (black), Frame rate 60, 75, 90 FPS
   c. Photoreceptors slow: 0.1-0.25 response time
   d. All eye movements shift image on retina!

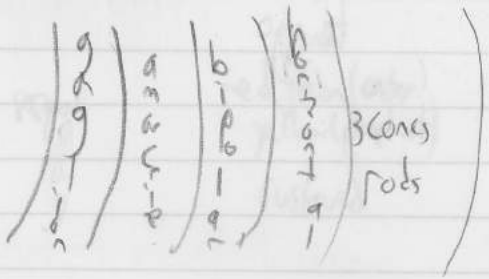
4. Vergence usually coupled with accommodation
But not in VR → always in focus

- Switch to slides on visual pathways
- Based on Mather Ch 7

Show:

- Retina

- Cross section



ON bipolar → cone and rod → connect to amacrine cells
OFF bipolar → cone only → connect to ganglion

Horizontal cells: Lateral inhibition

Pathways: Ganglion → LGN/thalamus part → visual cortex

Make alien circuit joke

- Receptive Field model → funneling, compression
- Opponency → important idea!
  - ↳ spectral vs spatial

Ganglia Cells

	Midget	Parasol	Bistratified
	70-80%	8-10%	<10%
	Photopic	P & Scotopic	Photopic
	opponent	opponent	nonopponent
spectral response	red-green (center) yellow (peripheral)	yellow (Red-green)	blue-yellow
temporal	sustained	transient	

ON-center

OFF-center

Ganglion Image

Path to visual cortex

Orientation Tuning

Visual Cortex

End slides

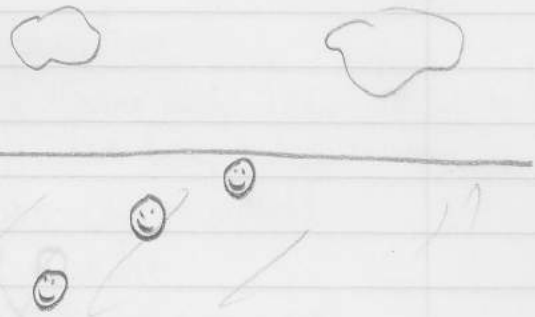
Depth Perception — Matlin Ch 10

Depth cues: 1) Metric - vary continuously with distance
2) Ordinal - ordering: near to far

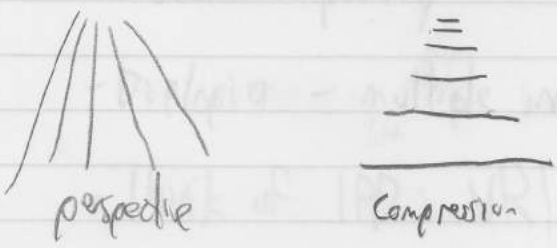
Multiplicity of depth cues — Not just stereo
(think about Google streetview)

1. Retinal image size

2. Height in visual field

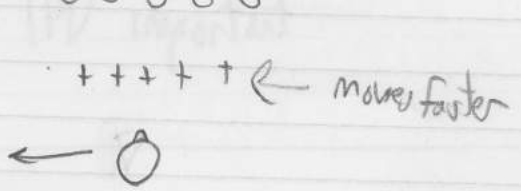


3. Texture gradients



Show
Slides!
(Dir 15)

4. Image blur
   Blur parts of image to appear outside of focal plane/depth
5. Atmospheric perspective
   Hazy mountains in distance
6. Accommodation
   for close objects and people under 50
7. Motion parallax (optical flow)



8. Shadows/shading
9. Interposition
10. Stereo cues
    - Vergence angle

    - Binocular disparity → left and right eye images are shifted
    - Diplopia - multiple images
    - Think of IPD in VR!
    - Horopter (LG explains curved TV)

Combination of Cues

Think Bayesian

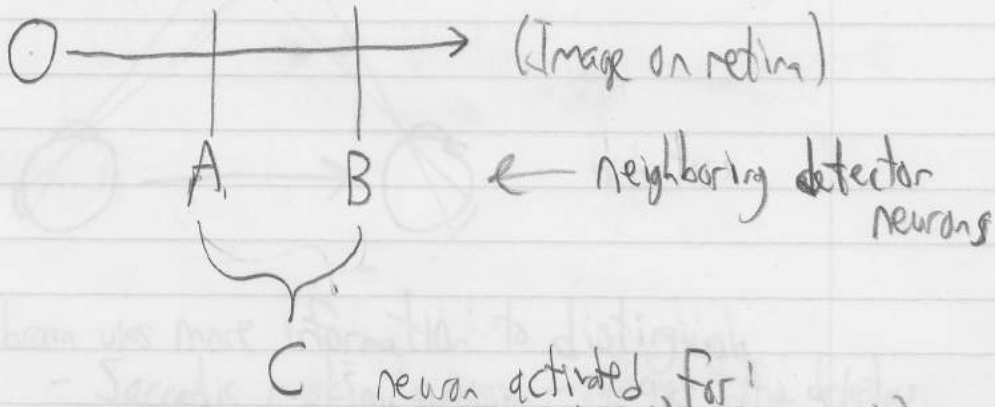
Scale perception vs depth perception

- Reach and grab something
- They affect each other in VR
- Context, IPD important

Motion Perception

Purposes:
- Segmentation / Segregation
  Extract moving body from background
- Extract 3D structure from motion
  (Spin the chair around)
- Visual guidance of action
  - Manipulation - grab a cup
  - Hand-eye coordination

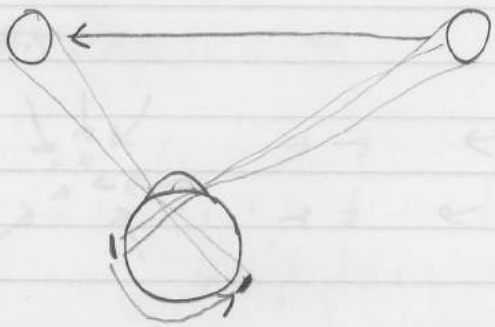
Neural circuitry for motion



Possible to confuse with wagon-wheel effect (video)

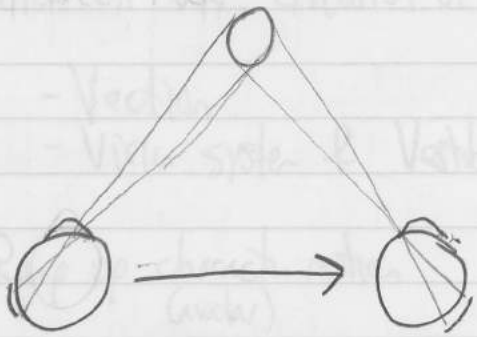
Object Motion vs Observer Motion

Retinal image moves same way in either case



Object moves, observer and eye fixed

Note: Hard to do because of smooth pursuit

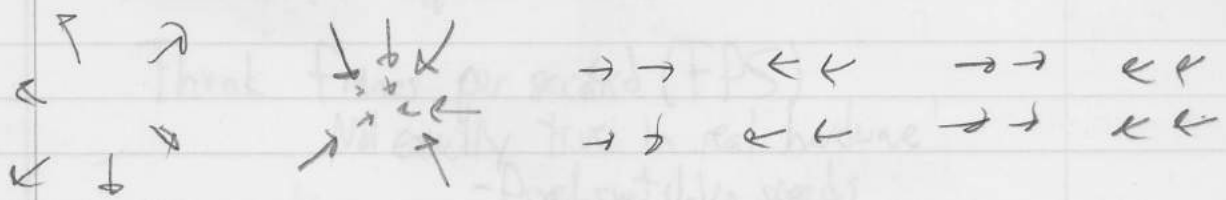


The brain uses more information to distinguish
- Saccadic masking/ supression – suppresses motion detectors
- Eye movement commands
- Large-scale motion – If eye moves, the whole scene moves
  (Recall big swing illusion)

Optical Flow

- Track movement of features on retina (or image plane or screen)
- A vectorfield on image plane (or sphere)

Self Motions



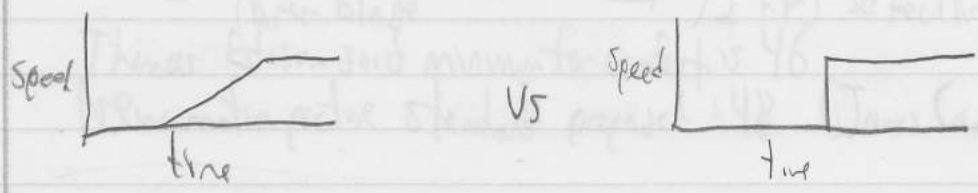
Forward Backward Left Right CCW CW

(Walk around room)

Big problem in VR: Illusion of self motion from optical flow

- Vection
- Vision system & Vestibular system in disagreement

Ex: Ramp up character motion (avatar)



Which is more comfortable?

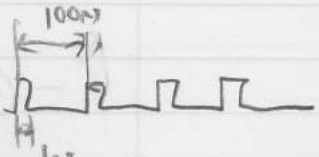
Small acceleration mismatch over long time
VS
Large acceleration mismatch over short time ← better!

Stroboscopic apparent motion

Think frames per second (FPS)
Not exactly true in real hardware!
- Pixel switching speeds
- Sequential scanout

See Wikipedia – "frame Rate"
"Flicker fusion threshold"

One ms enough for a frame to register
Up to 10 FPS, images perceived individually



Early silent films: 16-24 FPS (hand-turned speed)
Home movies: 16-18 FPS
Motion picture industry: 24 FPS standard

- Two-blade projector → 48 FPS effective } reduce
- Three-blade → 72 FPS effective } flicker

Thomas Edison said minimum for comfort is 46
New motion picture standards proposed: 48 (James Cameron wants)

CRT refresh rate → large monitors, peripheral vision

60 Hz - noticeable flicker
72 Hz - minimum ergonomic recommendation
85 Hz - comfort for all

Note: Detectable vs not, but headache, fatigue vs not, and comfortable

1 2 3 Levels

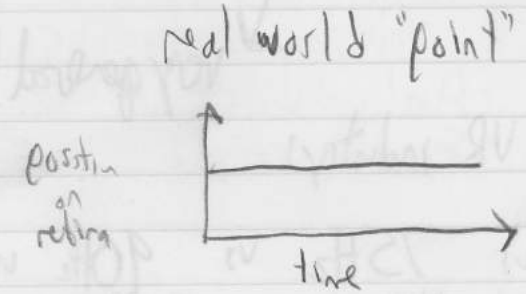
↓ very general!

Current talk in VR industry:

60 Hz vs 75 Hz vs 90 Hz vs 120 Hz
GearVR DK2 CrystalCoy/ Vive Sony?

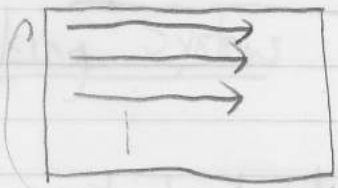
- Google for "blur busters"
  Everything better than 60Hz
- Abrash blogs

Works because photoreceptors need only 1ms of exposure, and hold for around 100ms



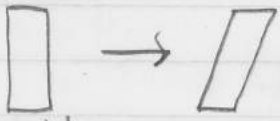
Problems with displays

Most scanout image line by line:



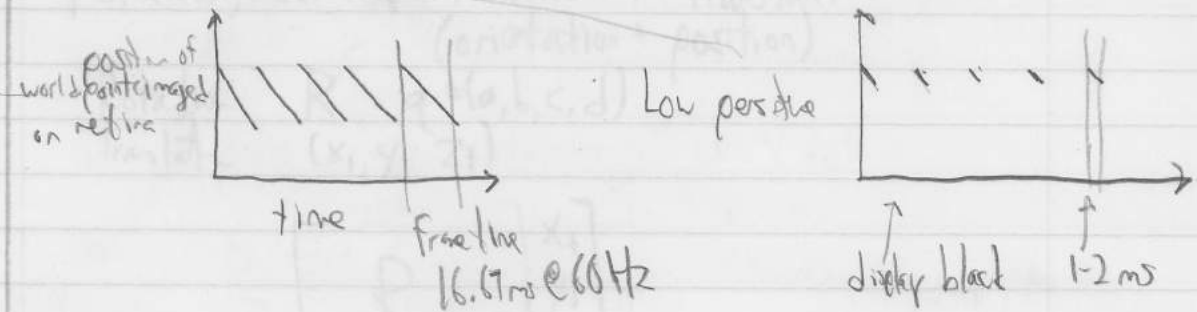
Motivated by original CRT - electron gun - analog
Need to keep phosphors illuminating

This causes a "waggle" when screen object moves



But not translating across retina in most cases

Recall smooth pursuit and VOR
Both supposed to keep object fixed on retina



Low persistence leads to flicker fusion problem; thus 90Hz or higher

Another problem: LCDs have slow pixel-switching speed
Result: Blur — see DK1

Tracking Systems

What do we want to track?

Think about rigid bodies

1) Head wearing HMD
2) Palms of hands
3) Eyes
4) Fingers
5) Entire body
6) Movable objects — controller, coffee cup, —
7) Other people in the space

For each, estimate rotation + translation
(orientation + position)

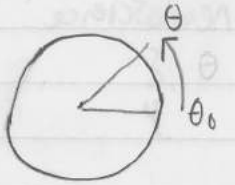
Rotation R, $q = (a, b, c, d)$
Translation (x_t, y_t, z_t)

$$\begin{bmatrix} R & \begin{matrix} x_t \\ y_t \\ z_t \end{matrix} \\ \hline \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{bmatrix}$$

Estimating orientation (Oculus Rift DK1, Samsung Gear VR)

→ Also DK2, but + position

2D case



Merry-go-round

$$\omega = \frac{d\theta}{dt}$$

At $t=0$, $\theta=\theta_0$
At $t>0$, rotation rate is ω rad/s
Thus, $\theta(t) = \theta_0 + \omega t$

What if ω varies over time?

$$\theta(t) = \theta_0 + \int_0^t \omega(s)\,ds$$

Discrete-time approximation: $\Delta t = 1\text{ms}$ $\quad \omega_i = \omega((i-1)\Delta t)$

$$\theta = \theta_0 + \sum_{i=1}^{k} \Delta\theta_i \approx \theta_0 + \sum_{i=1}^{k} \omega_i \Delta t = \hat{\theta} \quad \text{(Euler integration)}$$

Now suppose a sensor estimates ω_i every Δt

$$\hat{\theta} = \theta_0 + \sum_{i=1}^{k} \hat{\omega}_i \Delta t$$

$\hat{\theta}$: estimated rotation
$\hat{\omega}_i$: sensor reading from 1D gyroscope

[Continue in next notebook]

Blogger Beau Cronin

Oculus BPG: "O BPG may be the most substantial thing ever written on applied sensorimotor neuroscience"

Oculus Best Practices Guide

Goals:
1. Oculomotor comfort - avoid eyestrain
2. Bodily comfort - prevent feelings of disorientation, nausea
3. Positive user experience - fun, immersive, engaging
4. Minimal aftereffects - visual-motor functioning after use

Rendering

- Keep 1-1 correspondence between physical and virtual world head rotation & translation
- Don't make fixed splash screen or images w/o tracking
- R&L eye image differ only by viewpoint
- Supersampling, anti-aliasing — worst at center pixels

Latency

- Target ~20ms for sensor fusion reading to rendering
- Keep latency constant as possible
- Use predictive tracking

Tracking & viewpoint

- Never let tracking thread starve
- Don't make large rotating line (esp. horizon)

Positional tracking

- Mindful of users poking their heads anywhere in camera range
- Better to fade out when limit (e.g. wall) reached)

Acceleration

vection problems
- Make mismatches (physical & virtual) as short as possible
- Acceleration could be constant speed! Move along curve
- Have user control acceleration

Movement speed

- Real world 1.4 m/s walking
- Teleporting OK, but try to preserve orientation
- Don't make them look one way while moving another

Cameras

- Don't do zoom effects — bad optical flow/acceleration
- No head bobbing

Managing SimSickness

- Find fresh users, you are not good judge of your own
- Note that people vary in their tolerance
- Make adjustments or options to vary intensity of problems

Degree of Stereoscopic Depth

- Don't make things too close
- Don't make virtual camera R/L distance too large → Fatigue!

User Interfaces

- Make UI sit 2-3 meters in front of user,
- keep in middle ⅓ of screen
- Caution with head movement to select menu
- Embed info into world objects
- Draw crosshairs at object distance

Avatar control

- Acceleration curves
- Avoid keyboard options if possible
- Head mount for control → Dumpy the elephant

Sound

- Difference betwn headphones & speakers (6DOF transform)
- NPC (non-player character) speech should not be "center channel"
- Keep position (+ orientation) in mind

Content

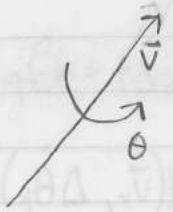
- Units: 1 meter in physical world = 1 meter in Unity
- Most objects betwn 0.75 - 3.5 m away
- Consider rendering avatar body, but beware of mismatch (eg sitting vs standing)
- Use monocular depth cues
- Avoid strafing, backstepping, spinning

Health & Safety

- Avoid 1-30Hz Flicker → photoepileptic seizures
- Avoid high contrast, high spatial frequency gratings →

Look up: Pokemon seizure episode

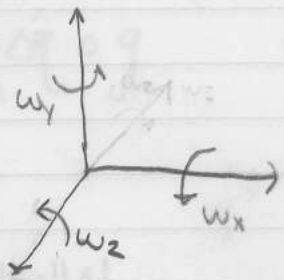
3D orientation case



Axis-angle $(\vec{v}, \theta)$

3 axis gyroscope measures

$\vec{\omega} = (\omega_x, \omega_y, \omega_z)$



If ω is constat over Δt, then rotation is

by $\Delta\theta = \|\vec{\omega}\| \Delta t = \left(\sqrt{\omega_x^2 + \omega_y^2 + \omega_z^2}\right) \Delta t$

and axis $\vec{v} = \frac{1}{\|\vec{\omega}\|} (\omega_x, \omega_y, \omega_z)$

during for Δt.

Let q_i be quaternion representation of $(\vec{v}_i, \Delta\theta_i)$

$q_i = \text{Quat}(\vec{v}_i, \Delta\theta_i)$

Recursively:

2D: $\hat{\theta}_{current} = \Delta\hat{\theta}_k + \hat{\theta}_{previous} = \hat{\omega}_k \Delta t + \hat{\theta}_{previous}$

3D $\hat{q}_{current} = \Delta\hat{q}_k \circ \hat{q}_{previous} = Quat(\hat{\vec{v}}_k, \Delta\hat{\theta}_k) \circ \hat{q}_{previous}$

Integrating sensor readings to estimate orientation

Recall 2D: $\hat{\theta} = \theta_0 + \sum_{i=1}^{k} \Delta\hat{\theta}_i = \theta_0 + \Delta\hat{\theta}_1 + \Delta\hat{\theta}_2 + \cdots + \Delta\hat{\theta}_k$

$= \Delta\hat{\theta}_k + \Delta\hat{\theta}_{k-1} + \cdots + \Delta\hat{\theta}_2 + \Delta\hat{\theta}_1 + \theta_0$

$\Delta\hat{\theta}_i = \hat{\omega}_i \Delta t$

match ordering

Now 3D:

$$\hat{q} = \Delta\hat{q}_k \circ \Delta\hat{q}_{k-1} \circ \cdots \circ \Delta\hat{q}_2 \circ \Delta\hat{q}_1 \circ q_0$$

← not commutative!

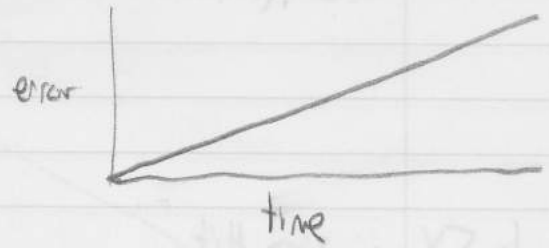
Each $\Delta\hat{q}_i = \text{Quat}(\hat{\vec{v}}_i, \Delta\hat{\theta}_i)$

$\hat{\vec{v}}_i$: $\frac{\hat{\vec{\omega}}_i}{\|\hat{\vec{\omega}}_i\|}$; $\Delta\hat{\theta}_i$: $\|\hat{\vec{\omega}}_i\| \Delta t$

Problem: Drift errors

2D: $d_k = \theta_k - \hat{\theta}_k$

3D: $d_k = q_k \circ \hat{q}_k^{-1}$



The error is a 3D rotation, with yaw, pitch, roll components

Note: Drift corrections can come from sensors at very low frequency.

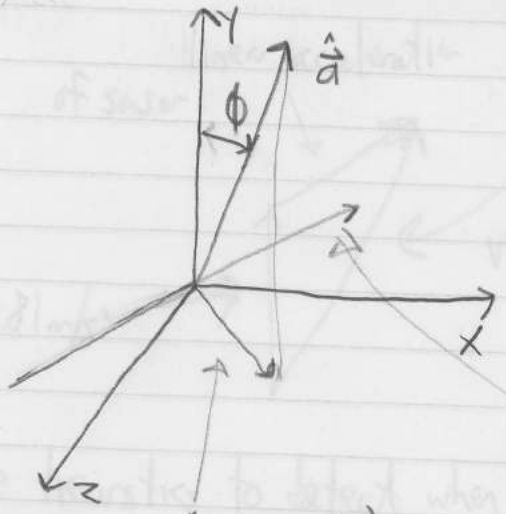
Let pitch + roll error be called <u>tilt error</u>.

Correcting for drift errors:

1) Use another sensor to provide world reference
2) Gradually apply corrections
   - Fast enough to fully compensate
   - Slow enough to avoid sim. sickness

For tilt error: Need a gravity or "up" sensor
For yaw error: Need a "compass"

<u>Use accelerometer to correct tilt error.</u>



Coordinate frame is estimated orientation

$\hat{a} = (\hat{a}_x, \hat{a}_y, \hat{a}_z)$

$(\hat{a}_x, 0, \hat{a}_z)$

$\vec{v}_{t.H} = (-\hat{a}_z, 0, \hat{a}_x)$ tilt axis in XZ plane

Need to rotate by ϕ about $\vec{v}_{t.H}$ to fix.

2D case

$\hat{\theta}_{corrected} = [illegible]\,\hat{\theta} + \alpha\phi$

ϕ: error measured in θ

Simple way to correct: Complementary filter

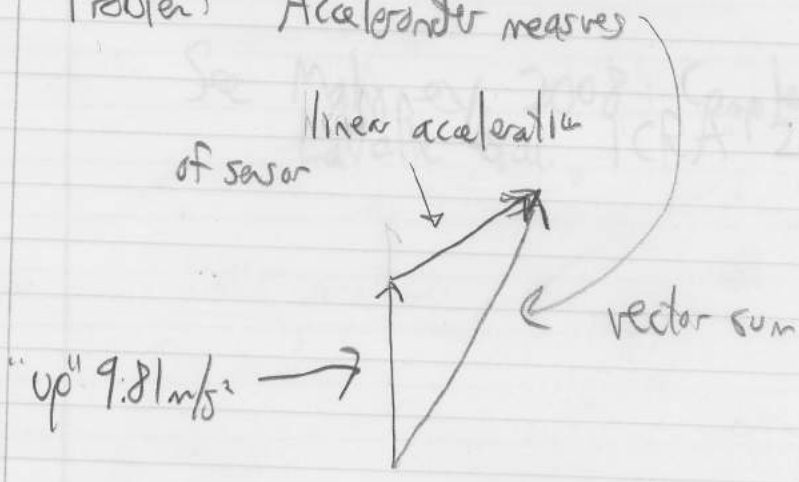
Gain coefficient $\alpha > 0$ $\alpha \approx 0$ (example $\alpha = 0.0001$)

In each Δt, ← changes every Δt

$$\hat{q}_{corrected} = Quat(\vec{V}_{tilt}, \alpha\phi) \circ \hat{q}$$

α needs to be large enough to cancel error, but small enough to avoid sim. sickness and perceived jitter.

Problem: Accelerometer measures



Use heuristics to detect when "not moving"

Example: $\|\hat{a}\| \approx 9.81$

↳ Can make mistakes.
Accelerate head downward

Use magnetometer to correct yaw error:

Similar to tilt correction:

- Calculate reference error
- Gradually apply using complementary filter

Problems:
- Vector sum of Earth's field, building field, board field
- Calibration hard
- Field might vary over time and position
- Soft iron bias

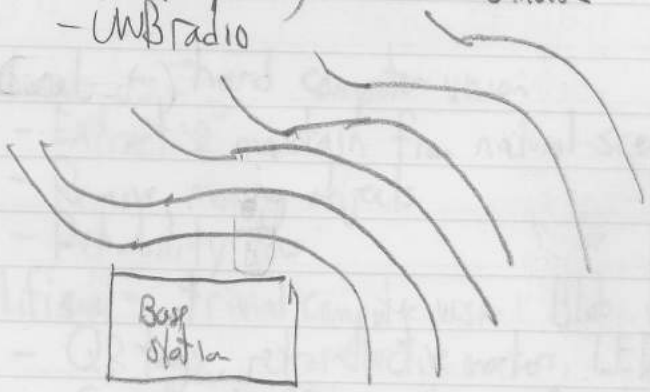
→ Accuracy ≈ 5 degrees

See Mahony 2008: Complementary filter on SO(3)
LaValle et al., ICRA 2014

Estimating position + orientation (Rift DK2, Sony Morpheus, Valve Vive)

- IMU (gyro + accel) not enough
  - Drift is too fast from double-integrating ∫
  - No way to detect drift error

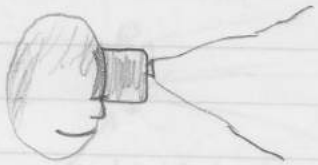
Solutions: 1) Generate non-constant magnetic or EM field
- Razer Hydra, STEM Sixense
- UWB radio



2) Visibility or line of sight methods

Camera arrangements

On headset

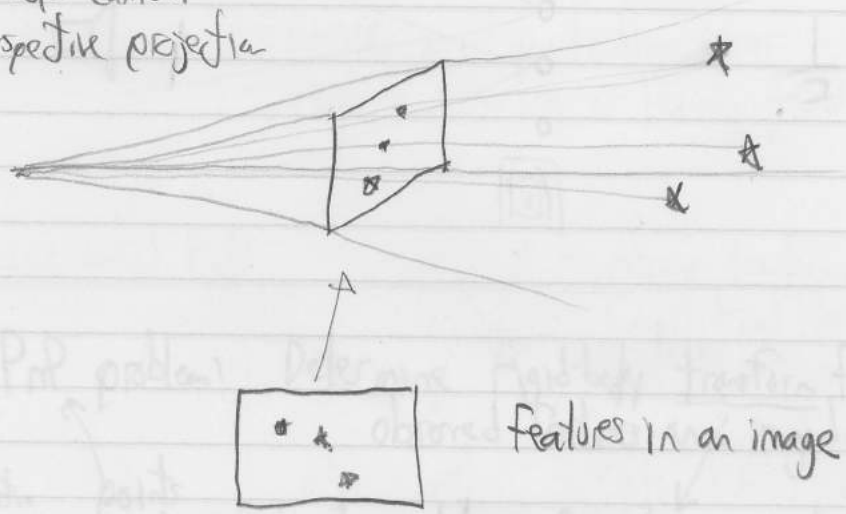


inside-out

In world

outside-in

Pinhole camera
Perspective projection

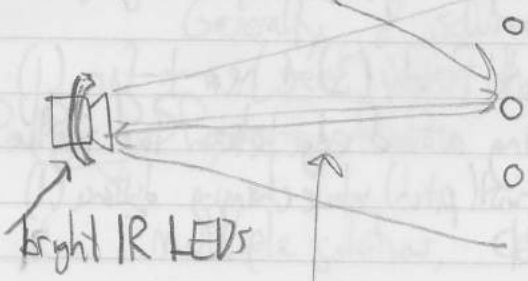


features in an image

Features: 1) Natural - hard computer vision
- Extract & maintain from natural scenes
- Remove moving objects
- Reliability low

2) Artificial - trivial computer vision: Blob detection
- QR tags, retroreflective markers, LEDs, laser-projector
- Can stay in IR spectrum (invisible to humans)

Retroreflective features

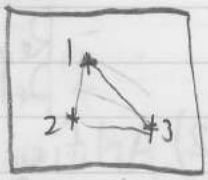


bright IR LEDs

power $\sim \frac{1}{d^2}$ and d is doubled

MoCap systems
Vicon
OptiTrack

Determine position & orientation of triagle from features in image

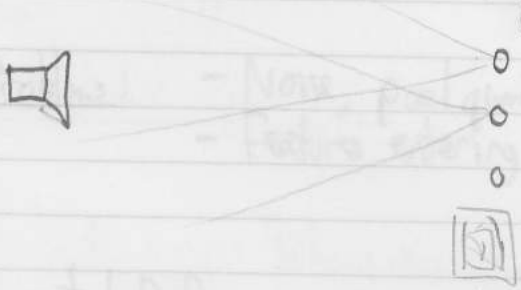


DOF analysis

- Start with 6 DOFs (rigid body)
- Each feature subtracts 2 DOFs (xy coords in image)

P1P: Object can rotate freely (3) + near-to-far (1) = 4
P2P: Object can swing around edge between two features (1) and twist (pitch) while changing distance (1) = 2
P3P: No freedom left

Illustrate with real object

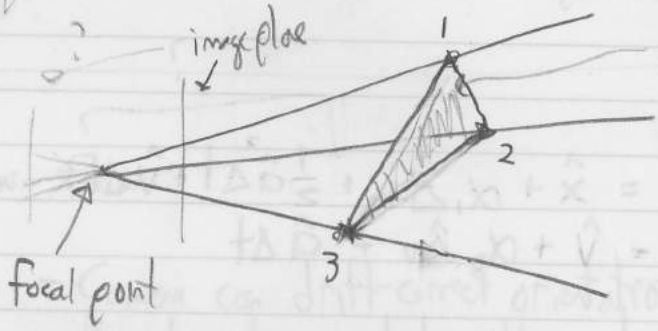


PnP problem: Determine rigid-body transform from identified, observed features on a rigid body.

perspective points

camera to body

P3P:



A system of polynomial equations can be obtained.
Generally, 8 solutions, but only 4 in front of camera
(See Wu, Hu, 2006 PnP Problem Revisited)

P4P, P5P, — PnP

multiple solutions, coplanarity causes difficulty

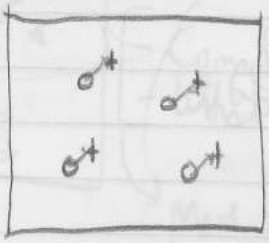
?

$$\hat{x}_{cor} = \hat{x} + \alpha_1 \hat{\Delta x} + \frac{1}{2}\hat{a}\Delta t + \hat{v}\Delta t$$

$$\vec{v}_{cor} = \hat{v} + \alpha_2 \hat{\Delta v} + \hat{a}\Delta t$$

Problems: - Noise, pixel quantization error
- Features entering or leaving FOV

Incremental PnP



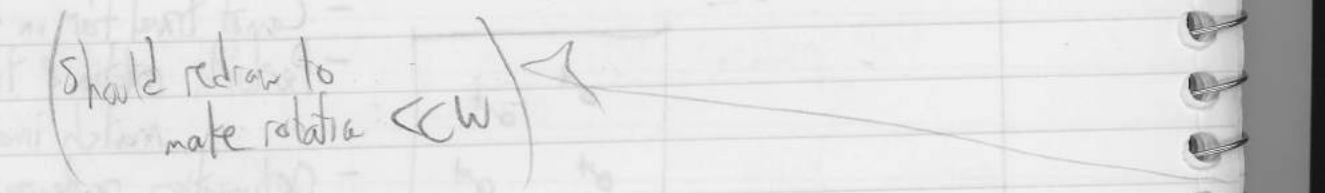
- Assume transform known already
- Cannot travel far in < 20ms
- Perturb rotation & translation to match image
- Optimization, randomized gradient descent
- Accelerometer helps improve starting point for optimization

---

Sensor Fusion:

- Camera can drift-correct orientation
- Accelerometer can help with position estimation
- Use complementary filter (2nd order)
  - ↳ Equivalent to Kalman filter for this problem (Higgins, 1975)
- Double integrator dynamical system model
  $\ddot{x} = v$ v = observed linear acceleration

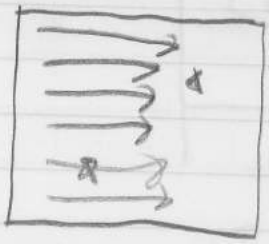
→ α coefficients for both position and velocity


(Should redraw to
make rotation CCW)

↗ Pharology

Lighthouse approach (Valve/HTC Vive)

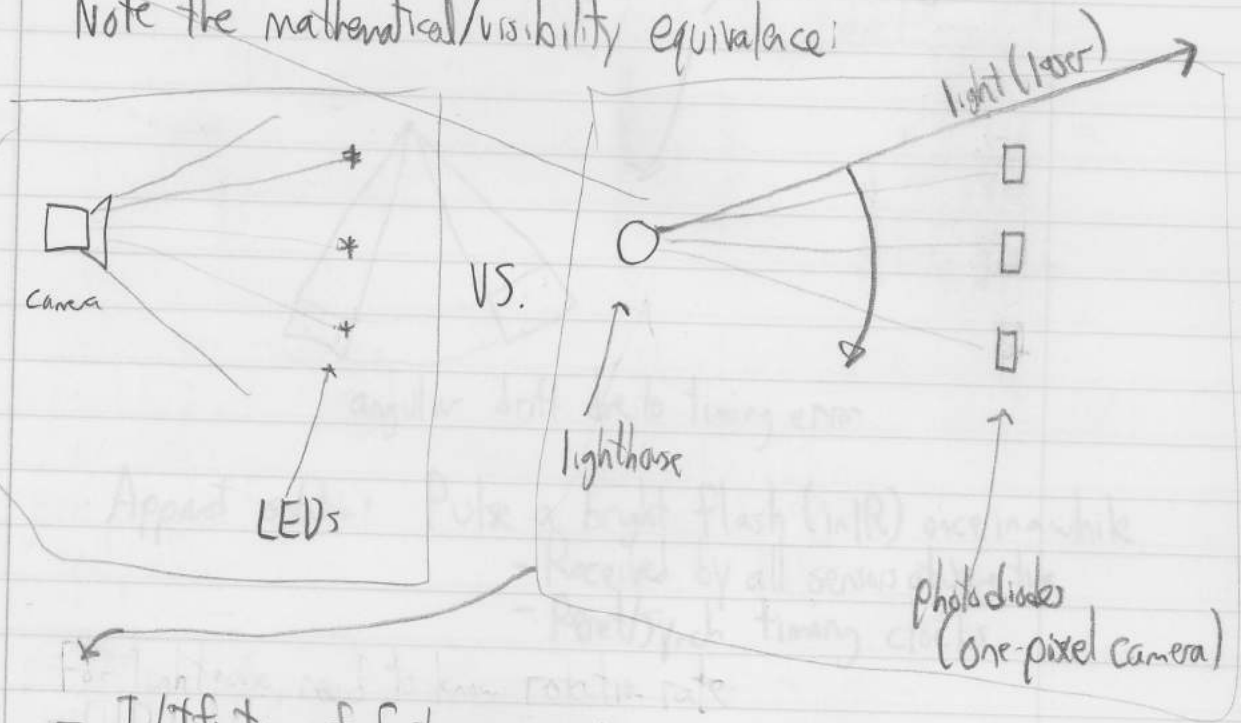
Why not bypass taking an image and scanning for features? (blob detection)



- Computationally costly (could do in hardware)
- Limited light per pixel

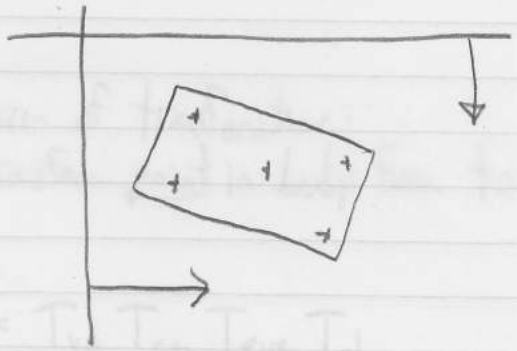
More resolution not necessarily better

Note the mathematical/visibility equivalence:

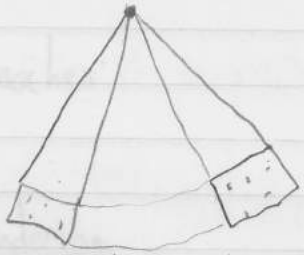


- Identification of features is easy
- Need to know rotation rate (100Hz?)
- No image processing
- Little power dissipation because laser
- Does have moving parts

Need horizontal and vertical beams



Problems: - Two beams hit same sensor at same time
- Maintaining timing to avoid angular drift



angular drift due to timing error

Apparent solution: Pulse a bright flash (in IR) once in a while
- Received by all sensors at same time
- Reset/Synch timing clocks

Rendering

Shirley et al., Ch 4, 8

Recall chain of trasformation:
Trasform point in body frame to world to screen space
(virtual) (pixel location)

$$T = T_{vp}\, T_{can}\, T_{eye}\, T_{rb}$$

T_{eye}: T_ℓ, T_r

Next problem: How to set the RGB values of each pixel?

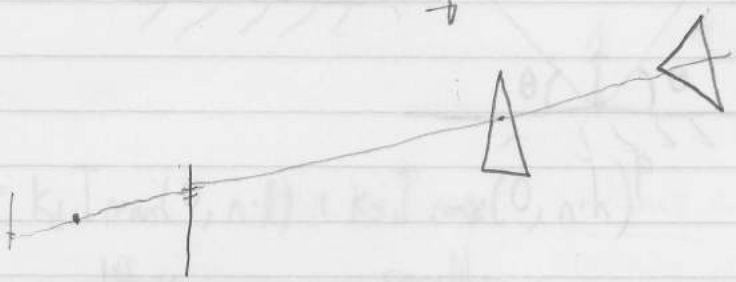
Two approaches:

Image-order rendering
- pixel-by-pixel
- Ray tracing
- easier to implement from scratch
- usually slower

object-order rendering
- triangle-by-triangle (or primitive)
- rasterization
- used by GPUs
- usually faster

Ray tracing stages!

1) ray generation
2) ray intersection
3) shading



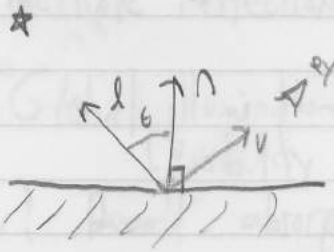
#1 is a ray of perspective projection

#2 must find <u>first</u> intersection from focal point

Intersection tests  Segment - Triangle

#3 Shading

Simplest: Lambertian - illumination proportional to cosine of



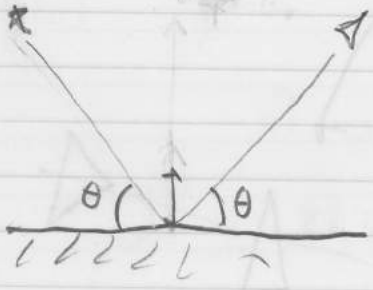
Unit vectors!

$$L = k_d I \max(0, n \cdot l)$$

pixel color — surface color — light intensity

$n \cdot l = \cos\theta$

When is light visible in mirror?

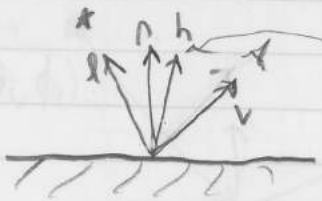


bisector = normal
(Law of Reflection)

diffuse, flat, matte surface

Handling highlights: Blinn-Phong shading

Unit vectors



bisector:

$$h = \frac{v + l}{\|v + l\|}$$

$$L = k_d I \max(0, n \cdot l) + k_s I \max(0, n \cdot h)^p$$

diffuse specular

(p. 83, Shirley)

$p = 10$ eggshell
100 mildly shiny
1000 really glossy
10000 nearly mirror like

What about ambient light?

Add $k_a I_a$ to L

What about multiple light sources?

Add diffuse and specular for each one.

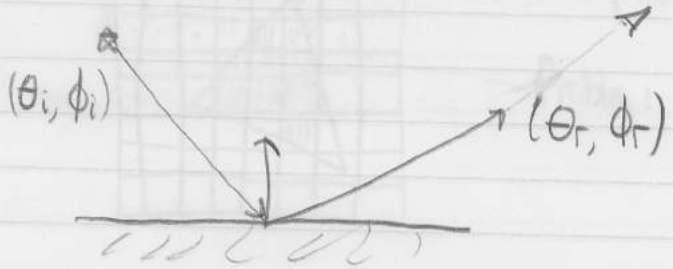
What about multiple reflections?

Global illumination — very expensive
(visibility with mirrors)

Transparency? Recall absorption, refraction, reflection picture from optics

More general model for shading: BRDF

Bidirectional Reflectance Distribution Function



$$f(\theta_i, \phi_i, \theta_r, \phi_r) = \frac{\text{radiance}}{\text{irradiance}}$$

radiance ← reflected energy

irradiance ← received energy

Rotational symmetry → $f'(\theta_i, \theta_r, \phi_i - \phi_r)$ – for physics consistency

Lambertian: Special case with f constant

Problem with VR: Stereo eye views

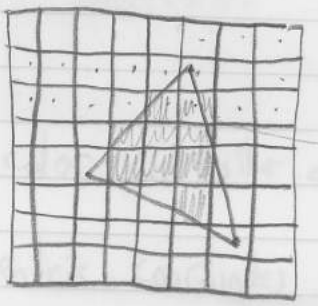
Need to do twice the work for shiny surfaces

Useful adaptations:

Z buffer from light source perspective can generate shadow map (shows parts of object observed by light)

Rasterization

Most common - GPUs
Object order rendering - triangle-by-triangle



Fill in pixels by sampling midpoint

Problem: Aliasing

- Depth ordering problem: Which triangle is in front?

Painters algorithm: Sort by furthest to closest; paint far to near order

Problem:  depth cycle - segment triangles

Problem: Sorting too costly

Z-buffer: Store depth value (z) at each pixel.
Render triangles in arbitrary order
Render pixel only if new z value is less than old one at that pixel

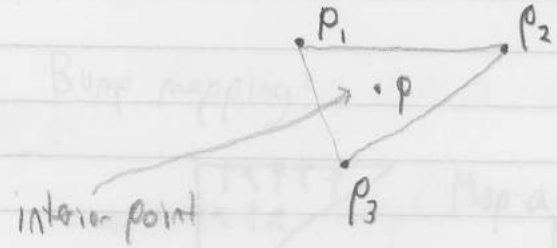
Clipping: Remove triangles (or parts of triangles) that are behind the eye, or too close (vertical clipping plane)

VR problem. Why?

Efficacy! Culling - Eliminate unnecessary rasterizations
Backface: Remove triangles on "back" of object (not in view)
View Frustum: Remove triangles not in viewing frustum

Propagating color, RGB, other attributes efficiently:

Barycentric coordinates



$p_i = (x_i, y_i)$

$p = \alpha_1 p_1 + \alpha_2 p_2 + \alpha_3 p_3$ For $0 \le \alpha_1, \alpha_2, \alpha_3 \le 1$ and $\alpha_1 + \alpha_2 + \alpha_3 = 1$

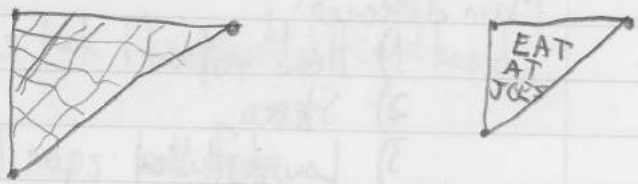
($\alpha_i = 0 \rightarrow$ on edge $\alpha_i, \alpha_j = 0 \rightarrow$ on vertex)

Efficient interpolation
$R = \alpha_1 R_1 + \alpha_2 R_2 + \alpha_3 R_3$
$G = \alpha_1 G_1 + \alpha_2 G_2 + \alpha_3 G_3$
$B = \alpha_1 B_1 + \alpha_2 B_2 + \alpha_3 B_3$
↑ at p_1 ↑ at p_2 ↑ at p_3

Regular Patterns present another resolution problem.
Wavelength and phase issue at boundaries

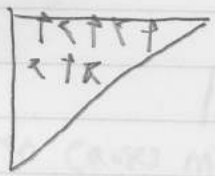
Other mappings over triangles:

Texture mapping (like painting a picture onto surface)



- Look up mipmapping for scaling textures (a form of antialiasing)

Bump mapping:



Map a pattern of normals to surface
Make it look rougher

Normal mapping: Use surface normals from high-resolution surface

→ Make plane look spherical (for example)

→ Use Lambertian shading to make surface look curved.

Main differences:

1) Head motion
2) Stereo
3) Low resolution

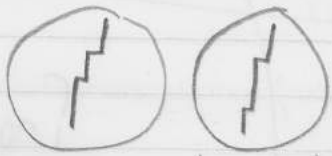
Problems with rendering for VR:

Shading: - Highlights (specularities) need stereo perspective → double work
- Texture maps look like painted cardboard
- Bump/normal maps look fake

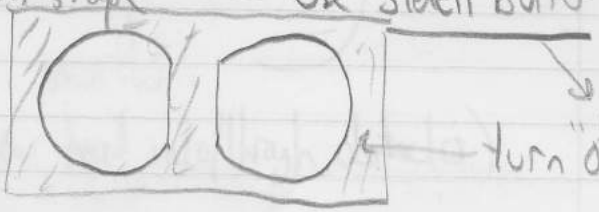
Aliasing: - Resolutions look low

Head motion turns "staircase" into "escalator"

- Stereo causes mismatched stairs per eye



Render target: - Unusual shape - Use stencil buffer in GPU



- Correct for optical distortion due to lenses
pincushion - barrel annihilation

Latency: - GPU pipeline optimized for triangle throughput, not latency.
(triangles/sec)

- Latency compensation by post-rendering image warp (time warp)
Visibility problems for 6DOF case

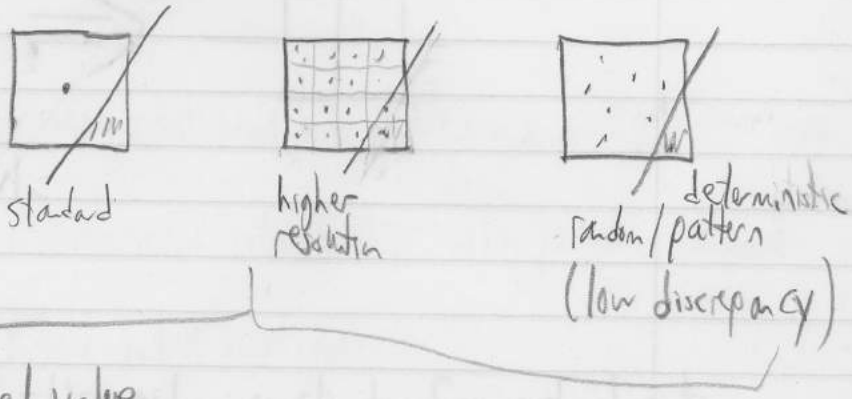
Geometry errors: - Thin objects look fake or implausible (leaves in Tuscany)
- Holes, isolated points more obvious (think about SLAM data)

- chromatic aberration?

- head in walls?
(you can move your head into/through obstacles)

Antialiasing - Get rid of "jaggies"

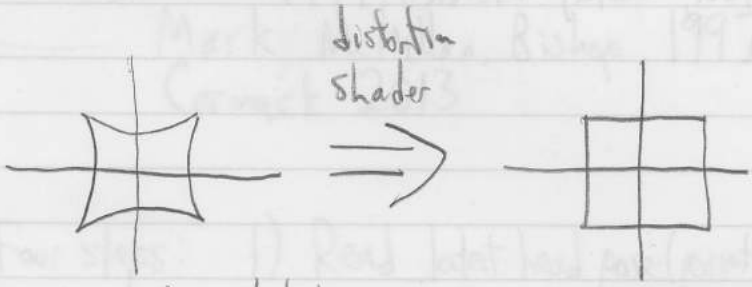
Supersampling

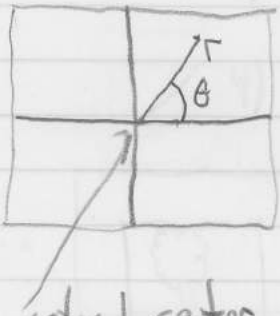


Common special case (hack): MSAA - multisample antialiasing

Only supersample depth (z) and stencil values (in or out of triangle)

Distortion Shading for High POV





- Use polar coordinates from optical center
- Assume rotation invariance for distortion
  (θ does not matter)
  → radially symmetry

$$(r, \theta) \mapsto (f(r), \theta)$$

Compute polynomial barrel distortion

$$f(r) = 1 + c_1 r^2 + c_2 r^4 + c_3 r^6 \ (+ c_4 r^8 + \cdots)$$

Odd powers not needed (theory of optics)

c_1, c_2, c_3 chosen by hand or laser measurements
↓
tune until grid lines straight

Apply $f(r)$ to every pixel to barrel warp image
Pincushion - Barrel annihilation!

Also, Darsa, Costa, Varshney, 1997

Post-rendering image warp (time warp)

Mark, McMillan, Bishop 1997
Carmack 2013

Four steps: 1) Read latest head pose (orientation & position)

How long? 10-50ms?

2) Render scene into buffer using pose for viewpoint

3) Read latest head pose

4) Adjust rendered output to fake the newer viewpoint

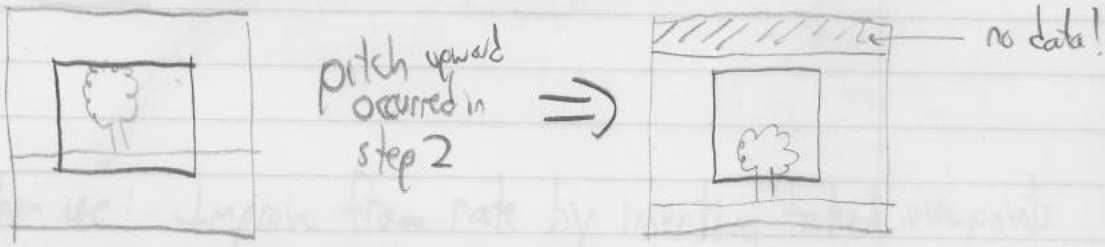


Image shifting accounts for small pitch (vertical) and yaws (horizontal)

Image rotation accounts for small roll.

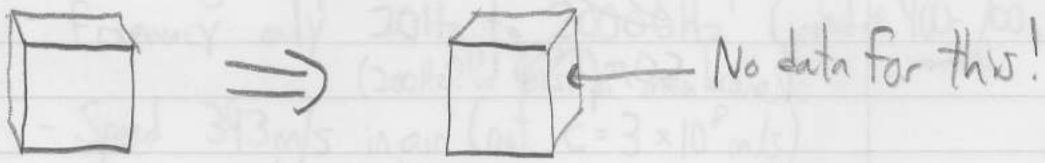
What about position?

Some faking possible:

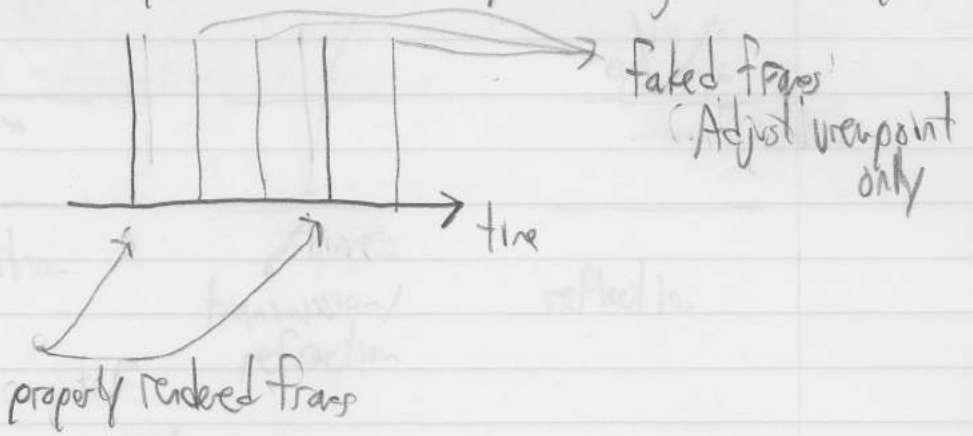
- Rescaling image to account for depth (z) change
- Shift image to account for x, y change

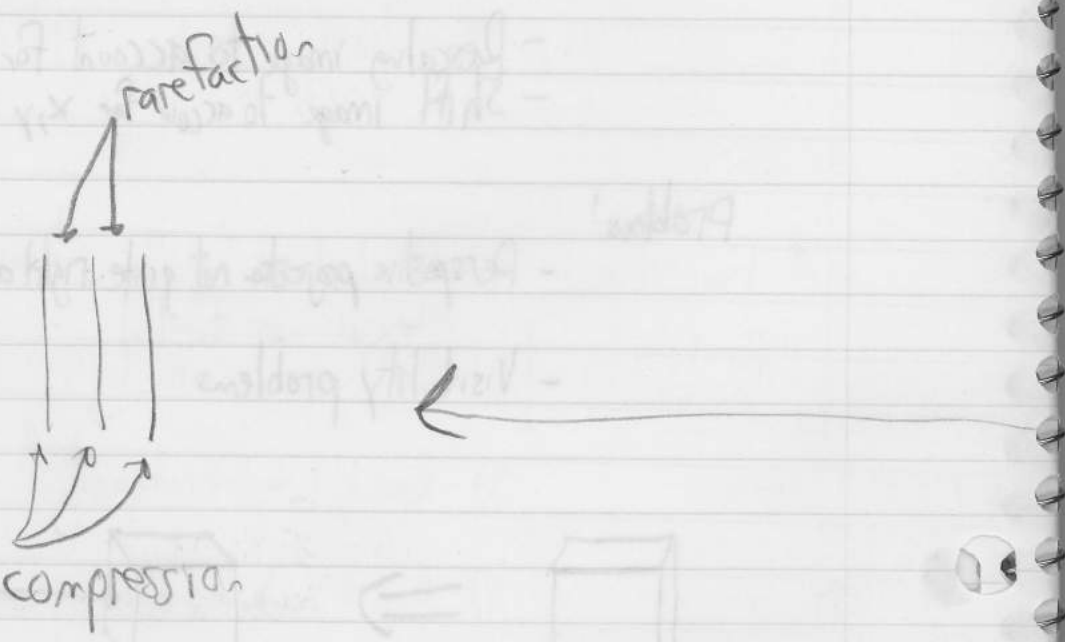
Problems!

- Perspective projects not quite right as z changes (small change OK)
- Visibility problems

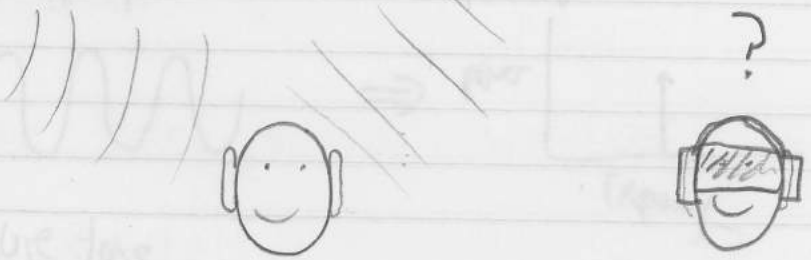


Another use: Improve frame rate by inserting faked viewpoints




rarefaction
compression

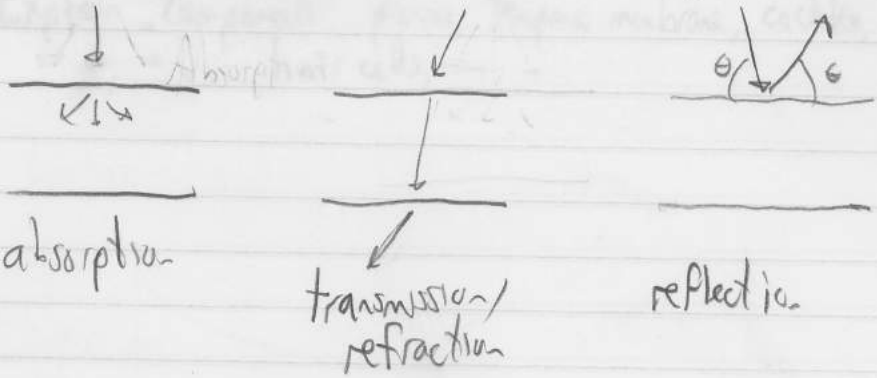
Audio For VR



Sound waves: Similar to light except:

- Fluctuating air pressure (instead of EM/photons) $\frac{8 \times 10^{14}\ Hz}{\text{wavelength}}$ $400\text{-}700\ nm$
- Frequency only 20Hz to 20000Hz (instead of
  (200Hz? in basilar membrane)
- Speed 343m/s in air (not $c = 3 \times 10^8\ m/s$)
  (wavelengths:

Interacting with other media: Just like light

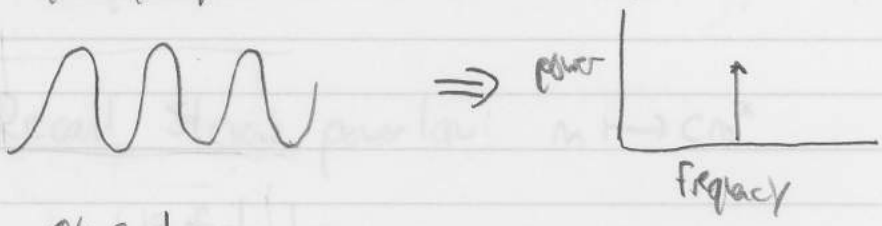


Note: We are not doing lenses, images

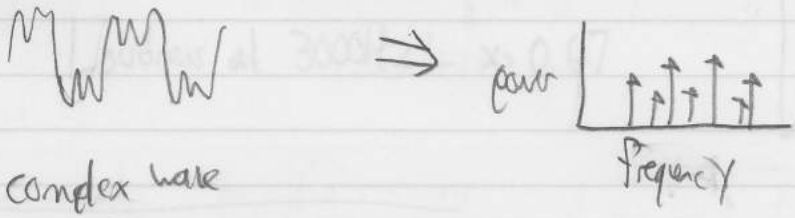
Frequency spectrum : Just like light

ROYGBIV

like in visible spectrum



pure tone



like mixed colors (white)

complex wave

Think filtering, Fourier transforms

Human auditory system

(show slide for pictures - dr 24)

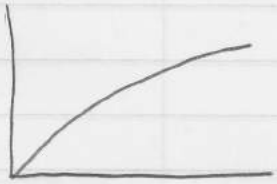
Explain components: pinna, tympanic membrane, cochlea, organ of Corti, hair cells, --

Perception of Sound

Recall Stevens power law: $n \mapsto cn^x$

n = magnitude of stimulus
p = perceived magnitude

Loudness at 3000Hz: $x > 0.67$



Loudness as function of frequency

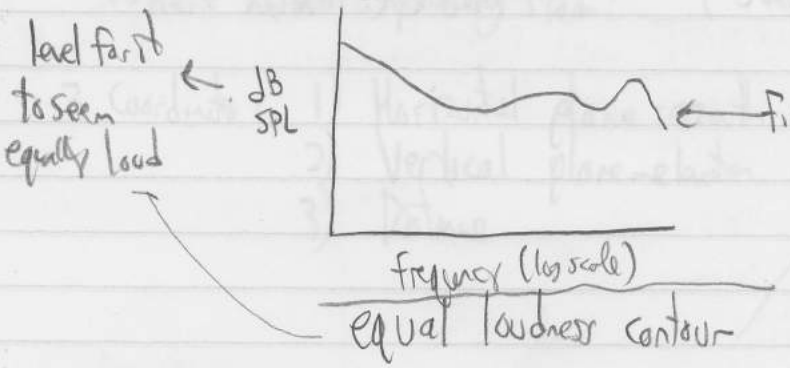
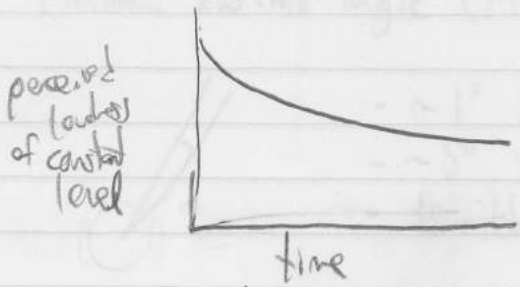


Fig 5.2 in Mather

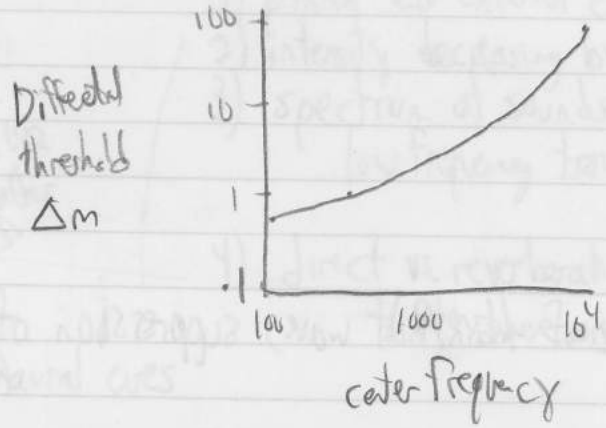
Loudness adaptation



Also, loudness above noise threshold

- ϕ and θ are known for vision because of position on retina, and eye rotation

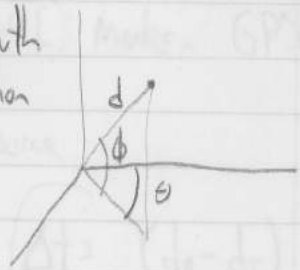
Pitch perception (recall Webers law and JND)



Auditory localization

Where is sound coming from? (Similar to depth perception in vision)

3 Coordinates: 1) Horizontal plane - azimuth
2) Vertical plane - elevation
3) Distance



Minimum audible angle (MAA) — JND

- ~1° below 1000Hz, straight ahead
- ~5° to the side
- terrible around 1500-1900Hz on side

Audio illusion:
Preference of main (first wave) suppression of later wave (reverberated)

vs.

only closer one

both heard

Monaural cues

1) pinna and external ear canal shape
2) intensity decreasing by inverse square law
3) spectrum of sounds
   low frequency travels further: Distant thunder is only low-frequency rumble
4) direct vs. reverberation energy
   (When there are reflecting surfaces)

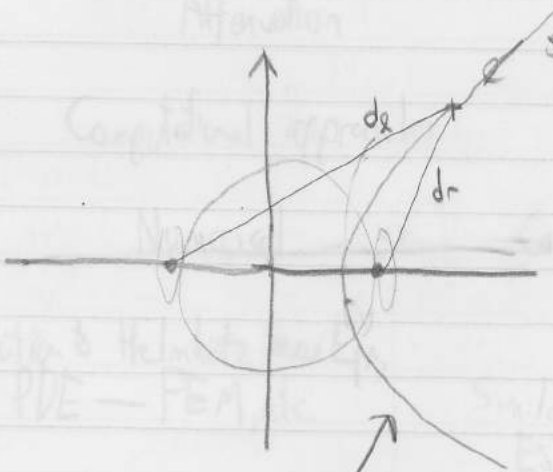
p. 142 Matter

Binaural cues

ILD: Interaural Level Differences – acoustic shadow of ear ("back face")

ITD: Interaural Time Difference – arrival time, distance between ears ~14cm

↳ In sensing: Multilateration from TDOA - time difference of arrival (DECCA system, WWII) Modern GPS)



$$\Delta t = \frac{(d_l - d_r)}{s} < .44\,ms$$

speed of sound ~ 340 m/s

Not sufficient (only ear-to-ear) ← maximum!!!

hyperboloid surface: Cone of Confusion

google for "audio rendering"
UNC talk Anish Chandak (2009)

Sound rendering

Four steps.

1. Modeling

- Geometric models – walls, obstacles
- Acoustic material properties – absorption, reflection, refraction
- Sound sources – point source, parallel wave source, loudness

→ Lower resolution than for graphics (not like pixels)
Simplified

2. Propagation

- Reflection – specular, scattering
- Diffraction
- Refraction
- Doppler effect
- Attenuation

Yes, there is an equivalent to BRDF for audio

Computational approaches

Numerical	vs	Combinatorial
		Visibility, ray-tracing
Solution to Helmholtz wave Eqn. PDE — FEM, etc.		Similar to graphics Estimate wave propagation
Accurate but expensive!		Fast, approximate

Kind of like applying barrel distortion
at the end for visual rendering

3. Rendering

Use head position and orientation to determine pressure signal at right and left ears

Problem: Account for scattering due to:

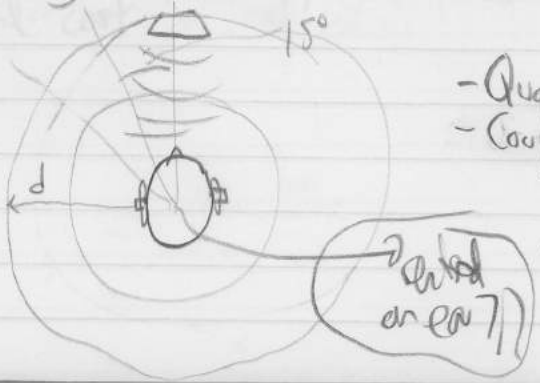
- Outer ears (pinna)
- Head
- Whole body (clothing?)

HRTF (Head-Related Transfer Function) → resolves cone of confusion!

$$H(\omega) = \frac{\text{Output}(\omega)}{\text{Input}(\omega)}$$

Adjust for scattering using linear transfer function (in frequency ω domain)

- Depends on individual ear, head, body shape (hair?)
- Depends on x, y, z of audio source
- Can estimate by recording in anechoic chamber various impulse responses



- Quantize angles, distance
- Could assume "far field" and distance is "at infinity"

$H(\omega, \theta, \phi, d)$ ← distance

or

$H(\omega, \theta, \phi)$ - far field approx

4. Display

Especially one "pixel" display per ear
Time is more important than space
Frequency, phase, amplitudes

! Surround sound (fixed in space) vs. headphones vs earbuds (eliminate pinna altogether)

What about head position & orientation?
Like CAVE

HRTF should account

Problems for developers:

1. How much modeling needed? How to accomplish?
   - Need good middleware
2. How to evaluate correctness or accuracy?
   - Ears not as sensitive as eyes (and eyes already play tricks)
   - HRTFs are person-dependent – like eyeglasses
3. Computational cost

3D UIs Interfaces
Bowman et al., 2005

Include vehicles, grappling, jumping high

Visualization of oneself:
- Torso
- Arms, legs, hands
- Face in mirror

Interfaces for VR

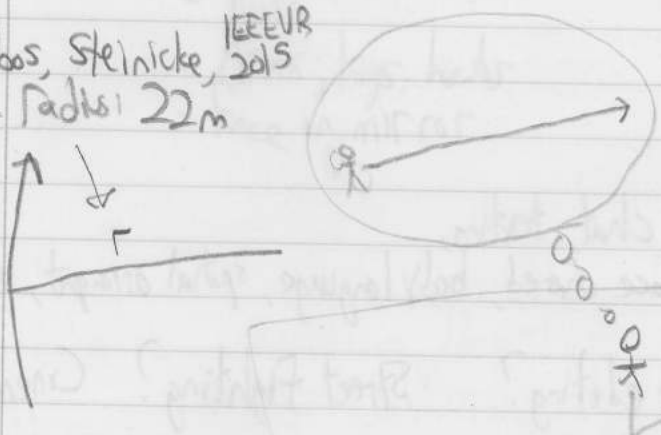
Some categories:

1. Locomotion - Moving oneself from place to place
   Issues: Speed, vestibular mismatch, getting lost
2. Manipulation - Touching, feeling, grabbing, carrying, discarding objects
   Issues: Selection, force feedback, grasping realism, depth perception
3. System Control - Menus, windows, text
   Issues: Readability, comfort, speed
4. Social - chat, texting,
   Issues: face covered, body language, spatial arrangt, uncanny valley
5. Others?? Text editing? Street Fighting? Cinematography?

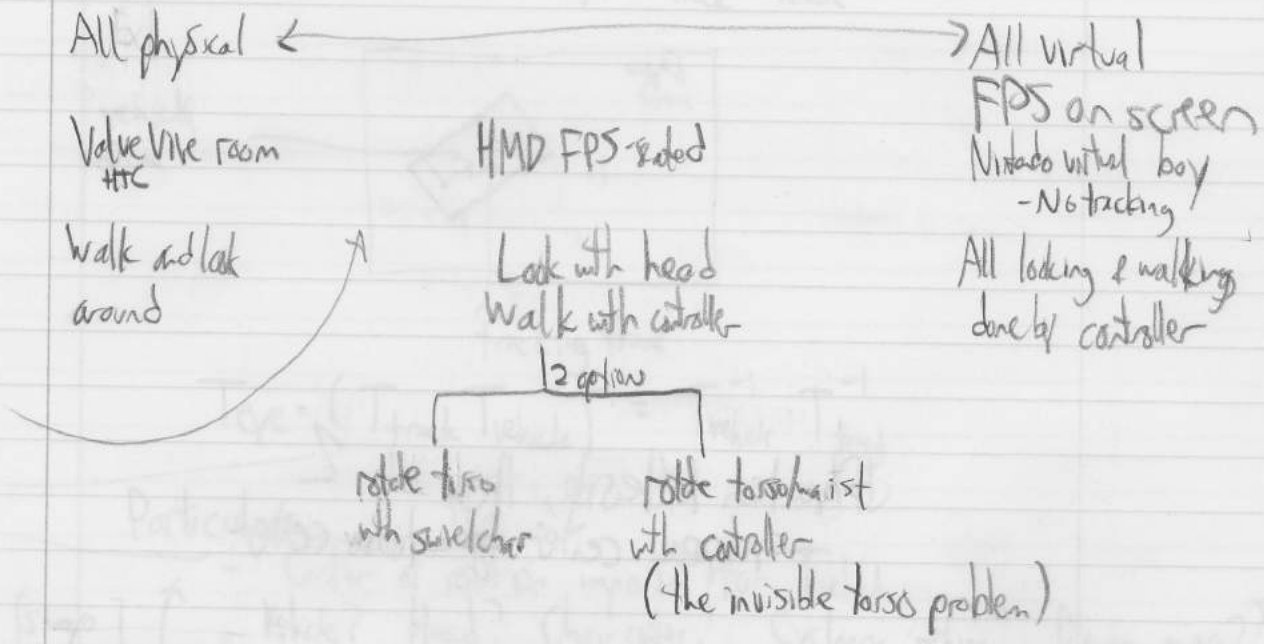
Simulating ANY interface that exists in the physical world!

- Stationary bicycle
- Omnidirectional treadmill

Bruder, Lubos, Steinicke, IEEEVR 2015
Minimum radius: 22m



Locomotion - Traveling in VR



How to apply it:

Recall chain of transforms: $T_{vp}\ T_{can}\ T_{eye}\ T_{rb}$

Consider how T_{eye} is formed from "look at" to get $\hat{x}, \hat{y}, \hat{z}, e$

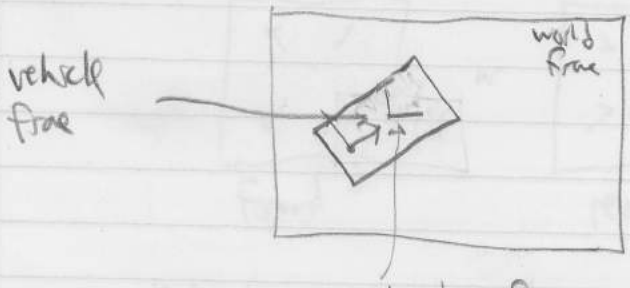
All physical: T_{eye} entirely determined by head tracking
All virtual: T_{eye} " " " controller

Cool trick for all physical walking in unbounded virtual worlds:
Redirected walking (Razzaque, Kohn, Whitton, 2001)
- Distort rotations, bend straight walks
- keep forcing user into center

If you lean to the side, then
cyclopean center ≠ torso center

For HMD FPS:

Lookat frame transformed by $T_{track}\ T_{vehicle}$



$$T_{eye} = (T_{track}\ T_{vehicle})^{-1} = T_{vehicle}^{-1}\ T_{track}^{-1}$$

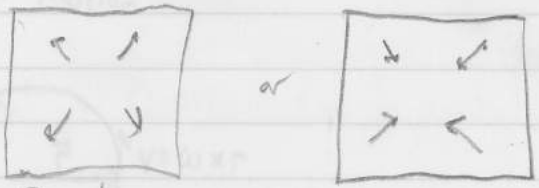
Particulars:

[swap order] ↕

- Center of rotation important for $T_{vehicle}$
  Vehicle? Head? Chair center? Cyclopean retina? Player nose??
- For virtual walking in a fixed chair, $T_{vehicle}$ is equivalent to 2D Rigid body transform
  Provides x, z offset, and yaw direction (torso)
- For free-flying ship, $T_{vehicle}$ is full 3D rigid body transform

Problems with vestibular mismatch (vection)

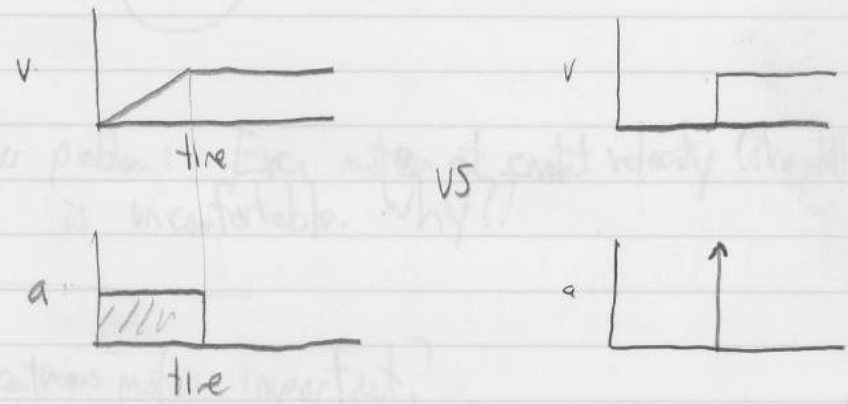
- Linear acceleration in VR causes perceived acceleration from visual



Note: Arrows <u>not</u> the accel. vectors

Vestibular organ knows there is no motion

Two strategies for initiating a walk



#1 #2

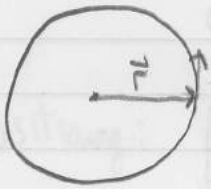
#2 is more comfortable because the brain rejects the brief, large mismatch as outlier

---

Idea! Do the same for rotations (Windlands)
Discrete jumps for orientation change
Too large jumps → confusion about new direction

- Angular motion, even when constant, induces motion

Circular motion



$$\vec{a} = \omega \times \vec{V} = \omega \times (\omega \times \vec{r})$$

< ex Beware!

Further problem: Even motion at constant velocity (straight line, constant speed) is uncomfortable. Why??

Is continuous motion important?
Depends on application — usually not important!
Teleporting - great for large areas; possible location confusion
VR Cinema - pick your seat from seating chart

Use a map or WIM (World In Miniature)

Issue! Wayfinding? — Learning a virtual city or forest
— Visual landmarks important — like real life!

Manipulation

Factors

1) Selection: Distance and direction to object
Size
Clutter around object
Occlusion (maybe partial)

2) Positioning: Distance and direction of initial and goal
Positioning precision requirement
Does goal have docking or basin of attraction?

3) Rotation: Amount and precision of rotation.
All 3 DOFs?

Input device categories:
1) Metric — Motions are tracked through space; consider DOFs
Desk mouse, air mouse (gyro, 3D), Leap Motion, gloves
2) Switch — Motions induced by pressing, pushing
Joystick, buttons

Even Atari 2600 (1977) had both: Metric was paddle  and Switch was joystick 

Which one is artificial steering wheel?

Could be either.

see Bowman et al book

Note that both have <u>scaling issues</u> over space and/or time

- How far to move mouse?
- How long to hold button?
- How much to rotate in a tap of joystick?

- Perceptual issue: Learnable motor program
- Precision issue: Can a high precision task be accomplished?
  Like press ☒ on window corner w/mouse
- Note: No vestibular mismatch with scaling for manipulation!

Selection idea: Do ray casting to point at objects - like laser pointer

Manipulation idea: Make a virtual hand on an extendable arm

Final advice: Reduce DOFs as much as possible in task

General issue: Haptic feedback
Rumble
Pressure/force

Gloves
Manipulator robot hand
3D printed objects

Important application: Medical simulation or telesurgery

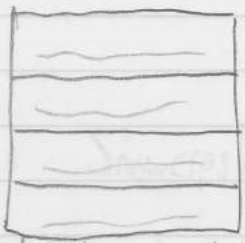
See Oculus BPG and
Bowman et al.

Got here? Or somewhere else?

System Control

Sending commands to the application

Most common: Graphical menu
(Others: Voice command, gesture command)



How to select? Head looking, controller, --

Guidelines:

- Embed menu in virtual world — like a billboard
  (Do not attach to face in front of eyes)
- Lazily pulling menu into looking direction is OK
- Distance in virtual world should be 2 to 3 meters away
- Fit inside middle 1/3 of viewing area — avoid head movements to read
- Embed info into the world — ammo count appears on gun

What about text entry?
Keyboard difficult
Split keyboard with separate
Glove) Chorded keyboard

Social Interaction

Virtual communities - sociology
↳ Ex book by Howard Rheingold
Includes Plato, chatrooms, email lists, Usenet, MUDs
multi user dungeon

Specialcase:
Virtual worlds

Sciencefiction: Snowcrash, Ready Player One, Neuromancer??

MMORPGs: Massively multiplayer online roleplaying games
teamwork!

SecondLife (OpenSimulator)
An entire economy formed - LindenDollars
GDP $64 million in 2009 - like Bitcoins
No government - damage, theft hard to achieve

Face-to-face interaction:

Avatars / Text	vs / vs	real faces / real audio

↗ Allows fantasy, alterations

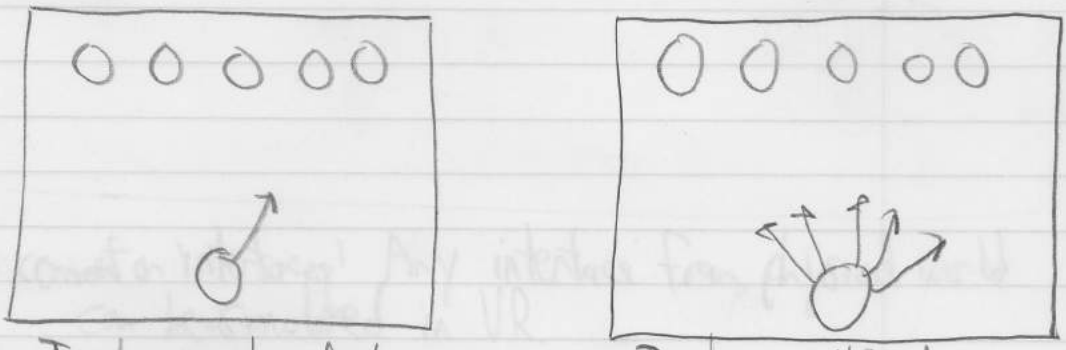
↑ Approaches reality

Achieving realism

- Use omnidirectional video camera
- Is person in real or placed into virtual world? Extracting problem
- What if an HMD occludes their face? Camera inside??
- Effect of delays – at least synchronize audio + video

With avatars, achieving the impossible (Transformed Social Interaction)

- Change gender, "race", species
- Change height, weight, eye color, hair color
- Empathy can be taught!!
  Police training videos
- Look at multiple people simultaneously (Bailenson et al 04)



Teacher in physical classroom Teacher in virtual classroom

In VR the teacher can look at every student simultaneously because the each get a unique world on their HMD

Also, if students aren't looking at you, then their avatar fades away

Bringing realism back into the avatar!

- Eye tracking (blinks, winks?)
  (More latency allowed than for foveated rendering)
- Face modeling, animating, and tracking
  See Paul Debevec, USC
- Beware the uncanny valley!
  People get creeped out
- Hand and body tracking
  Poor tracking worse than none at all (eg. Lego master, Kinect can be poor)
  Can communicate body language

Final comment on interfaces! Any interface from physical world can be simulated in VR.

Hence, a VR interface can be simulated in VR!
And on and on ——

(Put on a virtual HMD while already in VR)

Like the levels in Inception