Lecture 23 of 41

More Rotations; Visualization, Simulation Videos 4: Virtual & Augmented Reality, Viz-Sim

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

KSOL course pages: http://bit.ly/hGvXlH / http://bit.ly/eVizrE
Public mirror web site: http://www.kddresearch.org/Courses/CIS636
Instructor home page: http://www.cis.ksu.edu/~bhsu

Readings:
Today: Chapter 10, 13, §17.3 – 17.5, Eberly 2e – see http://bit.ly/ieUq45
Next class: §2.4.3, 8.1, Eberly 2e, GL handout
Wikipedia, *Visualization*: http://bit.ly/gVxRFp
Wikipedia on quaternions: http://bit.ly/f1GvTS, http://bit.ly/eBnCY4
Reference: Ogre Wiki quaternion primer – http://bit.ly/hv6zv0

Lecture Outline

- Reading for Last Class: §17.1 – 17.2, Eberly *2e*
- Reading for Today: Chapter 10, 13, §17.3 – 17.5, Eberly *2e*
- Reading for Next Class: §2.4.3, 8.1, Eberly *2e*, GL handout
- Last Time: Rotations in Animation
  - Flight dynamics: roll, pitch, yaw
  - Matrix, angles (fixed, Euler, axis), quaternions, exponential maps
- Quaternions Concluded
  - How quaternions work – properties (review)
    - Equivalent rotation matrix (RM)
    - Quaternion arithmetic
    - Composition of rotations by quaternion multiplication
  - Advantage: easy incremental rotation; camera, character animation
- Today: Intro to Visualization, Modeling & Simulation
  - Virtual reality (VR), virtual environments (VE)
  - Augmented reality (AR)

Where We Are

21	**Lab 4a: Animation Basics**	**Flash animation handout**
22	Animation 2: Rotations; Dynamics, Kinematics	Chapter 17, esp. §17.1 – 17.2
23	Demos 4: Modeling & Simulation; Rotations	Chapter 10[1], 13[2], §17.3 – 17.5
24	**Collisions 1: axes, OBBs, Lab 4b**	**§2.4.3, 8.1, GL handout**
25	Spatial Sorting: Binary Space Partitioning	Chapter 6, esp. §6.1
26	**Demos 5: More CGA; Picking; HW/Exam**	**Chapter 7[2]; § 8.4**
27	**Lab 5a: Interaction Handling**	**§ 8.3 – 8.4; 4.2, 5.0, 5.6, 9.1**
28	Collisions 2: Dynamic, Particle Systems	§ 9.1, particle system handout
	Exam 2 review; Hour Exam 2 (evening)	**Chapters 5 – 6, 7[2] – 8, 12, 17**
29	**Lab 5b: Particle Systems**	**Particle system handout**
30	**Animation 3: Control & IK**	**§ 5.3, CGA handout**
31	Ray Tracing 1: intersections, ray trees	Chapter 14
32	Lab 6a: Ray Tracing Basics with POV-Ray	RT handout
33	Ray Tracing 2: advanced topic survey	Chapter 15, **RT handout**
34	Visualization 1: Data (Quantities & Evidence)	Tufte handout (1)
35	**Lab 6b: More Ray Tracing**	**RT handout**
36	Visualization 2: Objects	Tufte handout (2 & 4)
37	Color Basics; Term Project Prep	**Color handout**
38	Lab 7: Fractals & Terrain Generation	**Fractals/Terrain handout**
39	**Visualization 3: Processes; Final Review 1**	Tufte handout (3)
40	**Project presentations 1; Final Review 2**	**–**
41	Project presentations 2	–
	Final Exam	**Ch. 1 – 8, 10 – 15, 17, 20**

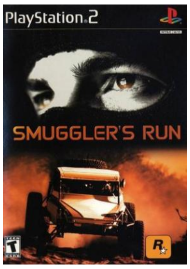
Lightly-shaded entries denote the due date of a written problem set; heavily-shaded entries, that of a machine problem (programming assignment); blue-shaded entries, that of a paper review; and the green-shaded entry, that of the term project.

Green, **blue** and **red** letters denote **exam review**, **exam**, and **exam solution review** dates.

Acknowledgements: CGA Rotations, Dynamics & Kinematics

Rick Parent
Professor
Department of Computer Science and Engineering
Ohio State University
http://www.cse.ohio-state.edu/~parent/

David C. Brogan
Visiting Assistant Professor, Computer Science Department, University of Virginia
http://www.cs.virginia.edu/~dbrogan/
Susquehanna International Group (SIG)
http://www.sig.com



Steve Rotenberg
Visiting Lecturer
Graphics Lab
University of California – San Diego
CEO/Chief Scientist, PixelActive
http://graphics.ucsd.edu

Review [1]: Representing 3 Rotational DOFs

3x3 Matrix (9 DOFs)

- Rows of matrix define orthogonal axes

Euler Angles (3 DOFs)

- Rot x + Rot y + Rot z

Axis-angle (4 DOFs)

- Axis of rotation + Rotation amount

Quaternion (4 DOFs)

- 4 dimensional complex numbers

University of Virginia

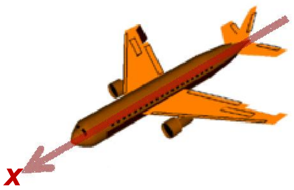
Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Computer Science at the University of Virginia

Review [2]: Method 1
Rotation Matrices – Roll, Pitch, & Yaw

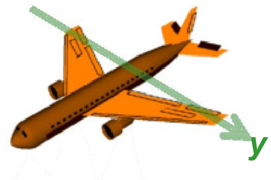
Rotation about *x* axis (Roll)

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) & 0 \\ 0 & \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Rotation about *y* axis (Pitch)

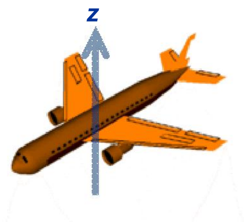
$$\begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



)06 Wikipedia,
ght Dynamics
/bit.ly/gVaQCX

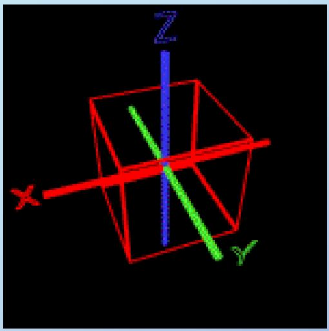
Rotation about *z* axis (Yaw)

$$\begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

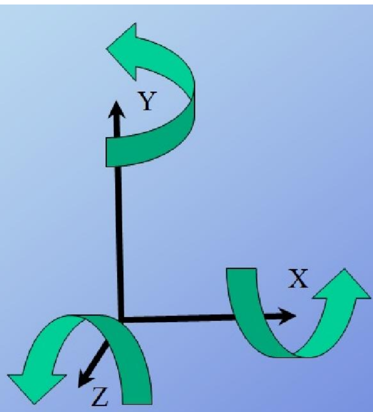


Adapted from slides ♥ 2007 – 2011 R. Parent, Ohio State University
CSE 682 (Computer Animation), http://bit.ly/feUESy
CSE 683/684A (Computer Animation Algorithms & Techniques), http://bit.ly/f8Myky

Review [3]: Method 2
Fixed Angles & Gimbal Lock



Anticz.com © 2001 M. Brown
http://bit.ly/6NIXVr



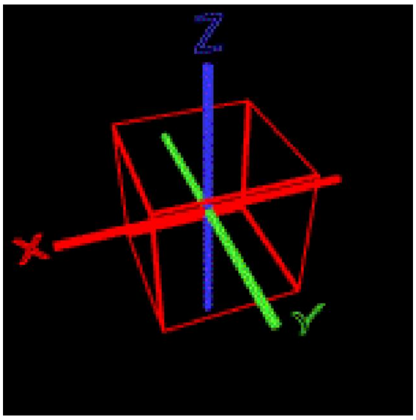
$$\begin{pmatrix}\alpha & \beta & \gamma\end{pmatrix} \longrightarrow P' = R_z(\gamma)R_y(\beta)R_x(\alpha)P$$

Fixed order: e.g., x, y, z; also could be x, y, x
Global axes

Adapted from slides ♥ 2007 – 2011 R. Parent, Ohio State University
CSE 682 (Computer Animation), http://bit.ly/feUESy
CSE 683/684A (Computer Animation Algorithms & Techniques), http://bit.ly/f8Myky

Gimbal Lock Illustrated [1]

- Gimbal Lock: Loss of DOF when 2 of 3 Gimbals Driven until Parallel
- Animated Examples
  - *e.g.*, *x* & *z* (left), *y* & *z* (right)
  - Caution: Seefeld (right) refers to these as "*x*" (red) & *z* (blue)
  - *y* (Pitch) = "x", *x* (Roll) = "*y*", *z* (Yaw) = "z" ("zed")

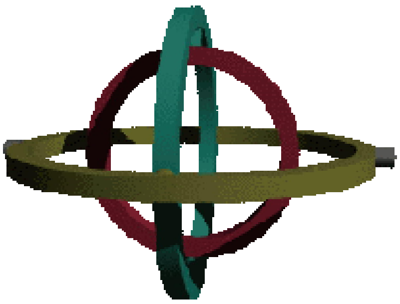


Anticz.com © 2001 M. Brown
http://bit.ly/6NIXVr

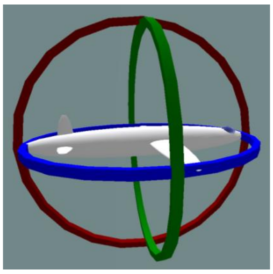
© 2007 S. Seefeld
http://bit.ly/e1nuo9

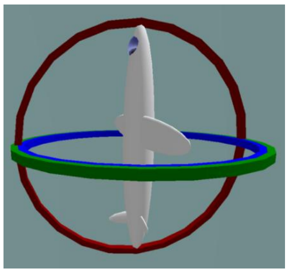
Gimbal Lock Illustrated [2]

- Gimbals: 2 of 3 Driven into Parallel Configuration
- Happens With Euler Angles Too: http://bit.ly/g32DQ5 (Wikipedia)
- Solution Approaches
  - Extra gimbal
  - Quaternions: $(\cos(\theta/2),\ x_0 \cdot \sin(\theta/2),\ y_0 \cdot \sin(\theta/2),\ z_0 \cdot \sin(\theta/2)).\quad \vec{N} = (x_0, y_0, z_0)$



Gimbal Lock figure © 2006 Wikipedia
(Rendered using POV-Ray)
http://bit.ly/hR88V2





Left: not locked
Right: *x* & *z* rotations locked (roll & pitch, no yaw)
Gimbal Lock figures © 2009 Wikipedia
http://bit.ly/he0LN9

Review [4]: Method 3
Euler Angles & Order Independence

$(\theta_x, \theta_y, \theta_z) = R_z R_y R_x$

- Rotate θ_x degrees about x-axis
- Rotate θ_y degrees about y-axis
- Rotate θ_z degrees about z-axis

Axis order is not defined

- (y, z, x), (x, z, y), (z, y, x)… are all legal
- Pick one

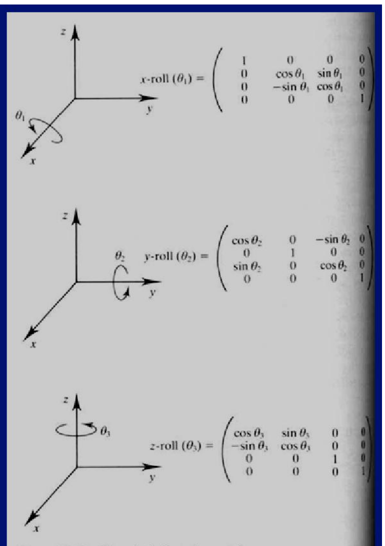


Figure 15.15 The principal rotation matrices.

Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Computer Science at the University of Virginia

Review [5]: Euler Angle Sequences

- This means that we can represent an orientation with 3 numbers
- A sequence of rotations around principal axes is called an *Euler Angle Sequence*
- Assuming we limit ourselves to 3 rotations without successive rotations about the same axis, we could use any of the following 12 sequences:

XYZ	XZY	XYX	XZX
YXZ	YZX	YXY	YZY
ZXY	ZYX	ZXZ	ZYZ

Adapted from slides ♥ 2004 – 2005 S. Rotenberg, UCSD
CSE169: Computer Animation, Winter 2005, http://bit.ly/f0ViAN

Using Euler Angles [1]: Representing Orientations

- This gives us 3! + C(3, 2) * 2 = 6 + 3 * 2 = 12 redundant ways to store an orientation using Euler angles
- Different industries use different conventions for handling Euler angles (or no conventions)

UCSD

Using Euler Angles [2]: Conversion: Euler Angle to RM

- To build a matrix from a set of Euler angles, we just multiply a sequence of rotation matrices together:

$$\mathbf{R}_x \cdot \mathbf{R}_y \cdot \mathbf{R}_z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_x & s_x \\ 0 & -s_x & c_x \end{bmatrix} \cdot \begin{bmatrix} c_y & 0 & -s_y \\ 0 & 1 & 0 \\ s_y & 0 & c_y \end{bmatrix} \cdot \begin{bmatrix} c_z & s_z & 0 \\ -s_z & c_z & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} c_y c_z & c_y s_z & -s_y \\ s_x s_y c_z - c_x s_z & s_x s_y s_z + c_x c_z & s_x c_y \\ c_x s_y c_z + s_x s_z & c_x s_y s_z - s_x c_z & c_x c_y \end{bmatrix}$$

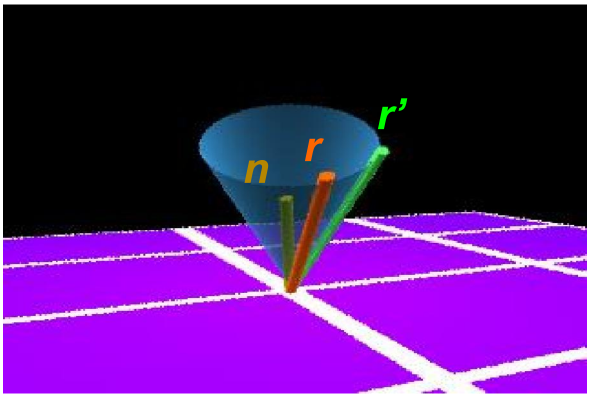
Review [6]: Method 4
Axis-Angle: Specification

Given

- *r* – vector in space to rotate
- *n* – unit-length axis in space about which to rotate
- ❑ –amount about *n* to rotate

Solve

- *r'* – rotated vector



Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Review [7]: Method 5
Quaternions to RM, Axis-Angle

$$Rot_{[s \quad x \quad y \quad z]} = \begin{bmatrix} 1-2y^2-2z^2 & 2xy-2sz & 2xz-2sy \\ 2xy-2sz & 1-2x^2-2z^2 & 2yz-2sx \\ 2xz-2sy & 2yz-2sx & 1-2x^2-2y^2 \end{bmatrix}$$

Axis-Angle

$$\begin{cases} \theta = 2\cos^{-1}(s) \\ (x, y, z) = v / \|v\| \end{cases}$$

Adapted from slides ♥ 2007 – 2011 R. Parent, Ohio State University
CSE 682 (Computer Animation), http://bit.ly/feUESy
CSE 683/684A (Computer Animation Algorithms & Techniques), http://bit.ly/f8Myky

Quaternions [1]: Basic Idea

Remember complex numbers: a + ib

- Where $i^2 = -1$

Invented by Sir William Hamilton (1843)

- Remember Hamiltonian path from Discrete Math?

Quaternion:

- $Q = a + bi + cj + dk$
  - Where $i^2 = j^2 = k^2 = -1$ and $ij = k$ and $ji = -k$
- Represented as: $q = (s, v) = s + v_x i + v_y j + v_z k$

Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Computer Science at the University of Virginia

Quaternions [2]: Definition

A quaternion is a 4-D unit vector q = [x y z w]

- It lies on the unit hypersphere $x^2 + y^2 + z^2 + w^2 = 1$

For rotation about (unit) axis v by angle θ

- vector part = (sin θ/2) *v* = *[x y z]*
- scalar part = (cos θ/2) = *w*
- $(\sin(\theta/2)\ n_x,\ \sin(\theta/2)\ n_y,\ \sin(\theta/2)\ n_{z,}\ \cos(\theta/2))$

Only a unit quaternion encodes a rotation - normalize

Quaternions [3]: Equivalent RM & Composition

Rotation matrix corresponding to a quaternion:

- [x y z w] = $\begin{bmatrix} 1-2y^2-2z^2 & 2xy+2wz & 2xz-2wy \\ 2xy-2wz & 1-2x^2-2z^2 & 2yz+2wx \\ 2xz+2wy & 2yz-2wx & 1-2x^2-2y^2 \end{bmatrix}$

Quaternion Multiplication

- $q_1 * q_2 = [v_1, w_1] * [v_2, w_2] = [(w_1 v_2 + w_2 v_1 + (v_1 \times v_2)), w_1 w_2 - v_1 . v_2]$
- quaternion * quaternion = quaternion
- this satisfies requirements for mathematical *group*
- Rotating object twice according to two different quaternions is equivalent to one rotation according to product of two quaternions

Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Quaternions [4]: Examples

X-roll (roll) of π

- $(\cos(\pi/2), \sin(\pi/2)(1, 0, 0)) = (0, (1, 0, 0))$

Y-roll (pitch) of π

- $(0, (0, 1, 0))$

Z-roll (yaw) of π

- $(0, (0, 0, 1))$

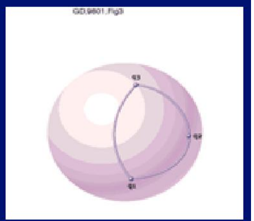
$R_y(\pi)$ followed by $R_z(\pi)$

- $(0, (0, 1, 0))$ times $(0, (0, 0, 1)) = (0, (0, 1, 0) \times (0, 0, 1)$
  $= (0, (1, 0, 0))$

Quaternions [5]: Interpolation

Biggest advantage of quaternions

- Interpolation
- Cannot linearly interpolate between two quaternions because it would speed up in middle
- Instead, Spherical Linear Interpolation, slerp()
- Used by modern video games for third-person perspective
- Why?



Hint: see http://youtu.be/-jBKKV2V8eU

Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Computer Science at the University of Virginia

Quaternions [6]: Spherical Linear Interpolation (SLERP)

Quaternion is a point on the 4-D unit sphere

- interpolating rotations requires a unit quaternion at each step
  - another point on the 4-D unit sphere
- move with constant angular velocity along the great circle between two points

Any rotation is defined by 2 quaternions, so pick the shortest SLERP

To interpolate more than two points, solve a non-linear variational constrained optimization

- Ken Shoemake in SIGGRAPH '85 (www.acm.org/dl)

Adapted from slides ♥ 2000 – 2004 D. Brogan, University of Virginia
CS 445/645, Introduction to Computer Graphics, http://bit.ly/h9AHRg

Quaternions [7]: Comparison with Euler Interpolation

- ***Quaternion (white) vs. Euler (black) interpolation***
- ***Left images are linear interpolation***
- ***Right images are cubic interpolation***

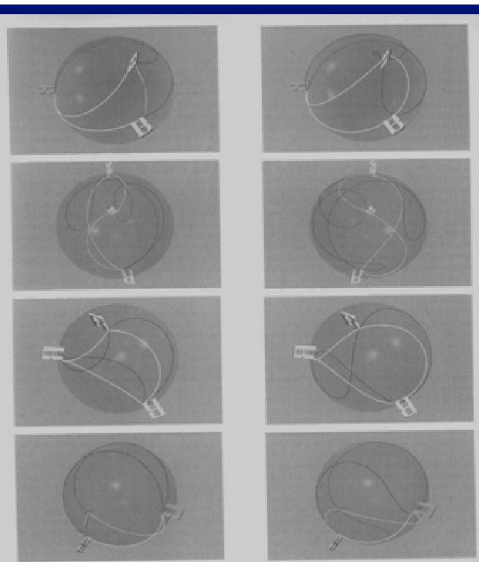


Figure 15.25 Shows how R moves through the three keys. In all cases the white line tracks the motion of R when the interpolation is carried out in quaternion space; the black line tracks the motion of R when Euler angles are interpolated. In each row the left illustration compares linear interpolation of Euler angles with spherical linear interpolation of quaternions. In each row the right illustration compares a cubic spline interpolation of Euler angles to the spherical cubic spline interpolation of quaternions (using squad()).

Quaternions [8]: Code

- Gamasutra (1998): http://bit.ly/dQy8Cp
- Nate Robins's Implementation: http://bit.ly/fcGufq
  - File `gltb.c`
  - `gltbMatrix`
  - `gltbMotion`

Spherical Interpolation [1]: Spheres

- Think of a person standing on the surface of a big sphere (like a planet)
- From the person's point of view, they can move in along two orthogonal axes (front/back) and (left/right)
- There is no perception of any fixed poles or longitude/latitude, because no matter which direction they face, they always have two orthogonal ways to go
- From their point of view, they might as well be moving on a infinite 2D plane, however if they go too far in one direction, they will come back to where they started!

Adapted from slides ♥ 2004 – 2005 S. Rotenberg, UCSD
CSE169: Computer Animation, Winter 2005, http://bit.ly/f0ViAN

Spherical Interpolation [2]: Hyperspheres

- Now extend concept to moving in *hypersphere* of unit quaternions
- Now have three orthogonal directions to go
- No matter how oriented in this space, can always go some combination of forward/backward, left/right and up/down
- Go too far in any direction: back to start point
- Location on unit hypersphere: orientation
- Moving in arbitrary direction corresponds to rotating around some arbitrary axis

Adapted from slides ♥ 2004 – 2005 S. Rotenberg, UCSD
CSE169: Computer Animation, Winter 2005, http://bit.ly/f0ViAN

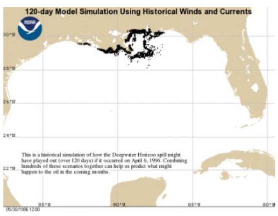
Review [8]: Dynamics & Kinematics

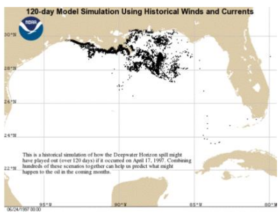
- **Dynamics**: Study of Motion & Changes in Motion
  - Forward: model forces over time to find state, *e.g.*,
    - Given: initial position p_0, velocity v_0, gravitational constants
    - Calculate: position p_t at time t
  - Inverse: given state and constraints, calculate forces, *e.g.*,
    - Given: *desired* position p_t at time t, gravitational constants
    - Calculate: position p_0, velocity v_0 needed
  - Wikipedia: http://bit.ly/hH43dX (see also: "Analytical dynamics")
  - For non-particle objects: rigid-body dynamics (http://bit.ly/dLvejg)
- **Kinematics**: Study of Motion without Regard to Causative Forces
  - Modeling systems – *e.g.*, articulated figure
  - Forward: from angles to position (http://bit.ly/eh2d1c)
  - Inverse: finding angles given desired position (http://bit.ly/hsyTb0)
  - Wikipedia: http://bit.ly/hr8r2u

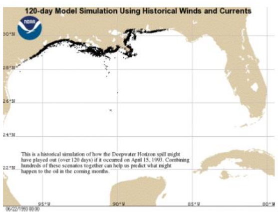
Forward Kinematics
© 2009 Wikipedia

Visualization [1]: Animating Simulations

Deepwater Horizon Oil Spill (20 Apr 2010)
http://bit.ly/9QHax4
120-day images © 2010 NOAA, http://1.usa.gov/c02xuQ







120-day simulation using
15 Apr 1993 weather conditions

132-day simulation using
2010 conditions
© 2010 National Center for Supercomputing Applications (NCSA)
http://youtu.be/pE-1G_476nA

Wilhelmson *et al.* (2004)
http://youtu.be/EgumU0Ns1YI
http://avl.ncsa.illinois.edu
http://bit.ly/eA8PXN

Visualization [2]: Virtual Reality (VR)

- Virtual Reality: Computer-Simulated Environments
- Physical Presence: Real & Imaginary
- Hardware: User Interface
  - Head-mounted display (HMD), gloves – see PopOptics goggles (left)
  - VR glasses, wand, *etc.* – see NCSA CAVE (right)

***Virtual Reality*, Wikipedia:**
http://bit.ly/fAvNeP
Image © 2007 National Air & Space Museum

CAVE (Cave Automatic Virtual Environment)
Image © 2009 D. Pape
HowStuffWorks article: http://bit.ly/feQxNK
© 2009 J. Strickland
Wikipedia: http://bit.ly/dKNEnU

Visualization [3]: Virtual Environments (VE)

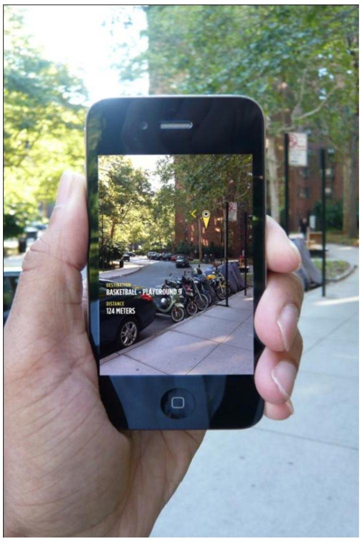
- Virtual Environment: Part of Virtual Reality Experience
- Other Parts
  - Virtual artifacts (VA): simulated objects – http://bit.ly/hskSyX
  - Intelligent agents, artificial & real – http://bit.ly/y2gQk

Experientia © 2006 M. Vanderbeeken *et al.*, http://bit.ly/hzfAQx
Second Life © 2003 – 2011 Linden Labs, Inc., http://bit.ly/wbvoL
Image © 2006 Philips Design

We Are Arcade © 2011 D. Grossett *et al.*, http://bit.ly/ftALjU
World of Warcraft: Cataclysm review © 2011
J. Greer, http://bit.ly/eENHXt
World of Warcraft © 2001 – 2011
Blizzard Entertainment, Inc.,
http://bit.ly/2qvPYF

Visualization [4]: Augmented Reality (AR)

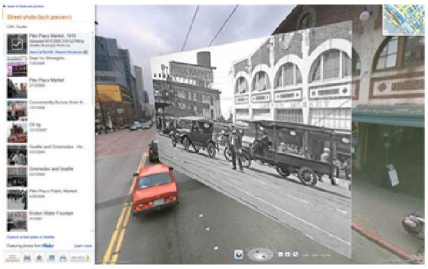
- **Augmented Reality: Computer-Generated (CG) Sensory Overlay**
- **Added to Physical, Real-World Environment**



"40 Best Augmented Reality iPhone Applications", © 2010 iPhoneNess.com, http://bit.ly/2qT35y
MyNav © 2010 Winfield & Co. http://bit.ly/dLTir7

Google goggles labs

Wikipedia, *Google Goggles*: http://bit.ly/gRRMLS



Bing Maps © 2010 – 2011
Microsoft Corporation
http://bit.ly/a9UviT
© 2010 TED Talks

Summary

- Reading for Last Class: §17.1 – 17.2, Eberly *2^e*
- Reading for Today: Chapter 10, 13, §17.3 – 17.5, Eberly *2^e*
- Reading for Next Class: §2.4.3, 8.1, Eberly *2^e*, GL handout
- Last Time: Rotations in Animation
  - Matrix, fixed angles, Euler angles, axis
  - Quaternions & how they work – properties, arithmetic operations
  - Gimbal lock defined & illustrated
- Quaternions Concluded
  - Incremental rotation: spherical linear interpolation (slerping)
  - Advantages of slerping *vs.* cubic interpolation between Euler angles
  - Uses: character animation, camera control (rotating Look vector)
- Dynamics & Kinematics (Preview of Lectures 28 – 30)
- Today: Modeling & Simulation
  - Virtual / augmented reality (VR/AR) & virtual environments (VE)
  - Visualization & simulation (Viz-Sim) preview

Terminology

- Last Time: Rotation using Matrices, Fixed Angles, Euler Angles
- Gimbal Lock
  - Loss of DOF
  - Reference (© 2007 S. Seefield): http://bit.ly/e1nuo9
- Axis-Angle – Rotate Reference Vector *r* about Arbitrary Axis (Vector) *A*/*n*
- Quaternions
  - Quaternions – different representation of arbitrary rotation
  - Exponential maps – 3-D representation related to quaternions
- Visualization – Communicating with Images, Diagrams, Animations
- Simulation – Artificial Model of Real Process for Answering Questions
- VR, VE, VA, AR
  - Virtual Reality: computer-simulated environments, objects
  - Virtual Environment: part of VR dealing with surroundings
  - Virtual Artifacts: part of VR dealing with simulated objects
  - Augmented Reality: CG sensory overlay on real-world images