Computergrafik 1

Andreas Butz

Ludwig-Maximilians-Universität München

Sommersemester 2015

Chapter 1 - Introduction, Motivation, Basics

- About this Class: Organization
- Tutorials
- Why Should I Learn about Computer Graphics?
- Very Brief History of Computer Graphics
- Math Recap: What We Need to Survive...

About this class: Organization

- Mainly Bachelor Medieninformatik, 4th semester
- „Vertiefende Themen“ in Bachelor Informatik + MI
- also Bachelor Kunst und Multimedia
- All others, please check how course can be counted

image source: mimuc.de

- Lecture: Andreas Butz
- Tuesday, 10-12 am, W201 Lehrturm
  - Lecture (2 hours) + tutorials (2 hours)
  - Start c.t., break?

- Web page http://www.medien.ifi.lmu.de/lehre/ss15/cg1/
- PDF of the slides: night before class, print out and bring to take notes and **fill in blanks**
- Podcast: night after class (if all goes well ;-)

Chapter 1 - Introduction, Motivation, Basics

- About this Class: Organization
- Tutorials
- Why Should I Learn about Computer Graphics?
- Very Brief History of Computer Graphics
- Math Recap: What We Need to Survive...

About the Tutorials: Organization

image source: XING

- Tutorials: Henri Palleis
  - Will start the week of May 04-08
  - Discussion of assignments, background info
- Weekly assignments, in sync with lecture
  - http://www.medien.ifi.lmu.de/lehre/ss15/cg1/
  - Submission **voluntary** (means: No bonus points!)
  - students who did the exercises statistically got better grades!
- Purpose:
  - In-depth understanding of concepts from lecture
  - Gaining some basic practical experience in low-level graphics programming
  - Preparation for written test (best preparation strategy: do the assignments)
  - Please note: Tutorials and assignments are a ***service*** for the students

Tutorial Schedule

- Tutorial dates:
  - Group 01: Monday 12-14, Amalienstr. 73A, Room 114
  - Group 02: Monday 14-16, Amalienstr. 73A, Room 114
  - Group 03: Monday 16-18, Amalienstr. 17, Room A 105
  - Group 04: Tuesday 08-10, Amalienstr. 73A, Room 114
  - Group 05: Wednesday 16-18, Amalienstr. 73A, Room 114
  - Group 06: Wednesday 18-20, Amalienstr. 73A, Room 114

- Registration for the tutorials via UniWorX:

  https://uniworx.ifi.lmu.de/

  - from today, 20:00 until next tuesday, 23:55
- First assignment is online as we speak
- First tutorial on Monday May 4th

Chapter 1 - Introduction, Motivation, Basics

- About this Class: Organization
- Tutorials
- Why Should I Learn about Computer Graphics?
- Very Brief History of Computer Graphics
- Math Recap: What We Need to Survive...

Why should I learn about Computer Graphics?

- Basis for graphical digital media
  - –in the heart of your study and many future jobs!
- Basis for recent CG movies and SFX
  - –practically no more movies without it!
- Basis for many computer games
  - –market bigger than the film industry

source: http://sketchup.google.com

2D vs. 3D graphics vs. Pixels (see „Digitale Medien“)

- Pixel-based graphics
  - given resolution, describe color at each pixel
  - basis for digital photography
  - whole research area of image processing

- 2D graphics (aka vector graphics)
  - uses 2D lines and areas to describe an image
  - 2D drawing programs: Inkscape, Illustrator, Corel Draw, ...

- 3D graphics
  - describe 3D objects of a scene
  - compute what light would do to these objects
  - compute pixel image from a virtual camera

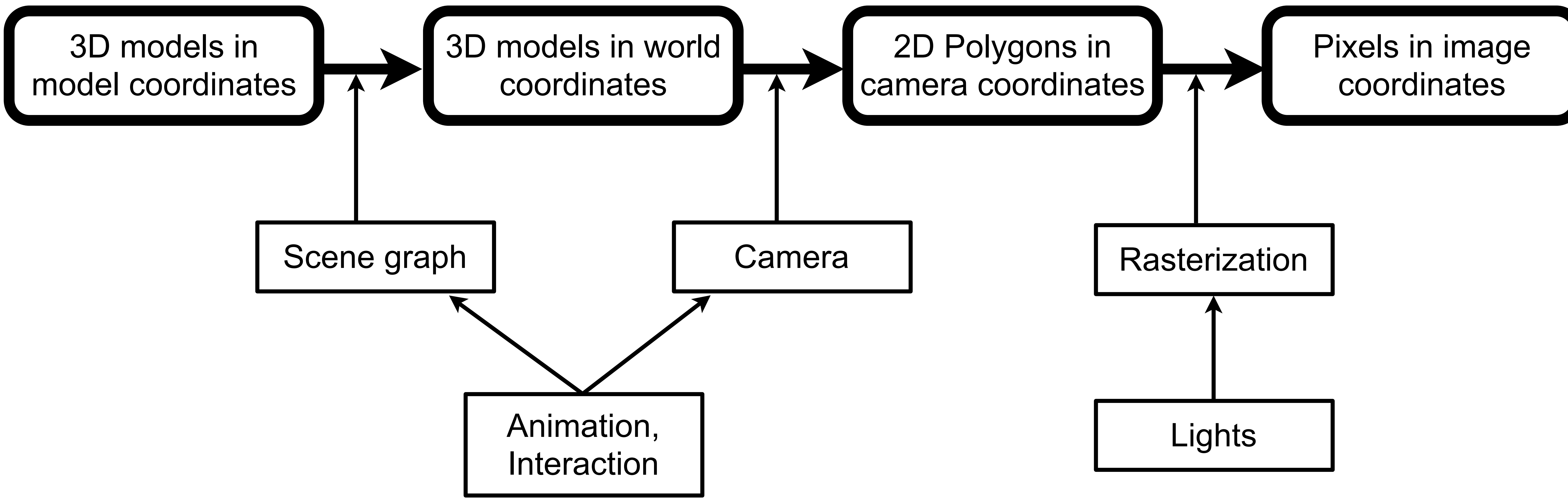
source: http://static.technorati.com/10/01/20/3467/Avatar-movie-Wallpapers.jpg

...so: 3D content on a 2D screen, huh?

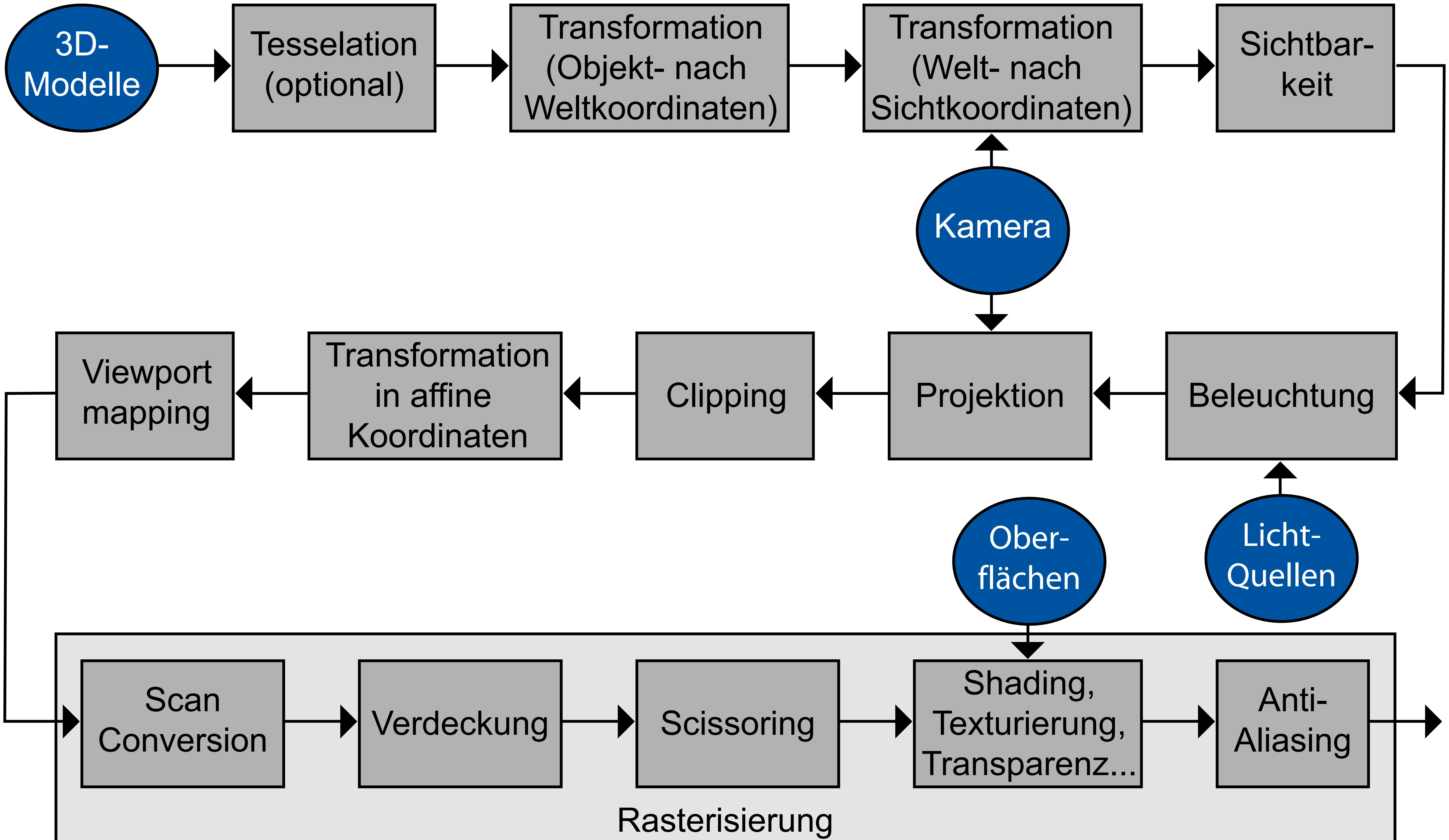
- General problem: current screens are 2D
  - for true 3D perception, we need 2 images for the 2 eyes (stereo)
  - this is technically still difficult (need glasses)
  - research area of volumetric or (auto)stereoscopic displays

- Content is 3D, display is 2D: what problems does this bring?
  -
  -
  -
  -
  -
  -
  -

The 3D rendering pipeline (our version for this class)

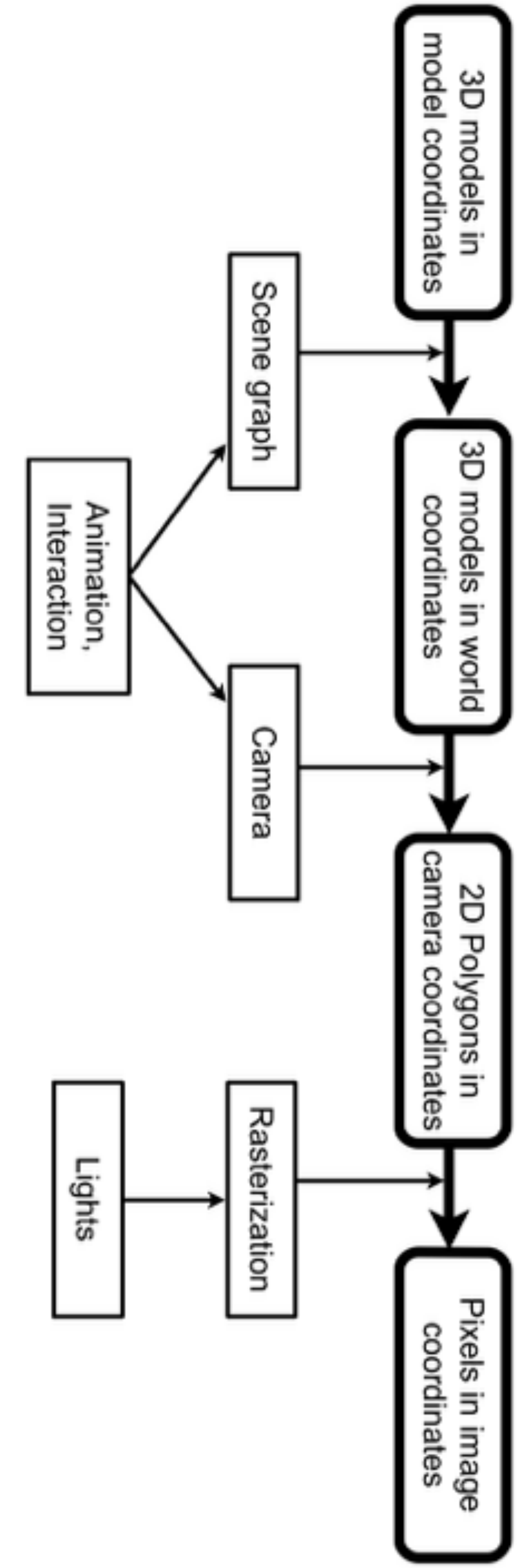


...this was not the only way to draw this pipeline...



Lecture Content & Schedule

Date	Nr	lecture topic
14.04.15		(AB at SWUN, Chengdu)
21.04.15		(AB at ACM CHI, Seoul)
28.04.15	1	Organization, Motivation, Basics
05.05.15	2	Mathematics for CG1
12.05.15	G	Guest lecture Marc Christie
19.05.15	3	3D Modeling
26.05.15		(holiday)
02.06.15	4	3D Camera, Culling, Rasterization (HP?)
09.06.15	5	Scene Graph
16.06.15	6	Light, Materials, Appearance
23.06.15	7	Shading and Rendering
30.06.15	8	Basics of 3D Animation
07.07.15	9	Interactive 3D Graphics
14.07.15	P	exam or final project presentation??



Literature Recommendations and links

- Malaka, Butz, Hussmann: Medieninformatik, Pearson Studium 2009
  - v.a. Kapitel 8: 3D-Grafik
- Bungartz, Griebel, Zenger: Einführung in die Computergraphik, 2. Auflage, Vieweg, 2002
- Hearn, Baker, Carithers: Computer Graphics with OpenGL, 4th edition, Pearson 2011
- Foley, Van Dam, Feiner: Computer Graphics – Principles and Practice, 2nd edition, Addison-Wesley, 1996
- Watt, A. et al.: Advanced Animation and Rendering Techniques.: Theory and Practice, Addison Wesley, 1992
- OpenGL: www.opengl.org
- Three.js: http://threejs.org/

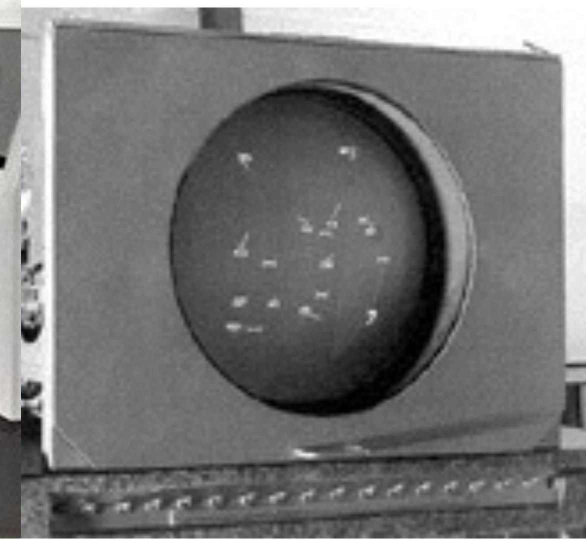
Chapter 1 - Introduction, Motivation, Basics

- About this Class: Organization
- Tutorials
- Why Should I Learn about Computer Graphics?
- Very Brief History of Computer Graphics
- Math Recap: What We Need to Survive...

Based on lecture material by Regina Pohle-Fröhlich

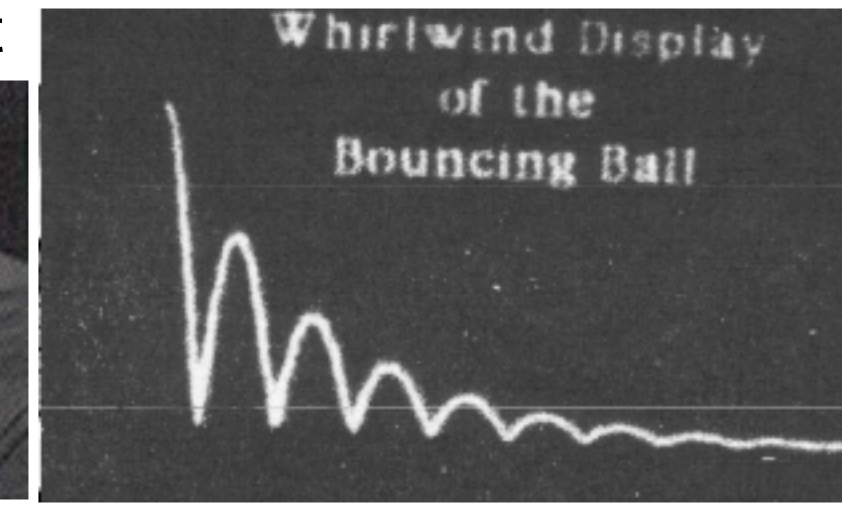
First Steps Towards Computer Graphics 1945 – 1963

1945-1952: “Whirlwind” computer (Jay Forrester, MIT)
Digital computer using oscilloscope screen displaying real-time aircraft data, later “SAGE” system



Using “light pen” for input

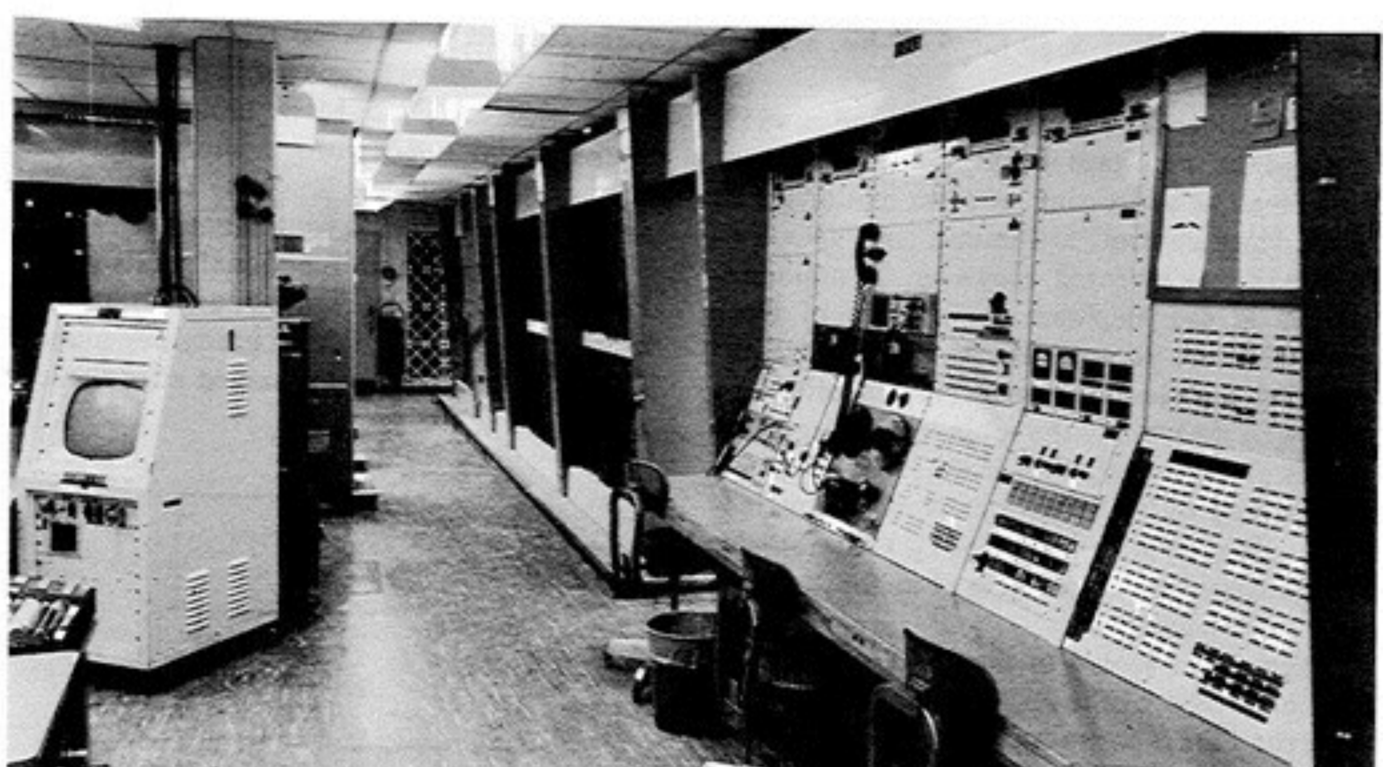
“Bouncing ball” (C. Adams)



wired.com

design.osu.edu/carlson/history

www.rendering.ovgu.de



research.microsoft.com

1957-1969: “TX-2” computer at MIT Lincoln Lab
Transistor-based computer providing interactive graphic displays
L.G. Roberts, 1962: 3D Graphics
Ivan Sutherland, 1963: Sketchpad

computerhistory.org

Theory Development in the 1970s

- 1971: Raster Scan Principle (M. Noll, Bell Labs)
  - Connecting a TV-like display with computer memory
- 1973: First ACM “SIGGRAPH” Conference
- 1971-1975: Shading algorithms (Gouraud 1971, Phong 1975)
- 1977-1978: Shadow computation (Crow, Williams)
- 1975: 3D Model “Utah Teapot” (M. Nevell, U. Utah)
- 1979: Raytracing (mirror reflection, transparency) (Kay, Whitted)
- 1984: Global illumination model “radiosity” (Goral et al., Nishita)

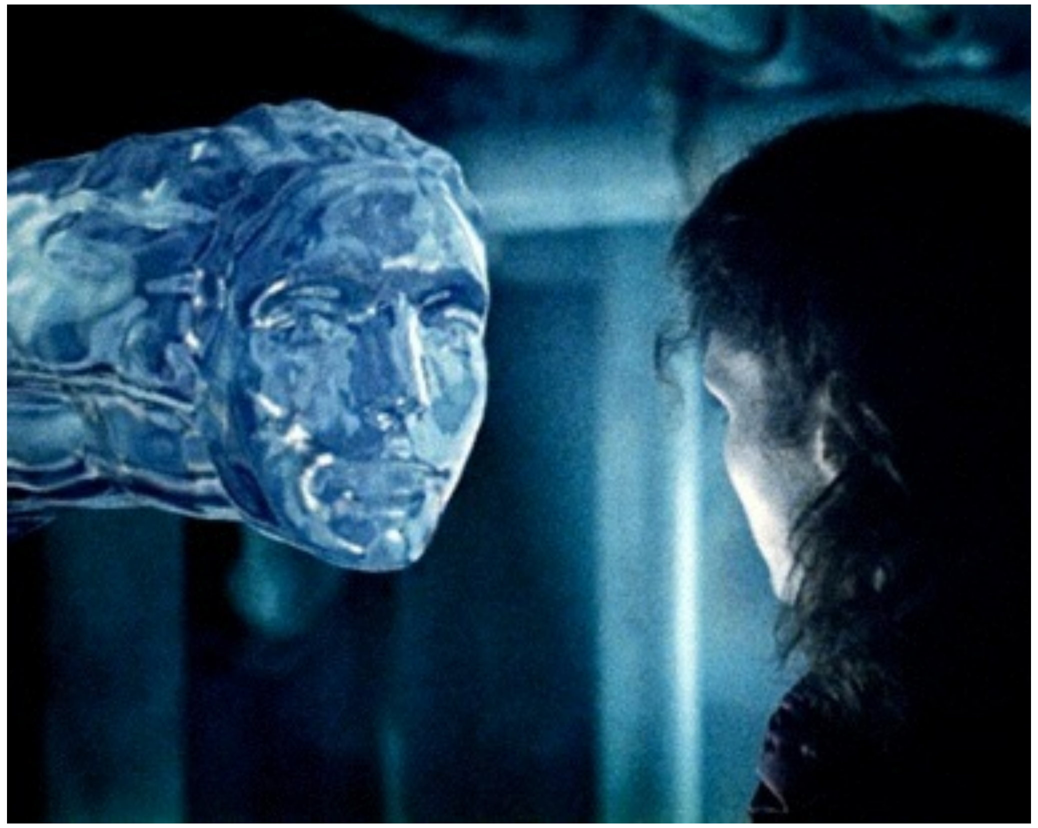
Utah Teapot
at Computer History Museum, Boston

wikipedia.org

Computer Graphics Goes to Cinema: 1980s

- 1979: Computer Graphics department of Lucas Film founded (ILM)
- 1980: Demonstration of video “Vol Libre” (L. Carpenter) at SIGGRAPH
- 1980: Computer Animations in movie “Tron”
- 1981: Predecessor of “Renderman” (REYES) by L. Carpenter at Lucas Film
- 1986: “Pixar” founded (Catmull, Smith), split off Lucas Film
- 1988: Movie “The Abyss” (Water creature by Lucas Film ILM)
- 1989: Motion Capturing (Jim Henson)
- 1995: Movie “Toy Story” (Pixar, fully computer-generated)

Vol Libre atariarchives.org



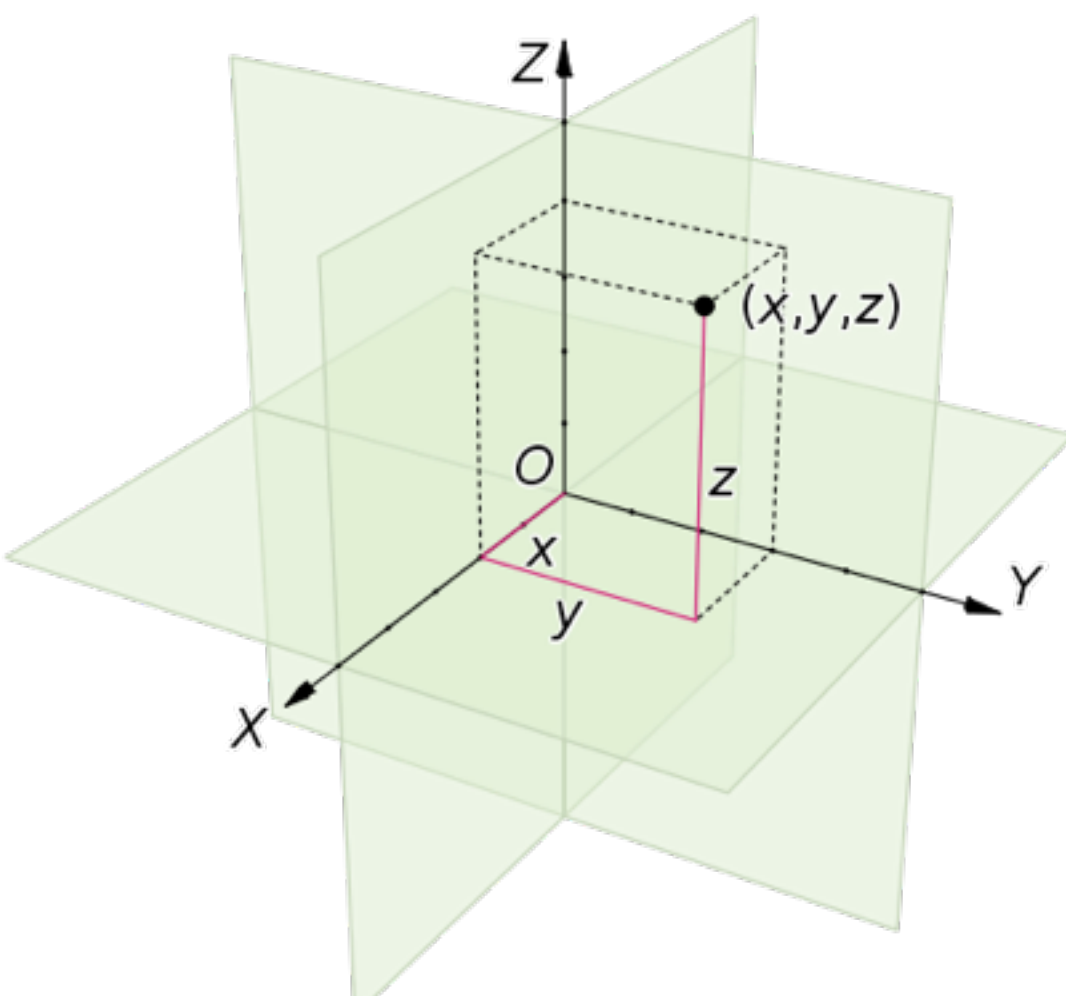
Abyss empireonline.com

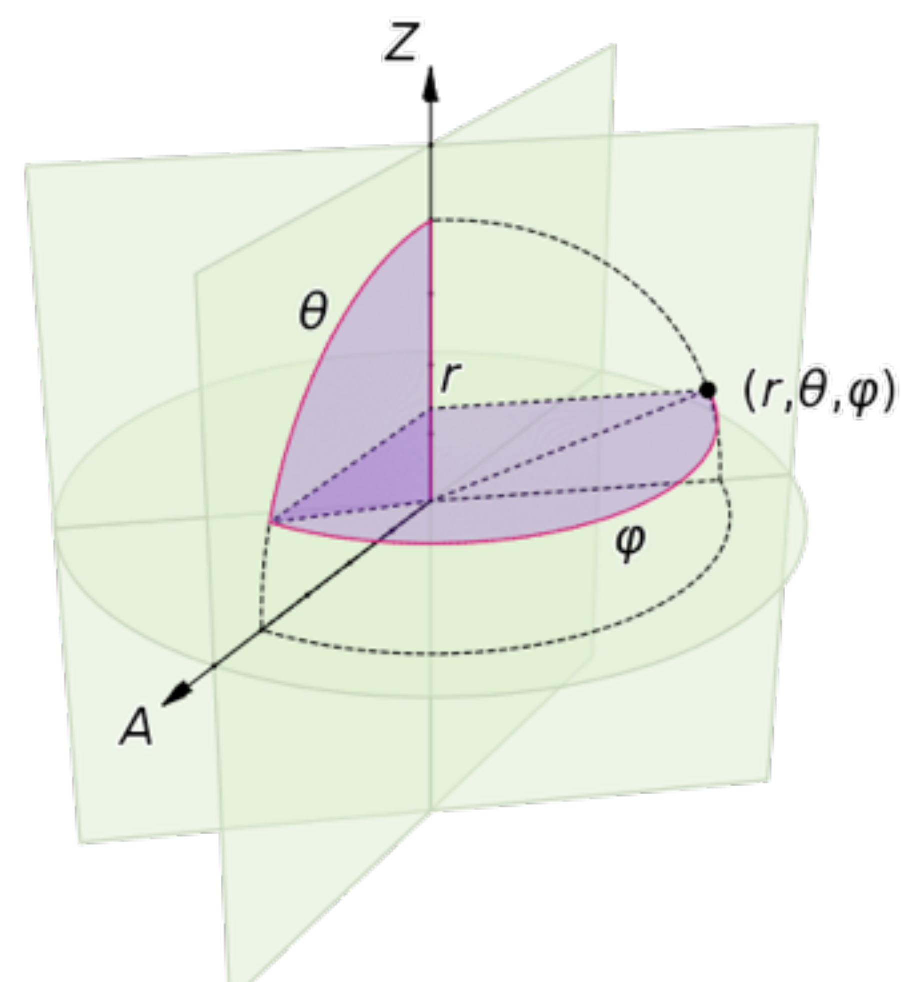
Chapter 1 - Introduction, Motivation, Basics

- About this Class: Organization
- Tutorials
- Why Should I Learn about Computer Graphics?
- Very Brief History of Computer Graphics
- Math Recap: What We Need to Survive...

Coordinate Reference Frames

- Dimensionality
  - We will meet: 2, 3 and 4 dimensions
- Types of coordinate systems
  - Cartesian (rectilinear): Pairwise orthogonal axes with (identical) linear scale
  - Non-cartesian (curvilinear): Many other systems
    - e.g. polar/spherical coordinates: angle plus distance

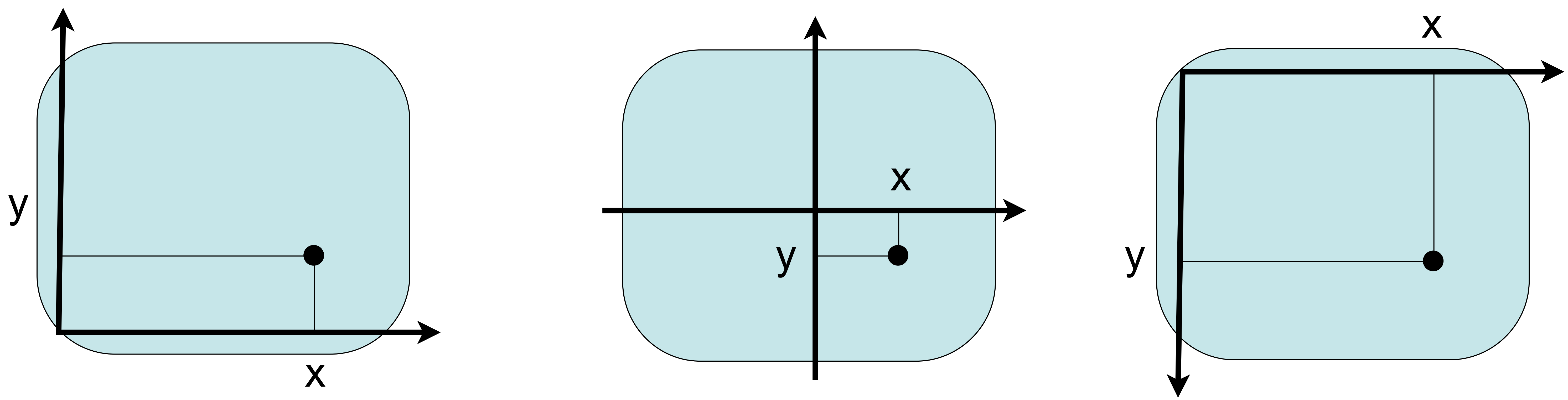




Images: Wikipedia

2D Cartesian Coordinate Reference Frames

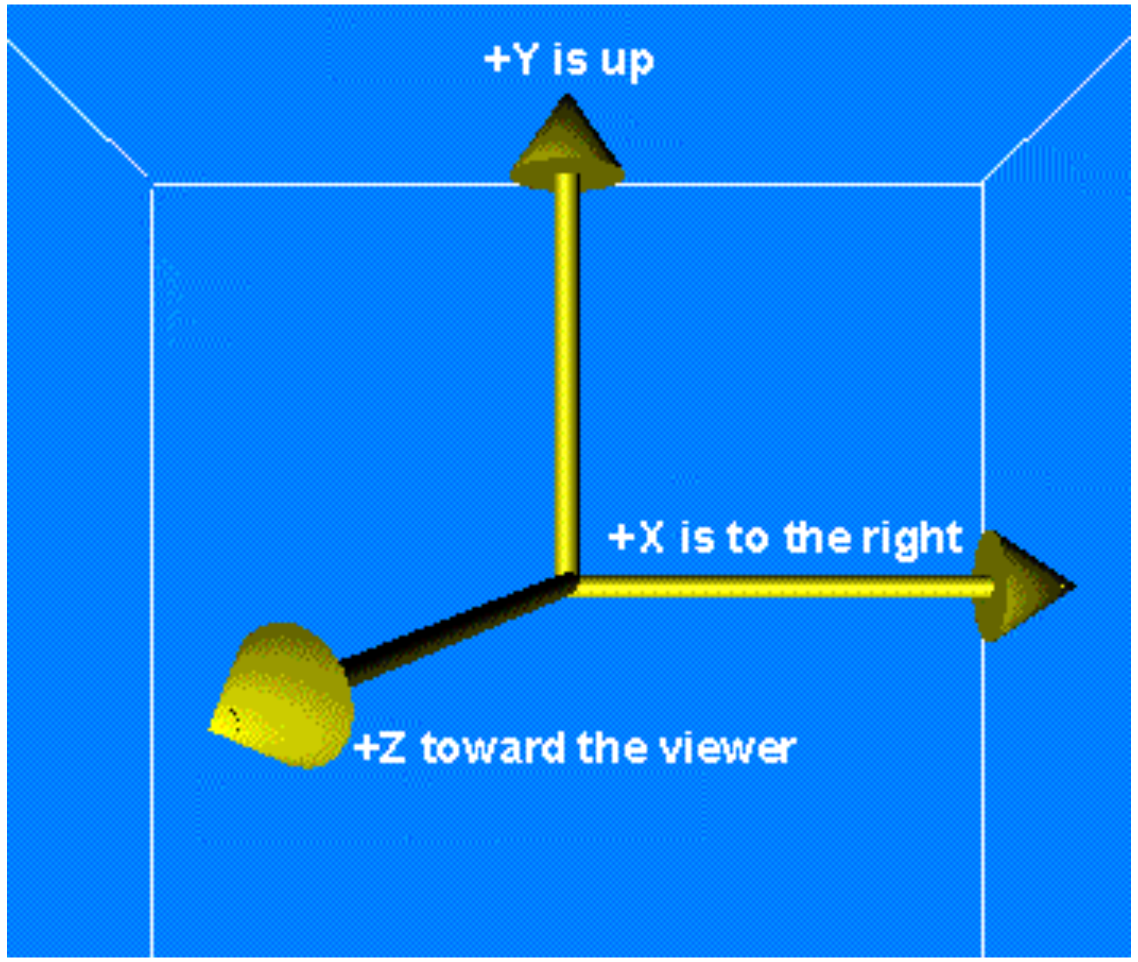
Device-independent commands of graphics packages:
Varying schemata: origin may be in lower-left corner, center, upper-left corner

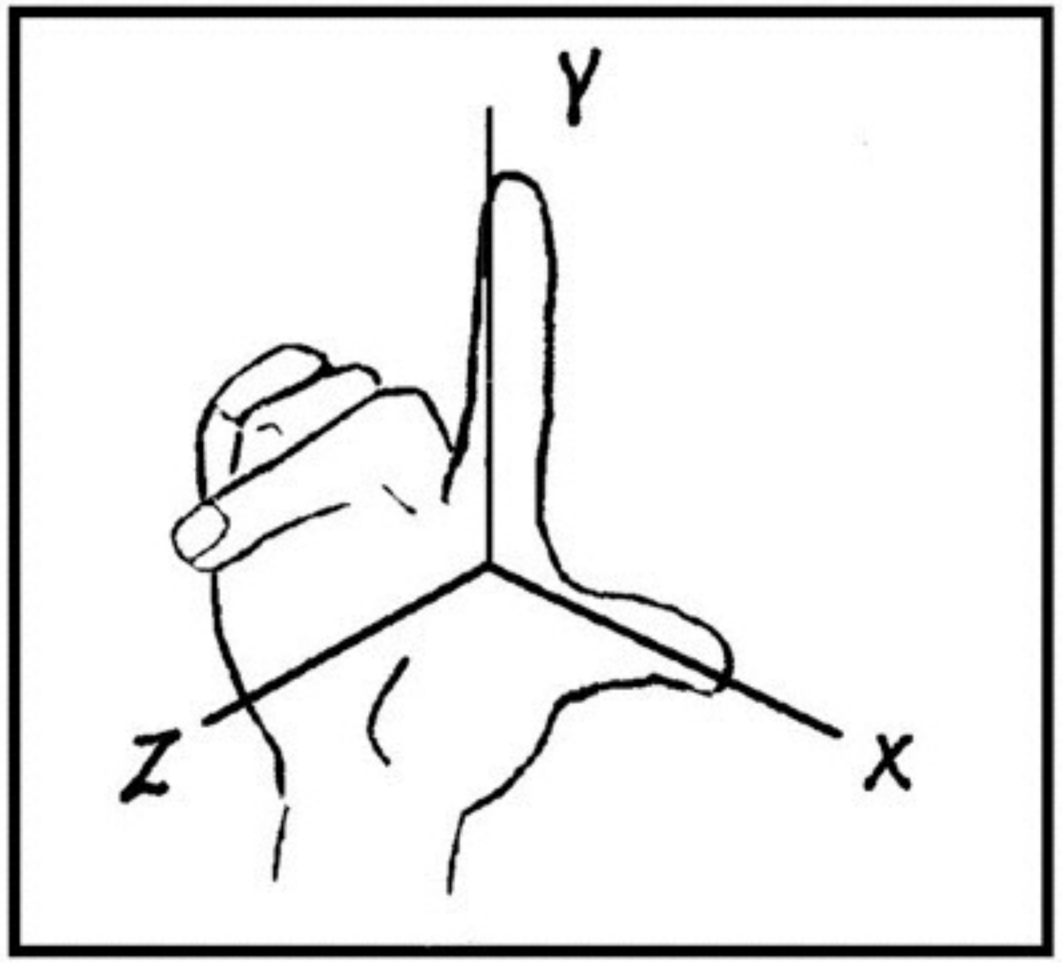


Device coordinates:
  Scan lines on cathode ray tubes, printers:
    origin in upper left corner, y axis points downwards
  Other devices: Origin in lower-left corner
Normalized device coordinates: Range from 0.0 to 1.0 (real number)
Physical device coordinates: Integers (pixel address)

Standard 3D Cartesian Coordinate Reference Frames

- Most frequently used "world coordinates" (e.g. in OpenGL): "Right handed" system, often depicted as looking from z axis





Pictures: euclidianspace.com, cornell.edu

- "Left handed" system used in special cases (e.g. 2D screen positions with additional depth information)

Points and Vectors

- *Point*
  - Position specified with coordinate values in some reference frame
  - e.g. in 3D Cartesian coordinates: (p_x, p_y, p_z)

$$\begin{pmatrix} p_x \\ p_y \\ p_y \end{pmatrix}$$

- *Vector*
  - Tuple of real numbers, considered as element of a vector space
  - Often written vertically (column vector)
  - In CG, people are sloppy about the diference between row and column vectors!

- Difference between two positions gives a vector
- Position can be specified by vector from origin in Cartesian system
- Vectors can be multiplied with a real number pointwise
- Two vectors of same length can be added pointwise

Properties of Vectors

- Magnitude (length)

$$a = \left(a_x, a_y, a_z\right) \qquad \|a\| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

- Direction angles

$$\cos\delta_x = \frac{a_x}{\|a\|} \qquad \cos\delta_y = \frac{a_y}{\|a\|} \qquad \cos\delta_z = \frac{a_z}{\|a\|}$$

Scalar Product (Dot Product)

- The *scalar product* computes a real (scalar) value from two coordinate vectors of equal length

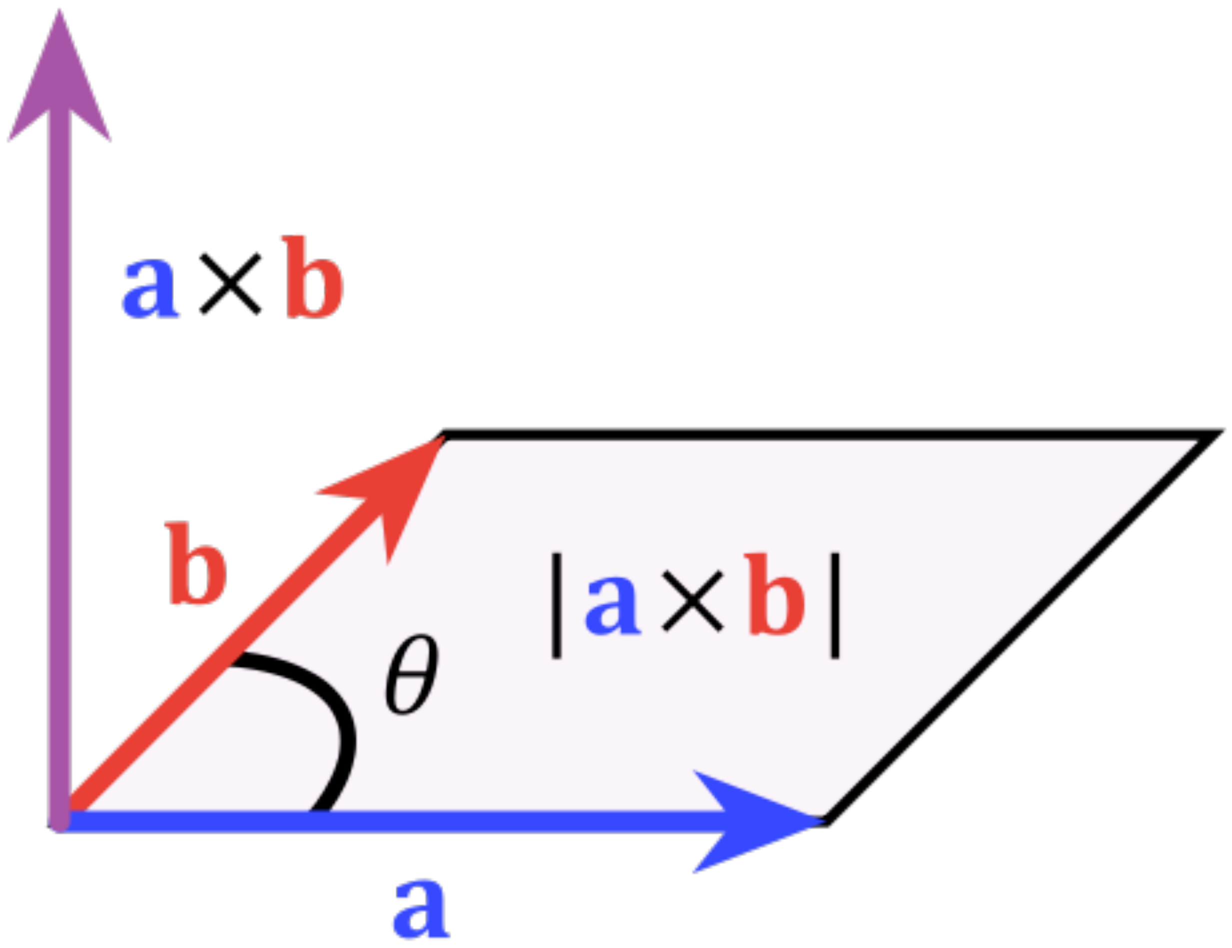
$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \bullet \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = a_x b_x + a_y b_y + a_z b_z$$

- Application: Computation of angle between two coordinate vectors

$$a \bullet b = \left\|a\right\| \cdot \left\|b\right\| \cdot \cos\alpha$$

Cross Product (Vector Product)

- The *cross product* of two coordinate vectors is a vector which is perpendicular to both given vectors
  - Direction: Right-hand rule
  - Magnitude: Equals spanned parallelogram

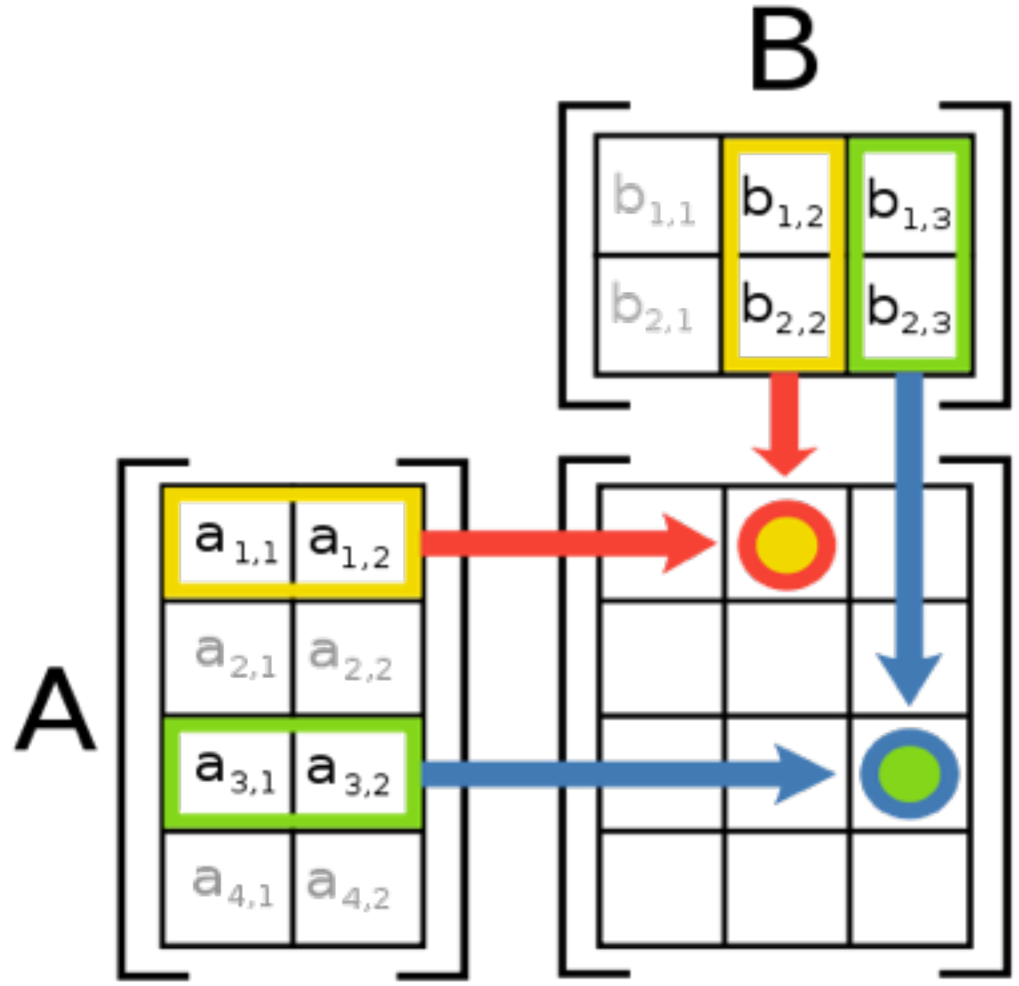


$$\begin{pmatrix} a_x \\ a_y \\ a_z \end{pmatrix} \times \begin{pmatrix} b_x \\ b_y \\ b_z \end{pmatrix} = \begin{pmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{pmatrix}$$

Matrices

- A *matrix* is an (*m*×*n*) arrangement of real numbers (*m* rows, *n* columns)
- Used in CG for expressing computations on coordinate vectors
- A matrix can be multiplied with a real number pointwise
- Two matrices of identical dimensions can be added pointwise
- Multiplying matrices:
  (*m*×*p*)-matrix *A* multiplied by (*p*×*n*)-matrix *B* gives (*m*×*n*)-matrix *C*

$$C_{i,j} = \sum_{k=1}^{p} A_{ik} \cdot B_{kj} \qquad 1 \le i \le m, \quad 1 \le j \le n$$



Multiplying a Matrix and a Vector

- Special case of matrix multiplication
- (m×p)-matrix A multiplied with vector v of length p gives vector w of length m

$$w_j = \sum_{k=1}^{p} A_{ik} \cdot v_k$$

- If this all sounded difficult or long-forgotten:
  - dig out your old school books
  - reread your LinAlg scripts
  - consider selecting another lecture

- There will be more math in the rest of the lecture
- There will be math in the exam!

http://jasinski.ukw.edu.pl/images/133b.jpg