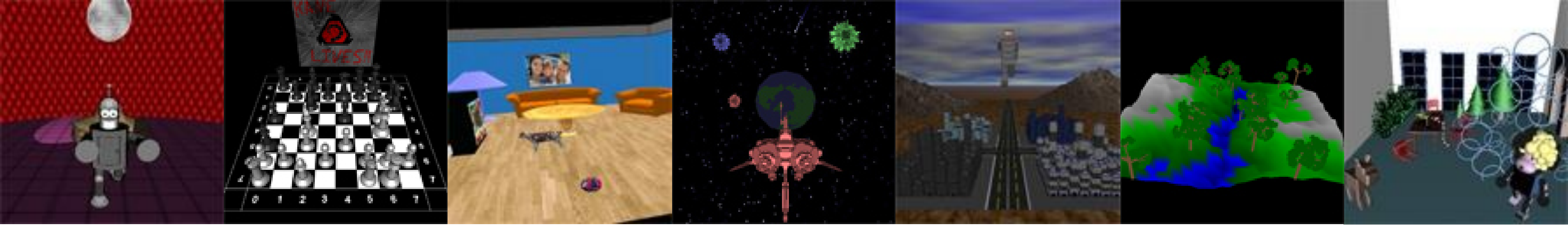


Computergrafik

Matthias Zwicker
Universität Bern
Herbst 2016

Staff

Instructor

- Matthias Zwicker (zwicker@iam.unibe.ch)

Teaching assistant

- Marco Manzi (manzi@iam.unibe.ch)
- Tiziano Portenier (portenier@iam.unibe.ch)

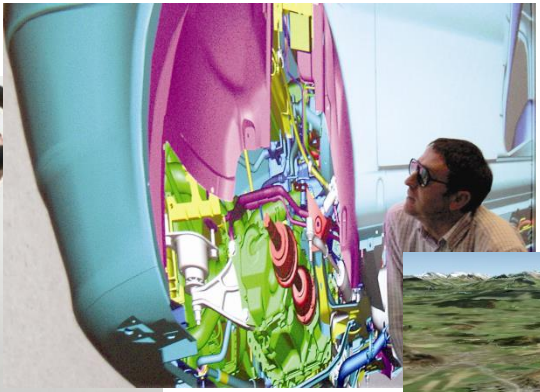
Student teaching assistants

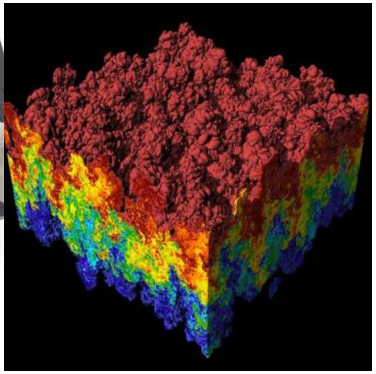
- Adrian Wälchli (adrian.waelchli@students.unibe.ch)
- Paul Frischknecht (paul.frischknecht@students.unibe.ch)

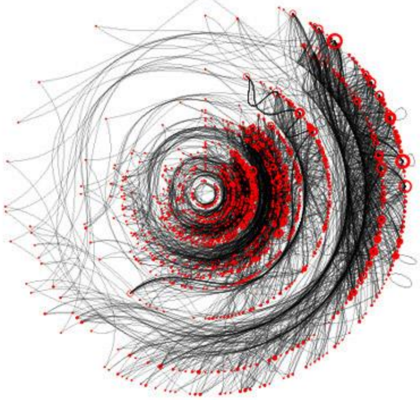
Today

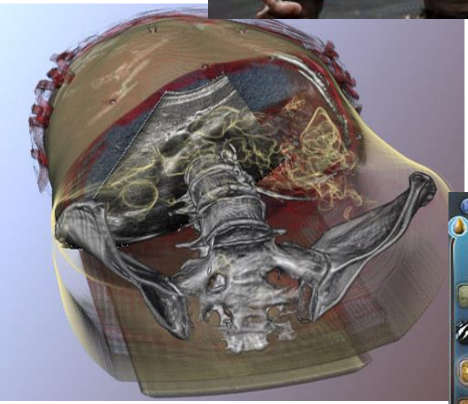
- Course overview
- Course organization
- Vectors and coordinate systems

Computer graphics









Computer graphics

- „Technology to create images using computers“
- Core areas
  - Rendering
  - Modeling
  - Animation

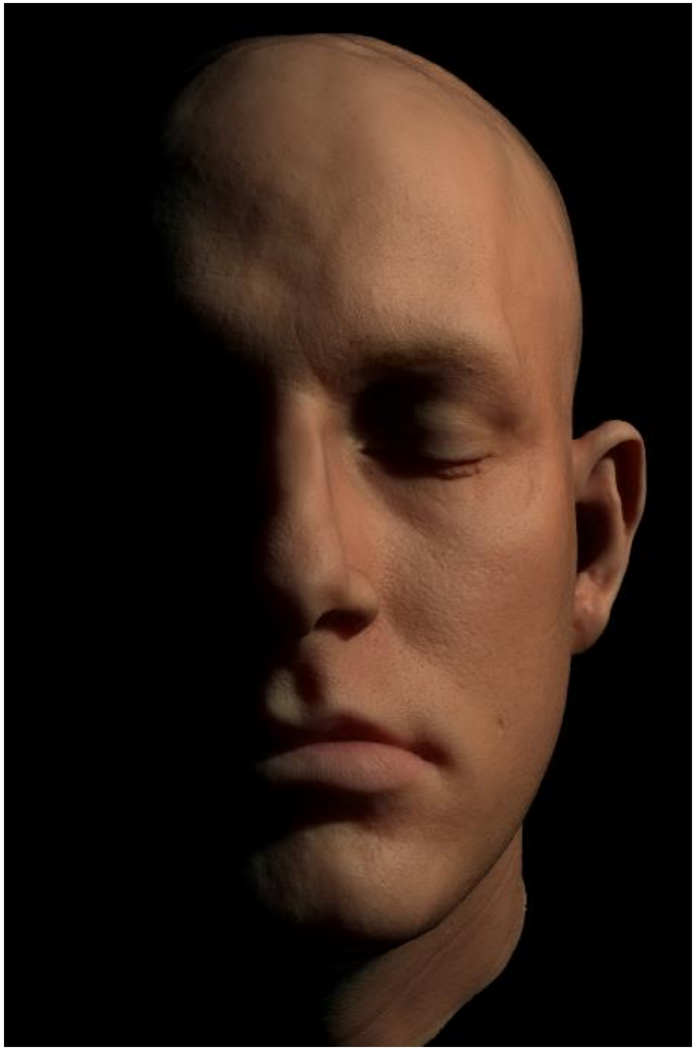
Rendering

- Synthesis of 2D image from 3D scene description
  http://en.wikipedia.org/wiki/Rendering_(computer_graphics)
  - Rendering algorithms interpret data structures that represent scenes using geometric primitives, material properties, and lights
- Input
  - Data structures that represent scene (geometry, material properties, lights, virtual camera)
- Output
  - 2D image (array of pixels)
  - Red, green, blue values for each pixel

Rendering

- Wealth of algorithms with different objectives
  - Photorealistic
  - Interactive
  - Artistic

Photorealistic rendering



See also http://en.wikipedia.org/wiki/Rendering_(computer_graphics)

Photorealistic rendering

- Physically-based simulation of light, materials, and camera
  - Physical model expressed using the rendering equation, http://en.wikipedia.org/wiki/Rendering_equation
  - Shadows, realistic illumination, multiple light bounces
- Slow, minutes to hours per image
- Special effects, movies
- Not in this class

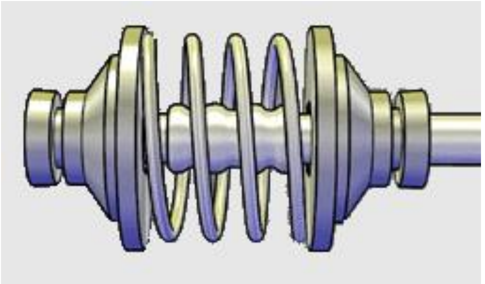
Interactive rendering

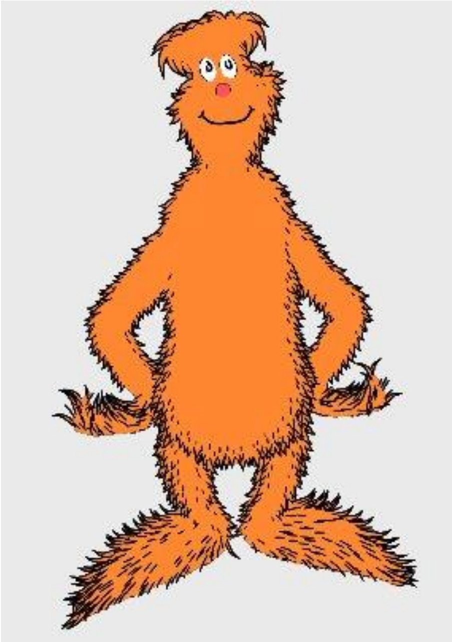
Interactive rendering

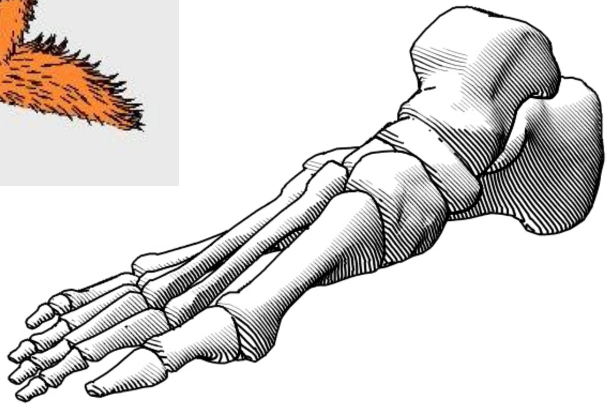
- Focus of this class
- Produce images within milliseconds
- Interactive applications (games, …)
- Using specialized hardware, graphics processing units (GPUs)
- Standardized APIs (OpenGL, DirectX)
- Often “as photorealistic as possible”
  - Hard shadows, fake soft shadows, only single bounce of light

Artistic rendering

- Also “non-photorealistic rendering”
  http://en.wikipedia.org/wiki/Non-photorealistic_rendering
- Stylized
- Artwork, illustrations, data visualization





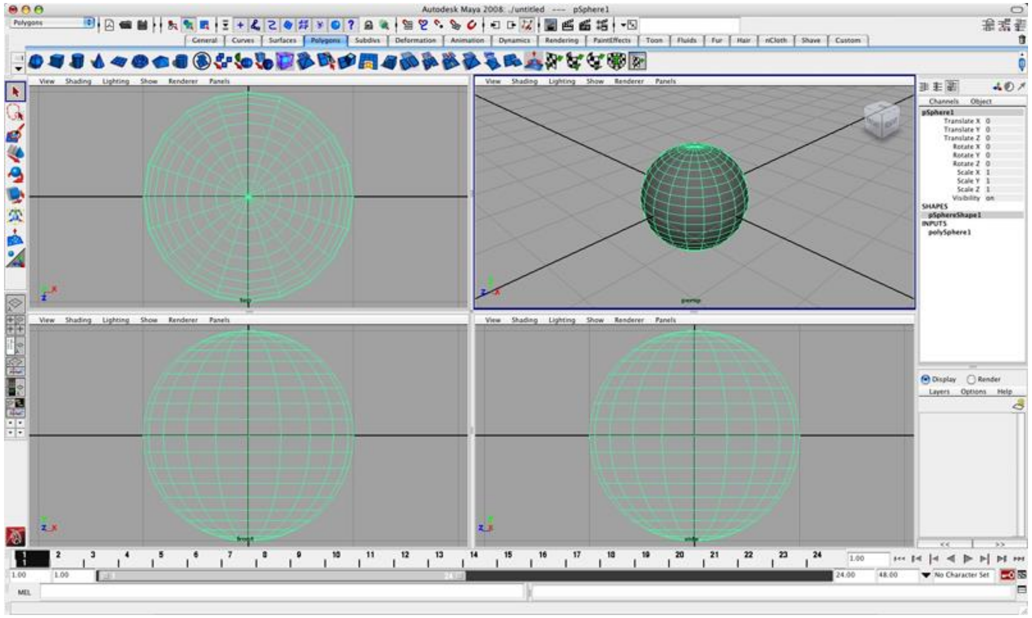


Modeling

- Creating 3D geometric data
  - The “model” or the “scene”
- Modeling techniques
  http://en.wikipedia.org/wiki/3D_modeling
  - Manually, using software tools
  - Algorithmic (procedural)
  - Physical measurements, 3D scanning

3D modeling tools

- Similar to plastic arts such as sculpting
- Professional tools
  - Autodesk (Maya, AutoCAD), LightWave 3D, …
- Free software
  - Blender
    http://www.blender.org/
- Not as easy to use as Notepad…



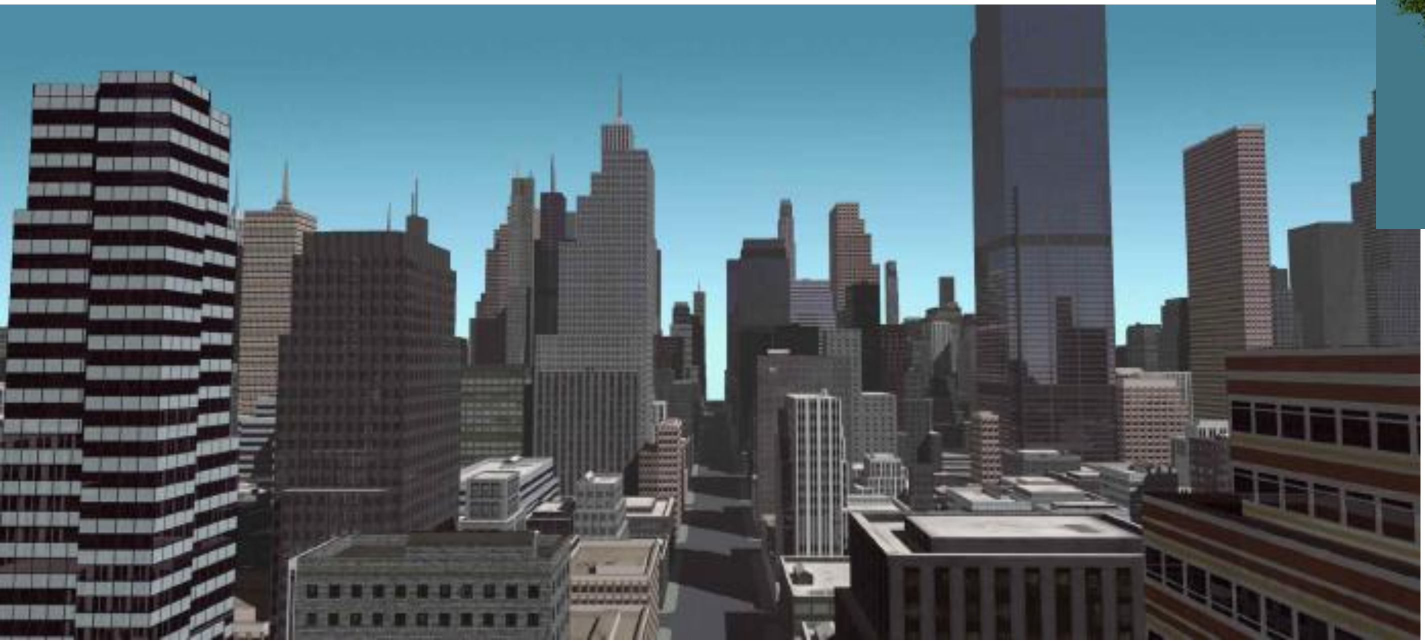
Maya Screenshot

Procedural modeling

- By writing programs, algorithms
  http://en.wikipedia.org/wiki/Procedural_modeling

Procedural tree

Procedural city

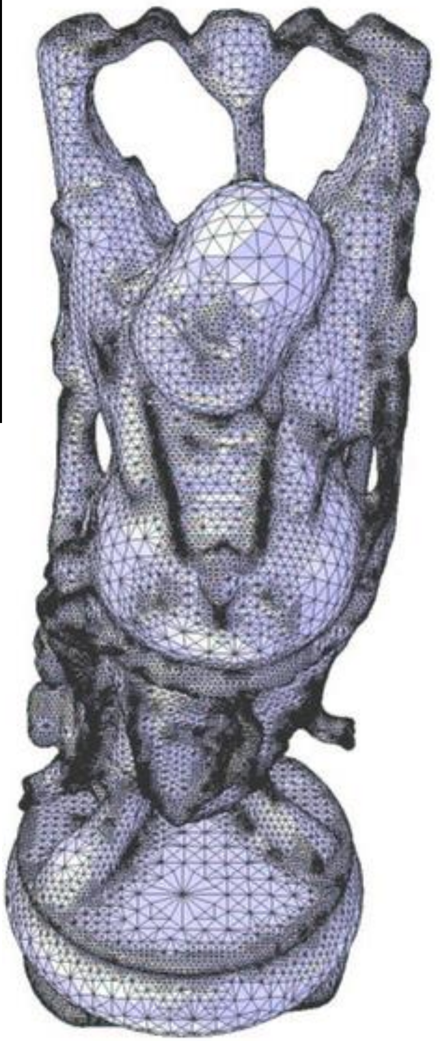


See also http://www.esri.com/software/cityengine/

Physical measurements

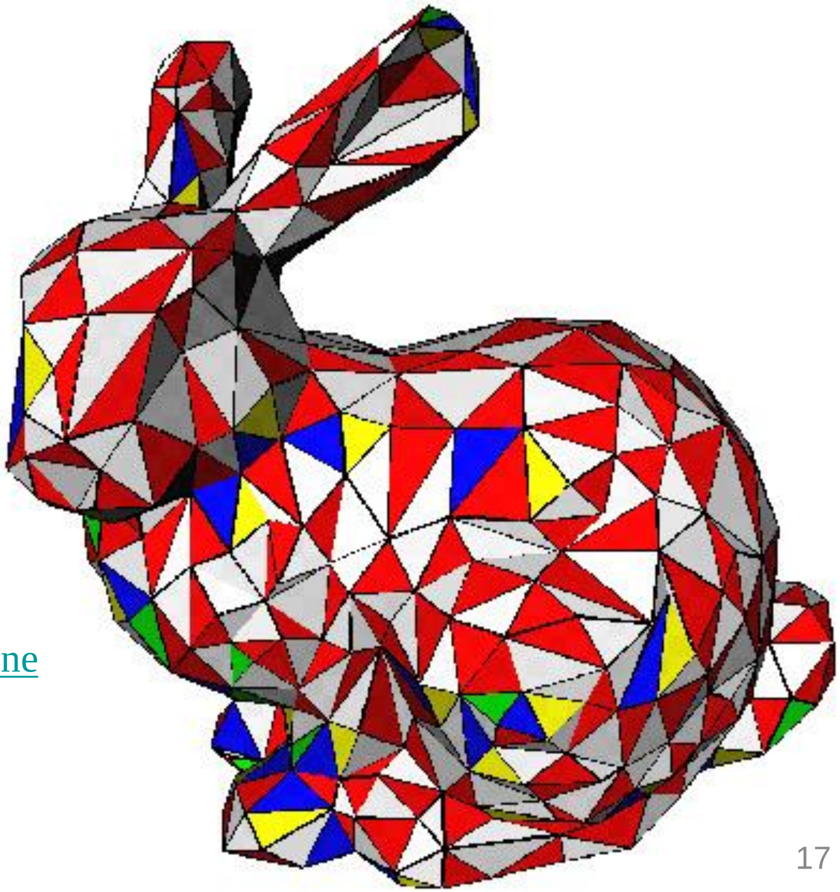
- 3D scanning
  http://en.wikipedia.org/wiki/3D_scanner
- Other imaging devices
  - Tomography
    http://en.wikipedia.org/wiki/Ct_scan
  - Magnetic resonance imaging (MRI scanning)
    http://en.wikipedia.org/wiki/Mri_scan
  - Etc.

Scanned statue (laser scan)
http://graphics.stanford.edu/data/3Dscanrep/



Surface representation

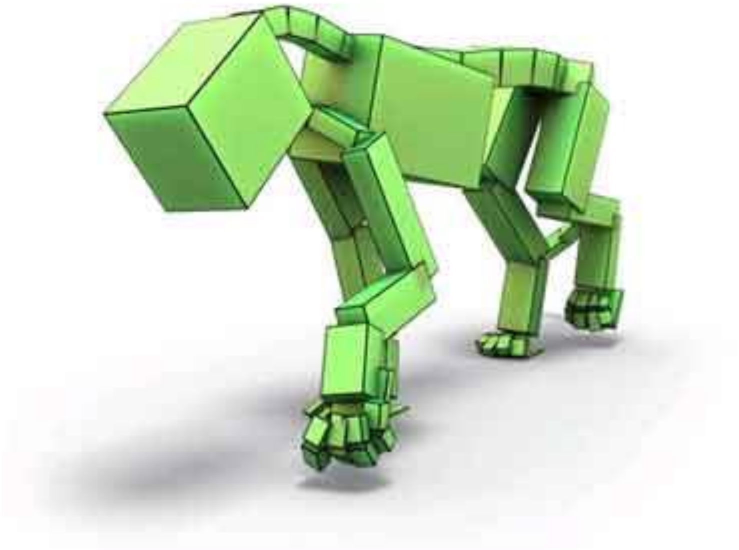
- Basic 3D models consist of collections of triangles (triangle mesh, https://en.wikipedia.org/wiki/Triangle_mesh)
- Each triangle consists of 3 vertices
- Each vertex contains
  - xyz position
  - Optionally other attributes such as color, etc.
- Many more sophisticated representations exist
  http://en.wikipedia.org/wiki/Nonuniform_rational_B-spline
  http://en.wikipedia.org/wiki/Subdivision_surface

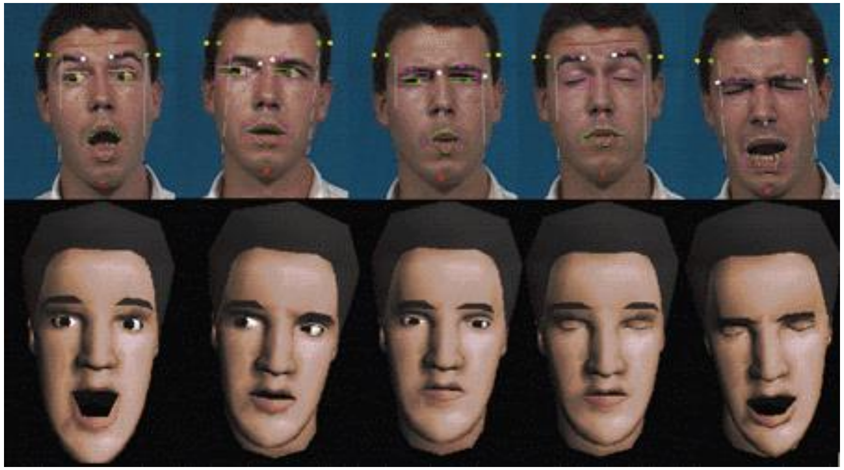


Animation

- Deforming or editing the data
- Change over time
- Faces, articulated characters, fire, water, rigid objects, elastic objects, fracturing objects, …

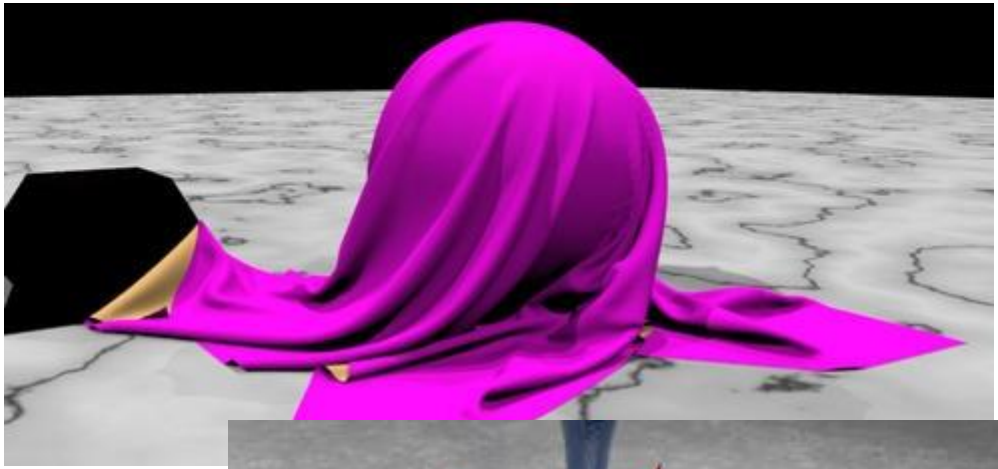
Animation

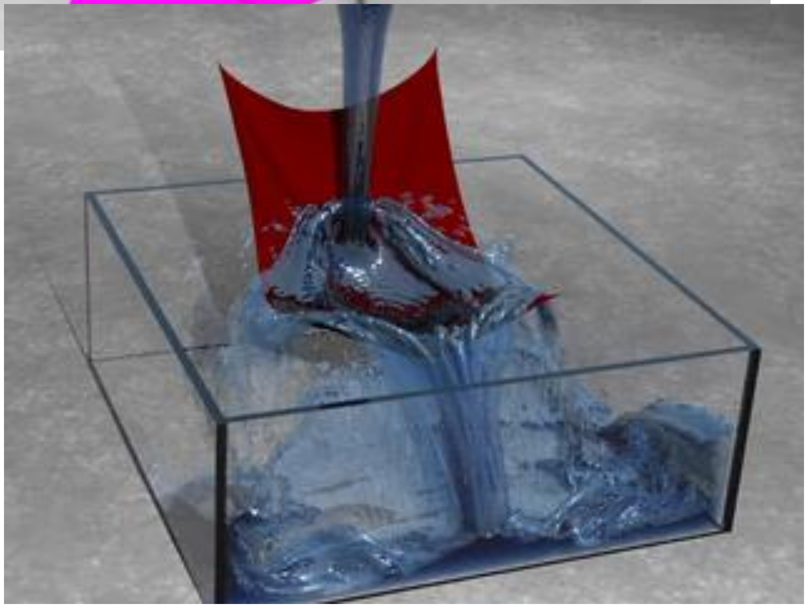




http://www.youtube.com/watch?v=bLiX5d3rC6o

Physics simulation





http://www.youtube.com/watch?v=XH28gAb8FiQ

See also http://www.nvidia.com/object/physx_new.html

In this class

The Basics…

- Rendering 3D models
  - Camera simulation
  - Interactive viewing
  - Lighting, shading
- Modeling
  - Triangle meshes
  - Smooth surfaces
- Experience with linear algebra, Java, OpenGL, GPU programming
- Background for advanced topics

Schedule

1. Introduction
2. Homogeneous coordinates, transformations
3. Projection
4. Rasterization
5. Color
6. Shading I
7. Shading II
8. Textures
9. Scene management
10. Curves
11. Surfaces
12. Advanced shading, shadows
13. Virtual reality
14. Reserve

Questions?

Today

- Course overview
- Course organization
- Vectors and coordinate systems

Course organization

Lecture

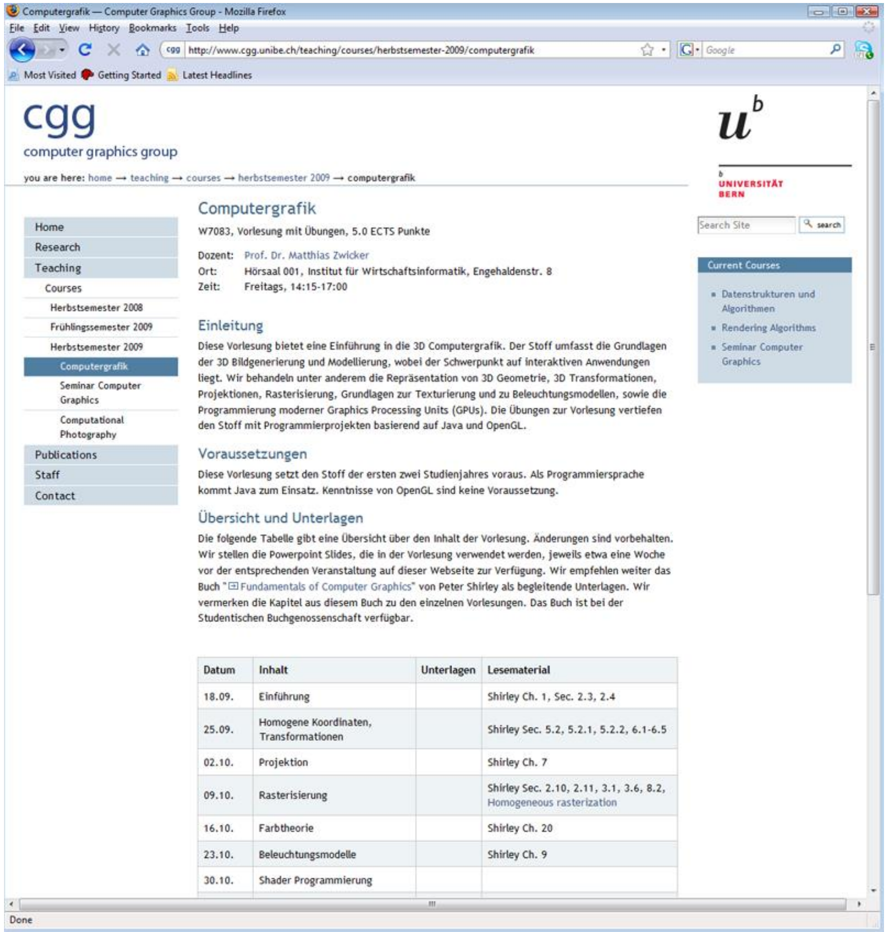
- Fridays, 14:00-16:00
- Engehaldenstrasse 8, Raum 001

Exercises

- Fridays, 16:00-17:00
- Engehaldenstrasse 8, Raum 001

Class web page

- Overview of topics
  http://www.cgg.unibe.ch/teaching/computergrafik

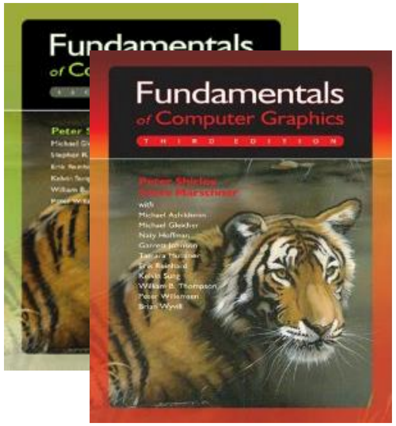


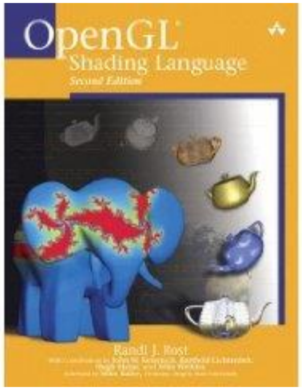
Ilias

- Search for “Computergrafik HS2016” in https://ksl.unibe.ch, Ilias link provided
- Use your campus account to log in
- Join without password
- All relevant content for the class
  - Schedule
  - Slides
  - Project descriptions
  - Scanned chapters from textbook
  - Online forum for questions and discussion
- Take advantage of online forum to get advice!

Textbooks

- Fundamentals of Computer Graphics, Peter Shirley, 2nd or 3rd edition (recommended)
  - Available in the student bookstore
  - Scanned chapters on Ilias
- Opengl Shading Language, Rost, Addison Wesley, 2nd edition (optional)





Exercises

- Six programming projects
- Two exercise series on paper
- Successful completion of exercises is requirement for exam
  - Each assignment is worth 10 points
  - Total 80 points
  - Requirement is 75% (60 points)
- Late penalty
  - 50% of original score
  - Exceptions for military service, illness

Programming Projects

- Assignments and schedule on Ilias
- Java base code and documentation on github
  https://github.com/mzwicker/Computergrafik-Basecode
- Turn-in electronically on Ilias and demonstration to TA in ExWi pool
  - More details, how to sign-up in exercise session

Programming Projects

- Use ExWi pool or your own computer
- Need support of OpenGL 3.0 or later
  - Update your graphics driver!
- Older Intel integrated graphics processors do not currently support OpenGL 3.0
  - "HD graphics" series, Sandy Bridge processors are first to provide support
    http://en.wikipedia.org/wiki/Intel_GMA
  - For Intel integrated graphics on Linux, drivers may not be available that support our code

Programming Projects

Build your own 3D rendering engine

- **Project 1**: Matrices, Vectors, and Coordinate Transformations
- **Project 2**: Interactive Viewing
- **Project 3**: Rasterization
- **Project 4**: Lighting and Texturing
- **Project 5**: Scene Graphs
- **Project 6**: Curves and Surfaces, Virtual Reality

Exercises on paper

- Two exercise series on paper
- Schedule TBA
- As preparation for exam

Prerequisites

Familiarity with

- Linear algebra (matrix calculations)
- Java
- Object oriented programming

Questions?

Today

- Course overview
- Course organization
- Vectors and coordinate systems

3D scene representation

- Goal: describe 3D scenes
  - Position, orientation, motion of objects
  - Relation of objects to virtual camera
  - Projection of scene onto image plane
- Linear algebra provides mathematical tools
  - Vectors, coordinate systems, matrices, etc.
  - As little abstract theory as possible in this class

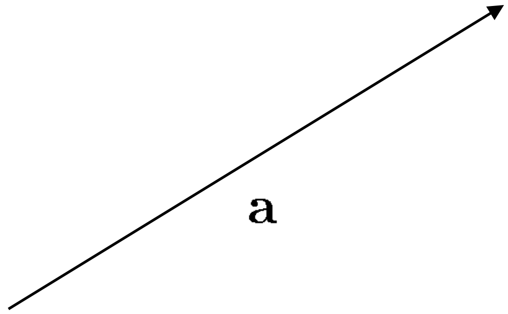
Topics today

Linear algebra & vector geometry review

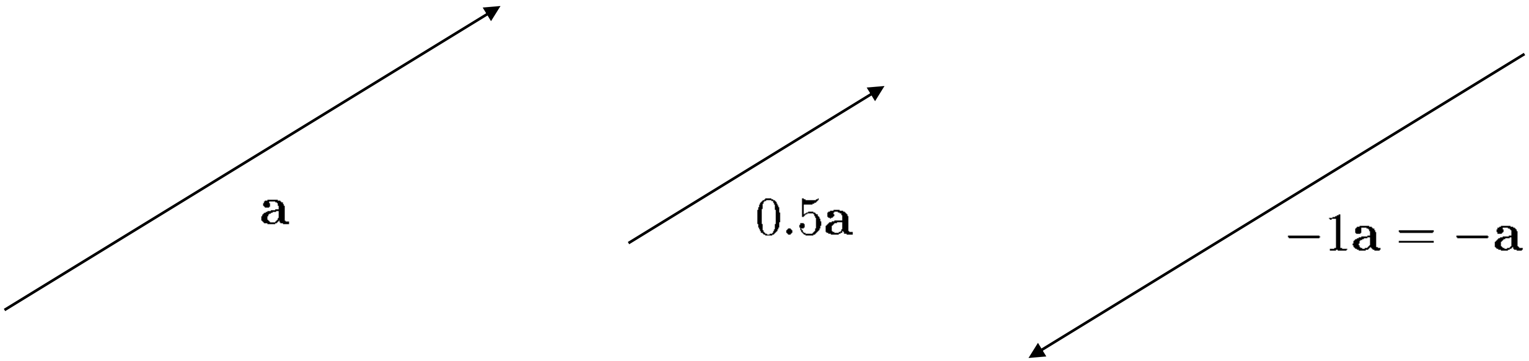
- Vectors
- Linear combination, linear dependency
- Coordinate systems
- Dot product, cross product
- Normal vectors
- Representation of planes using vectors

Vectors

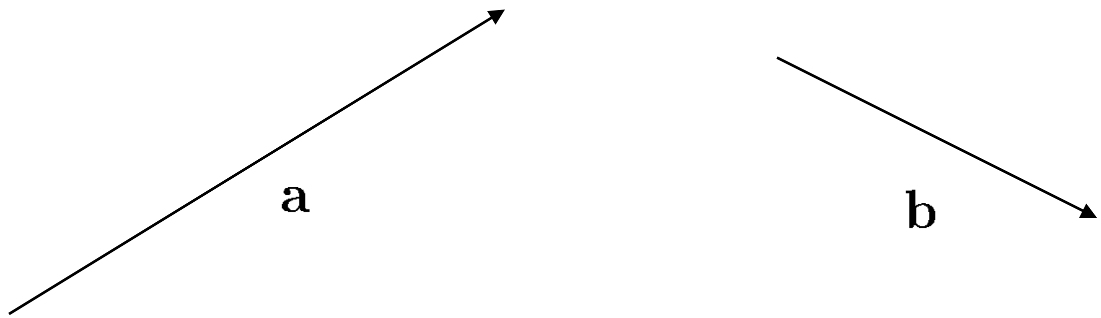
- Direction and length in 3D
  - No anchor point
- Vectors can describe
  - Difference between two 3D points
  - Speed of an object
- Vectors are in bold-face

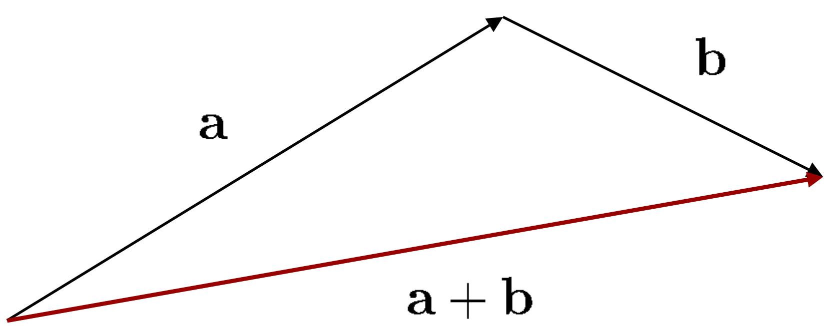


Multiplication by scalar



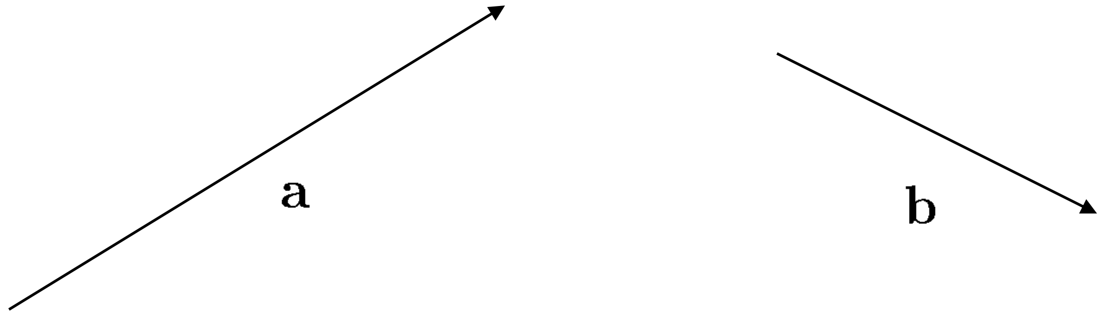
Addition

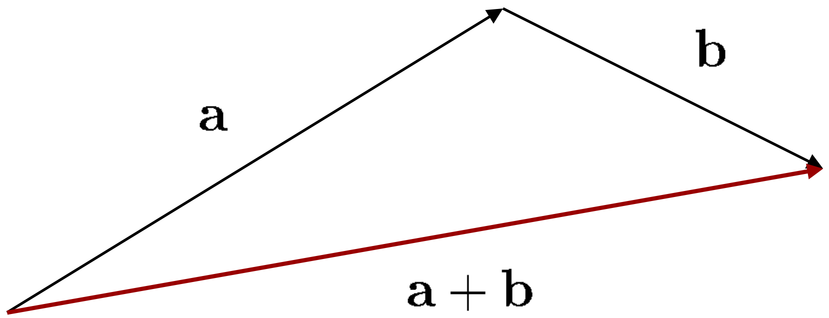


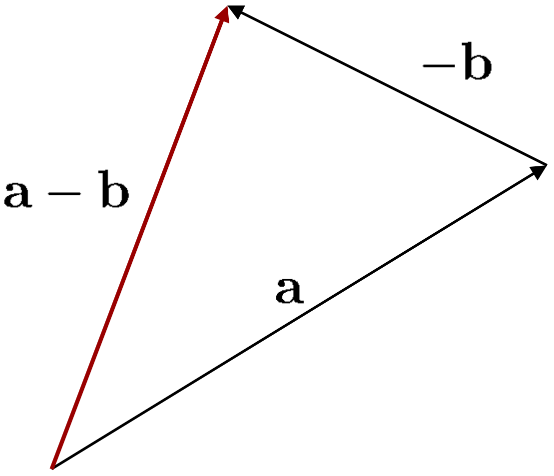


Vectors

Addition







Vectors

Linear combination

$$s\mathbf{a} + t\mathbf{b}, \quad s, t \in \mathbf{R}$$

$$\sum_{i=1}^{n} s_i \mathbf{a}_i \quad s_i \in \mathbf{R}$$

Vectors

Linear combination

$$s\mathbf{a} + t\mathbf{b}, \quad s, t \in \mathbf{R}$$

$$\sum_{i=1}^{n} s_i \mathbf{a}_i \quad s_i \in \mathbf{R}$$

Linearly dependent vectors

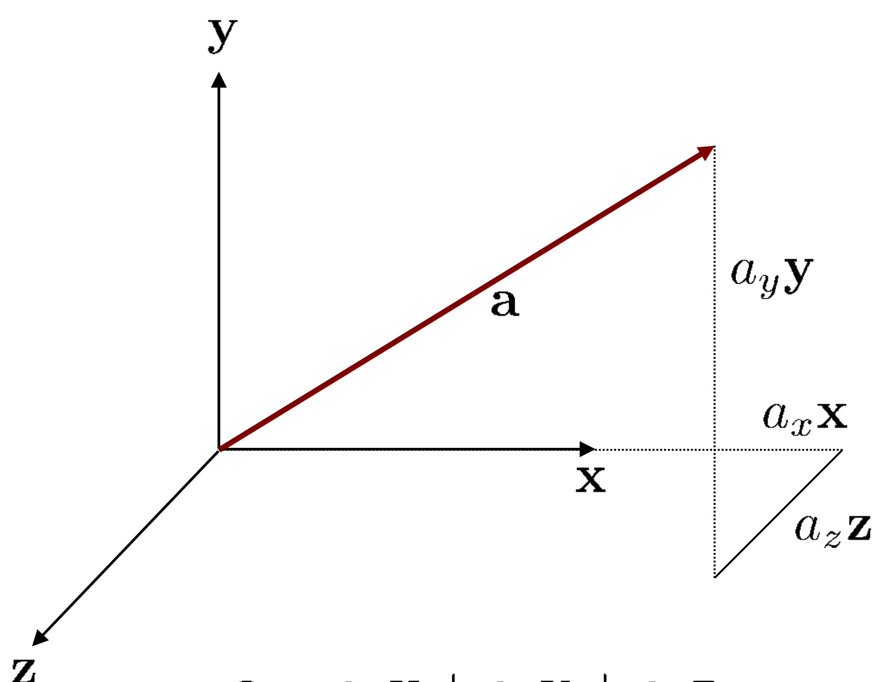
- A set of vectors $\mathbf{a}_i, i = 1 \ldots n$ is linearly dependent if there exist scalars s_i such that

$$\mathbf{a}_j = \sum_{i=1, i \neq j}^{n} s_i \mathbf{a}_i$$

- Otherwise, they are linearly independent

Coordinate systems

- Describe any vector with respect to three basis vectors $\mathbf{x}, \mathbf{y}, \mathbf{z}$



$$\mathbf{a} = a_x\mathbf{x} + a_y\mathbf{y} + a_z\mathbf{z}$$

- The basis vectors form a coordinate system

Coordinate systems

- Any three vectors that are linearly independent could be used as a basis
  - Different lengths
  - Not perpendicular to each other

Coordinate systems

- Any three vectors that are linearly independent could be used as a basis
  - Different lengths
  - Not perpendicular to each other
- Why linearly independent?
- Why exactly three vectors?
- Other coordinate systems?

Coordinate systems

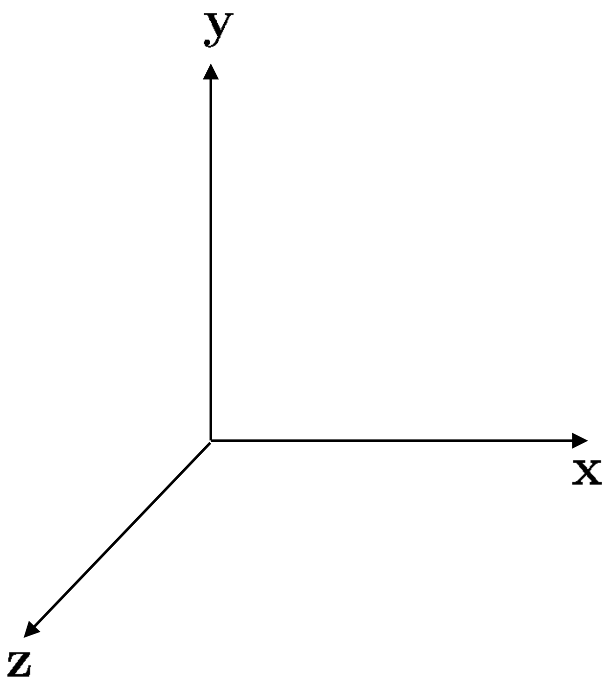
Cartesian coordinate systems

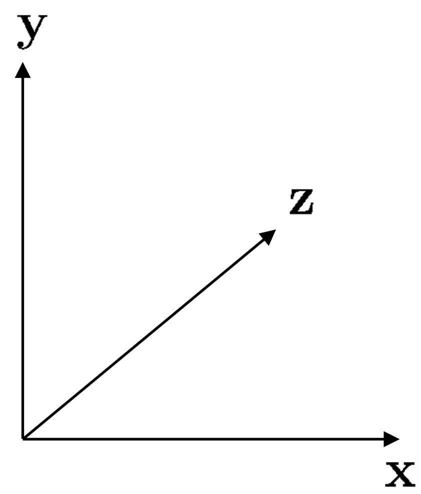
http://en.wikipedia.org/wiki/Cartesian_coordinate_system

- Basis vectors
  - Have unit length
  - Are perpendicular to each other
- Orthonormal

Coordinate Systems

Handedness





Right handed Left handed

Vector arithmetic using coordinates

$$\mathbf{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \qquad \mathbf{b} = \begin{bmatrix} b_x \\ b_y \\ b_z \end{bmatrix}$$

$$\mathbf{a} + \mathbf{b} = \begin{bmatrix} a_x + b_x \\ a_y + b_y \\ a_z + b_z \end{bmatrix} \qquad \mathbf{a} - \mathbf{b} = \begin{bmatrix} a_x - b_x \\ a_y - b_y \\ a_z - b_z \end{bmatrix}$$

$$-\mathbf{a} = \begin{bmatrix} -a_x \\ -a_y \\ -a_z \end{bmatrix} \qquad s\,\mathbf{a} = \begin{bmatrix} s a_x \\ s a_y \\ s a_z \end{bmatrix}$$

Vector Magnitude

- The magnitude (length) of a vector is:

$$|\mathbf{v}|^2 = v_x^2 + v_y^2 + v_z^2$$

$$|\mathbf{v}| = \sqrt{v_x^2 + v_y^2 + v_z^2}$$

- A vector with length=1.0 is called a *unit vector*
- We can also *normalize* a vector to make it a unit vector

$$\frac{\mathbf{v}}{|\mathbf{v}|}$$

- Unit vectors are often used as **surface normals**

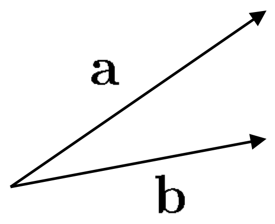
Questions?

Dot product

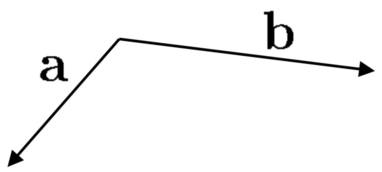
- Scalar value that tells us something about the relationship between two vectors
  - Product of lengths of vectors and cosine of angle between vectors
- Definition does not refer to a coordinate system
  - Result is independent of Cartesian coordinate system

Dot product

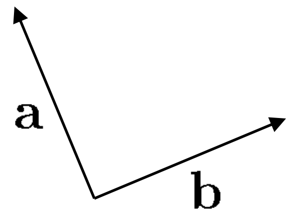
- If $\mathbf{a}\cdot\mathbf{b} > 0$ then $\theta < 90^\circ$
  - Vectors point in the same general direction



- If $\mathbf{a}\cdot\mathbf{b} < 0$ then $\theta > 90^\circ$
  - Vectors point in opposite direction



- If $\mathbf{a}\cdot\mathbf{b} = 0$ then $\theta = 90^\circ$
  - Vectors are perpendicular
  - (or one or both of the vectors is degenerate (0,0,0))



Dot product using coordinates

- Result is independent of coordinate system!

$$\mathbf{a} \cdot \mathbf{b} = \sum a_i b_i$$

$$\mathbf{a} \cdot \mathbf{b} = a_x b_x + a_y b_y + a_z b_z$$

$$\mathbf{a} \cdot \mathbf{b} = |a||b| \cos \theta$$

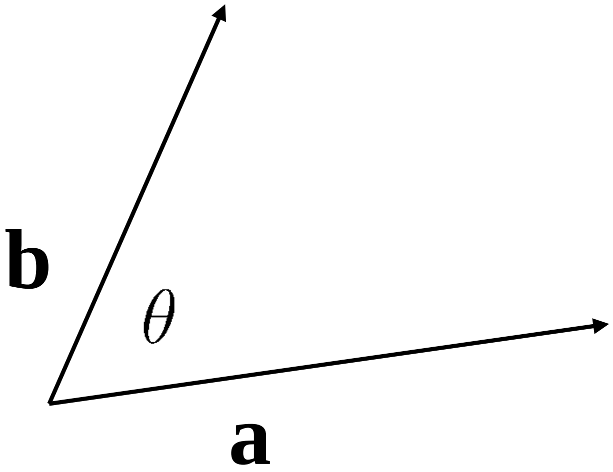
- What is the meaning of $\mathbf{a} \cdot \mathbf{a}$?

Angle between vectors

$$\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}|\cos\theta$$

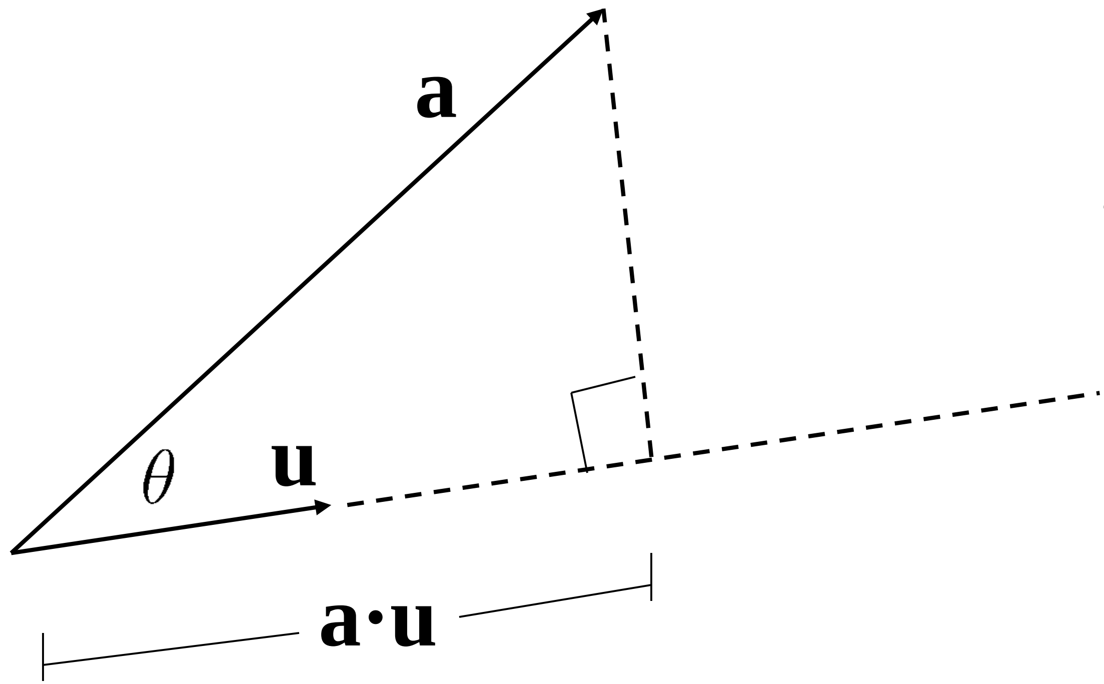
$$\cos\theta = \left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|} \right)$$

$$\theta = \cos^{-1}\left(\frac{\mathbf{a} \cdot \mathbf{b}}{|\mathbf{a}||\mathbf{b}|} \right)$$



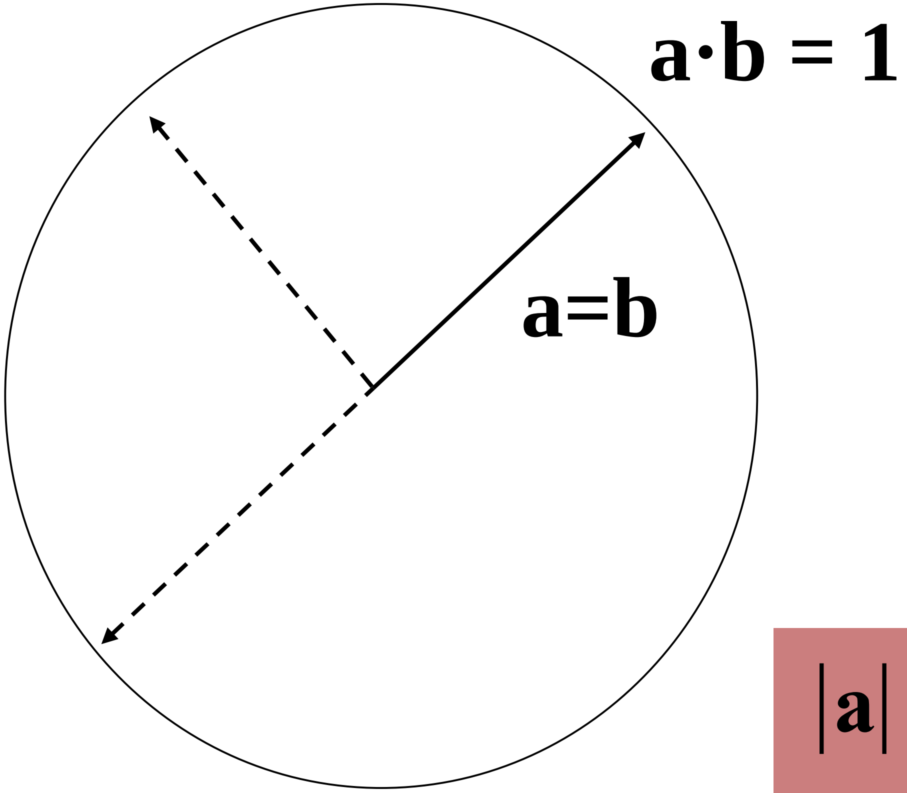
Dot products with unit vector

- If $|\mathbf{u}|=1.0$ then $\mathbf{a}\cdot\mathbf{u}$ is the length of the *orthogonal projection* of $\mathbf{a}$ onto $\mathbf{u}$



$$\mathbf{a}\cdot\mathbf{u} = |\mathbf{a}|\cos\theta$$

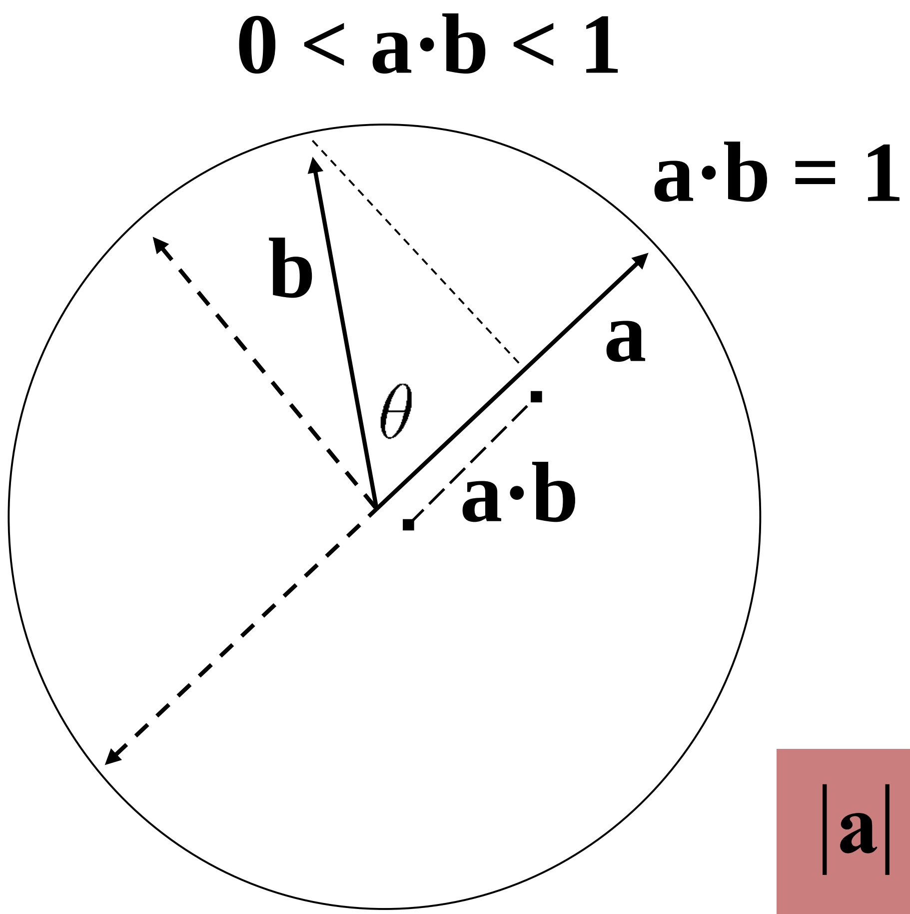
Dot products with unit vectors



$$|\mathbf{a}| = |\mathbf{b}| = 1.0$$

$$\mathbf{a} \cdot \mathbf{b} = \cos(\theta)$$

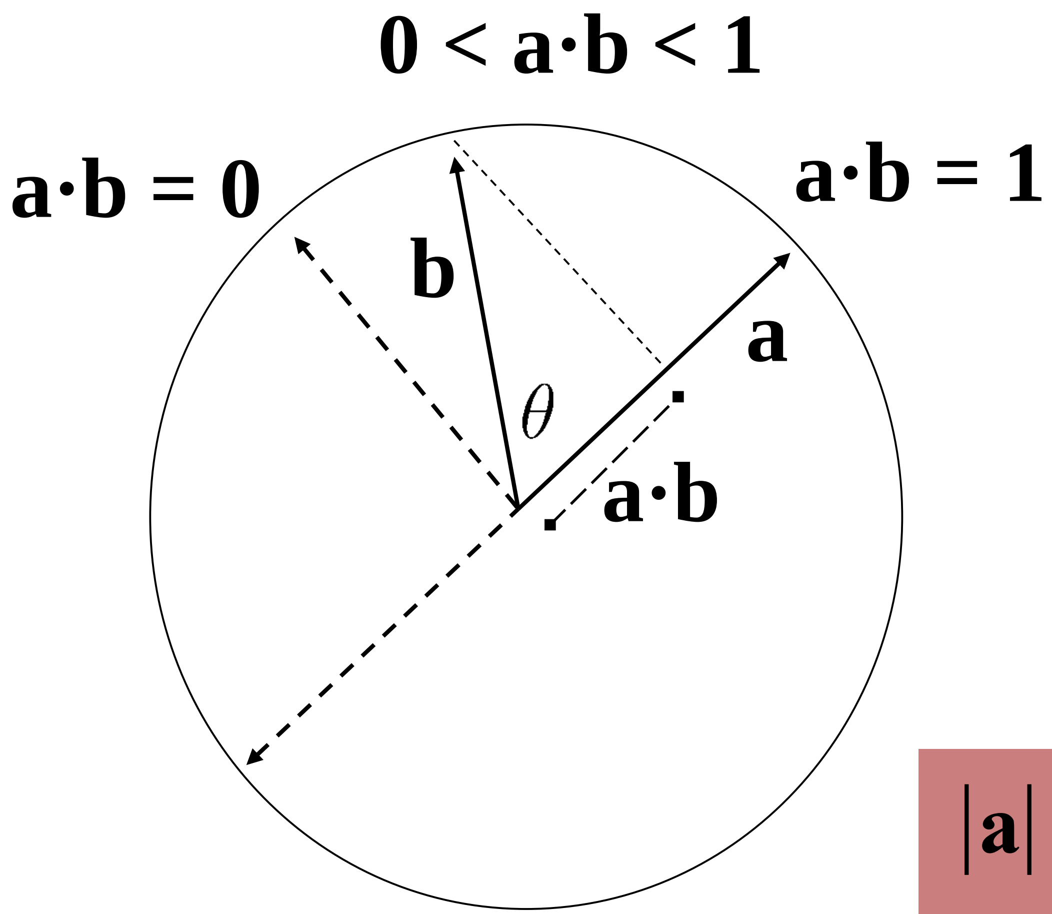
Dot products with unit vectors



$$|\mathbf{a}| = |\mathbf{b}| = 1.0$$

$$\mathbf{a} \cdot \mathbf{b} = \cos(\theta)$$

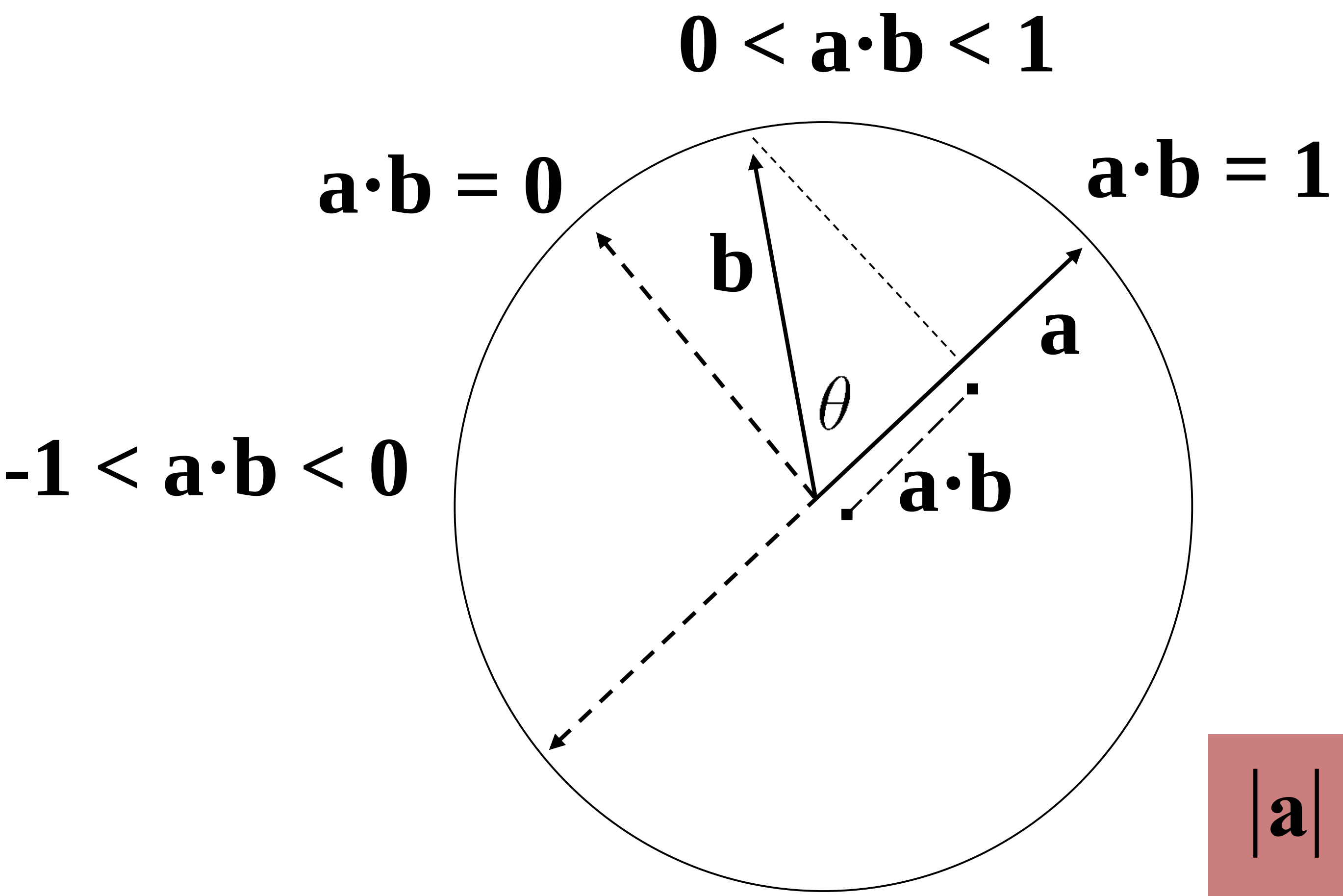
Dot products with unit vectors



$$|\mathbf{a}| = |\mathbf{b}| = 1.0$$

$$\mathbf{a} \cdot \mathbf{b} = \cos(\theta)$$

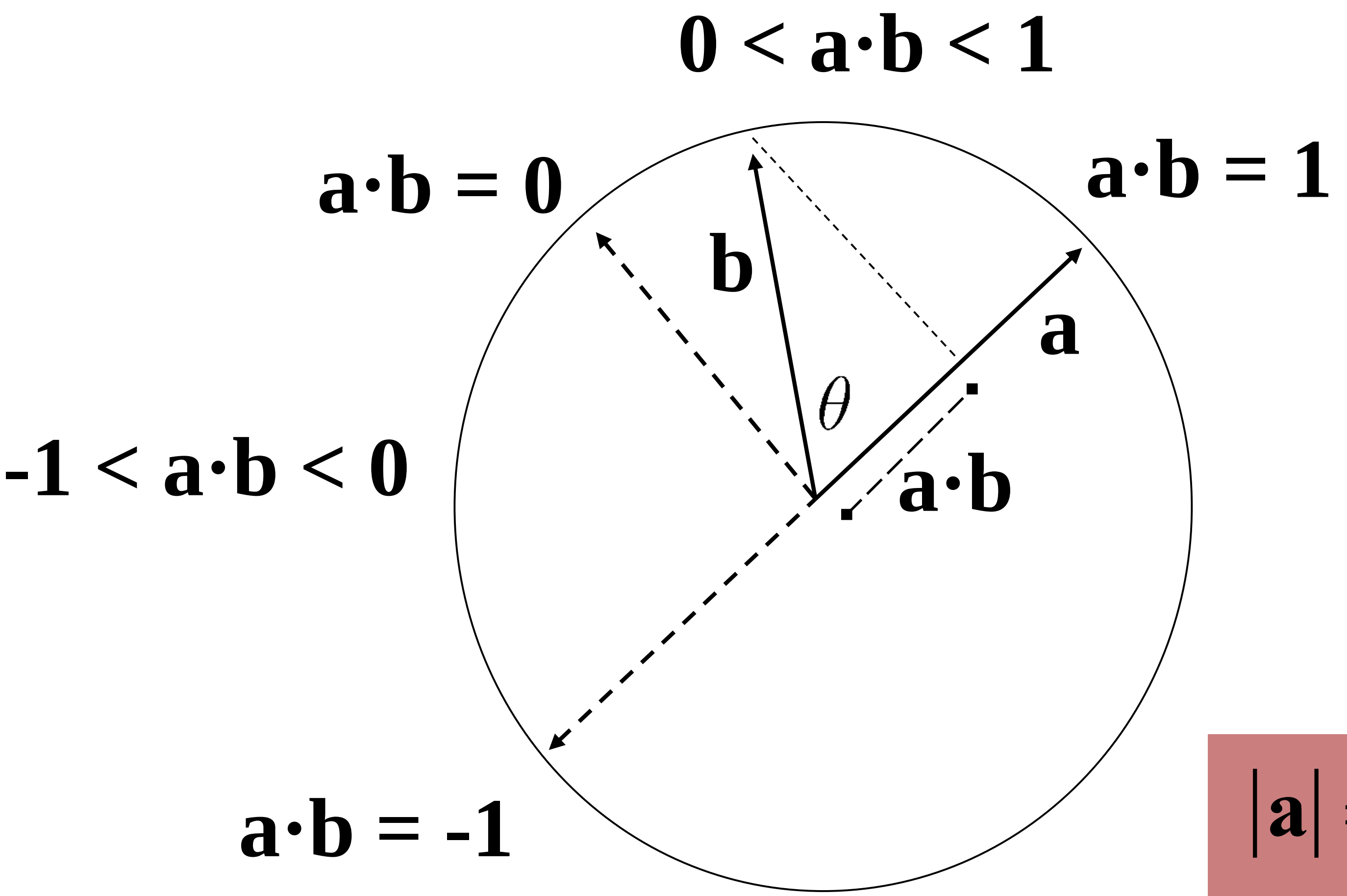
Dot products with unit vectors



$$|\mathbf{a}| = |\mathbf{b}| = 1.0$$

$$\mathbf{a} \cdot \mathbf{b} = \cos(\theta)$$

Dot products with unit vectors

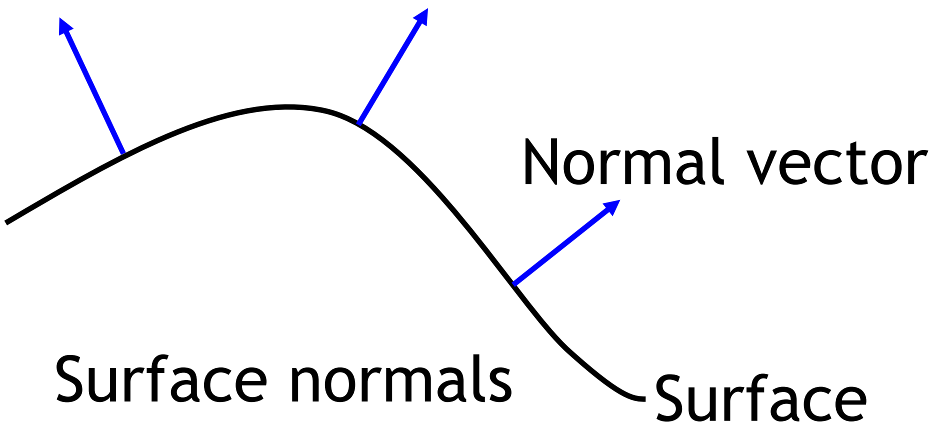


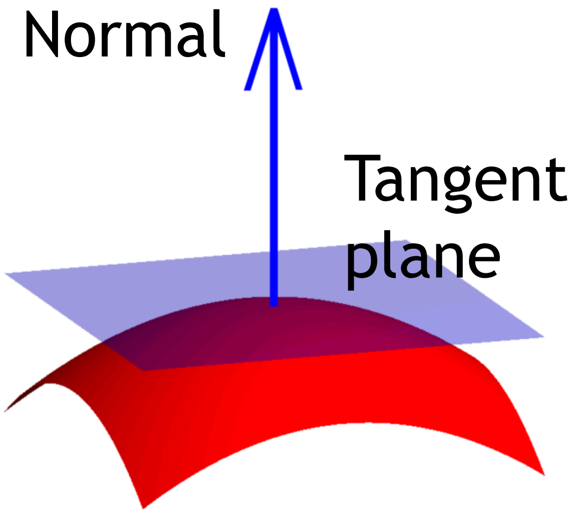
$$|\mathbf{a}| = |\mathbf{b}| = 1.0$$

$$\mathbf{a} \cdot \mathbf{b} = \cos(\theta)$$

Surface normals

- Vectors are direction and length in 3D
- Can describe
  - Difference between two 3D points
  - Speed of an object
  - Surface normals: directions perpendicular to tangent plane of surfaces





http://en.wikipedia.org/wiki/Normal_(geometry)

Representing planes

- A plane can be defined by
  - Its closest distance to the origin
  - and its normal vector
- How can we determine if a point lies on the plane using the dot product?

Cross product

- Written as **a** x **b**
- A vector perpendicular to **a** and **b**
  - In the direction defined by the right hand rule
  - Length is area of parallelogram spanned by **a** and **b**
- Definition does not refer to coordinate system!

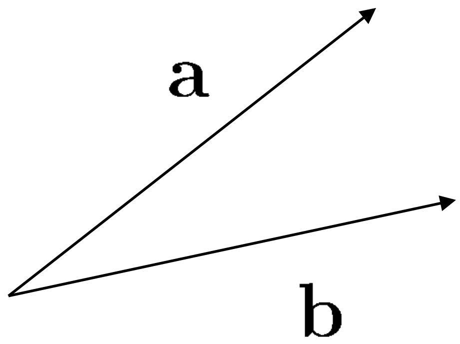
Cross product

- If vectors **a, b** are unit length and perpendicular, then **a, b, a** x **b** is a right handed coordinate system
- **a** x **b** = -(**b** x **a**)

Cross product

$\mathbf{a} \times \mathbf{b}$ is a vector perpendicular to both $\mathbf{a}$ and $\mathbf{b}$, in the direction defined by the right hand rule

Vectors $\mathbf{a}, \mathbf{b}$ lie in the plane of the projection screen. Does $\mathbf{a} \times \mathbf{b}$ point towards you or away from you? What about $\mathbf{b} \times \mathbf{a}$?



Cross product

$\mathbf{a} \times \mathbf{b}$ is a vector *perpendicular* to both $\mathbf{a}$ and $\mathbf{b}$, in the direction defined by the right hand rule

$$|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}||\mathbf{b}|\sin\theta$$

$|\mathbf{a} \times \mathbf{b}| =$ area of parallelogram $\mathbf{ab}$

$|\mathbf{a} \times \mathbf{b}| = 0$ if $\mathbf{a}$ and $\mathbf{b}$ are parallel (or one or both degenerate)

Cross product

- Using coordinates

$$\mathbf{a} \times \mathbf{b} = \begin{bmatrix} a_y b_z - a_z b_y \\ a_z b_x - a_x b_z \\ a_x b_y - a_y b_x \end{bmatrix}$$

Questions?

Coming up

Exercise session

- Introduction to the Java base code
- Representation of 3D shapes using triangle meshes

Next class

- Matrices and transformations