**Computer Graphics**

**Time:** Tuesday 5-6:50pm  
**rm 102, Warren Weaver Hall**

**Instructor:** Denis Zorin,  
**office hours Tuesday, 3-4 or by appointment**

**Class-related (also read by TA, if there is one):**  
intro-graphics@mrl.nyu.edu

**Confidential and really, really urgent**  
dzorin@mrl.nyu.edu

**Web page**  
http://www.mrl.nyu.edu/dzorin/cg05

---

**What is this class about?**

This is a **programming class**.  
It is about algorithms that are used to create computer graphics images.

We will **not learn how to use animation or rendering software** to create animations. Our goal is to learn the basics that are necessary to develop such software.
Computer Graphics

Prerequisites

- Basic linear algebra and calculus
- Solid C/C++ programming
- no Java

Hardware/Software

- PC, Linux or Windows, MacOS X
- other requires permission
- need OpenGL and GLUT; instructions for installation will be available

Grading

- 1-2 written assignments
- 6 programming assignments (subject to change)

Programming

- up to 30% quality of code
- 50% off for crashes without assert
Computer Graphics

- Late policy:
  - 24 hours: 20% off
  - 48 hours: 40% off
  - 72 hours: 80% off

Extensions are for exceptional circumstances only and have to be requested in advance, meaning at least 4 days before the due date;

NO extensions on the last day.

Prerequisites

Programming:
Good programming skills are essential.
Good working knowledge of C++ (or at least C) is assumed.
The programming load is high; the grade is primarily determined by programming.

Math:
Elementary geometry and linear algebra.
Topics

- Emphasis on OpenGL rendering pipeline
- Scan conversion
- Image processing
- Basic modeling
- Lighting
- Rendering algorithms

What is computer graphics?

- **Computer science**: software and hardware systems, vision, computational geometry
- **Mathematics**: transformations, curves, surfaces, PDEs, numerical integration
- **Physics**: light, dynamics
- **Psychology**: perception
- **Art**
Applications

Entertainment
- Animation and special effects
- Games

CAD
Scientific Visualization
Medicine
System Visualization

Animation
Games

Games
Games

Merge real and virtual objects
Computer-Aided Design

Conceptual design

Simulation
Visualization

Thunderstorm Simulation
NASA’s FAST

Airflow around a Harrier Jet
FAST System
Nasa Ames

Visible Human

© 1995 IPDM University of Hamburg, Germany

© 2005, Denis Zorin
Software Visualization

SeeSoft
S. Eick
Lucent
Shows code age:
new -> red
old -> blue

Modeling

Create the environment
- shapes
- appearance
- views

make it move
- define parameters
- compute how object shape, position, appearance changes
Rendering

Physics-based: simulate light propagation
Empirical: use trial and error to get pictures that are good enough
Nonphotorealistic: imitate artistic styles
Image-based: generate directly from photos or video (no modeling)

Image Processing

Output: typically a raster device (CRT, LCD, printer)
Discrete colors/intensities
Need to convert continuous data to discrete
Combine real and synthetic
Geometry review, part I

Vectors and points
- points and vectors
- Geometric vs. coordinate-based (algebraic) approach
- operations on vectors and points

Lines
- implicit and parametric equations
- intersections, parallel lines

Planes
- implicit and parametric equations
- intersections with lines
**Geometry vs. coordinates**

**Geometric view:** a vector is a directed line segment, with position ignored.

Different line segments (but with the same length and direction) define the same vector.

A vector can be thought of as a translation.

**Algebraic view:** a vector is a pair of numbers.

---

**Vectors and points**

Vector = directed segment with position ignored

Operations on points and vectors:
- point - point = vector
- point + vector = point

\[ v + w \]

\[ -w \]

\[ w \]

\[ cw \]
Dot product

Dot product: used to compute projections, angles and lengths.

Notation: \((w \cdot v) = \) dot product of vectors \(w\) and \(v\).

\[
\begin{align*}
(w \cdot v) &= |w| |v| \cos \alpha, \\
|v| &= \text{length of } v
\end{align*}
\]

Properties:

- if \(w\) and \(v\) are perpendicular, \((w \cdot v) = 0\)
- \((w \cdot w) = |w|^2\)
- angle between \(w\) and \(v\): \(\cos \alpha = (w \cdot v)/(|w||v|)\)
- length of projection of \(w\) on \(v\): \((w \cdot v)/|v|\)

Coordinate systems

For computations, vectors can be described as pairs (2D), triples (3D), ... of numbers.

Coordinate system (2D) =

point (origin) + 2 basis vectors.

Orthogonal coordinate system:

basis vectors perpendicular.

Orthonormal coordinate system:

basis vectors perpendicular and of unit length.

Representation of a vector in a coordinate system:

2 numbers equal to the lengths (signed) of projections on basis vectors.
Operations in coordinates

\[ v = (v \cdot e_x)e_x + (v \cdot e_y)e_y \]

works only for orthonormal coordinates!

\[ v = v_x e_x + v_y e_y = [v_x, v_y] \]

Operations in coordinate form:

\[ v + w = [v_x, v_y] + [w_x, w_y] = [v_x + w_x, v_y + w_y] \]

\[-w = [-w_x, -w_y] \]

\[ \alpha w = [\alpha w_x, \alpha w_y] \]

Dot product in coordinates

\[ |v||w| \cos \alpha = |v||w| \cos(\alpha_v - \alpha_w) \]

\[ = |v||w| (\cos \alpha_v \cos \alpha_w + \sin \alpha_v \sin \alpha_w) \]

\[ = |v||w| (v_x w_x / |v||w| + v_y w_y / |v||w|) \]

\[ = v_x w_x + v_y w_y \]

Linear properties become obvious:

\[(v+w) \cdot u = (v \cdot u) + (w \cdot u) \]

\[(a v \cdot w) = a(v \cdot w) \]
3D vectors

Same as 2D (directed line segments with position ignored), but we have different properties.

In 2D, the vector perpendicular to a given vector is unique (up to a scale).

In 3D, it is not.

Two 3D vectors in 3D can be multiplied to get a vector (vector or cross product).

Dot product works the same way, but the coordinate expression is

\[(v \cdot w) = v_x w_x + v_y w_y + v_z w_z\]

Vector (cross) product

\[v \times w\] has length \[|v||w| \sin \alpha\]

= area of the parallelogram with two sides given by \(v\) and \(w\), and is perpendicular to the plane of \(v\) and \(w\).

\[(v + w) \times u = v \times u + w \times u\] Direction (up or down) is determined by the right-hand rule.

\[(cv) \times w = c(v \times w)\]

\[v \times w = -w \times v\] unlike a product of numbers or dot product, vector product is not commutative!
Vector product

Coordinate expressions

\( \mathbf{v} \times \mathbf{w} \) is perpendicular to \( \mathbf{v} \), and \( \mathbf{w} \):

\[
(u \cdot v) = 0 \quad \quad (u \cdot w) = 0
\]

the length of \( u \) is \( |\mathbf{v}| |\mathbf{w}| \sin \alpha : \\
(u \cdot u) = |\mathbf{v}|^2 |\mathbf{w}|^2 \sin^2 \alpha = |\mathbf{v}|^2 |\mathbf{w}|^2 (1 - \cos^2 \alpha) \\
= |\mathbf{v}| |\mathbf{w}| (|\mathbf{v}| |\mathbf{w}| - (\mathbf{v}, \mathbf{w}))
\]

Solve three equations for \( u_x, u_y, u_z \)

---

Vector product

Physical interpretation: torque

\[ \text{torque} = \mathbf{r} \times \mathbf{F} \]

axis of rotation

force \( \mathbf{F} \)

displacement \( \mathbf{r} \)
Vector product

Coordinate expression:

\[
\begin{vmatrix}
e_x & e_y & e_z \\
v_x & v_y & v_z \\
w_x & w_y & w_z \\
\end{vmatrix}
= \begin{vmatrix}
v_y & v_z \\
w_x & w_z \\
\end{vmatrix}
- \begin{vmatrix}
v_x & v_z \\
w_x & w_z \\
\end{vmatrix}
+ \begin{vmatrix}
v_x & v_y \\
w_x & w_y \\
\end{vmatrix}
\]

Notice that if \( v_z = w_z = 0 \), that is, vectors are 2D, the cross product has only one nonzero component (z) and its length is the determinant

\[
\begin{vmatrix}
v_x & v_y \\
w_x & w_y \\
\end{vmatrix}
\]

Vector product

More properties

\((a \cdot (b \times c)) = b(a \cdot c) - c(a \cdot b)\)

\(((a \times b) \cdot (c \times d)) = (a \cdot c)(b \cdot d) - (b \cdot c)(a \cdot d)\)