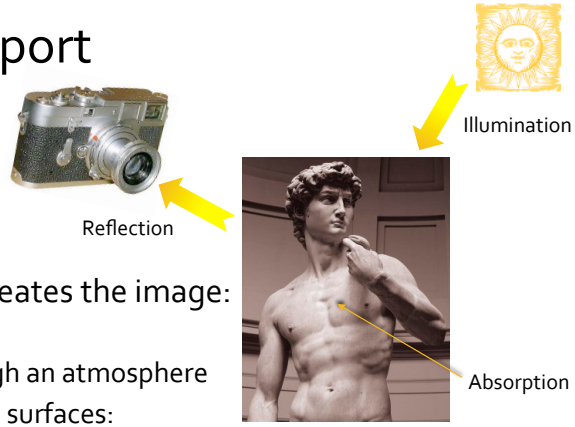
EECS 487: Interactive Computer Graphics

Lecture 1: Introduction to
- Computer Graphics
- the Course

Light Transport



Reflected light creates the image:
- light leaves source
- light travels through an atmosphere
- light interacts with surfaces: absorption, reflection, refraction
- light arrives at receiver (eye, film, sensor)

In the real world, light transport "happens naturally"

Computer Graphics

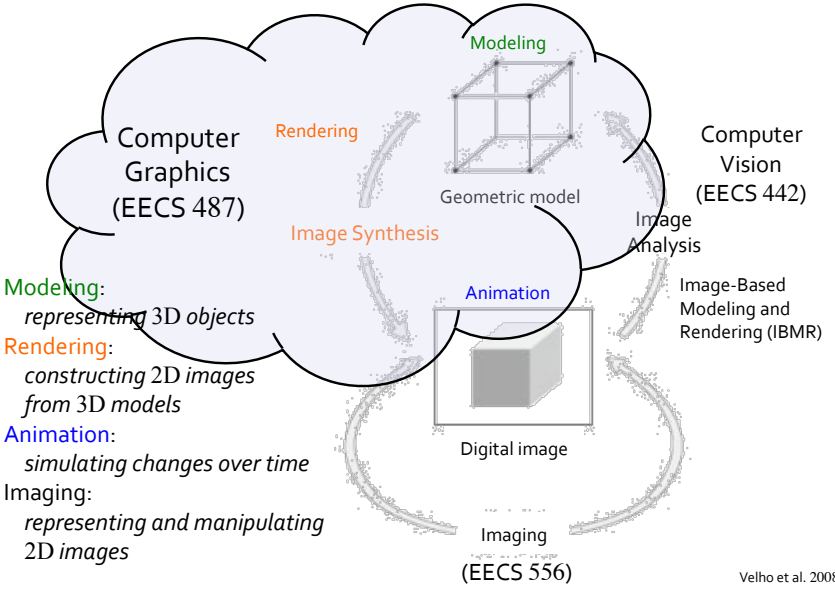
Algorithms for image synthesis
- define a set of objects
- arrange them in a scene
- illuminate the scene

  ⇒ light emission, transmission, absorption, reflection, refraction must all be simulated!
- convert the 3D scene into a 2D view

All done in a computer, operating on an imaginary scene with internal representations for simulated objects, lights, and cameras

Emulating light transport in this imaginary scene requires enormous amount of mathematical, physics, and programming sophistication

What This Course Covers

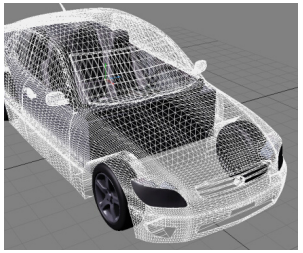


Geometric Modeling

Why do geometric modeling?

What is a 3D model?

How are objects represented internally in a computer?



"Computer graphics models are like movie sets in that usually only the parts that will be seen are actually built."
— *Cook, Carpenter, Catmull*

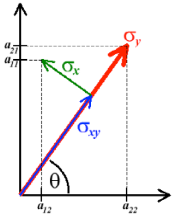
Geometric Modeling

How are the models created?

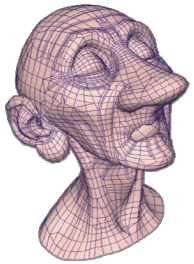
first "physically-realistic" computer generated image

We don't cover model creation . . . (ARTDES 300)

Geometric Modeling Topics



How are the models represented?
- points, vectors, triangles
  - interpolation, barycentric coordinates
  - polygonal meshes
- implicit curves and surfaces
- splines
- subdivision surfaces



How are the models arranged in a scene?
- 2D and 3D rigid transformations
- 3D viewing and perspective projections, homogeneous coordinates
- modeling hierarchies

We will briefly cover procedural models

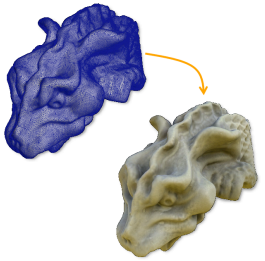
Rendering

What is rendering?

Synthesizing a 2D image from a 3D geometric model

Given:
- geometry
- lighting and shading
- and material:
  - colors
  - textures

produce an image

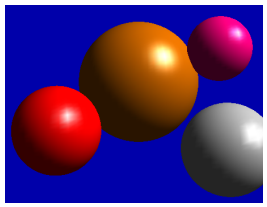


Determine how much light is reflected from each point to the viewer

"All it takes is for the rendered image to look right."
— *Blinn*

Gargoyle: Daschbacher&Kautz o8

Rendering Topics



Determine how light interacts with objects/surfaces: what color each pixel

- scan-line conversion, anti aliasing
- illumination and reflection models for different surfaces
- GPU programming
- texture mapping
- ray tracing and shadowing
- radiosity and global illumination

Different rendering objectives:

- Photorealism
- Interactive performance
- Artistic expression
- CAD and Scientific Visualization

Photorealistic Rendering

Physically-based simulation of light

Realistic global illumination:

- shadows, caustics, color bleed

Be as true to the physics of light *as necessary*

Slow, minutes to hours per image

Used for special effects, movies

- big budgets, tight schedules ⇒ server farm

(not covered . . .)

Interactive Rendering

Want to be "as photorealistic as possible"

But must produce each image within msecs

Common tricks to speed up lighting computations:

- "faking it" in image/screen space, e.g., shadow maps
- pre-computation, e.g., environment map
- probabilistic approximations
- exploits GPUs

Games ⇒ ~$200 consumer electronics!

Artistic Rendering

Stylized, non-photo realistic
Artwork
Illustration
Data visualization
(not covered . . .)

Data Visualization

Presentation and interpretation of data
- makes vast quantities of data accessible
- perhaps the only way the data could be interpreted

New data representations

Precision and correctness

No cheating allowed!

Physics Simulation

Cloth as mass-spring network

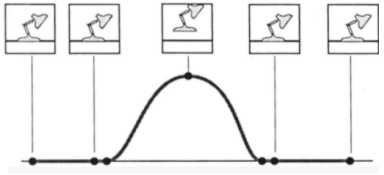
Fluid dynamics

(not covered . . .)

Animation Topics

How to represent motion?
- position, facing, etc. as a function of time

How to specify and control motion?
- keyframing: generate poses by hand or motion capture, control interpolation with spline
- dynamics: particle systems, physically-based simulation: faces, skin, hair, cloth, fluid
- behavioral simulation ("brains")

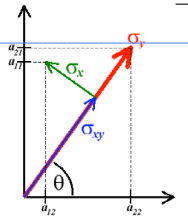


How Much Math?

"Computer graphics: Mathematics made visible."
— *James*

Vectors and matrices:
2D, 3D, 4D

Dot and cross products



Examples:
- Dot product: $\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + a_3 b_3$
- What is $\arccos\left(\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}\right)$, where **a** and **b** are vectors?
- What is $0.5|(\mathbf{q}-\mathbf{p}) \times (\mathbf{r}-\mathbf{p})|$, where **p**, **q**, **r** are points in 3D?
- What is described by this set of points in 3D: $\alpha\mathbf{p} + (1-\alpha)\mathbf{q}$, where $0 \le \alpha \le 1$
- Read the course note on *Vectors and Matrices*

3D Transformations

"Much of graphics is just translating math directly into code. The cleaner the math, the cleaner the resulting code."

— *Shirley*

3D transformations used in
- positioning of the camera
- positioning of lights within a scene
- positioning of objects within a scene
- transformation of objects in 2D and 3D

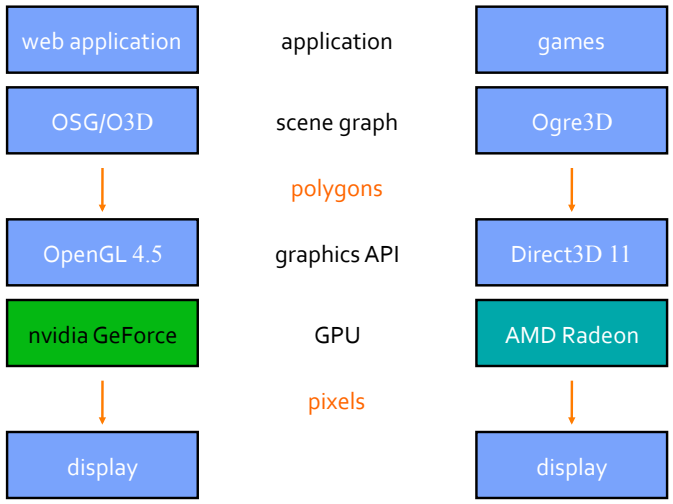
Most of these operations can be achieved by applying 4×4 matrices to 4D vectors
- linear algebra and homogeneous coordinates

Will try to introduce math "as needed", "just-in-time"

Notation

Description	Notation Example
angle	$\phi, \rho, \theta, \omega$
scalar	$a, b, t, u_k, v, w_{i,j}$
vector (handwritten)	$\mathbf{a}, \mathbf{u}, \mathbf{v}_s, \mathbf{h}(\rho), \mathbf{h}_z$ ($\vec{a}, \vec{u}, \vec{v}_s, \vec{h}(\rho), \vec{h}_z$) or $\langle v_x, v_y, v_z \rangle$
point (alternative, handwritten)	$\mathbf{p}, \mathbf{q}, \mathbf{r}$ $(P, Q, R, \tilde{p}, \tilde{q}, \tilde{r})$ or $\langle p_X, p_Y, p_Z \rangle$
unit or normal vector (handwritten)	$\hat{\mathbf{u}}$ $(\hat{u})$
line segment	$\mathbf{uv}$, $\overline{a_i b_j}$, $\overline{PQ}$
plane	$\pi : \mathbf{n} \cdot \mathbf{p} + d = 0$
triangle	$\triangle \mathbf{v}_0 \mathbf{v}_1 \mathbf{v}_2$ $(\triangle PQR)$
matrix (handwritten)	$\mathbf{M}, \mathbf{R}_x(\rho), \mathbf{I}, \mathbf{T}(\mathbf{t})$ $(\hat{M}, \hat{R}_x(\rho), \hat{I}, \hat{T}(\vec{t}))$
dot product	$\mathbf{u} \cdot \mathbf{v}$
cross product	$\mathbf{u} \times \mathbf{v}$
transpose of vector $\mathbf{v}$ or matrix $\mathbf{M}$	$\mathbf{v}^T$ or $(v_x \; v_y \; v_z)^T$ and $\mathbf{M}^T$
direct product (of color vectors)	$\mathbf{u} \otimes \mathbf{v} = (u_x v_x \;\; u_y v_y \;\; u_z v_z)^T$
unary, perpendicular dot product	$\mathbf{v}^\perp = (-v_y \; v_x)^T$
absolute value of a scalar	$\lvert a \rvert$
length or norm or magnitude of a vector	$\lvert \mathbf{v} \rvert$ or $\lVert \mathbf{v} \rVert$ $(\lVert \vec{a} \rVert)$
determinant of a matrix	$\lvert \mathbf{M} \rvert$

Graphics Software Stacks



Akeley

What this Course is NOT About

We do NOT cover:
- webpage design/rich Internet application: HTML, CSS, Ajax, Flex/Flash, Silverlight, etc.
- graphics packages for:
  - presentation: PowerPoint, Keynote
  - 2D drawing/painting: Photoshop, GIMP, OmniGraffle
  - 3D modeling: 3dsmax, Blender, Maya, AutoCAD
  - scene graph management and rendering: Renderman, Ogre3D, OpenSceneGraph, Java3D, O3D, three.js
- GUI programming:
  - Qt, Gtk, Carbon, Cocoa, Quartz, .NET, X11, etc.
- Mobile graphics programming (we will introduce OpenGL ES and WebGL)

Teaching Staff

Instructor: Sugih Jamin
email: jamin@eecs.umich.edu
Office: 4737 BBB
Office hours: MWF 10:30-11, Th 11:30-12 and by appt.

GSI: Lisa Dion
uniqname: lisadion
Office hours: Tu 4-5, and by appt. in Learning Center of BBB
Extra lab grading time: Thu 6-7
Will lead lab sessions

Grader: Haohuan Wang (haohuanw)

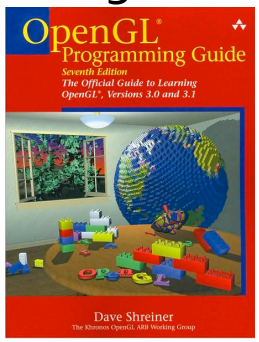
Course Web Site

http://www.eecs.umich.edu/~sugih/courses/eecs487/

Last year's lecture slides posted on web site, some will be updated *after* lecture

- always grab a fresh copy if you need to consult a lecture note
- don't bother to keep a printed copy

Recommended Readings



Redbook

Available online?
http://it-ebooks.info/book/2138/

Prerequisites:
- EECS 281
- C++
- Linear algebra and basic geometry

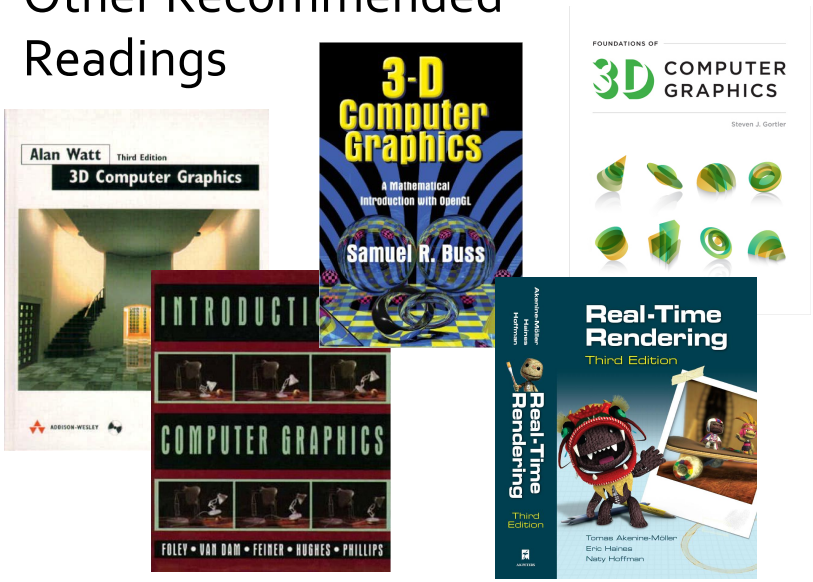
Recommended Readings

OpenGL:
- OpenGL API Tables accessible from the course web site
- many other books and sources listed on course web site (Links)

We only touch upon Direct3D* tangentially (the fundamentals should be familiar)

*Direct3D is the 3D graphics API of DirectX, which includes other APIs such as for input and audio control. The closest equivalent of DirectX in the free world is SDL

Other Recommended Readings



How to Read the Syllabus Page

Mon 09/22 Assigned HW1 PA0 Due Linked to MFile	2D transforms: linear and affine transforms Recommended reading: • TP3 Sections 3.1-3.6, 3.10, Appendix A. Demoes: • 2D Geometric Transformations • Brown's Affine Transforms Additional reading: • Course note on *Transforms*, 2009. Optional reading: • *Matrices and Determinants*.

I won't be nagging you; check the syllabus yourself

Labs posted 7-5 days prior to due date (announced on ctools)

Course Directory (ITCS AFS/MFile)

https://mfile.umich.edu/?path=/afs/umich.edu/class/eecs487/f15/

```
FILES/
FOLDERS/
```

Available now:

```
lab0.tgz
```

Course Announcements

Announcement page on course web site (ctools)

Both course web site and Announcement page are "required readings"

Send *both* Lisa and myself email if you have any questions

We will post FAQ's on the Announcements page, please check it first before asking your questions

Grading Policy

- 1 Final Exam*: 15% Mon 12/21, 10:30-12:30
- 1 Midterm Exam*: 15% Mon 10/26, 6-8 pm, 1013 EECS
- 2 Homeworks and *n* Pop Quizzes: 16%

Hand in hardcopies

- 10 Lab Assignments: 20% Graded in person
- 4 Programming Assignments: 32% Turn in online
- Class Participation: 2%

Do not email us any of your assignments!

* Make-up time must be requested within 2 weeks from today

Lab Grading

During lab session and lab grading time ONLY

No lab grading in lecture

Lab grading during office hours only if no other students are waiting with questions

Do NOT accost Lisa outside the lab

Remote Lab Submission

Upload to 487 CTools' Drop Box:

the result of running

```
% openssl sha1 [files …]
```

Windows (not MD5):
http://www.nirsoft.net/utils/hash_my_files.html
or install Cygwin along with Net→openssh package

Once you have computed the SHA1, don't make any more changes to the files, or your SHA1 will become invalid

Must still be graded in person by Lisa before the next lab is due or get a zero for the lab

Computer Support

We provide `Makefile`s, but not IDE project files

You need to have your own laptop with at least OpenGL 2.1 and GLSL 1.2 support (later versions ok)

You'll need to have OpenGL and GLUT installed on your laptop, see relevant section of http://www.eecs.umich.edu/~sugih/courses/eecs487/common/notes/ide/

Collaboration

- All work must be done individually
- Cheating and plagiarizing are not tolerated
- To pass off the implementation of an algorithm as that of another is also considered cheating:
  - e.g., insertion sort is *not* heap sort
  - if you can not implement a required algorithm, you *must inform the teaching staff when turning in your assignment*
- Homework: consultation of online and offline sources allowed, but must not be copied verbatim, you need to show that you have understood the material. Cite your sources, including classmates and roommates, but not teaching staff or required readings
- Exams: see course Grading Policy web page

Grading Policy

Re-grade:
- within 5 working days (except Final Exam, same day)
- written request
- whole work will be re-graded

Late days:
- 4 free late days in total for all programming assignments together
  - including weekends
  - NOT per assignment
- no need to inform us to use any of your free late days
- keep track of your own free late day usage

Help with PAs stops 2 days before due date

Late Penalty

Applied to HW and PAs *after* free late days are used up
Labs will not be accepted late

Penalty schedule:
- ≤ 24 hours: 4% of the assignment's total points
- ≤ 48 hours: 8+4=12%
- HW will not be accepted more than 2 days late
- ≤ 72 hours: 12+12=24%
- ≤ 96 hours: 16+24=40%
- PAs more than 4 days late will not be accepted

Example:
- PA worths 100 points, work late by 24 hours and 10 mins
  - if no free late days left: 12 points late penalty
  - if 1 free late day left: 8 points late penalty
- turning in HW after lecture has started is considered one day late