

174a Lecture 2

Today's Videos

SIGGRAPH trailers from 2017

Going backwards,

https://www.youtube.com/watch?v=3OGKh_9Rj_8

And

<https://www.youtube.com/watch?v=5YvIHREdVX4>




One Example of today's industry “hacks”

<https://www.youtube.com/watch?v=ct3vWWI86f8>

- Tools packaged with production software suites
- Video does not claim to use any math that converges to real-life physics formulas
 - Approximation is not convergence!





The Difficulties of Learning Computer Graphics

(And teaching it!)

Background

- Most of today's approaches to creating a low-level, math-based graphics program come with a substantial learning curve
 - Numerous complex steps
- In terms of preparation and learning, it is costly to build prototypes using low-level 3D graphics programs today



Background

- A common alternative is to:
 - Forgo low-level control
 - Employ overpowered industrial tools to wrap basic graphics functionality
 - Simple mathematics of projecting 3D triangles onto a 2D plane of pixels



Background

- Graphics beginners are faced with long lists of setup steps
 - Especially in the case of newer "shader-based" approaches.
 - These approaches are both more complex and harder to learn.




Background

- The graphics learning curve is extreme.
- Drawing just a single triangle requires:
 - Secondary "shader" programs
 - Multiple types of GPU memory management
 - At least three computer languages (in the case of WebGL).



WARNING

- Graphics tutorials online abound that still use outdated “pre-shader” coding paradigms
 - They're alluring due to their simplicity.
 - However, support for their older commands has been:
 - Removed from new graphics cards
 - Never existed in web browser implementations.
 - Eventually it dawns on newcomers that they must commit to learning the new way.
- 

Background

- Mandatory setup steps initialize the graphics card (GPU):
 - Give it the shader program code
 - Give it raw data buffers pertinent to the 3D scene
 - Obtain pointers to these in GPU memory.



Background

- Even after setup:
 - All the subsequent shape-drawing actions of the programmer are still cluttered with boilerplate code
 - Loading and switching between pointers to the GPU.
- Nothing is drawn without these steps. There is no built-in way to organize them.



Background

- Common graphics card interfaces exposed for doing the above steps are called:
 - DirectX
 - OpenGL
 - More widely available and more prevalent in education.



Background

- Regardless, graphics methods all follow a similar pattern.
- The phrase ``OpenGL program" is widely taken to imply C++ due to the ubiquity of the language in early graphics education, but it need not be:
 - Python, Java, and JavaScript can make the same OpenGL calls.



Background

- C++
- Python
- JavaScript
- Java
- 3D Graphics programming in one language feels familiar in all the others.



Background

- JavaScript is currently the only means of running code on browsers inside of web pages.
- When JavaScript is used, the OpenGL commands are called WebGL
 - They are still the same API function calls as would appear in C++.



Your Textbook

- Edward Angel's "Interactive Computer Graphics...with WebGL" is the leading gentle introduction for those wanting to do graphics programming.
- Used at:
 - UCLA and other schools
 - SIGGRAPH's official intro course
- Has supplemental code and links to demos that run on the web
- Chapter 2 is a suggested organization of WebGL programs
 - hello world
- Other web tutorials share program structure

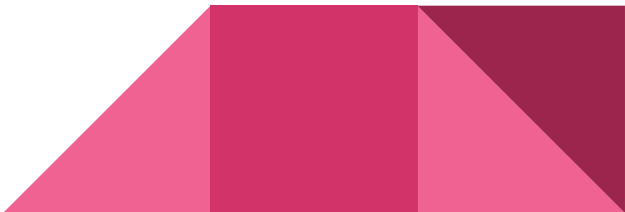


Edward Angel's The Case for Teaching Computer Graphics with WebGL: A 25-Year Perspective

- Described his rationale for eventually moving his helpful C++ based libraries over to WebGL.
 - Worsening learning curve of C++ graphics setup
 - Difficulty in setting up uniform C++ compiling environments for all students
 - Your all's inconsistent hardware



Edward Angel's The Case for Teaching Computer Graphics with WebGL: A 25-Year Perspective

- He found that WebGL has:
 - Comparable performance to C++
 - The advantages of a standardized environment on all platforms (including phones)
 - An interpreted code engine that aids development
 - Advanced coding tools built right into modern web browsers.
- 

Sounds good

- WebGL is the current best platform for graphics training.
 - Code examples on the web are more easily:
 - Run as demos
 - Analysed
 - Packaged with inline tutorials
 - Compiled to any machine
 - Debugged
 - Hosted
 - Shared
 - Remixed




JavaScript

- JavaScript's functional programming styles
 - Specific graphics applications
 - Also tend toward smaller total source code.



JavaScript

- We make full use of the 2015 “es6” version of JavaScript
 - Adds further brevity and power to the language.
 - Be sure to include “es6” in all Google searches!!
 - Performance workarounds such as WebAssembly and Emscripten can run at near native speeds.
 - JavaScript's benefits for prototyping and sharing code on the web are clear.
- 

Example sites that allow graphics programming right inside the browser

- JSFiddle
- Dwitter
- WebGL Playground
- Shadertoy
- The 174a web server



The Textbook's web demos

- <https://www.cs.unm.edu/~angel/WebGL/7E/Common/>



Linear Algebra Review

Linear Algebra: The Algebra of Vectors and Matrices (and Scalars)

Vector spaces

Matrix algebra

Coordinate systems

Affine transformations

Vectors

N-tuple of scalar elements

$$\mathbf{v} = (x_1, x_2, \dots, x_n), \quad x_i \in \mathbb{R}$$

Vector:

Bold lower-case

Scalar:

Italic lower-case

Vectors

N-tuple:

$$\mathbf{v} = (x_1, x_2, \dots, x_n), \quad x_i \in \mathbb{R}$$

Magnitude:

$$|\mathbf{v}| = \sqrt{x_1^2 + \dots + x_n^2}$$

Unit vectors

$$\mathbf{v} : |\mathbf{v}| = 1$$

Normalizing a vector

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

Operations with Vectors

Addition

$$\mathbf{x} + \mathbf{y} = (x_1 + y_1, \dots, x_n + y_n)$$

Multiplication with scalar (scaling)

$$a\mathbf{x} = (ax_1, \dots, ax_n), \quad a \in \mathbb{R}$$

Properties

$$\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$$

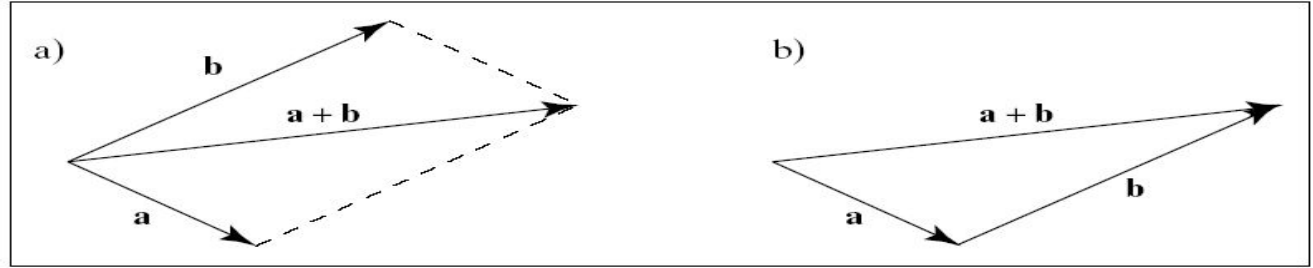
$$(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w})$$

$$a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}, \quad a \in \mathbb{R}$$

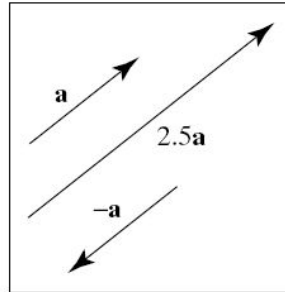
$$\mathbf{u} - \mathbf{u} = \mathbf{0}$$

Visualization of 2D and 3D Vectors

Addition

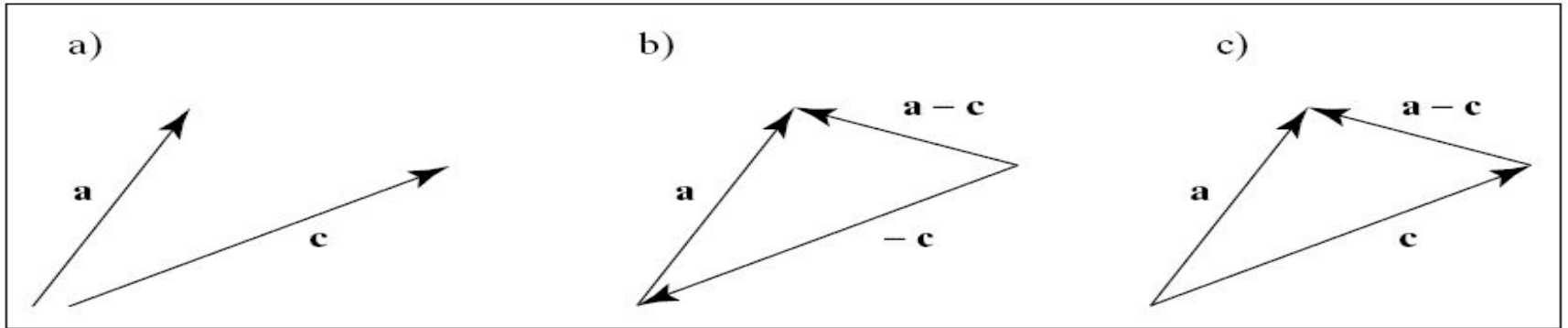


Scaling



Subtraction

Adding the negatively scaled vector



Linear Combination of Vectors

Definition

A linear combination of the m vectors $\mathbf{v}_1, \dots, \mathbf{v}_m$ is a vector of the form:

$$\mathbf{w} = a_1 \mathbf{v}_1 + \dots + a_m \mathbf{v}_m, \quad a_1, \dots, a_m \text{ in } \mathbb{R}$$

Special Cases

Linear combination

$$\mathbf{w} = a_1 \mathbf{v}_1 + \dots + a_m \mathbf{v}_m, \quad a_1, \dots, a_m \text{ in } \mathbb{R}$$

Affine combination:

A linear combination for which $a_1 + \dots + a_m = 1$

Convex combination

An affine combination for which $a_i \geq 0$ for $i = 1, \dots, m$

Linear Independence

For vectors $\mathbf{v}_1, \dots, \mathbf{v}_m$

$$\text{If } a_1 \mathbf{v}_1 + \dots + a_m \mathbf{v}_m = \mathbf{0} \quad \text{iff} \quad a_1 = a_2 = \dots = a_m = 0$$

then the vectors are linearly independent

Generators and Base Vectors

How many vectors are needed to generate a vector space?

- Any set of vectors that generate a vector space is called a generator set
- Given a vector space \mathbb{R}^n we can prove that we need minimum n vectors to generate all vectors \mathbf{v} in \mathbb{R}^n
- A generator set with minimum size is called a basis for the given vector space

Standard Unit Vectors

$$\mathbf{v} = (x_1, \dots, x_n), \quad x_i \in \mathbb{R}$$

$$\begin{aligned}(x_1, x_2, \dots, x_n) &= x_1(1, 0, 0, \dots, 0, 0) \\ &\quad + x_2(0, 1, 0, \dots, 0, 0) \\ &\quad \dots \\ &\quad + x_n(0, 0, 0, \dots, 0, 1)\end{aligned}$$

Standard Unit Vectors

For any vector space \mathbb{R}^n :

$$\mathbf{i}_1 = (1, 0, 0, \dots, 0, 0)$$

$$\mathbf{i}_2 = (0, 1, 0, \dots, 0, 0)$$

\dots

$$\mathbf{i}_n = (0, 0, 0, \dots, 0, 1)$$

The elements of a vector \mathbf{v} in \mathbb{R}^n are the scalar coefficients of the linear combination of the basis vectors

Standard Unit Vectors in 2D & 3D

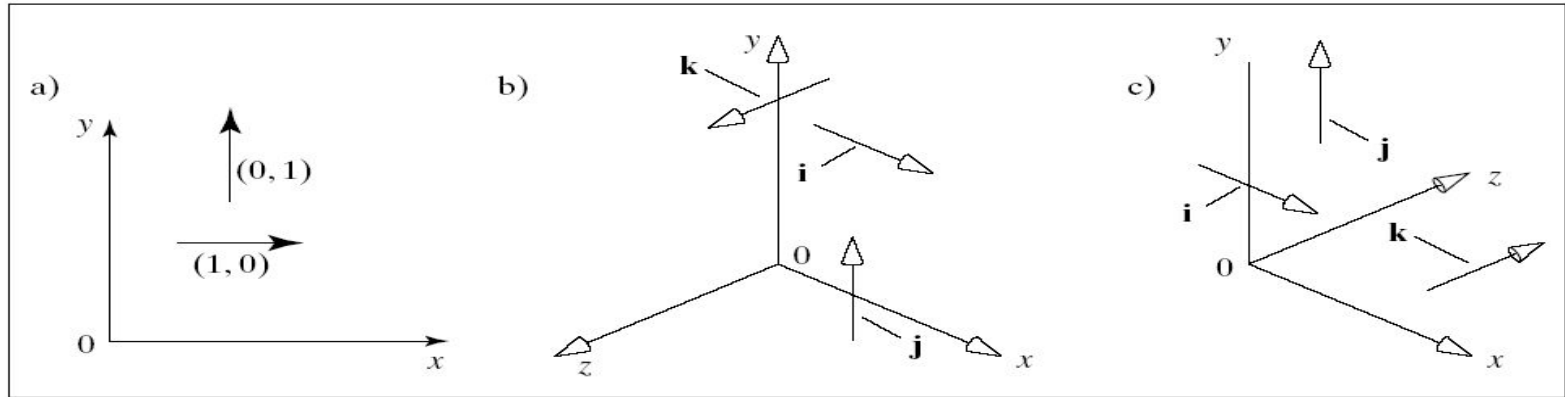
$$\mathbf{i} = (1, 0)$$

$$\mathbf{j} = (0, 1)$$

$$\mathbf{i} = (1, 0, 0)$$

$$\mathbf{j} = (0, 1, 0)$$

$$\mathbf{k} = (0, 0, 1)$$



Right handed

Left handed

Representation of Vectors Through Basis Vectors

*Given a vector space R^n ,
a set of basis vectors $B = \{\mathbf{b}_i \text{ in } R^n, i=1, \dots, n\}$ and
a vector \mathbf{v} in R^n
we can always find scalar coefficients such that:*

$$\mathbf{v} = a_1 \mathbf{b}_1 + \dots + a_n \mathbf{b}_n$$

So, vector \mathbf{v} expressed with respect to B is:

$$\mathbf{v}_B = (a_1, \dots, a_n)$$

Dot Products in Graphics

- The problem dot products solve in graphics:
 - A linear function that maps a point onto a scalar

$$3x + 4y + 5z = ?$$

- Predictable effect as you adjust a coordinate.
- Another problem they solve: Comparing Vectors
 - Trig measurements!

Dot Products and Matrices

- What if we want a function that produces not a scalar, but a new point?
 - *This would become a tool for moving points somewhere new!*
- How do we generate three scalar outputs instead of one?

Dot (Scalar) Product

Definition:

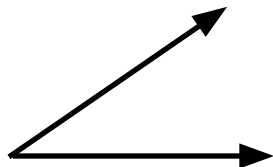
$$\mathbf{w}, \mathbf{v} \in \mathbb{R}^n$$
$$\mathbf{w} \cdot \mathbf{v} = \sum_{i=1}^n w_i v_i$$

Properties

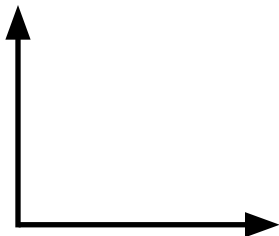
1. Symmetry: $\mathbf{a} \cdot \mathbf{b} = \mathbf{b} \cdot \mathbf{a}$
2. Linearity: $(\mathbf{a} + \mathbf{b}) \cdot \mathbf{c} = \mathbf{a} \cdot \mathbf{c} + \mathbf{b} \cdot \mathbf{c}$
3. Homogeneity: $(s\mathbf{a}) \cdot \mathbf{b} = s(\mathbf{a} \cdot \mathbf{b})$
4. $|\mathbf{b}|^2 = \mathbf{b} \cdot \mathbf{b}$
5. $\mathbf{a} \cdot \mathbf{b} = |\mathbf{a}||\mathbf{b}| \cos(\theta)$

Dot Product and Perpendicularity

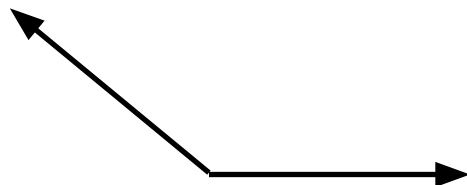
From Property 5:



$$\mathbf{a} \cdot \mathbf{b} > 0$$



$$\mathbf{a} \cdot \mathbf{b} = 0$$



$$\mathbf{a} \cdot \mathbf{b} < 0$$

Perpendicular Vectors

Definition

Vectors \mathbf{a} and \mathbf{b} are perpendicular iff $\mathbf{a} \cdot \mathbf{b} = 0$

Also called “normal” or “orthogonal” vectors

It is easy to see that the standard unit vectors form an orthogonal basis:

$$\mathbf{i} \cdot \mathbf{j} = 0, \quad \mathbf{j} \cdot \mathbf{k} = 0, \quad \mathbf{i} \cdot \mathbf{k} = 0$$

Cross (Vector) Product

Defined only for 3D vectors and with respect to the standard unit vectors

Definition

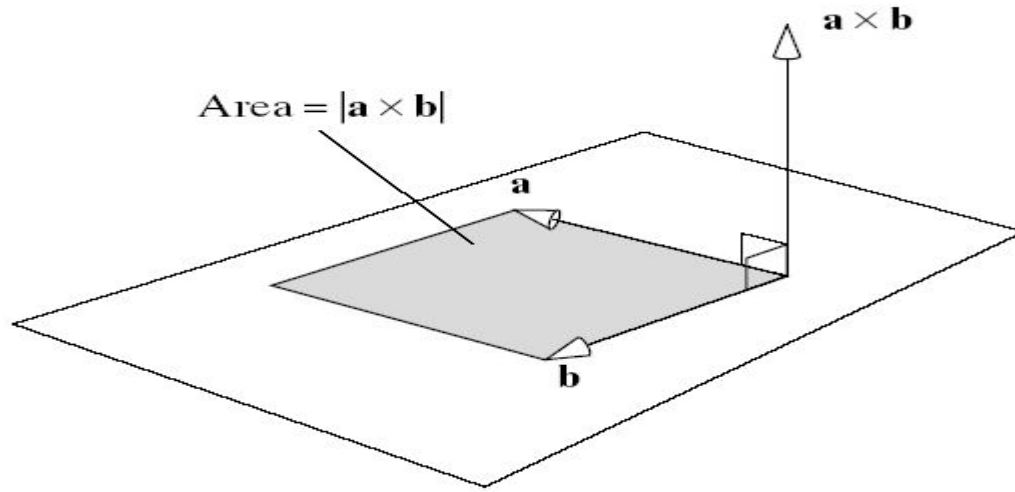
$$\mathbf{a} \times \mathbf{b} = (a_y b_z - a_z b_y)\mathbf{i} + (a_z b_x - a_x b_z)\mathbf{j} + (a_x b_y - a_y b_x)\mathbf{k}$$

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ a_x & a_y & a_z \\ b_x & b_y & b_z \end{vmatrix}$$

Properties of the Cross Product

1. $\mathbf{i} \times \mathbf{j} = \mathbf{k}, \quad \mathbf{i} \times \mathbf{k} = -\mathbf{j}, \quad \mathbf{j} \times \mathbf{k} = \mathbf{i}$
2. Antisymmetry: $\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$
3. Linearity: $\mathbf{a} \times (\mathbf{b} + \mathbf{c}) = \mathbf{a} \times \mathbf{b} + \mathbf{a} \times \mathbf{c}$
4. Homogeneity: $(s\mathbf{a}) \times \mathbf{b} = s(\mathbf{a} \times \mathbf{b})$
5. The cross product is normal to both vectors: $(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{a} = (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{b} = 0$
6. $|\mathbf{a} \times \mathbf{b}| = |\mathbf{a}||\mathbf{b}|\sin(\theta)$

Geometric Interpretation of the Cross Product




Matrices

Rectangular arrangement of scalar elements

Matrix:

Bold upper-case


$$\mathbf{A}_{3 \times 3} = \begin{pmatrix} -1 & 2.0 & 0.5 \\ 0.2 & -4.0 & 2.1 \\ 3 & 0.4 & 8.2 \end{pmatrix}$$

$$\mathbf{A} = (\mathbf{A}_{ij})$$

Special Square ($n \times n$) Matrices

Zero matrix: $\mathbf{A}_{ij} = 0$ for all i, j

Identity matrix: $\mathbf{I}_n = \begin{cases} \mathbf{I}_{ii} = 1 & \text{for all } i \\ \mathbf{I}_{ij} = 0 & \text{for } i \neq j \end{cases}$

Symmetric matrix: $(\mathbf{A}_{ij}) = (\mathbf{A}_{ji})$

Operations with Matrices

Addition:

$$\mathbf{A}_{m \times n} + \mathbf{B}_{m \times n} = (a_{ij} + b_{ij})$$

Properties:

1. $\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$
2. $\mathbf{A} + (\mathbf{B} + \mathbf{C}) = (\mathbf{A} + \mathbf{B}) + \mathbf{C}$
3. $f(\mathbf{A} + \mathbf{B}) = f\mathbf{A} + f\mathbf{B}$
4. Transpose: $\mathbf{A}^T = (a_{ij})^T = (a_{ji})$

Multiplication

Definition:

$$\mathbf{C}_{m \times r} = \mathbf{A}_{m \times n} \mathbf{B}_{n \times r}$$

$$(\mathbf{C}_{ij}) = \left(\sum_{k=1}^n a_{ik} b_{kj} \right)$$

Properties:

1. $\mathbf{AB} \neq \mathbf{BA}$
2. $\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$
3. $f(\mathbf{AB}) = (f\mathbf{A})\mathbf{B}$
4. $\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC},$
 $(\mathbf{B} + \mathbf{C})\mathbf{A} = \mathbf{BA} + \mathbf{CA}$
5. $(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$

Inverse of a Square Matrix

Definition

$$\mathbf{M}\mathbf{M}^{-1} = \mathbf{M}^{-1}\mathbf{M} = \mathbf{I}$$

Important property

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1} \mathbf{A}^{-1}$$

Dot Product as a Matrix Multiplication

Representing vectors as column matrices:

$$\begin{aligned} \mathbf{a} \cdot \mathbf{b} &= \mathbf{a}^T \mathbf{b} \\ &= (a_1 \ a_2 \ a_3) \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} \\ &= a_1 b_1 + a_2 b_2 + a_3 b_3 \end{aligned}$$