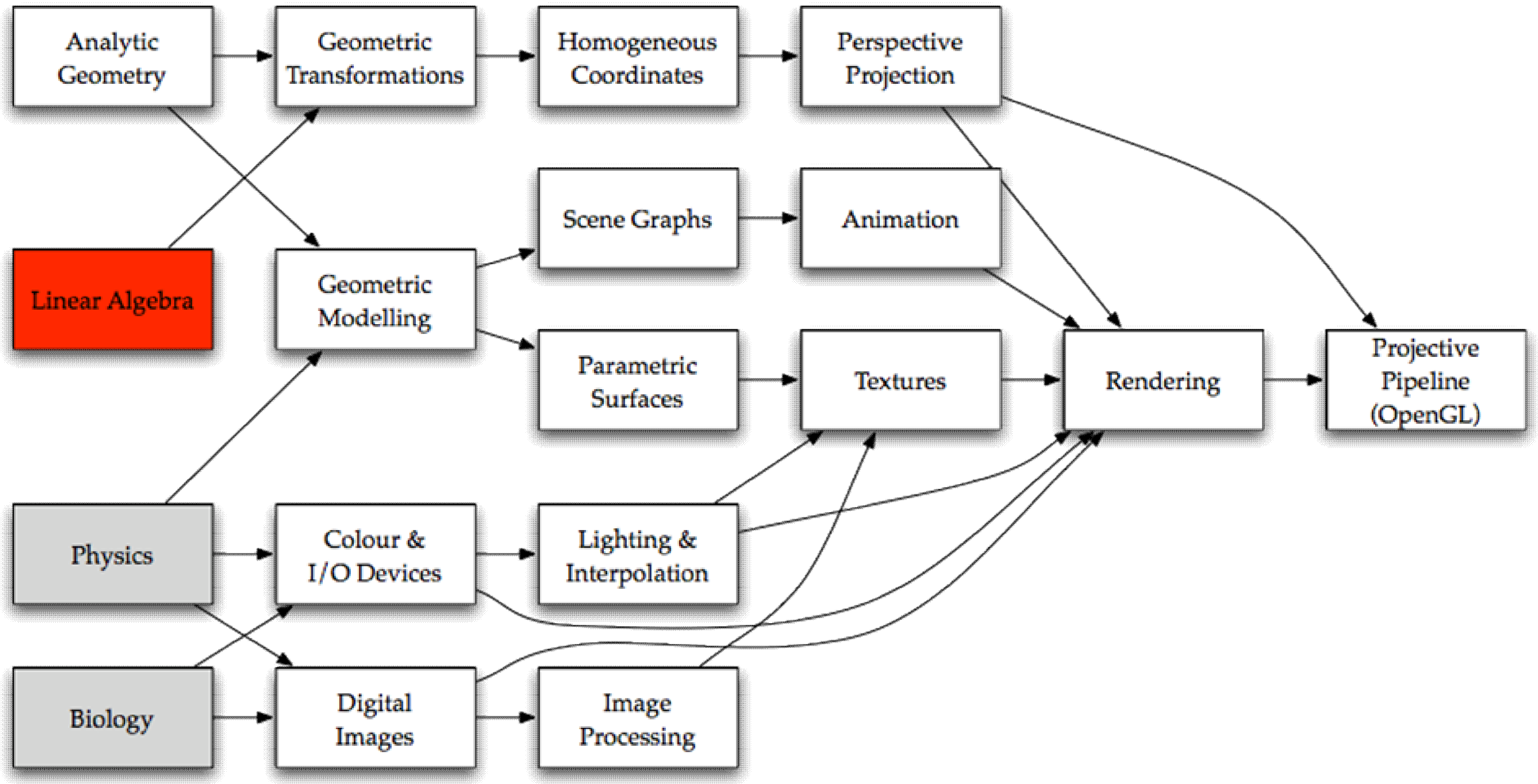
Vectors & Linear Algebra (Review)

Where We Are



Analytic Geometry
Geometric Transformations
Homogeneous Coordinates
Perspective Projection
Scene Graphs
Animation
Linear Algebra
Geometric Modelling
Parametric Surfaces
Textures
Rendering
Projective Pipeline (OpenGL)
Physics
Colour & I/O Devices
Lighting & Interpolation
Biology
Digital Images
Image Processing

Cartesian Coordinates

- Invented by René Descartes (1596-1650)
  - Along with algebraic notation
  - And the x,y,z axes
- Originally used for mathematics
  - to describe locations accurately in 3D
- Developed into linear algebra

Map Location

- Dublin is at 53º 20’ N, 6º 15’ W
  - 53º 20’ N of the equator
  - 6º 15’ W of Greenwich
- All locations are *relative*
  - to a point called the *origin*
  - which is arbitrary

The Origin

- Can be anywhere you like
  - but *pick* one and *stick* to it
- Always written as *O* or *(0,0)*
- *All* locations are relative to the origin
  - move the origin, and you move the world

Map Coordinates

- Are measured in
  - degrees
  - metres / kilometres
  - miles / yards
- I.e. the choice of unit is important
  - but you can use any unit you like

Map Directions

- We only need two directions on a map
  - North & East
  - South & West are negative
- But could we choose other directions?
  - e.g. NW & NNE?

Coordinate Systems

- A *coordinate* system consists of:
  - an origin
  - a set of units
  - a set of directions, called *axes*
    - preferably perpendicular to each other
    - preferably one unit long
    - *x, y, & z*

Standard Coordinates

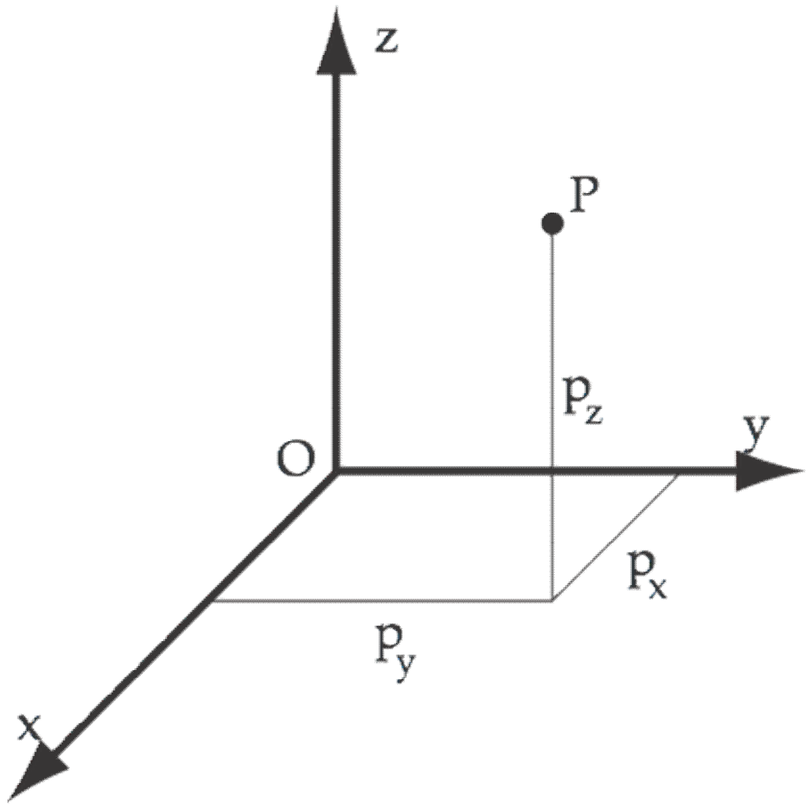
- *x*-axis usually runs from left to right
  - why?
- *y*-axis usually runs from bottom to top
  - why?
- *z*-axis runs *out* of the page (or screen)

Right-hand rule

- *x*-axis follows the right thumb
- *y*-axis follows the right index finger
- *z*-axis sticks out of your palm
- What happens if you use your left hand?
  - you get the *left-hand rule* instead

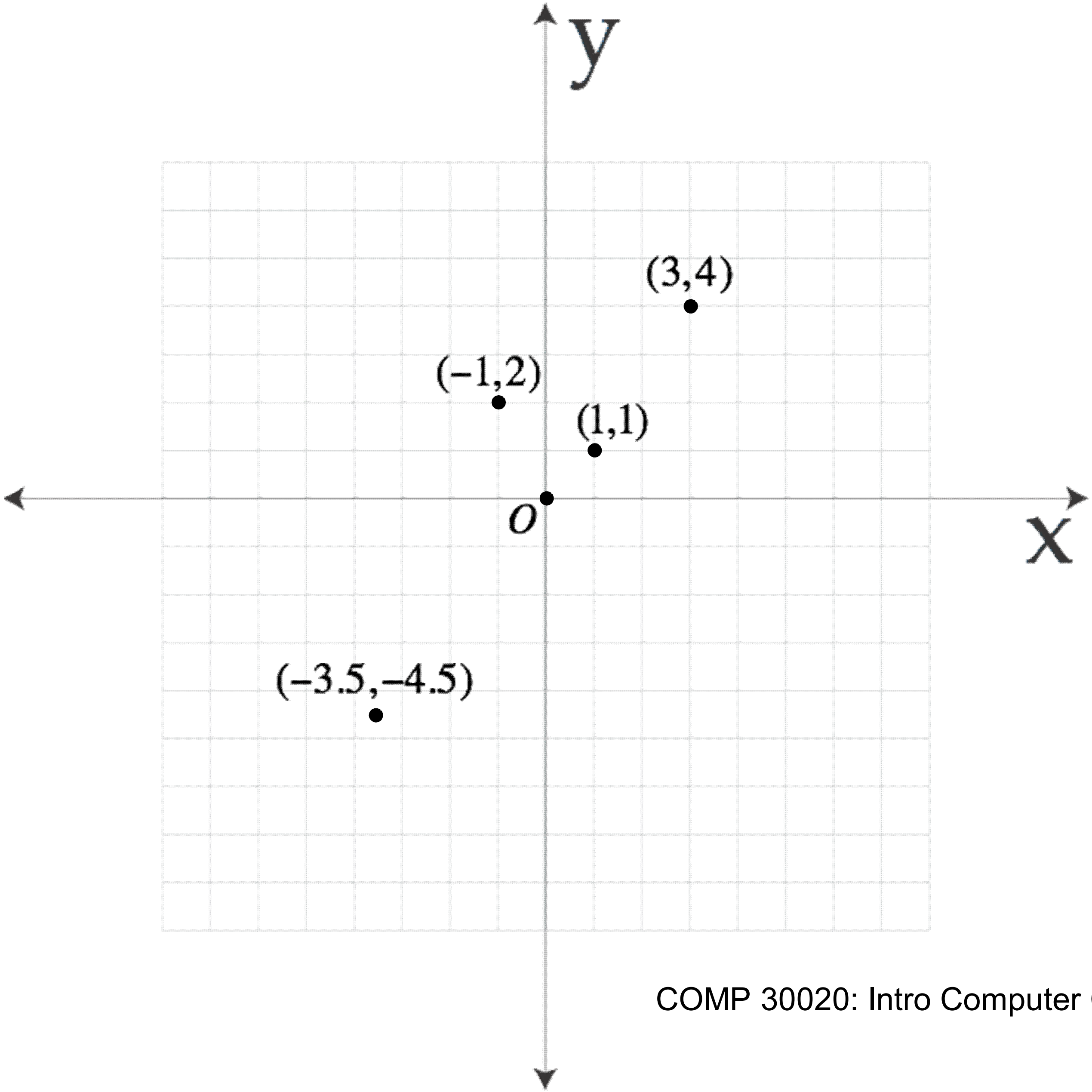
Cartesian Coordinates

- A point p is just a set of *coordinates*
  - distance p_x parallel to x
  - distance p_y parallel to y
  - distance p_z parallel to z
- relative to the origin O



$$p = \left(p_x, p_y, p_z\right)$$

Examples

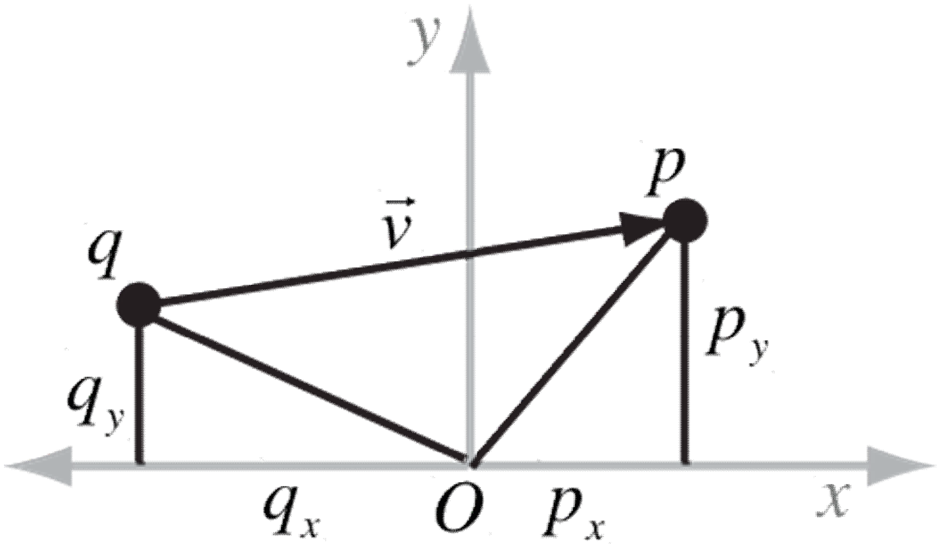


Vectors

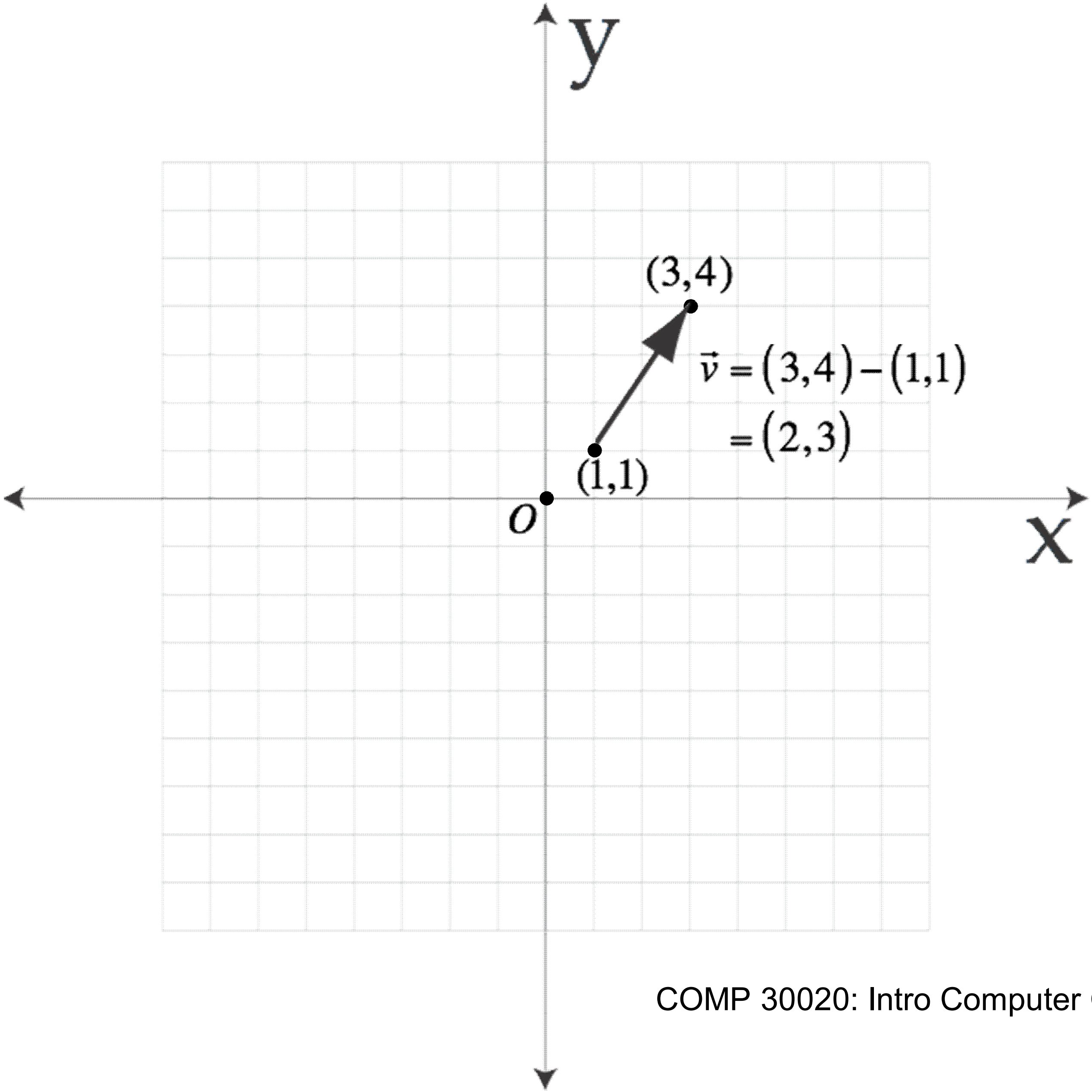
- A vector measures direction and distance between two points
- Obtained by subtracting tail q from head p

$$\vec{v} = p - q$$

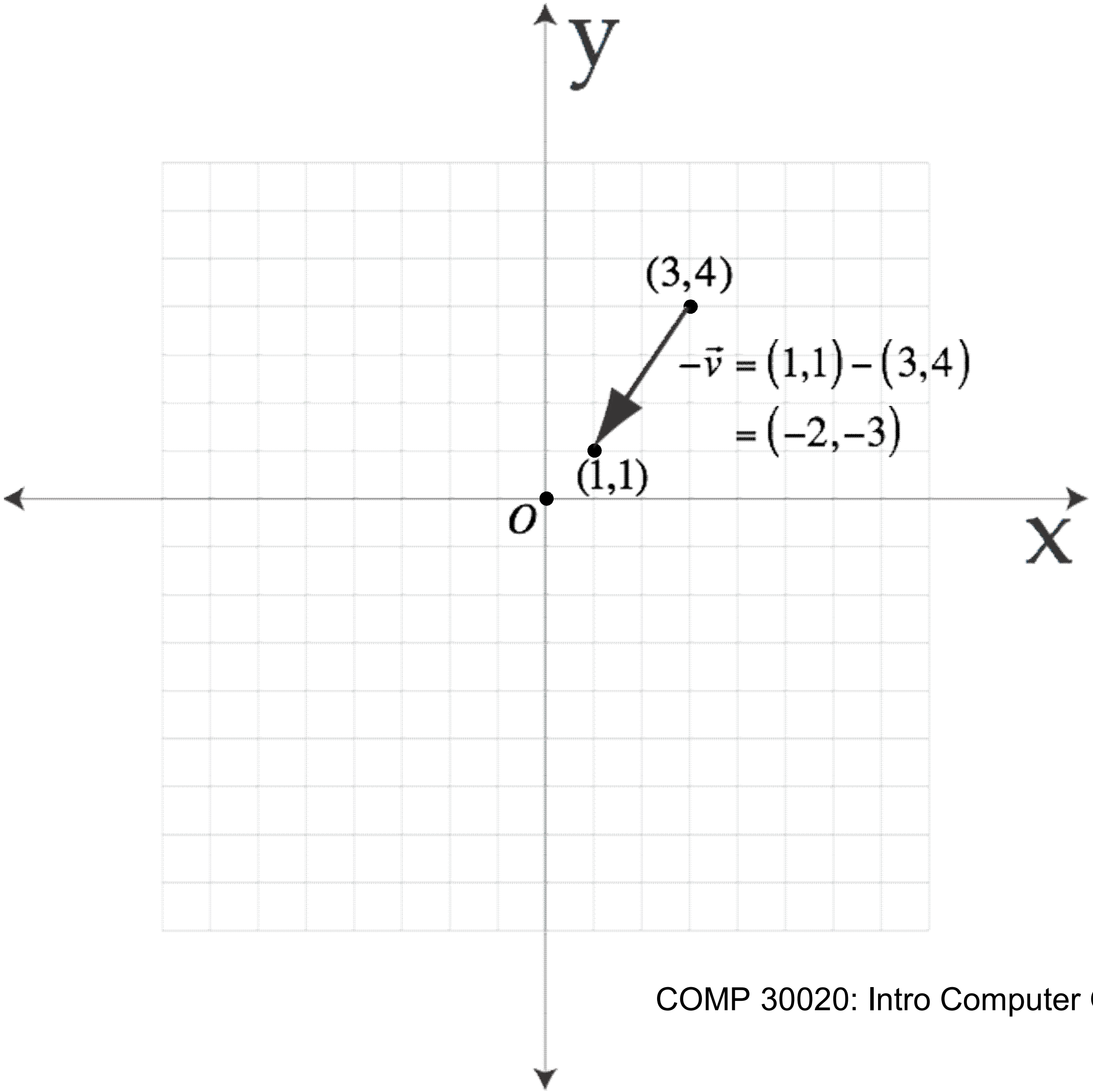
$$\begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} p_x - q_x \\ p_y - q_y \end{bmatrix}$$



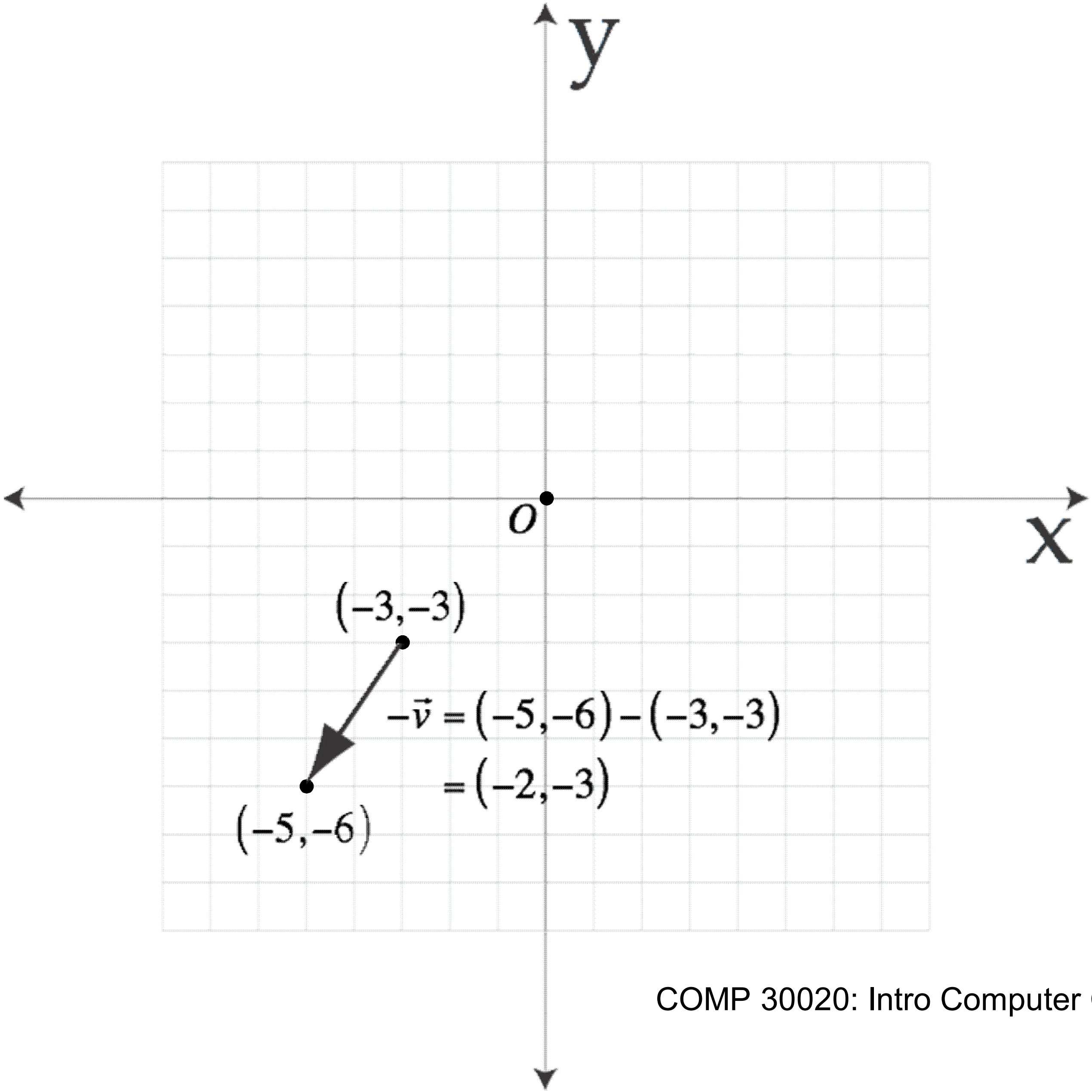
Example



Negative Vector



Movable Vector



Points vs. Vectors

- Points are not the same as vectors
  - points are position
  - vectors are direction
- Point + Vector = Point
- Point - Point = Vector

Addition / Subtraction

- Add the coordinates

$$\vec{v} + \vec{w} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} + \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} = \begin{bmatrix} v_x + w_x \\ v_y + w_y \\ v_z + w_z \end{bmatrix}$$

  - associative
  - commutative
  - inverse
  - neutral

$$\vec{u} + \left(\vec{v} + \vec{w}\right) = \left(\vec{u} + \vec{v}\right) + \vec{w}$$

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

$$\vec{u} + -\vec{u} = \vec{0}$$

$$\vec{u} + \vec{0} = \vec{u}$$

Addition

- Undefined:
  - Point + Point
- Point result:
  - Point + Vector, Vector + Point
- Vector result:
  - Vector + Vector

Subtraction

- Undefined:
  - Vector - Point
- Point result:
  - Point - Vector
- Vector result:
  - Point - Point, Vector - Vector

Scalar Multiplication

$$r\vec{v} = r\begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} = \begin{bmatrix} rv_x \\ rv_y \\ rv_z \end{bmatrix}$$

- Multiply each coordinate
  - distributive

$$r\left(\vec{v} + \vec{w}\right) = r\vec{v} + r\vec{w}$$

$$\left(r + s\right)\vec{v} = r\vec{v} + s\vec{v}$$

Scalar Multiplication

- Undefined:
  - Scalar * Point, Point * Scalar
- Point result:
  - None
- Vector result:
  - Scalar * Vector, Vector * Scalar

Vector Multiplication?

- Scalar Multiplication:
  - Scalar * Vector, Point * Vector
- Point Multiplication:
  - None
- Vector times vector:
  - Dot Product, Cross Product

Dot Product

- Also called scalar or inner product:
- Computes a scalar from two vectors:
  - commutative
  - distributive
- VERY useful

$$\vec{v} \cdot \vec{w} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \cdot \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} = v_x w_x + v_y w_y + v_z w_z$$

$$\vec{v} \cdot \vec{w} = \vec{w} \cdot \vec{v}$$

$$(r\vec{v}) \cdot \vec{w} = r(\vec{v} \cdot \vec{w})$$

$$(rs)\vec{v} = r(s\vec{v})$$

Cross Product

- Computes a vector from two vectors

$$\vec{v} \times \vec{w} = \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \times \begin{bmatrix} w_x \\ w_y \\ w_z \end{bmatrix} = \begin{bmatrix} v_y w_z - v_z w_y \\ v_z w_x - v_x w_z \\ v_x w_y - v_y w_x \end{bmatrix} = \begin{vmatrix} i & j & k \\ v_x & v_y & v_z \\ w_x & w_y & w_z \end{vmatrix}$$

  - anti-commutative $\quad \vec{v} \times \vec{w} = -\vec{w} \times \vec{v}$
  - distributive $\quad (r\vec{v}) \times \vec{w} = r(\vec{v} \times \vec{w})$
  - also very useful $\quad (rs)\vec{v} = r(s\vec{v})$

Cross Product

Cross Product gets a line perpendi cular to both lines
This is the surface normal
“Normal” is normally the surface or face of a geometric object
Using the “Normal” , are able to calculate the angle of reflection
This allows us to correctly shade the surface. (Multiple approaches)

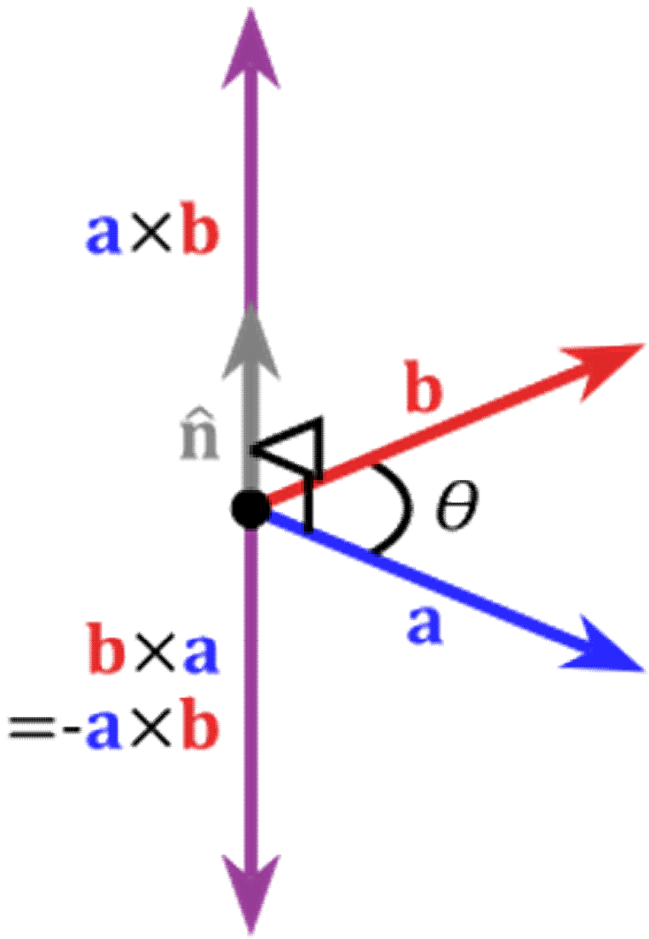


Image from Wikipedia (referencing is important)

Normal (per pixel or face)

Image from X-Wing (luasArts)

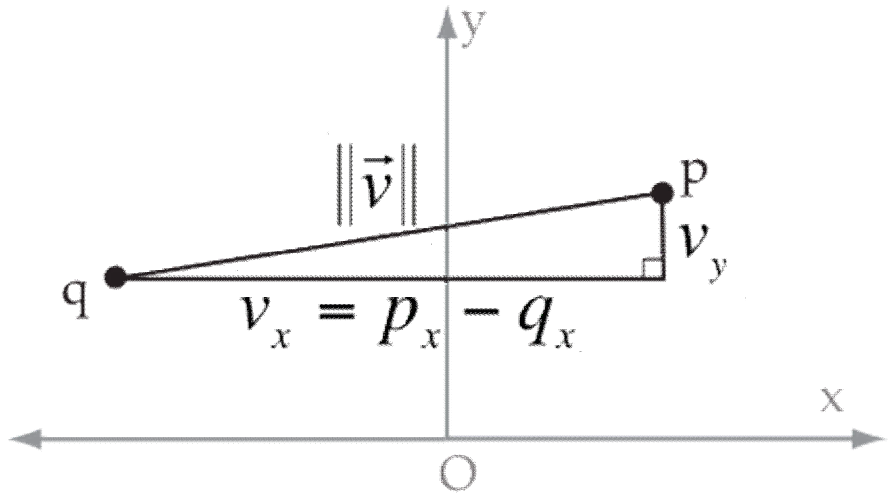
More Vector Operations

- Length / Norm / Magnitude
- Normalize / Unit (convert to length 1)
- Can we do division?

Vector Length (Magnitude)

- Using Pythagoras

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

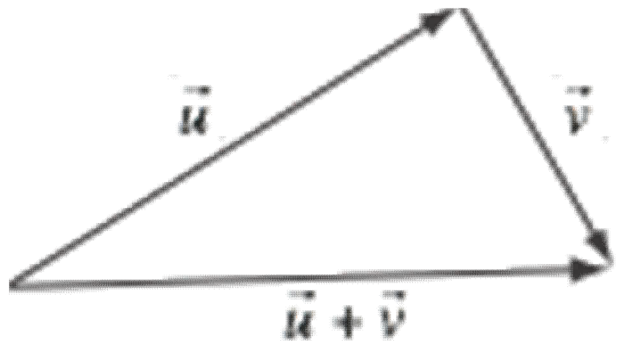


- Scalar multiplication:

$$\|r\vec{v}\| = r\|\vec{v}\|$$

- Triangle Inequality:

$$\|\vec{u} + \vec{v}\| \leq \|\vec{u}\| + \|\vec{v}\|$$



Normalized Vectors

- A normalized vector has length of 1
  - Multiply vector by inverse of length:

$$\vec{v}' = \frac{1}{\|\vec{v}\|}\vec{v}$$

$$= \frac{\vec{v}}{\|\vec{v}\|}$$

Dependent Vectors

- Dependent vectors depend on each other

$$\begin{bmatrix}1\\0\\1\end{bmatrix} = \begin{bmatrix}1\\0\\0\end{bmatrix} + \begin{bmatrix}0\\0\\1\end{bmatrix}$$

- I.e. they are redundant
  - you don't need all of them

Independent Vectors

- Vectors that are not dependent

$$a\begin{bmatrix}1\\0\\0\end{bmatrix} = b\begin{bmatrix}0\\0\\0\end{bmatrix} + c\begin{bmatrix}0\\0\\1\end{bmatrix} \text{ if and only if } a = b = c = 0$$

- A basis is a set of 3 independent 3-D vectors

$$\begin{bmatrix}1\\0\\0\end{bmatrix} \quad \begin{bmatrix}1\\1\\0\end{bmatrix} \quad \begin{bmatrix}1\\0\\-3\end{bmatrix}$$

Orthonormal Basis

- A normal basis is a basis of unit vectors
- An orthogonal basis is a basis of mutually perpendicular vectors
- An orthonormal basis is a basis of mutually perpendicular unit vectors
- I.e. a set of axes for a coordinate system

Matrices

- Rectangular arrays of numbers
  - we will always use square ones

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & -3 \end{bmatrix}$$

- And we can define operations with them

Notation

- Matrices are always in square brackets
- The entries are indexed by row & column

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} 3 & -2 & 1 \\ 14 & 1.7985 & 0 \\ 1 & 8 & 3 \end{bmatrix}$$

Matrix Addition

- Like vectors, just add the elements

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & -3 \end{bmatrix} + \begin{bmatrix} 3 & 1 & -9 \\ 5 & 2 & 6 \\ 4 & 0 & 8 \end{bmatrix} = \begin{bmatrix} 1+3 & 1+1 & 1-9 \\ 0+5 & 1+2 & 0+6 \\ 0+4 & 0+0 & -3+8 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 2 & -8 \\ 5 & 3 & 6 \\ 4 & 0 & 5 \end{bmatrix}$$

Matrix Addition

- Assuming the dimensions match,
- Point + Matrix, Vector + Matrix undefined
- Matrix + Matrix returns Matrix
- Associative, commutative, distributive, inverse & neutral

Additive Zero

- There is a zero matrix
  - adding it to a matrix does nothing

$$0_{3,3} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Additive Inverse

- The *negative* of a matrix

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} 3 & -2 & 1 \\ 14 & 1.7985 & 0 \\ 1 & 8 & 3 \end{bmatrix}$$

$$-M = \begin{bmatrix} -m_{11} & -m_{12} & -m_{13} \\ -m_{21} & -m_{22} & -m_{23} \\ -m_{31} & -m_{32} & -m_{33} \end{bmatrix} = \begin{bmatrix} -3 & 2 & -1 \\ -14 & -1.7985 & 0 \\ -1 & -8 & -3 \end{bmatrix}$$

$$M + (-M) = (-M) + M = 0$$

Matrix Multiplication

- Assuming the dimensions match,
- Point * Matrix, Vector * Matrix undefined
- Matrix * Point returns Point
- Matrix * Vector returns Vector
- Matrix * Matrix returns Matrix
- But the *order* of multiplication matters
- Additional Examples in the Red book on Page 214

Matrix * Point or Vector

- For each *row* of matrix
  - compute dot product with vector
  - gives one coordinate of result

$$M \times \vec{v} = \begin{bmatrix} \begin{bmatrix} m_{11} \\ m_{12} \\ m_{13} \end{bmatrix} \bullet \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \\ \begin{bmatrix} m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} \bullet \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \\ \begin{bmatrix} m_{31} \\ m_{32} \\ v_{33} \end{bmatrix} \bullet \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} m_{11}v_1 + m_{12}v_2 + m_{13}v_3 \\ m_{21}v_2 + m_{22}v_2 + m_{23}v_3 \\ m_{31}v_3 + m_{32}v_2 + m_{33}v_3 \end{bmatrix}$$

Example

$$\begin{bmatrix}1 & 1 & 1\\0 & 1 & 0\\0 & 0 & -3\end{bmatrix}\begin{bmatrix}-5\\7\\-1\end{bmatrix} = \begin{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix}\bullet\begin{bmatrix}-5\\7\\-1\end{bmatrix}\\\begin{bmatrix}0\\1\\0\end{bmatrix}\bullet\begin{bmatrix}-5\\7\\-1\end{bmatrix}\\\begin{bmatrix}1\\0\\-3\end{bmatrix}\bullet\begin{bmatrix}-5\\7\\-1\end{bmatrix}\end{bmatrix}$$

$$= \begin{bmatrix}1(-5)+1(7)+1(-1)\\0(-5)+1(7)+0(-1)\\0(-5)+0(7)-3(-1)\end{bmatrix} = \begin{bmatrix}1\\7\\3\end{bmatrix}$$

Matrix Multiplication

- Similar to what we've just seen
- Take a row i of A and a column j of B
- Put their dot product at their intersection

$$C = AB$$

$$c_{ij} = a_{i?} \bullet b_{?\,j}$$

$$= \sum_{k=1}^{n} a_{ik} b_{kj}$$

Matrix Multiplication (2x2)

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}\begin{bmatrix} 3 & 5 \\ 1 & 6 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 1 & 2 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \end{bmatrix} & \begin{bmatrix} 1 & 2 \end{bmatrix}\begin{bmatrix} 5 \\ 6 \end{bmatrix} \\ \begin{bmatrix} 3 & 4 \end{bmatrix}\begin{bmatrix} 3 \\ 1 \end{bmatrix} & \begin{bmatrix} 3 & 4 \end{bmatrix}\begin{bmatrix} 5 \\ 6 \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} 1(3)+2(1) & 1(5)+2(6) \\ 3(3)+2(1) & 3(5)+4(6) \end{bmatrix}$$

$$= \begin{bmatrix} 5 & 17 \\ 11 & 39 \end{bmatrix}$$

The Other Way Round

$$\begin{bmatrix} 3 & 5 \\ 1 & 6 \end{bmatrix}\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 3 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ 3 \end{bmatrix} & \begin{bmatrix} 3 & 5 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \end{bmatrix} \\ \begin{bmatrix} 1 & 6 \end{bmatrix}\begin{bmatrix} 1 \\ 3 \end{bmatrix} & \begin{bmatrix} 1 & 6 \end{bmatrix}\begin{bmatrix} 2 \\ 4 \end{bmatrix} \end{bmatrix}$$

$$= \begin{bmatrix} 3(1)+5(3) & 3(2)+5(4) \\ 1(1)+6(3) & 1(2)+6(4) \end{bmatrix}$$

$$= \begin{bmatrix} 11 & 26 \\ 19 & 26 \end{bmatrix}$$

Oops . . .

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 3 & 5 \\ 1 & 6 \end{bmatrix} = \begin{bmatrix} 5 & 17 \\ 11 & 39 \end{bmatrix}$$

$$\begin{bmatrix} 3 & 5 \\ 1 & 6 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 11 & 26 \\ 19 & 26 \end{bmatrix}$$

- Like subtraction, the order matters
  - *worse*, we don't get the opposite result

Multiplicative Identity

- The *identity* matrix
  - multiplying by it does nothing

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$IM = MI = M$$

Matrix Division

- Can we *divide* by a matrix?
  - Yes, if we multiply by the *inverse*
  - the equivalent of 1/x
  - hard to calculate in general
  - easy for the ones we will use
  - exists *only* for square matrices

$$MM^{-1} = M^{-1}M = I$$

Matrix Transpose

- Flips a matrix along it's diagonal
- Useful for calculating inverses

$$M = \begin{bmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{bmatrix} = \begin{bmatrix} 3 & -2 & 1 \\ 14 & 1.7985 & 0 \\ 1 & 8 & 3 \end{bmatrix}$$

$$M^T = \begin{bmatrix} m_{11} & m_{21} & m_{31} \\ m_{12} & m_{22} & m_{32} \\ m_{13} & m_{23} & m_{33} \end{bmatrix} = \begin{bmatrix} 3 & 14 & 1 \\ -2 & 1.7985 & 8 \\ 1 & 0 & 3 \end{bmatrix}$$

Some Comments

- Matrix arithmetic is repetitive
  - repeated multiplication & addition
  - ideal for doing on a computer
  - especially in object-oriented C++
  - but not so great in JAVA but Computer have gotten very fast so for simple examples it runs the same speed
  - Also C++ gets slow too, Shaders are written in C and get compiled at runtime by the GPU.