Computer Graphics (COMP0027) 2023/24

Linear Algebra

Tobias Ritschel

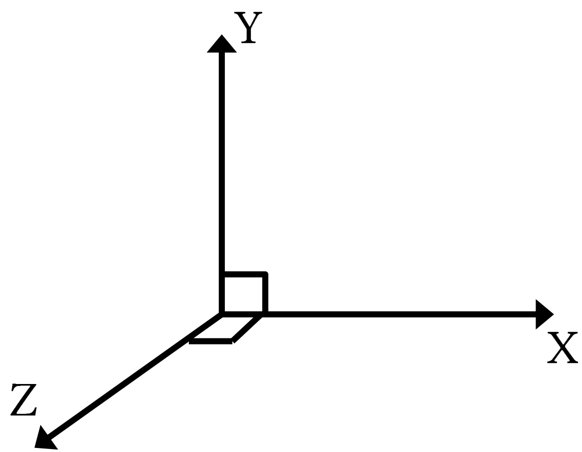
Overview

- Points
- Vectors
- Lines
- Matrices
- 3D transformations as matrices
- Homogenous coordinates
- Rays
- Spheres

Basic Maths

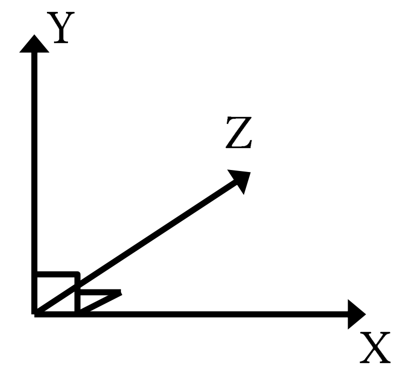
- In computer graphics we need mathematics both
  - for describing our scenes and also
  - for performing operations on them, such as projection and various transformations.
- Coordinate systems (right- and left-handed), serve as a reference point.
- 3 axes labelled x, y, z at right angles.

Co-ordinate Systems



Right-Handed System

(Z comes out of the screen)



Left-Handed System

(Z goes into the screen)

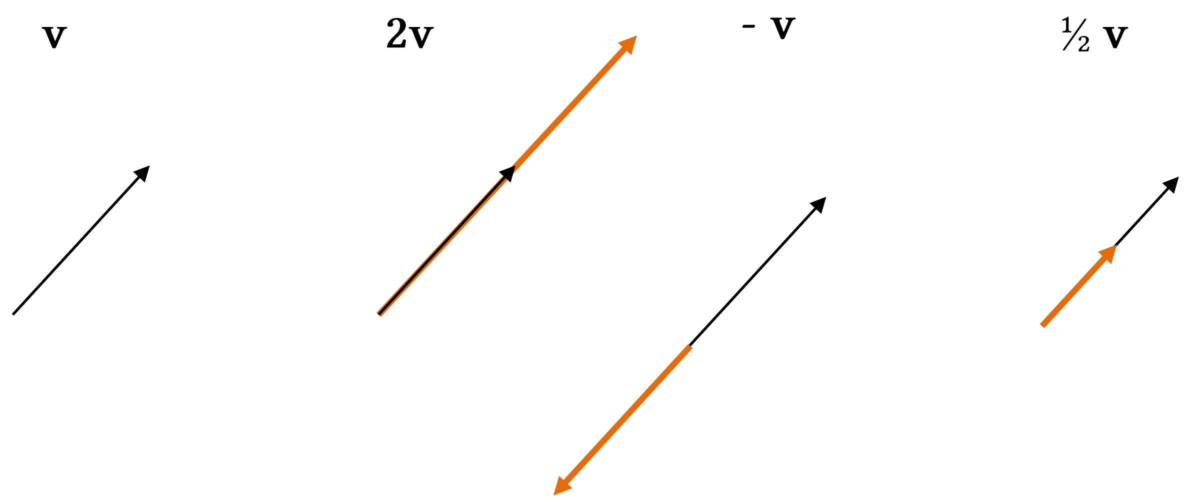
Points, $\mathrm{P}\,(x, y, z)$

- Vector from the origin of our coordinate system to the point
- x, y and z are the coordinates

Vectors, $\mathbf{v}\ (x, y, z)$

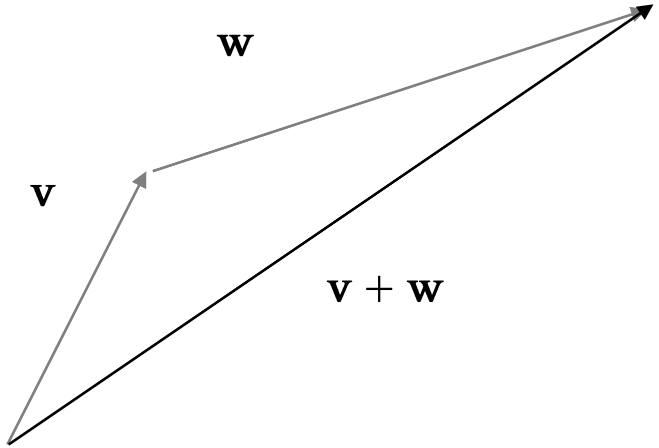
- Represent a *direction* (and magnitude) in 3D space
- Points != Vectors
  Vector + Vector = Vector
  Point – Point = Vector
  Point + Vector = Point
  Point + Point = ?

Vectors operations



Scalar multiplication of vectors (they remain parallel)

Vector addition



Vector addition $\mathbf{v} + \mathbf{w}$

Vector subtraction

Vector difference $\mathbf{v} - \mathbf{w} = \mathbf{v} + (-\mathbf{w})$

Vector length

- Length of a vector v, a tuple (x, y, z)

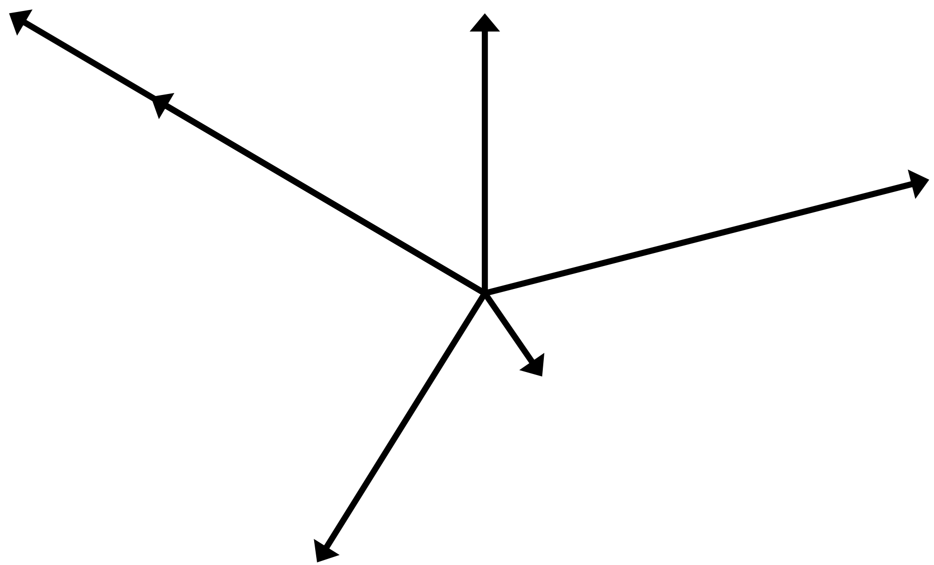
$$|\mathbf{v}| = \sqrt{x^2 + y^2 + z^2}$$

Vector normalization

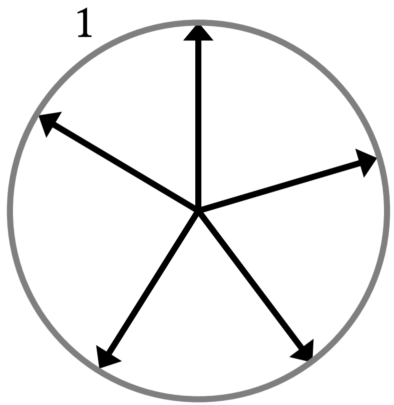
- A unit vector: a vector can be normalised such that it retains its direction, but is scaled to have unit length

$$\mathbf{v}_{\text{unit}} = \frac{\mathbf{v}}{|\mathbf{v}|}$$

Normalization, visually



Normalization, visually



Dot Product

$$\mathrm{u} \cdot \mathrm{v} = \mathrm{x_u x_v + y_u y_v + z_u z_v}$$

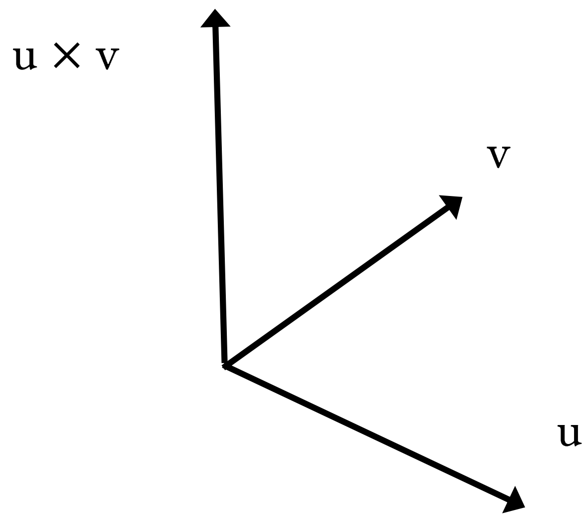
$$\mathrm{u} \cdot \mathrm{v} = |\mathrm{u}||\mathrm{v}| \cos \theta$$

$$\cos \theta = \mathrm{u} \cdot \mathrm{v} \; / \; |\mathrm{u}||\mathrm{v}|$$

- Result is purely a scalar number not a vector
- What happens when the vectors are unit
- What does it mean if dot product is 0 or 1?

Cross Product

- The result is not a scalar but a vector which is normal to the plane of the other two



Matrices

Vectors and Matrices

- Matrix is an array of numbers with dimensions m (columns) by n (rows)
  - Example: 6 by 3 matrix
  - Element [3][2] is 3

$$\begin{pmatrix} 3 & 0 & 0 & -2 & 1 & -2 \\ 1 & 1 & 3 & 4 & 1 & -1 \\ -5 & 2 & 0 & 0 & 0 & 1 \end{pmatrix}$$

- Vector can be considered a $1 \times n$ matrix

Types of Matrix

- Identity matrices - I

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \qquad \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Symmetric

$$\begin{pmatrix} a & b & c \\ b & d & e \\ c & e & f \end{pmatrix}$$

- Diagonal

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -4 \end{pmatrix}$$

1. Symmetric matrix is equal to its transpose
2. Diagonal matrices are symmetric
3. Identity matrices are diagonal

Operation on Matrices

- Addition
  - Done elementwise

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} + \begin{pmatrix} p & q \\ r & s \end{pmatrix} = \begin{pmatrix} a+p & b+q \\ c+r & d+s \end{pmatrix}$$

- Transpose
  - “Flip”
    (m by n becomes n by m)

$$\begin{pmatrix} 1 & 4 & 9 \\ 5 & 2 & 8 \\ 6 & 7 & 3 \end{pmatrix}^{\mathrm{T}} = \begin{pmatrix} 1 & 5 & 6 \\ 4 & 2 & 7 \\ 9 & 8 & 3 \end{pmatrix}$$

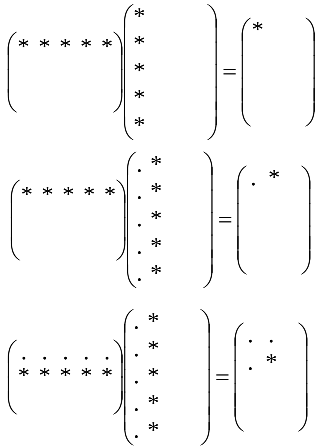
Operations on Matrices

- Multiplication
  - Only possible to multiply of dimensions
    - m_1 by n_1 and m_2 by n_2 if $m_1 = n_2$
      - I.e., if number of rows in first matrix equals number of columns in second matrix
      - Resulting matrix is m_2 by n_1
    - e.g., Matrix A is 3 by 4 and Matrix B is by 2 by 3
      - resulting matrix is 2 by 4
    - Just because $\mathrm{A} \times \mathrm{B}$ is possible doesn't mean $\mathrm{B} \times \mathrm{A}$ is!

Matrix Multiplication Order

- A is k by n
- B is n by k
- $\mathrm{C} = \mathrm{A} \times \mathrm{B}$ defined by

$$c_{i,j} = \sum_{l=1}^{k} a_{i,l} b_{l,j}$$



- $\mathrm{B} \times \mathrm{A}$ not necessarily equal to $\mathrm{A} \times \mathrm{B}$

Inverse

- If $A \times B = I$ then $B = A^{-1}$
- We will not discuss how to compute it here
- GLSL / Matlab / Eigen (C++) will do it for you for 4x4 and 3x3 etc

3D Transforms

In 3-space vectors and points are transformed by 3-by-3 matrices

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a\mathrm{x} + \mathrm{b}y + \mathrm{c}z \\ \mathrm{dx} + \mathrm{ey} + fz \\ \mathrm{gx} + \mathrm{hy} + iz \end{pmatrix}$$

Scale

- Scale uses a diagonal matrix

$$\begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} ax \\ by \\ cz \end{pmatrix}$$

Scale: Example

- Scale by 2 along x and -2 along z

$$\begin{pmatrix} 2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \\ 5 \end{pmatrix} = \begin{pmatrix} 6 \\ 4 \\ -10 \end{pmatrix}$$

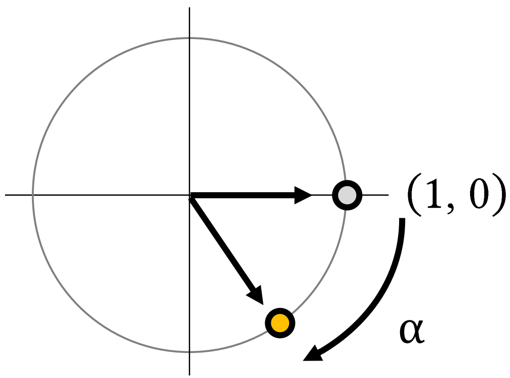
Rotation

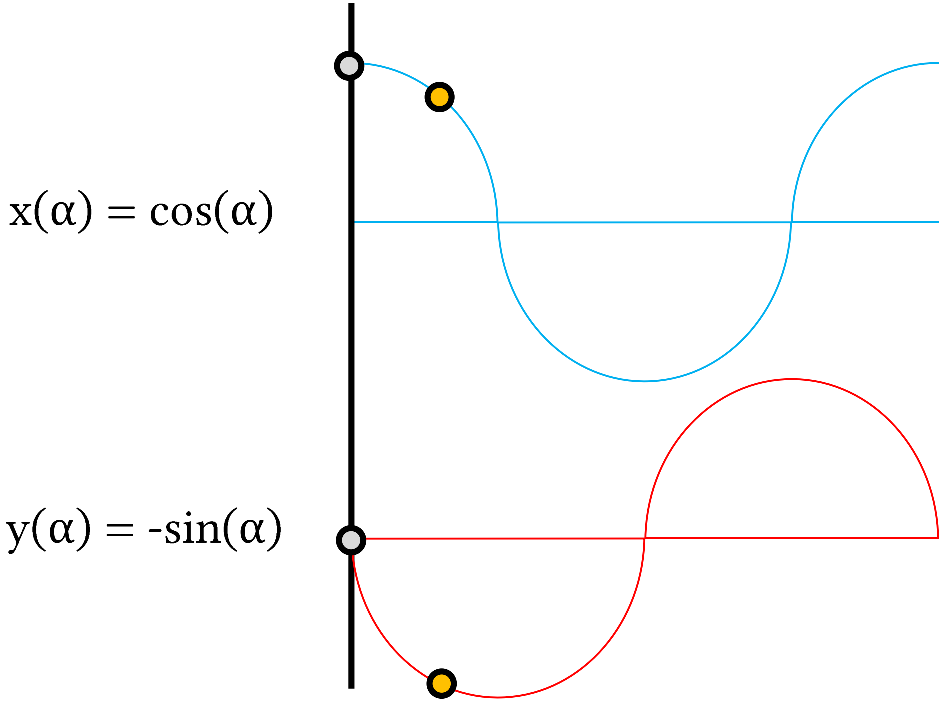
Rotation

- First 2D
- Then 3D one axis
- Then 3D multiple axis

2D Rotation

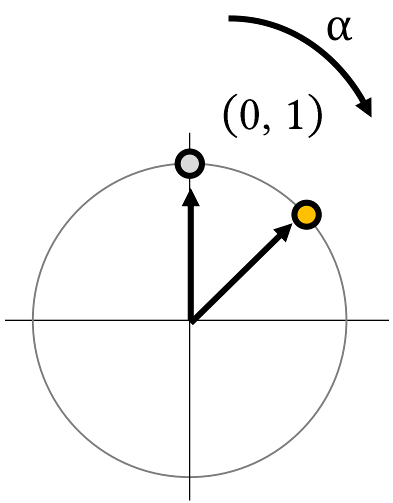
- Idea: What does the basis on circles do?

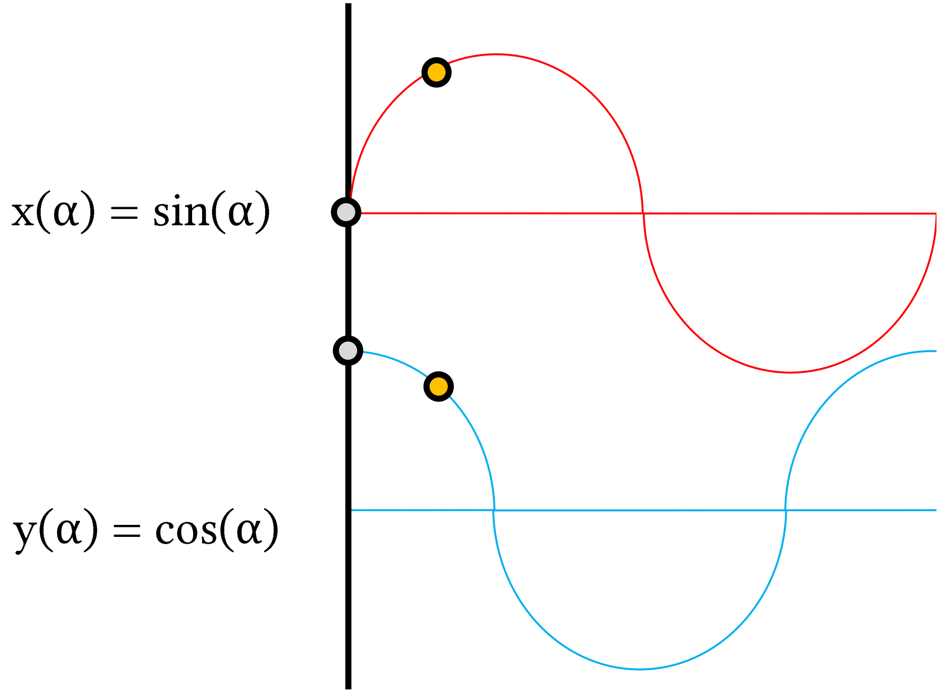




2D Rotation

- Idea: What does the basis on circles do?





2D Rotation Matrix Construction

- If you know what happens to the basis, you know it all

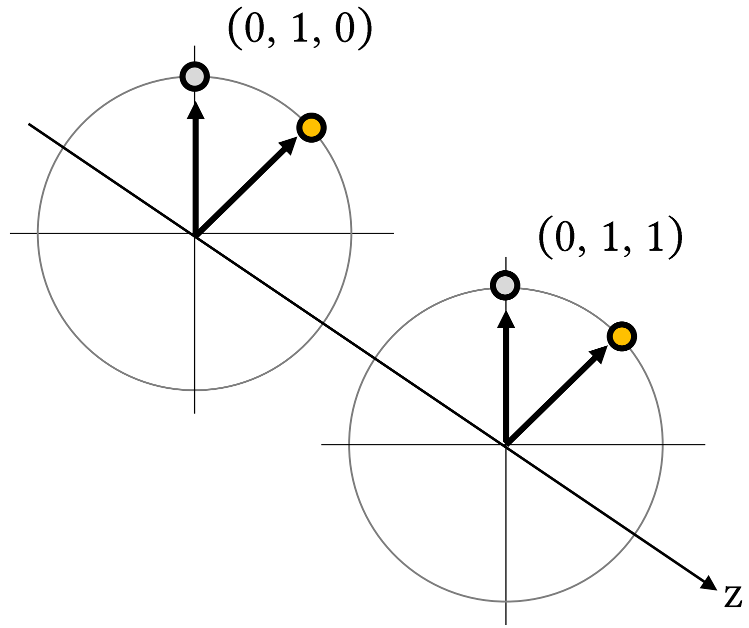
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos(\alpha) \\ \sin(\alpha) \end{pmatrix}$$

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -\sin(\alpha) \\ \cos(\alpha) \end{pmatrix}$$

$$\mathrm{R}(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

3D Rotation, z

- When rotating around z, z stays fixed



3D Rotation, z

- Just leave the axis we rotate around alone

$$\mathrm{R}_z(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3D Rotation all

- All other axis are similar

$$\mathrm{R}_x(\alpha) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{pmatrix}$$

$$\mathrm{R}_y(\alpha) = \begin{pmatrix} \cos(\alpha) & 0 & \sin(\alpha) \\ 0 & 1 & 0 \\ -\sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix}$$

$$\mathrm{R}_z(\alpha) = \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Homogenous coordinates

Homogenous Points

- Add dimension, constrain that to be equal to 1 $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$
- Why? 4D allows us to include 3D translation in matrix form!
- Homogeneity means that any point in 3-space can be represented by an infinite variety of homogenous 4D points
  - $\begin{pmatrix} 2 \\ 3 \\ 4 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 6 \\ 8 \\ 2 \end{pmatrix} = \begin{pmatrix} 3 \\ 4.5 \\ 6 \\ 1.5 \end{pmatrix}$

Homogenous Form

- Homogenous component is preserved $\begin{pmatrix} * \\ * \\ * \\ 1 \end{pmatrix}$, and
- Aside from the translation the matrix is I

$$\begin{pmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} a + x \\ b + y \\ c + z \\ 1 \end{pmatrix}$$

Homogenous division

- We call two vectors $\begin{pmatrix} a \\ b \\ c \\ w_1 \end{pmatrix}$ and $\begin{pmatrix} x \\ y \\ z \\ w_2 \end{pmatrix}$ **homogenous** if
- $\begin{pmatrix} a/w_1 \\ b/w_1 \\ c/w_1 \\ w_1/w_1 \end{pmatrix} = \begin{pmatrix} x/w_2 \\ y/w_2 \\ z/w_2 \\ w_2/w_2 \end{pmatrix}$
- Will later be used in perspective division

Putting it together

- R is rotation and scale components
- T is translation component

$$\begin{pmatrix} R1 & R2 & R3 & T1 \\ R4 & R5 & R6 & T2 \\ R7 & R8 & R9 & T3 \\ 0 & 0 & 0 & 1 \end{pmatrix} = RT$$

Order Matters

- Composition order of transforms matters
  - Remember that basic vectors change so “direction” of translations changed

$$\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & 0 & 2\\ 0 & 1 & 0 & 3\\ 0 & 0 & 1 & 4\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ z\\ 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x+2\\ y+3\\ z+4\\ 1\end{pmatrix}=\begin{pmatrix}x+2\\ z+4\\ -y-3\\ 1\end{pmatrix}$$

$$\begin{pmatrix}1 & 0 & 0 & 2\\ 0 & 1 & 0 & 3\\ 0 & 0 & 1 & 4\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & -1 & 0 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ z\\ 1\end{pmatrix}=\begin{pmatrix}1 & 0 & 0 & 2\\ 0 & 1 & 0 & 3\\ 0 & 0 & 1 & 4\\ 0 & 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ z\\ -y\\ 1\end{pmatrix}=\begin{pmatrix}x+2\\ z+3\\ -y+4\\ 1\end{pmatrix}$$

Matrix conventions

- Using OpenGL conventions here, but you might also see a different one out there.
- Column Vectors:
  - What we use on the slides
  - Matrix-times-vector
- Row Vectors:
  - Often used in maths
  - Vector-times-matrix

- Also: Beware that OpenGL matrices are defined column-major too!

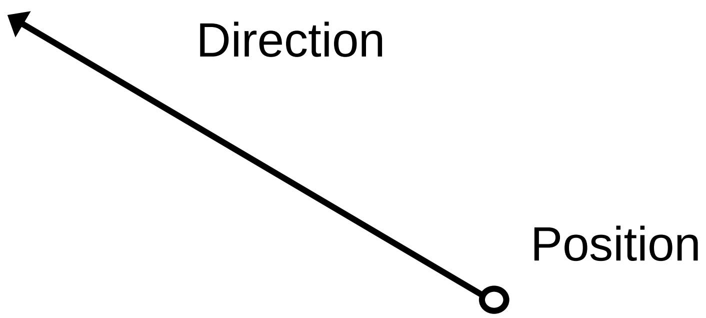
Matrix Summary

- Rotation, Scale, Translation
- Composition of transforms
- The homogenous form

Rays

Ray

- **Position**: You are somewhere
- **Direction**: You look somewhere
- **Ray**: You are somewhere *and* look somewhere



Parametric line / ray equation

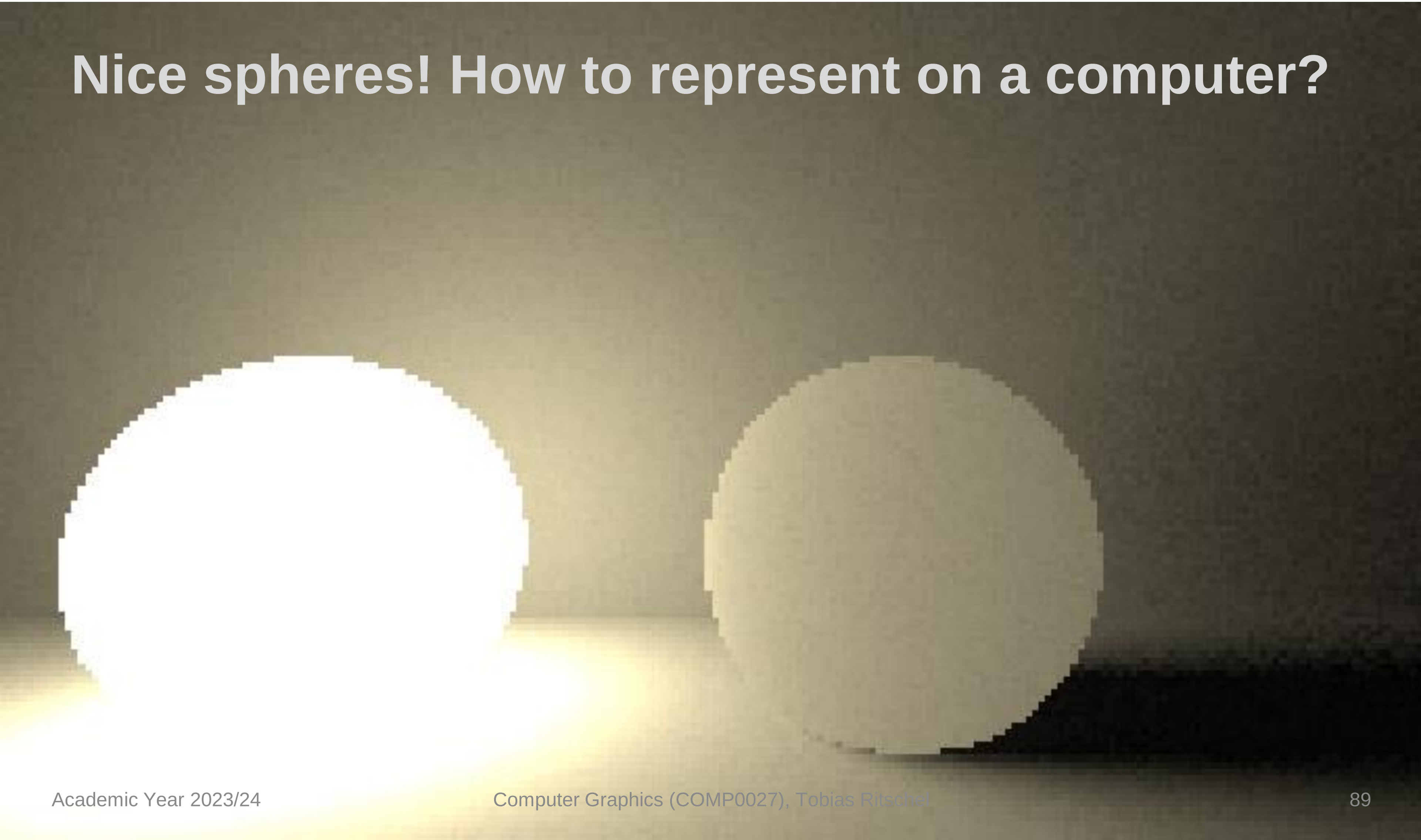
- Given two points $P_0 = (x_0, y_0, z_0)$ and $P_1 = (x_1, y_1, z_1)$
- the line passing through them can be expressed as:

$$P(t) = P_0 + t\,(P_1 - P_0) = \begin{cases} x(t) = x_0 + t\,(x_1 - x_0) \\ y(t) = y_0 + t\,(y_1 - y_0) \\ z(t) = z_0 + t\,(z_1 - z_0) \end{cases}$$

With the parameter $-\infty < t < \infty$

Spheres

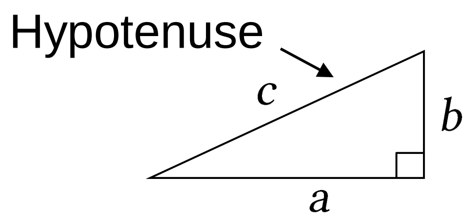
Nice spheres! How to represent on a computer?



Equation of a sphere

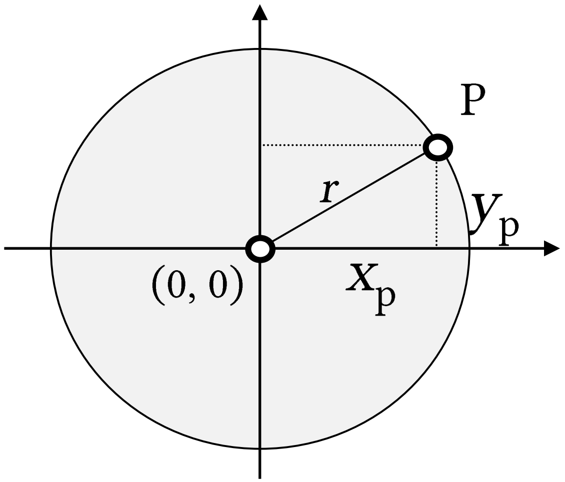
- Pythagoras Theorem:

$$a^2 + b^2 = c^2$$



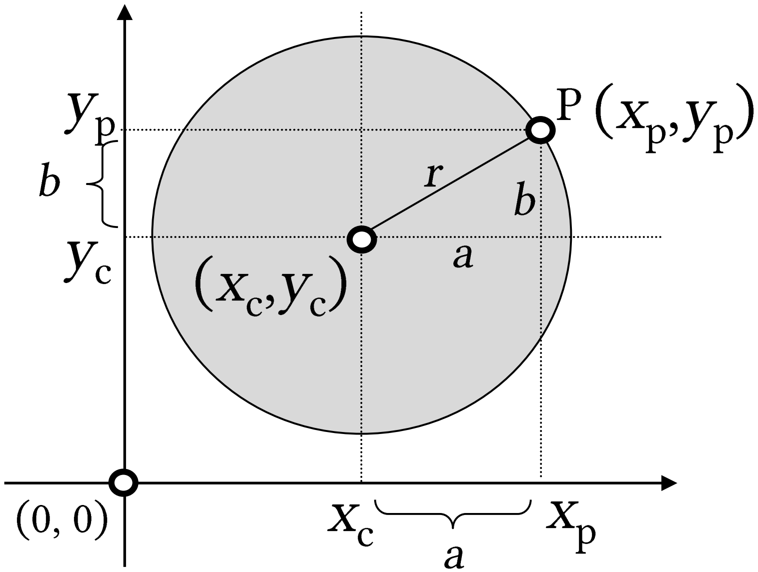
- Given a circle through the origin with radius r, then for any point P on it we have:

$$x^2 + y^2 = r^2$$



Equation of a sphere

If the circle is not centred on the origin, we still have: $a^2 + b^2 = r^2$



where
$$a = x_p - x_c$$
$$b = y_p - y_c$$

So for the general case $(x - x_c)^2 + (y - y_c)^2 = r^2$

Equation of a sphere

Pythagoras theorem generalises to 3D giving

$$a^2 + b^2 + c^2 = d^2$$

Sphere not at the origin $$(x - x_c)^2 + (y - y_c)^2 + (z - z_c)^2 = r^2$$

Sphere at the origin $$x^2 + y^2 + z^2 = r^2$$

Slightly not symmetric

- Line:
  Mapping from scalar t to 3D points
- Sphere:
  Set of all points that are solution to an equation

Wrapping up

Math Typography

- Scalar are lowercase Italic Roman x
- Points and Directions are uppercase Roman X
- Vectors are lower case Bold Roman $\mathbf{x}$
- Matrices are uppercase Sans X
- Operators are uppercase bold Roman $\mathbf{X}$

- … best done in TeX. For project reports etc.

Do I need to remember all this?

- Conceptually you should understand
- These operations have been implemented for you
- GLSL (used in coursework) has types for this
  - `vec2, vec3, vec4`
  - `mat2, mat3, mat4, even mat2x3`
  - operators like `dot(), cross(), inverse()`
- In C++, Eigen is a good lib for all this