Introduction (Computer Graphics)

Revision Notes

These revision notes serve to complement the lectures, ongoing assessments, tutorials, and practical sessions throughout the module. This material will aid you in your understanding of core theoretical concepts and prepare you for the end of semester exam. If you have any questions or problems, please don't hesitate to contact me (Ben Kenwright - b.kenwright@napier.ac.uk).

Computer Graphics (SET08116)

Tutorial Revision Notes Outline:

Contents

Week 1 - Introduction

Refresh basic principles, symbols, trigonometry.
e.g., Ask how we distinguish a unit-vector from a scalar when we write them down?

- What is Pythagoras theorem (2D & 3D)?
- What are the three formulae: “sin”, “cos”, “tan”?

  sin = opposite / hypotenuse

  cos = adjacent / hypotenuse

  tan = opposite / adjacent

  Tip: There are a couple of ways to help you remember which formula to use. Remember SOHCAHTOA (it sounds like ’Sockatoa’) or Some Old Hag Cracked All Her Teeth On Apples.
- Sum of angles inside a triangle (basic geometry)?
- What is PI (π)?
- What are radians and degrees?
- Algebra (manipulation)
- Comprehend mathematical concepts (sqrt, power, linear, algebra, equations)
- Common sense & logical thinking ability (numerical manipulation)

> These fundamental mathematical concepts are important and you need to know them for the exam!

Week 2 - Vectors

Give the equations for:

- Dot Product
- Cross Product
- Plane Equation
- Shortest distance between a point and a plane
- Projection Equation
  (Apply this later for reflection calculations see lecture slides)

Dot

$$\begin{aligned} \vec{a} \cdot \vec{b} &= a_x b_x + a_y b_y + a_z b_z \\ &= |\vec{a}|\ |\vec{b}| cos\theta \end{aligned} \tag{1}$$

Cross

$$\begin{aligned} \vec{a} \cdot \vec{b} &= (a_y b_z - a_z b_y, a_z b_x - a_x b_z, a_x b_y - a_y b_x) \\ &= |\vec{a}|\ |\vec{b}| sin\theta \end{aligned} \tag{2}$$

Plane Equation

$$\vec{p} \cdot \hat{n} = d \tag{3}$$

where $\vec{p}$ is a point on the plane, and $\hat{n}$ is the plane normal, and d is the shortest distance from the plane to the origin.

Week 3 - Matrices

Give the 4x4 matrix for:

- Identity
- Translation
- Transpose
- Matrix-Matrix Multiplication
- Scale
- Rotation around X/Y/Z Axis
- View Matrix
- Projection Matrix

Matrix Identity

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

Matrix Scale

$$I = \begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

where s_x, s_y, and s_z are the scales along the x, y, and z axis.

Matrix Translation

$$I = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

where t_x, t_y, and t_z is the translation along the x, y, and z axis.

Rotation Z

$$\mathbf{R}_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

Rotation X

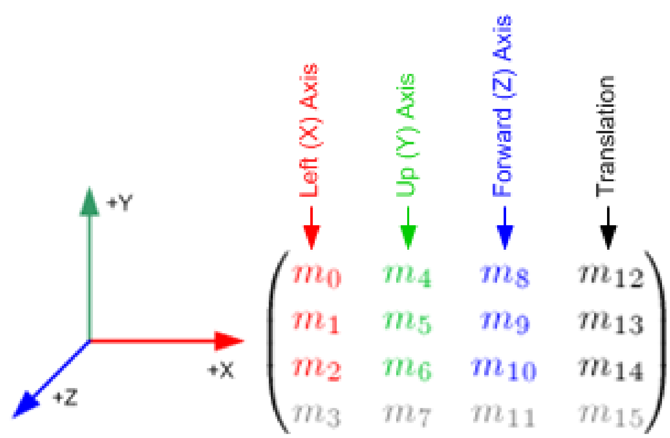
$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{8}$$

Rotation Y

$$\mathbf{R}_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{9}$$

Matrix Multiplication

Model View

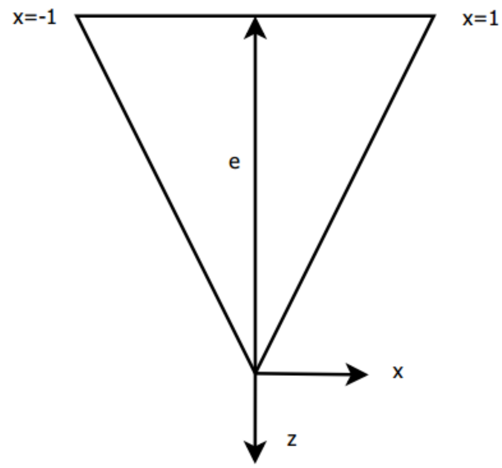


Field of View We define a camera, or a view frustum, as six planes:

- Left and Right
- Top and Bottom
- Near and Far

Camera space or eye space is the coordinate space where the camera location is centred at the origin, the x axis to the right, and y axis upward. The z axis depends on the library used (left handed and right handed space).

We can view illustrate our field of view as follows:



$$e = \frac{1}{tan\frac{\alpha}{2}} \tag{10}$$

where e is the near plane distance given the horizontal field of view angle α.

Our vertical field of view is determined by our aspect ratio - the screen height divided by the width. We denote this value as the aspect ratio (a). We also need our actual near plane distance n and far plane distance f.

We find the plane equation for each of the six frustum planes as:

$$\begin{aligned}
\text{Near} &=< 0,0,1,n > \\
\text{Far} &=< 0,0,1,f > \\
\text{Left} &=< \frac{e}{\sqrt{e^2+1}}, 0, -\frac{e}{\sqrt{e^2+2}} > \\
\text{Right} &=< -\frac{e}{\sqrt{e^2+1}}, 0, -\frac{e}{\sqrt{e^2+2}} > \\
\text{Bottom} &=< 0, \frac{e}{\sqrt{e^2+a^2}}, -\frac{a}{\sqrt{e^2+a^2}} > \\
\text{Top} &=< 0, -\frac{e}{\sqrt{e^2+a^2}}, -\frac{a}{\sqrt{e^2+a^2}} >
\end{aligned} \tag{11}$$

Projection Matrix

$$M_{\text{frustum}} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2nf}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{12}$$

where r,l,t,b,f and n are the right, left, top, and bottom values for the screen and the far and near values for the frustum.

Projection We transform a camera space coordinate to screen space by multiplying by this matrix:

$$\vec{P}' = M_{\text{frustum}}\vec{P} = \begin{bmatrix} \frac{2n}{r-l} & 0 & \frac{r+l}{r-l} & 0 \\ 0 & \frac{2n}{t-b} & \frac{t+b}{t-b} & 0 \\ 0 & 0 & -\frac{f+n}{f-n} & -\frac{2nf}{f-n} \\ 0 & 0 & -1 & 0 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \tag{13}$$

We then divide the x, y and z components of $\vec{p}'$ by the w component to produce the final screen space coordinate. For the point to be visible, it must lie in the cube defined by the minimum and maximum points $< 1,1,1 >$ and $< 1,1,1 >$.

You need to be able to create, concatenate and manipulate matrix transforms and be able to apply them (e.g., transforming vector coordinates).

Week 4 - Quaternions

Give the equation for:

- Identity
- Quat-Quat Multiplication
- Axis-Angle to Quaternion
- Conjugate

Quaternion Identity

$$\begin{aligned} q &= (v_x, v_y, v_z, s) <=> q = (\vec{v}, s) \\ &= (0, 0, 0, 1) \end{aligned} \tag{14}$$

Quaternion Multiplication

$$\begin{aligned} q_a\ q_b &= (\vec{v}_a, s_a)\ (\vec{v}_b, s_b) \\ &= s_a\ s_b + \vec{v}_a\ s_b + \vec{b}\ s_a - \vec{v}_a \cdot \vec{v}_b + \vec{v}_a \times \vec{v}_b \end{aligned} \tag{15}$$

Quaternion from Axis-Angle

$$q = \left(\vec{v} \sin\left(\frac{\theta}{2}\right), \cos\left(\frac{\theta}{2}\right) \right) \tag{16}$$

where $\vec{v}$ is the axis, and θ is the angle of rotation.

Week 5 - Lighting Part 1

Revision

- Vectors
- Matrices
- Quaternions

Lighting (Part 1)

Give equation for:

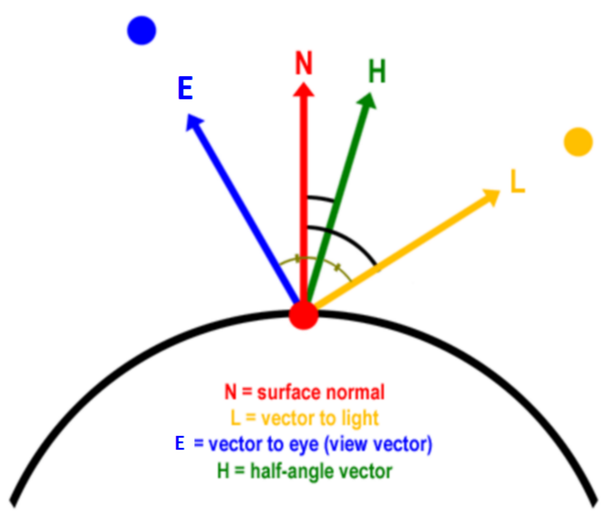
- Ambient
- Diffuse
- Specular (Half-Angle)

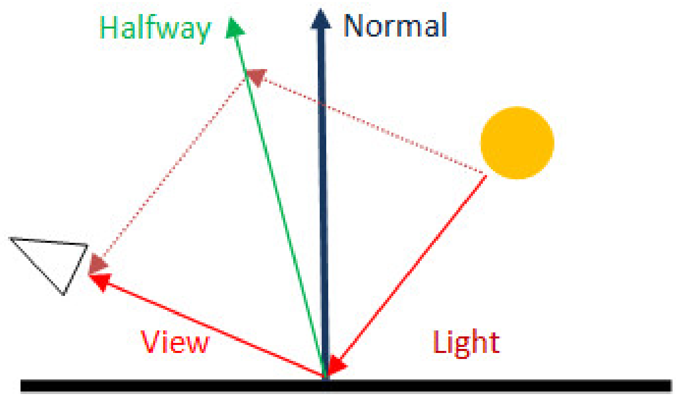
Explain each of the lighting equation components and what it contributes to the scene.

Lighting Equations

$$\begin{aligned}
\text{Ambient} &= DA \\
\text{Diffuse} &= (DC)max(\hat{N} \cdot \hat{L}, 0.0) \\
\text{Specular} &= (SD)max(\hat{N} \cdot \hat{H}, 0.0)^m \\
\text{Final Colour} &= \text{Ambient} + \text{Diffuse} + \text{Specular}
\end{aligned} \tag{17}$$

$$\begin{aligned}
\hat{H} &= \frac{\hat{L} + \hat{E}}{||\hat{L} + \hat{E}||} \\
\hat{E} &= \frac{\vec{e} - \vec{p}}{||\vec{e} - \vec{p}||}
\end{aligned} \tag{18}$$





Point Light The standard equations for lighting is modified to take into account distance (attenuation) by scaling the value of C.

$$K = DA + (DC)max(\hat{N} \cdot \hat{L}, 0) + (SC)max(\hat{N} \cdot \hat{H}, 0)^m \tag{19}$$

We include **attenuation** as:

$$C = \frac{1}{k_c + k_l d + k_q d^2} C' \tag{20}$$

where d is the distance from the point to the light source, and k_c, k_l, and k_d are constants.

Phong-Gourad Shading: Describe the difference between normal interpolation ('Phong-shading') and color-interpolation ('Gouraud-shading'). Name one advantage for each method over the other.

Answer:

Gouraud shading computes the lighting model (such as the Phong model) at the vertices of the triangle and interpolates the color across the triangle. Phong shading interpolates the normals of the vertices and re-executes the lighting calculation at every pixel. Before the advent of Teraflop-capable pixel-shader batteries in consumer GPUs, Phong-shading was too costly for real-time applications, because much more computation was necessary at each pixel. The model is still more costly today, but it is now feasible in real-time.

Lighting is an essential component of the computer graphics module. You need to be able to give and explain the fundamental equations for lighting. This includes using them in a practical context, with numerical values.

Week 6 - Cameras & Matrices

Revision

- Matrices
- Vectors
- Lighting

Cameras & Further Matrices

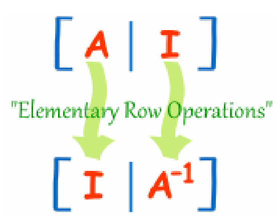
- Write down determinant (3x3) matrix
- Explain why we need to inverse a matrix
- Show how the Gauss-Jordan elimination technique works (inverse)
- Why are scene graphs important for rendering complex scenes?

- Review Cameras
- Write down view matrix
- Write down perspective matrix
- Write down frustum planes
- Explain the 'Model-View-Projection' matrix

Matrix Determinant The determinant tells us things about the matrix that are useful in systems of linear equations, calculus, and can also help us find the inverse of a matrix, and much more. For us to find the determinant, the matrix must be square (i.e. have the same number of rows as columns). We calculate the determinant using basic arithmetic concepts.

Exercises - find the determinant of a 2x2 matrix and a 3x3 matrix.

Gauss-Jordan Elimination (Inverse) We can find the inverse of a matrix using the Gauss-Jordan method. It is a straightforward technique that using elementary row operations. The Gauss-Jordan method starts with a matrix - we then write it down an Identity Matrix next to it (known as the 'Augmented Matrix'). We then manipulate the rows (e.g., by adding, multiplying or swapping) until we make our staring Matrix into an Identity Matrix I - and the Identity matrix next to it into the Inverse Matrix.

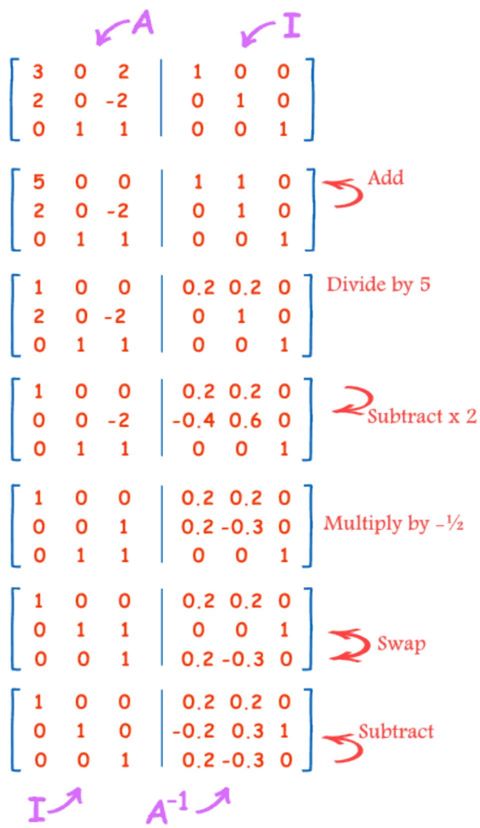


Example: find the Inverse of "A" using the Gauss-Jordan method:

$$A = \begin{bmatrix} 3 & 0 & 2 \\ 2 & 0 & -2 \\ 0 & 1 & 1 \end{bmatrix}$$

We start with the matrix A, and write it down with an Identity Matrix I next to it:

Now we do our best to turn A (the Matrix on the left) into an Identity Matrix. The goal is to make Matrix A have 1s on the diagonal and 0s elsewhere (an Identity Matrix) ... and the right hand side comes along for the ride, with every operation being done on it as well. We can only do these "Elementary Row Operations":

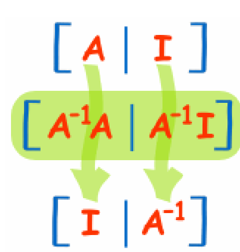


Hence, matrix A has been made into an Identity Matrix, and at the same time an Identity Matrix got made into A^{-1}:

$$A^{-1} = \begin{bmatrix} 0.2 & 0.2 & 0 \\ -0.2 & 0.3 & 1 \\ 0.2 & -0.3 & 0 \end{bmatrix}$$

$$A\ A^{-1} = I$$

How the Gauss-Jordan method works?



The total effect of all the row operations is the same as multiplying by A^{-1}. Hence, matrix A becomes an identity (because $A^{-1}A = I$). And the identity matrix becomes A^{-1} (because $A^{-1}I = A^{-1}$).

You need to be able to explain what the determinant is and be able to calculate the determinant for simple matrices (e.g., 2x2 or 3x3). You need to be able to use the Gauss-Jordan elimination technique. Scene graphs are an essential element for complex scenes - you should be able to explain why. You also need to be able to write down the view/perspective matrices.

Week 7 - Lighting Part 2

Revision

- Vectors
- Matrices
- Quaternions
- Lighting (Part 1
- Cameras
  Write down the view matrix
  Write down perspective matrix

Lighting (Part 2)

Spot Light For spot light we modify our colour calculation to the following:

$$C = \frac{max(-\hat{R} \cdot \hat{L}, 0)^p}{k_c + k_l d + k_q d^2} C' \tag{21}$$

where R is the direction the spot light is facing and p is the spot light power, d is the distance from the point to the light source, and k_c, k_l, and k_d are constants.

- Explain how the lighting equations are adapted for artistic control
- Explain how the reflection angle is calculated using the projection equation (dot)

You need to be able to understand how artists control the lighting equations.

Week 8 - Intersections (Geometry)

Give the equations for:

- Intersection examples (simple geometry/triangles/rays/points)
- Plane-Point
- Plane-Ray
- Plane-Sphere
- Ray-Sphere
- View-Frustum

Plane-Point A plane is represented by $ax + by + cz = d$ or $\hat{n} \bullet \vec{p} = d$; where $\hat{n}$ (or a, b, c) is the plane normal , and $\vec{p}$ (or x, y, z) is a point on the plane. We can derive the shortest distance from an arbitrary point $\vec{q}$ to the plane, given by:

$$\hat{n} \bullet (\vec{p} - \vec{q}) = d_q \tag{22}$$

where d_q is the shortest distance to the plane. We multiply the plane normal and subtract it from the arbitrary point to get the point on the plane surface:

$$\vec{q}_{plane} = \vec{q} - d_q \hat{n} \tag{23}$$

Plane-Ray

$$\begin{aligned} \text{Ray} \quad & \vec{p} = \vec{p}_0 + t\vec{v} \\ \text{Plane} \quad & \vec{p} \bullet \hat{n} = d \\ \text{Point-Plane} \quad & (\vec{p} - \vec{q}) \bullet \hat{n} = d \end{aligned} \tag{24}$$

Substitute for q in point-plane with the ray, we get:

$$\begin{aligned} (\vec{p} - \vec{p}_0 + t\vec{v}) \bullet \hat{n} &= d \\ \text{d=0 if on the plane} & \\ (\vec{p} - \vec{p}_0) \bullet \hat{n} + (t\vec{v} \bullet \hat{n}) &= 0 \\ (t\vec{v} \bullet \hat{n}) &= (\vec{p} - \vec{p}_0) \bullet \hat{n} \\ t &= ((\vec{p} - \vec{p}_0) \bullet \hat{n})/(\vec{v} \bullet \hat{n}) \end{aligned} \tag{25}$$

Solution:

$$\begin{aligned} t = ((\vec{p} - \vec{p}_0) \bullet \hat{n})/(\vec{v} \bullet \hat{n}) \\ \vec{p} = \vec{p}_0 + t\vec{v} \end{aligned} \tag{26}$$

Plane-Sphere We calculate the distance between the centre of the sphere and the plane. If the distance is less than the radius, the sphere is intersecting the plane. To push the sphere out of intersection, we use the plane normal and the penetration depth (i.e., radius minus the distance to the plane).

Ray-Sphere As shown in Figure 1, we can use geometric principles, such as, Pythagorean theorem and the projection equation, to calculate the intersection of a ray and a sphere.

With reference to Figure 1, here are the steps:

1. we can calculate $\vec{L}$ (i.e., $\vec{C} - \vec{O}$)
2. we use the projection equation to calculate d

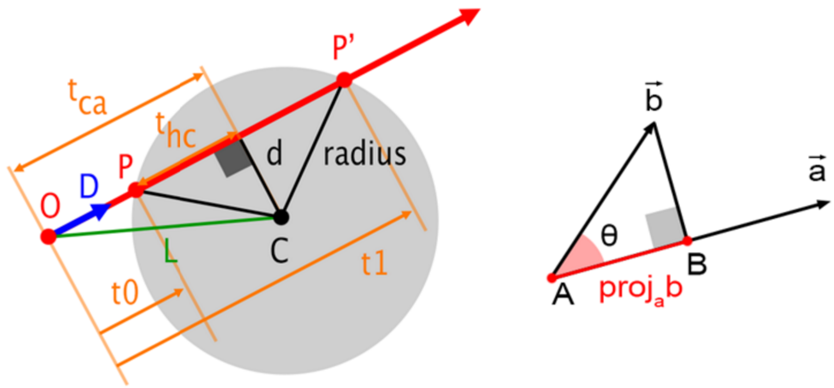


Figure 1: Geometric principles to calculate the intersection of a ray and sphere.

3. Pythagoras lets us calculate the length t_{ca} (i.e., $|L|^2 = |t_{ca}|^2 + |d|^2$)
4. we also use Pythagoras for the 'inner' triangle to calculate t_{hc} (i.e., $|t_{hc}|^2 = d^2 + r^2$) - this works because the distance between $P - C$ is the same as the sphere radius
5. subtract $|t_{ca}| - |t_{hc}|$ to get the distance from the ray origin to the intersection point

View-Frustum In 3D computer graphics, the view-frustum (also called viewing frustum) is the region of space in the modelled world that may appear on the screen; it is the field of view of the notional camera. We model the view-frustum as a collection as planes.

> You must be able to calculate the view-frustum and be able to determine if a point is inside or outside of the view (i.e., plane equation).

Week 9 - Interpolation (LERP/Slerp/Splines)

Give the LERP/Slerp equations and examples of where you would use them

- Equation for interpolating along a straight line
- Explain & show the equation for a Bezier spline (refer to lecture slides)
- Understand and reproduce simple interpolation equations (linear and angular)

Week 10 - Texturing

- Explain texturing (e.g., u,v coordinates)
- Explain mip-mapping
- Texture projection techniques
- Explain procedural texturing (with examples)

> You need to be able to explain texturing in the context of computer graphics, such as, mip-mapping, projection techniques, procedural approaches, and uv mapping.

Week 11 - 15 Revision

- Examine material from previous sections, while discussing practical graphical aspects, such as, floating point precision, texturing, coordinate transforms, optimisations. Apply numbers to the equations e.g., using a calculator to prepare you for the exam
- Practice revision questions for the exam