Computer Graphics 1 - Introduction

Tom Thorne

thomas.thorne@ed.ac.uk
www.inf.ed.ac.uk/teaching/courses/cg/

What is in the course?

Graphics algorithms, data structures and rendering

- object representations
- 3D transformations
- rasterisation
- lighting and shading

OpenGL

- Lab session

Photorealistic rendering

- Ray tracing
- Monte Carlo methods
- Photon mapping

Overview for today

- Topics and applications in computer graphics
- SIGGRAPH
- Quick refresher on vector and matrix algebra
- Graphics pipelines, lectures overview

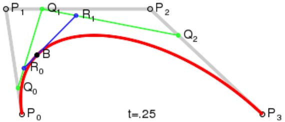
Streams of computer graphics

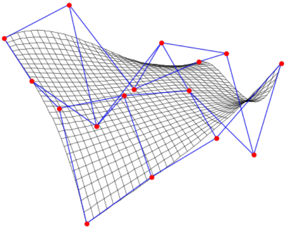
- 2D and 3D modeling
- Interactive applications
- Realistic rendering

Modeling

Designing curved surfaces – car and aircraft manufacturers

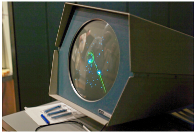
- Renault, Pierre Bézier
- Citroën, Paul de Casteljau

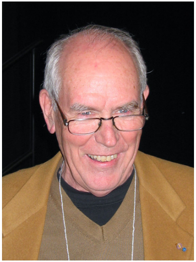


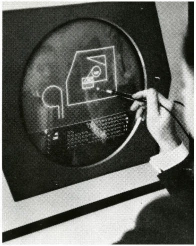


Interactive applications

- Sketchpad (1963, Ivan Sutherland)
- Flight simulations
- Games: Tennis (1958), Spacewar (1962)



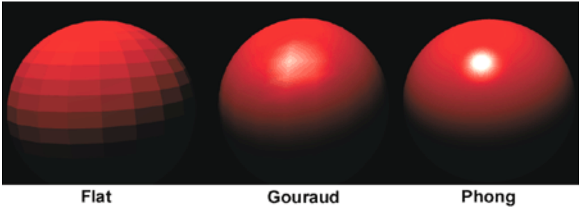




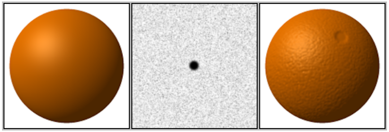
Realistic rendering

Lots of work at the Univeristy of Utah

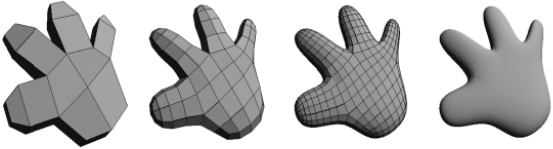
- Gouraud shading (1971)
- Phong illumination model (1973)
- Phong shading (1975)



Realistic rendering



- Bump mapping (Blinn, 1978)
- Subdivision surfaces (Catmull, Clark, 1978)



Applications - Animation and film

Applications - Animation and film

Realistic lighting

- Reflection of light around the scene

Physical simulation

- making fluids, smoke, solid objects etc. move realistically

Character animation

- realistic or natural looking movement and facial expressions

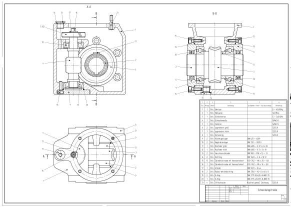
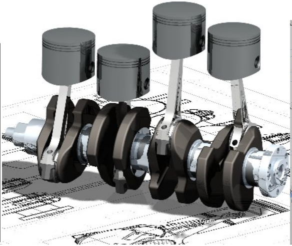
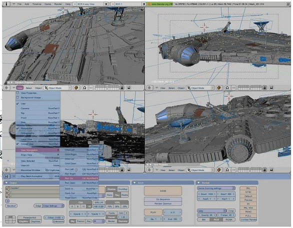
Applications - Games

Real time, usually at 30 or 60 frames per second

- Rendering
- Character movement, AI
- Physical simulations
- User input

Modeling

Computer aided design and 3D modeling

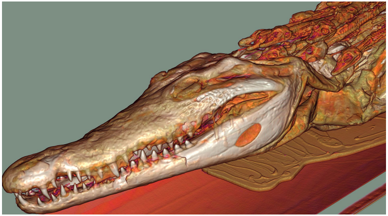
Applications - Scientific visualisation

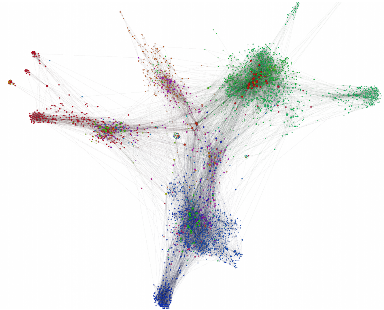
Transforming large amounts of data into forms that are easy to interpret

- volume rendering
- complex networks

Challenges

- reducing number of dimensions
- interactivity
- visual perception





SIGGRAPH

Special Interest Group on Graphics and Interactive Techniques

- The world's biggest computer graphics conference, running since 1974
- Many import techniques published as SIGGRAPH papers
- `http://http://www.siggraph.org`
- Collection of recent papers here `http://kesen.realtimerendering.com`

Maths for computer graphics

Computer graphics involves lots of maths

- 2D and 3D coordinates
- Matrix transformations
- Reflection and refraction
- Curved surfaces

Vector spaces

Laws

$$\begin{aligned}
\boldsymbol{a}+\boldsymbol{b} &= \boldsymbol{b}+\boldsymbol{a} \\
(\boldsymbol{a}+\boldsymbol{b})+\boldsymbol{c} &= \boldsymbol{a}+(\boldsymbol{b}+\boldsymbol{c}) \\
\lambda(\boldsymbol{a}+\boldsymbol{b}) &= \lambda\boldsymbol{a}+\lambda\boldsymbol{b} \\
(\lambda\mu)\boldsymbol{a} &= \lambda(\mu\boldsymbol{a}) \\
\boldsymbol{a}+\boldsymbol{0} &= \boldsymbol{a} \\
\forall\boldsymbol{a}\exists\boldsymbol{b} : \boldsymbol{a}+\boldsymbol{b}=\boldsymbol{0} & \\
1\boldsymbol{a} &= \boldsymbol{a}
\end{aligned}$$

E.g. two dimensions

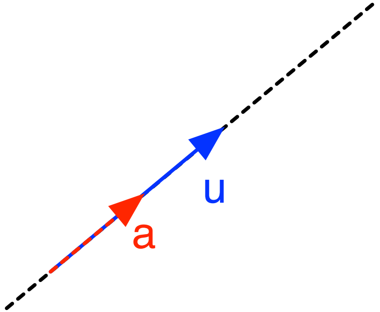
$$\begin{aligned}
\boldsymbol{u} &= \begin{pmatrix} x \\ y \end{pmatrix} \\
\|\boldsymbol{u}\| &= \sqrt{x^2+y^2}
\end{aligned}$$

N dimensions

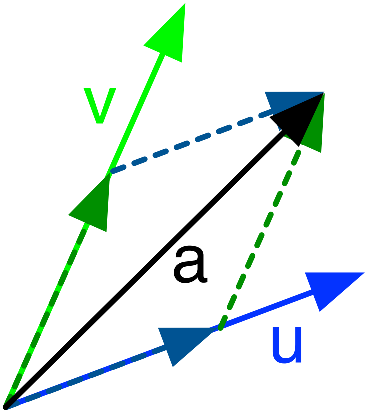
$$\begin{aligned}
\boldsymbol{v} &= \begin{pmatrix} x_1 \\ \vdots \\ x_N \end{pmatrix} \\
\|\boldsymbol{v}\| &= \sqrt{x_1^2+x_2^2+\ldots+x_N^2}
\end{aligned}$$

Subspaces

Line
$\boldsymbol{a} = \lambda \boldsymbol{u}$

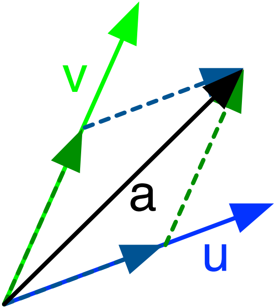


Plane
$\boldsymbol{a} = \lambda \boldsymbol{u} + \mu \boldsymbol{v}$



3D space
$\boldsymbol{a} = \lambda \boldsymbol{u} + \mu \boldsymbol{v} + \nu \boldsymbol{w}$

Linear dependence



Two vectors are *linearly dependent* if:

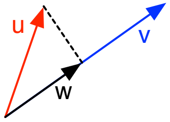
- $\alpha \boldsymbol{a} + \beta \boldsymbol{b} = \mathbf{0}$

for some α and β not both zero. More generally for *n* vectors:

- $\alpha \boldsymbol{a} + \beta \boldsymbol{b} + \ldots + \nu \boldsymbol{n} = \mathbf{0}$

Dot product

$$\boldsymbol{u} \cdot \boldsymbol{v} = \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \cdot \begin{pmatrix} v_1 \\ \vdots \\ v_n \end{pmatrix} = u_1 v_1 + u_2 v_2 + \cdots + u_n v_n$$



$$\begin{aligned} \boldsymbol{u} \cdot \boldsymbol{v} &= \|\boldsymbol{u}\| \|\boldsymbol{v}\| \cos\theta \\ \boldsymbol{u} \cdot \boldsymbol{v} &= \boldsymbol{v} \cdot \boldsymbol{u} \\ \boldsymbol{u} \cdot (\lambda \boldsymbol{v} + \mu \boldsymbol{w}) &= \lambda(\boldsymbol{u} \cdot \boldsymbol{v}) + \mu(\boldsymbol{u} \cdot \boldsymbol{w}) \end{aligned}$$

$$\|w\| = u \cdot v$$

Coordinate systems

In an N dimensional vector space V, a set of N linearly independent vectors forms a *basis*. For some vector $\boldsymbol{u} \in V$:

$$\boldsymbol{u} = \alpha \boldsymbol{e}_1 + \beta \boldsymbol{e}_2 + \ldots + \nu \boldsymbol{e}_N$$

A basis $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_N$ is *orthogonal* if for all $\boldsymbol{e}_i, \boldsymbol{e}_j, i \neq j$:

$$\boldsymbol{e}_i \cdot \boldsymbol{e}_j = 0$$

A basis $\boldsymbol{e}_1, \ldots, \boldsymbol{e}_N$ is *orthonormal* if it is orthogonal and for all $\boldsymbol{e}_i$:

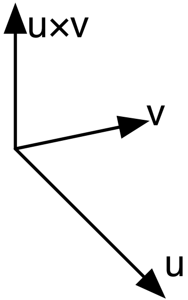
$$\|\boldsymbol{e}_i\| = 1$$

Cross product

$$\|\boldsymbol{u} \times \boldsymbol{v}\| = \|\boldsymbol{u}\|\|\boldsymbol{v}\| \sin\theta$$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \times \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} bz - cy \\ cx - az \\ ay - bx \end{pmatrix}$$

$$\boldsymbol{v} \times \boldsymbol{u} = -\boldsymbol{u} \times \boldsymbol{v} \neq \boldsymbol{u} \times \boldsymbol{v}$$



Matrix algebra

$$A = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & \dots & \dots & a_{2n} \\ \vdots & & & \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

$$\begin{aligned} A(BC) &= (AB)C \\ AB &\neq BA \\ (A+B)C &= AC+BC \\ AI &= IA = A \\ (AB)^T &= B^T A^T \\ A^{-1}A &= I \end{aligned}$$

Light transport

- Ultimate aim is to simulate light transport

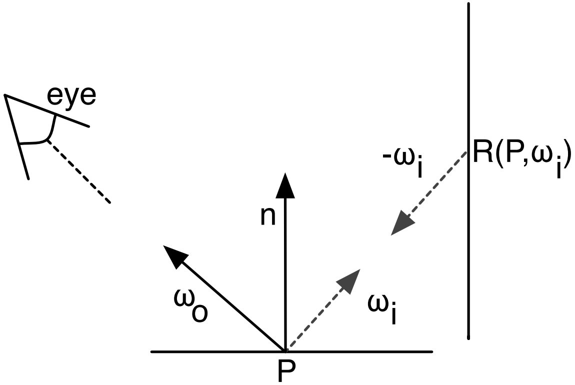
Real time rendering:

- Various approximations, smoke and mirrors (tricks)

Photorealistic rendering:

- Ray tracing
- Approximation through Monte Carlo integration
- Hybrid methods

Light transport equation



$$
\begin{aligned}
L(P,\omega_o) &= L^e(P,\omega_o) + L^{\text{ref}}(P,\omega_o) \\
&= L^e(P,\omega_o) \\
&\quad + \int_{\omega_i \in S(P)} L(R(P,\omega_i), -\omega_i) f_r(P,\omega_i,\omega_o)(\omega_i \cdot \boldsymbol{n}_P) d\omega_i
\end{aligned}
$$

Graphics pipeline

Geometry

- Transformation
- Perspective projection
- Hidden surface removal

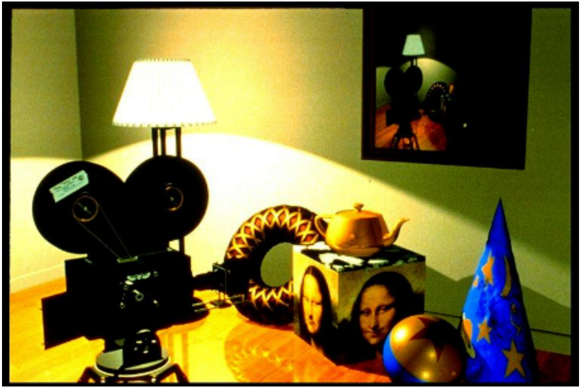
Shading and lighting

- Reflections
- Shadows

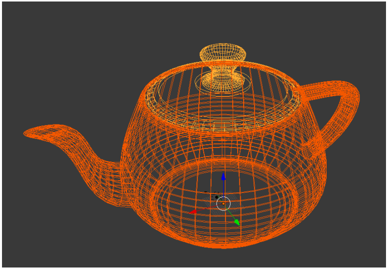
Rasterisation

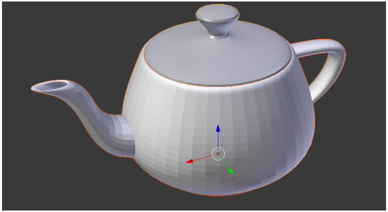
- Anti aliasing
- Texture mapping
- Bump mapping
- Ambient occlusion

Example scene



Object models (lecture 2)





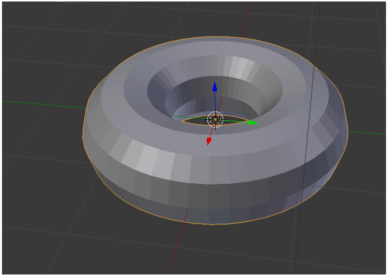
Geometry stage

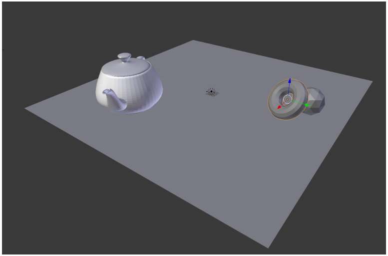
- Transforming objects
- View transformation
- Illumination and shading (for vertices)

Object transformation (lecture 3)

Placing objects in the scene

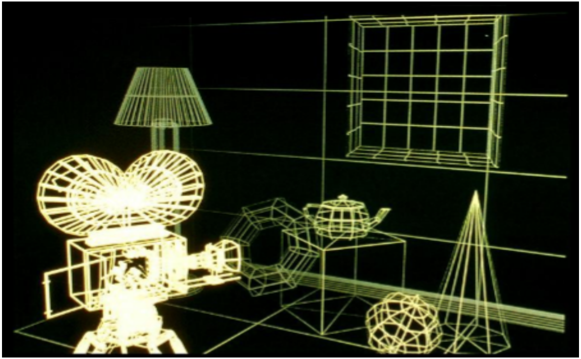
- Rotation
- Scaling
- Translation



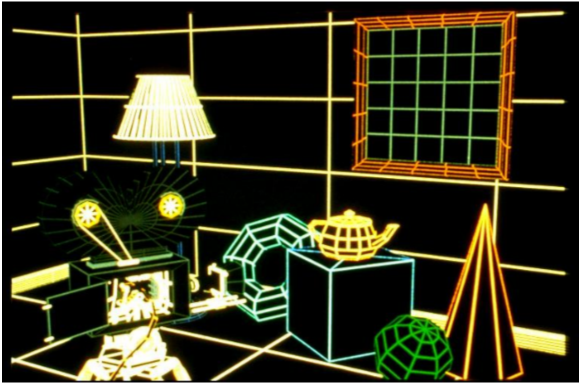


Perspective projection

Viewing the scene from a specific camera position - perspective transformation of 3D coordinates to 2D screen coordinates

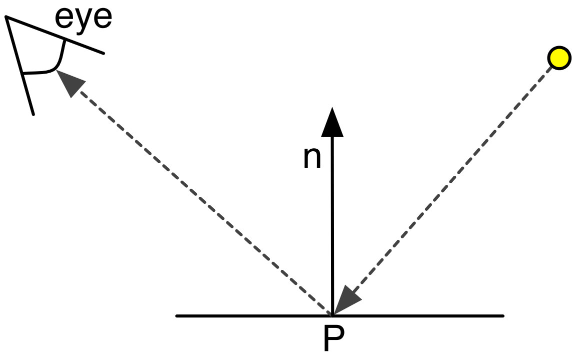


Hidden surface removal (lecture 13)

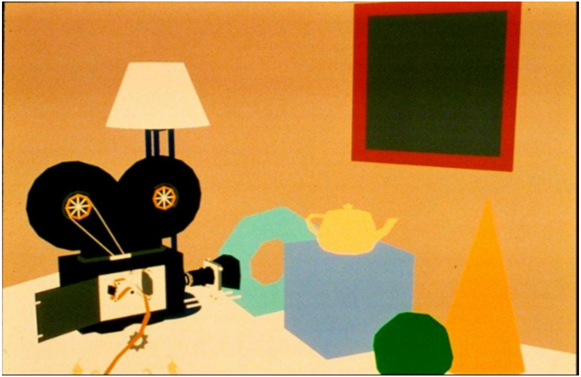


Shading and lighting (lecture 5)

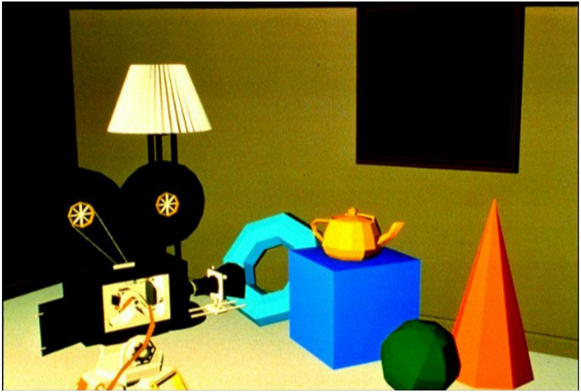
Determining the lighting of each pixel



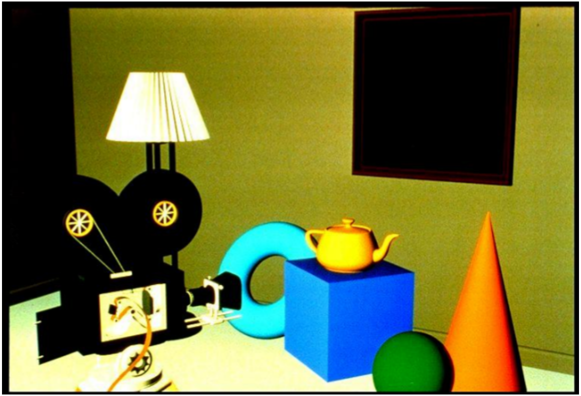
Constant shading



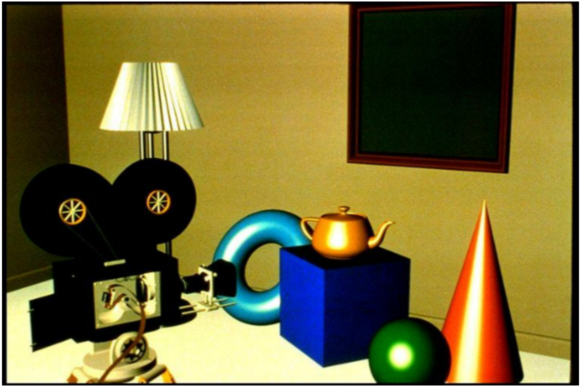
Flat shading



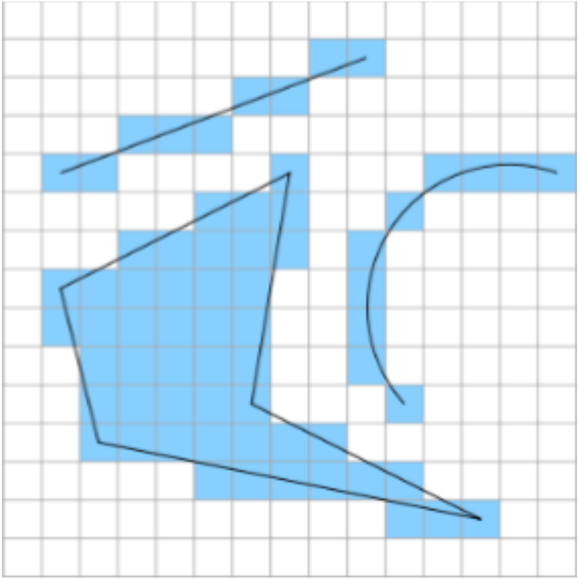
Gouraud shading



Specular highlights



Rasterisation (lecture 7)

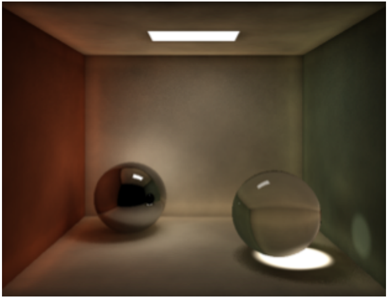


Texture mapping (lecture 8)

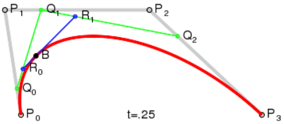
Other effects

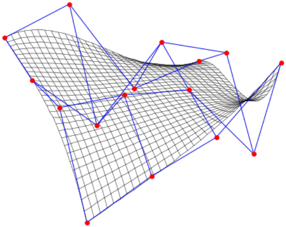
- Bump mapping (lecture 14)
- Reflections (lecture 9)
- Shadows (lecture 12)

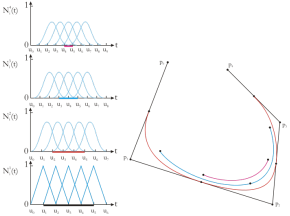
Photorealistic rendering (lectures 14/15)

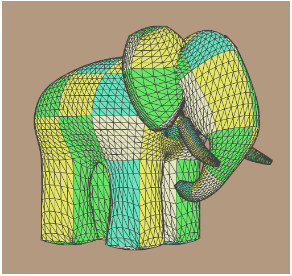


Curves and surfaces (lectures 16/17)









Coursework

Coursework 1:

- OpenGL
- Lecture 4 is a lab session for coursework 1 and to introduce OpenGL
- Deadline 4pm on 24/10/14

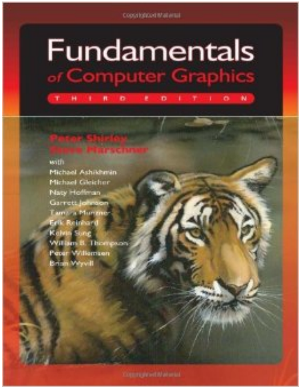
Coursework 2:

- Raytracing
- Lecture 11 is a lab session for coursework 2
- Deadline 4pm on 21/11/14

For both:

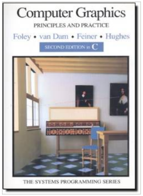
- Written in C++
- **Must** compile and run on DICE (Scientific Linux 6.5)
- See course page for submission details

Books

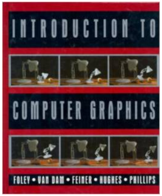


Fundamentals of Computer Graphics
Shirley and Marschner, CRC Press, 2010.

Books



Computer Graphics Principles and Practice
Foley, van Dam, Feiner and Hughes, Addison Wesley, 1997.



Introduction to Computer Graphics
Foley, van Dam, Feiner, Hughes and Phillips, Addison Wesley, 1995.

References

Reading

- Shirley, Chapter 2.4 (Miscellaneous Math – Vectors)
- Foley, Appendix A.1-A.5 (Maths for Computer Graphics)