CZ2003
Computer Graphics and Visualization

SCSE, NTU

Dr. Alexei SOURIN

Associate Professor
Deputy Director of Fraunhofer IDM@NTU
N4-02a-10 www.ntu.edu.sg/home/assourin

Introduction to computer graphics and foundation mathematics

Module 1

Lecture 1

Rationale for the Course

- Information visualization and visual analytics as graphics rendering of abstract data becomes more and more important in modern science and technology.
- Graphic rendering of data with non-graphical nature is well beyond using plots and diagrams, and it requires rendering various mathematical models.
- Many engineered software systems employ graphical presentations as part of user interfaces.
- Computer graphic finds applications across many discipline including scientific, interior designs, and architectural designs among many others.

Learning Objectives

- You will learn how to see geometry and colors beyond mathematical formulas and to represent geometric shapes and motions by analytical functions and algorithmic procedures.
- You will develop spatial visualization skills which will permit you to perform 3D rotations and other transformations and motions of geometric objects.
- You will also learn how a common personal computer can be used for solving complex computer graphics problems.

Course

- Lectures (26 hours), tutorials (12 tutorials), labs (5 experiments).
- The lectures mostly concentrate on visualization of mathematics: using mathematics for defining geometric shapes, colors and transformations.
- Labs (in SW1B Lab) start on weeks 3-4 according to the individual lab schedules (see the course site).
- 5 assignments. Assessment electronically after week 13. Lab mark contributes 30% to the final mark.
- Tutorials begin next week (week 2). Written solutions must be submitted to the tutors at the beginning of each tutorial. Tutorial solutions will be marked:
  1 mark (~>60% correct), 0.5 mark (~30-60% correct),
  0 mark(~<30% correct, or plagiarism, or no solution is submitted).
  Tutorial marks will contribute 10% to the final mark.

Lectures

1. **Introduction to computer graphics and foundation mathematics – 2 hours**
   Principles of visualization. Coordinate systems. Time as another dimension. Vectors and matrices. Vector and matrix algebra with application to geometric coordinate space. Geometric meaning of dot and cross products. Surface normal calculation.
2. **Programming computer graphics and visualization – 2 hours**
   Introduction to common computer graphics software used for solving engineering and information visualization problems. Introduction to the software used in the coursework.
3. **Geometric shapes – 9 hours**
   Points, polygons, voxels, and procedural models. Analytical definitions of curves, surfaces, and solid objects. Shape animation and morphing. Principles of graphics rendering geometry.
4. **Transformations and motions – 5 hours**
   Affine transformations. Translation, Scaling, Rotation. Composition of transformations. Motions by parametric functions and time-dependent affine transformations.
5. **Illumination and Texture Mapping – 6 hours**
   RGB color model and light sources. Illumination calculation. Image texture mapping. 3D geometric textures. Appearance assignment to geometry.
6. **Efficient rendering – 2 hours**
   Hierarchical representation. Spatial partitioning. Bounding volumes. Level of detail.
7. **Revision – 2 hours**

Tutorials (from week 2)

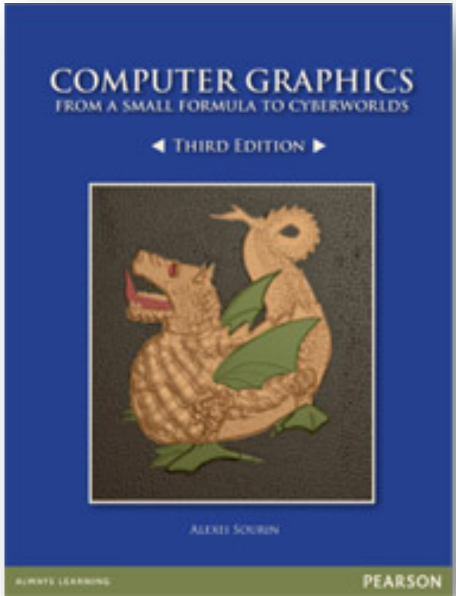
- T1. Visual mathematics and visualization pipeline
- T2. Mathematical functions in computer graphics
- T3. Geometry: Curves
- T4. Geometry: Planes, Polygons and Bilinear Surfaces
- T5: Geometry: Surfaces by Sweeping
- T6. Geometry: Solid objects
- T7. Electronic week tutorial: Blobby shapes

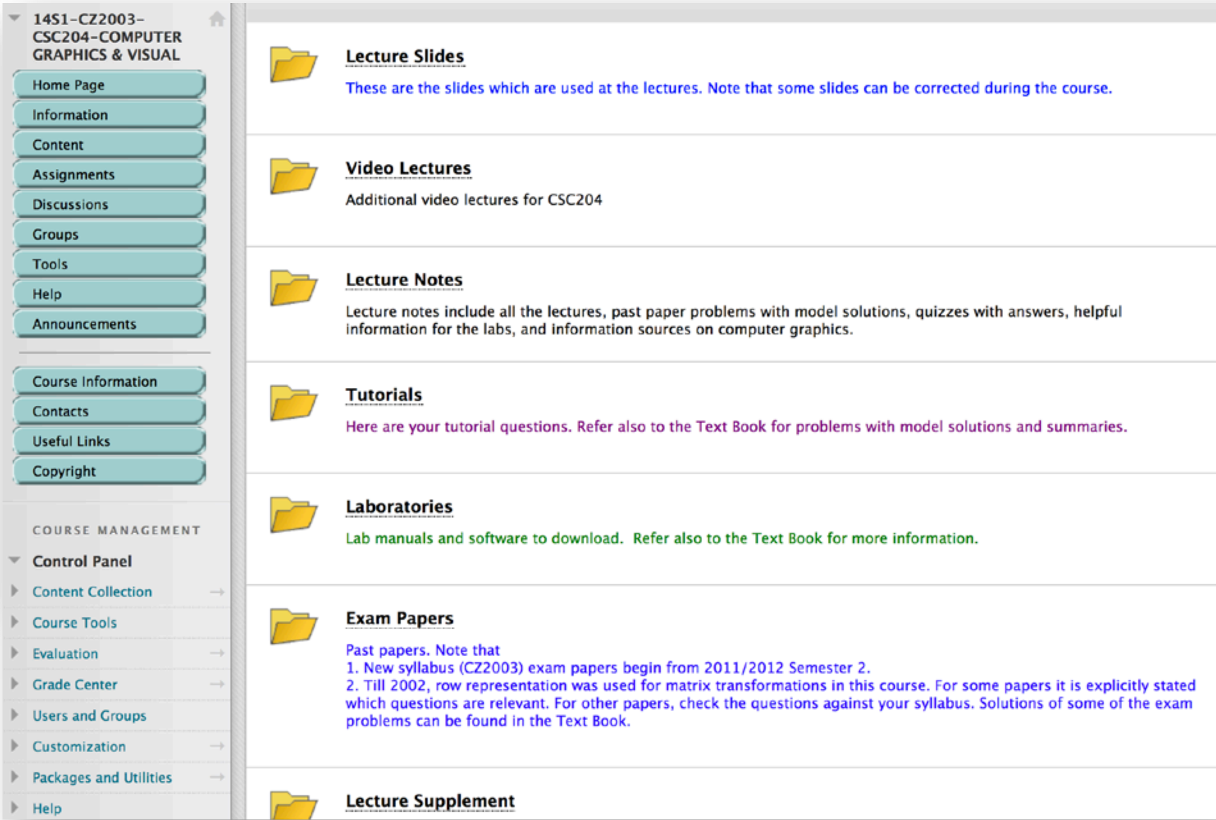
- T8. Transformations: 2D
- T9. Transformations: 3D
- T10. Motions
- T11. Illumination
- T12. Surface texture mapping
- T13. Efficient rendering

Labs (from week 3): Visual Mathematics

1. Polygons
2. Parametric curves
3. Parametric surfaces and solids
4. Implicit Solids
5. Morphing

Text Book and Course Site





http://www.ntu.edu.sg/home/assourin/Book

Introduction to Computer Graphics and Visualization

- Definitions
- Visualization steps

Definitions

- *Computer Graphics* concerns the pictorial *synthesis* of real or imaginary objects from their computer-based models
- *Visualization*

  To form an image of something; envisage; to make a physical 3D model

- *Image Processing* is the converse process: the *analysis* of scenes, or *reconstruction* of models of 2D or 3D objects from their pictures

Graphics Display

- **Pixel** – picture element
- **Resolution:** number of pixels which can be displayed horizontally and vertically.
- **Number of colors:**
  Maximum actual number of colors depends on the graphics display device and the video card. Device independent colors are programed as real numbers between 0 and 1

LCD Display

- **RGB color model**: addition of *red+green+blue=white*
  used with graphics displays (used in this course)

- **CMYK color model**: subtraction of *cyan+magenta+yellow=black (key)*
  used when printing (not used in the course)

- **Polygonization:** to calculate polygons interpolating surfaces of shapes
- **Shading**: filling in surface of polygons with colored pixels

Visualization Steps

- Define Objects: 3D shapes, defined by polygonal surfaces (triangle sets)
- Define a Viewpoint (viewer position and orientation) and viewing parameters
- Define light source(s): ambient, point, directional, etc.
- Define visible surface material properties

Summary

- Computer graphics makes images with computers
- Visualization requires: object model (geometry + material), light source(s), observer

Introduction to computer graphics and foundation mathematics

Module 1. Lecture 2

Learning objectives

- To understand that definition of geometry with a computer requires using its digital representation in a form of coordinates as well as various mathematical functions defining the coordinates in given coordinate domains with selected sampling resolutions.

Foundation Mathematics

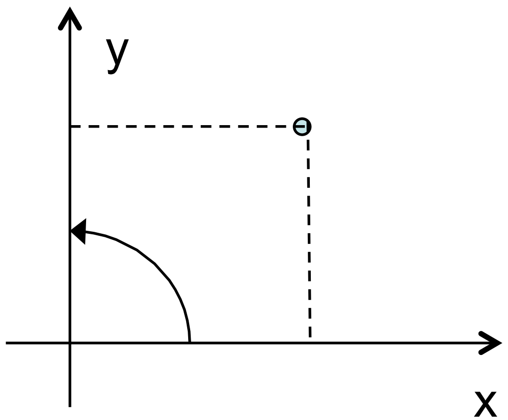
- Coordinate systems
- Analytic functions
- Pythagorean theorem
- Trigonometry
- Matrices
- Vectors

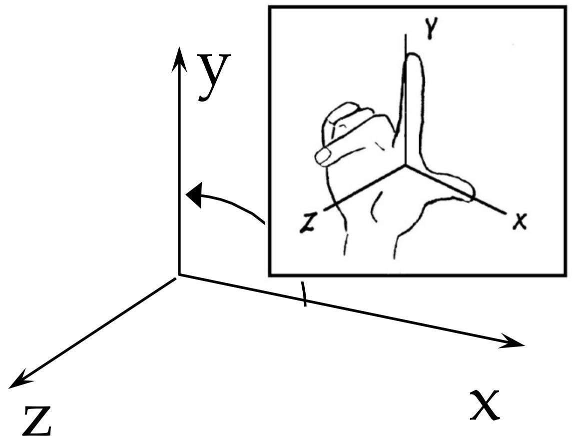
Coordinate systems

- René Descartes in 1637 revolutionized mathematics by providing the first systematic link between Euclidean geometry (“Elements” 300 BC) and algebra. He discovered an ability to define geometry by algebraic formulas
- This required some system of measuring locations of points in space by numbers
- Coordinates are signed numbers used to **uniquely** determine the position of a point or other geometric elements in the modeling space
- Number of coordinates = dimension of space, i.e. plane – 2 coordinates, 3D space – 3 coordinates
- Coordinate systems:
  - Cartesian Coordinate System (2D and 3D).
  - Polar
  - Cylindrical
  - Spherical
- Time will be considered as yet another dimension (coordinate)

Cartesian Coordinate system

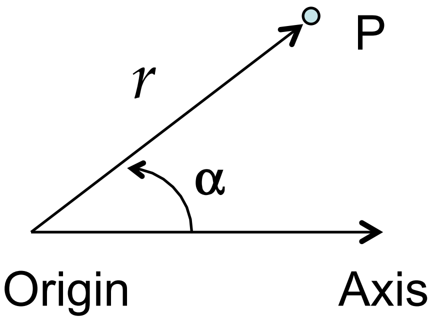
- Cartesian Coordinate System (2D and 3D). Cartesius is a Latinized name of Descartes.
- Coordinate axes are located at 90 degrees to each other
- Location of point P is defined by coordinates which are distances from the point to the axes measured along the lines orthogonal to the axes
- Cartesian coordinates are the foundation of analytic geometry linear algebra, complex analysis, differential geometry, multivariate calculus, group theory, and more.
- *Right-handed* 2D coordinate system: if rotation of the first axis towards the second axis is counter-clockwise.
- *Right-handed 3D coordinate system:* if the three vectors are situated like the thumb, index finger and middle finger pointing straight up from your palm. Also, rotation of the first axis towards the second axis is counter-clockwise as seen from the third axis.
  Also, while curling fingers from the first to the second axis, the extended thumb will show the direction of the third axis.

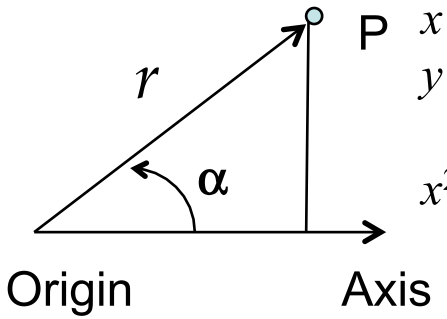




Polar Coordinate system

- Defined by an axis and an origin point on it.
- Location of a 2D point P is defined as a distance *r* from the origin to the point and an angle α between the axis and the vector cast towards the point
- Distance *r* is positive
- Angle α is from 0 to 2π
- Positive α is measured counter-clockwise



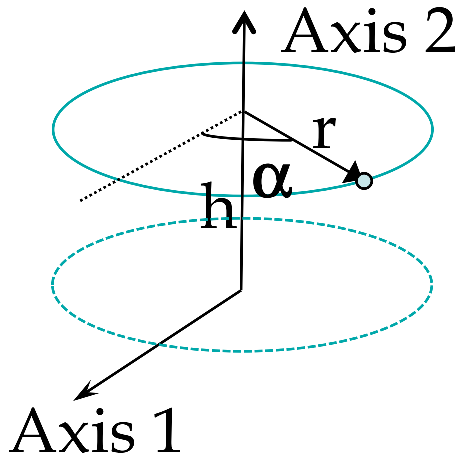


$x = r\cos\alpha$
$y = r\sin\alpha$

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

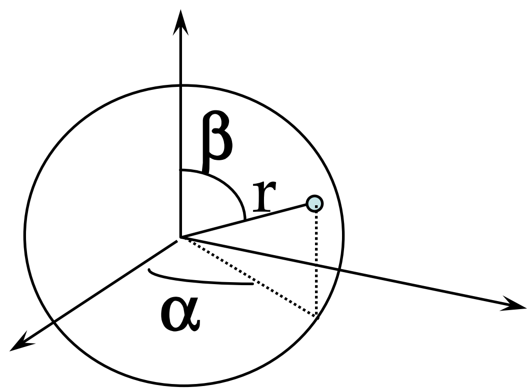
Cylindrical Coordinate system

- Extension of polar system to 3D
- Defined by two orthogonal axes and an origin which is their intersection
- Location of a 3D point P is defined as a displacement *h* along the second axis towards the plane orthogonal to the two axes and containing the point P, a distance *r* from the second axis to the point, an angle α between the first axis and the vector cast towards the point
- Distance *h* can be positive and negative
- Distance *r* is positive
- Angle α is from 0 to 2π
- Positive α is measured counter-clockwise as seen towards the origin



Spherical Coordinate system

- Extension of polar system to 3D
- Defined by three orthogonal axes and an origin which is their intersection
- Location of a 3D point P is defined by a distance *r* from the origin to the point, an two angles α (azimuth) and β (zenith) between the first axis and the vector cast towards the point
- Distance r is positive
- Angle α is from 0 to 2π
- Angle β is from 0 to 1π



Check point: Previously we learnt

- Coordinate systems are used for defining individual objects, groups of objects, the whole simulated scene, the observer location and output graphics objects (e.g., pixels) on the graphics devices
- Coordinates are signed numbers used to **uniquely** determine the position of a point or other geometric elements in the modeling space
- Number of coordinates = dimension of space,
  i.e. plane – 2 coordinates, 3D space – 3 coordinates
- Coordinate systems:
  - Cartesian Coordinate System 2D and 3D (x, y and x, y, z).
  - Polar 2D (r, α)
  - Cylindrical 3D (r, α, h)
  - Spherical 3D (r, α, β)
- Time will be considered as yet another dimension (coordinate)

Mathematical functions

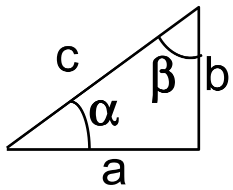
- In mathematics, a **function** associates one quantity, the *argument* of the function, also known as the *input*, with another quantity, the *value* of the function, also known as the *output*. A function assigns exactly one output to each input. Values from the input domain map to the values in the function range.
- Coordinate functions:
  - Explicit way of function definition
    - $x=f(y),\quad y=f(x),\quad z=f(x,y),\quad g=f(x,y,z)$
  - Implicit way of function definition
    - $f(x)=0,\quad f(x,y)=0,\quad f(x,y,z)=0$
  - Parametric way of function definition (pa'ram'etric, pa'rameter)
    - $x=f_x(t)\quad y=f_y(t)\quad z=f_z(t)\quad t=[t_1,\ t_2]$
    - $x=f_x(u,v)\quad y=f_y(u,v)\quad z=f_z(u,v)\quad u=[u_1,\ u_2]\quad v=[v_1,v_2]$
    - $x=f_x(u,v,w)\ y=f_y(u,v,w)\ z=f_z(u,v,w)\ u=[u_1,u_2]\ v=[v_1,v_2]\ w=[w_1,w_2]$

Mathematical functions

- Conversion from explicit $y=f(x)$ to implicit $f(x,y)=0$:
  - by making an equation $y-f(x)=0$
- Conversion from explicit $y=f(x)$ to parametric $x=f_x(u)$ $y=f_y(u)$:
  - by introducing parameter(s). The simplest, but not always the best way, is to assign $x=u$, then $y=f(u)$
- Conversion from parametric $x=f_x(u)$ $y=f_y(u)$ to explicit $y=f(x)$ or implicit $f(x,y)=0$
  - by expressing parameter u as a function of x from the first equation and then by substituting it into the second equation
  - by eliminating parameter u while doing algebraic manipulations with the two equations (raising to power, multiplications, additions, subtractions, divisions, etc.).

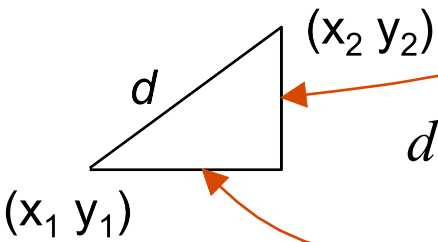
Pythagoream theorem

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



$a=c\cdot\cos\alpha \qquad b=c\cdot\sin\alpha$
$b=c\cdot\cos\beta \qquad a=c\cdot\sin\beta$
$b=a\cdot\tan\alpha \qquad a=b\cdot\tan\beta$

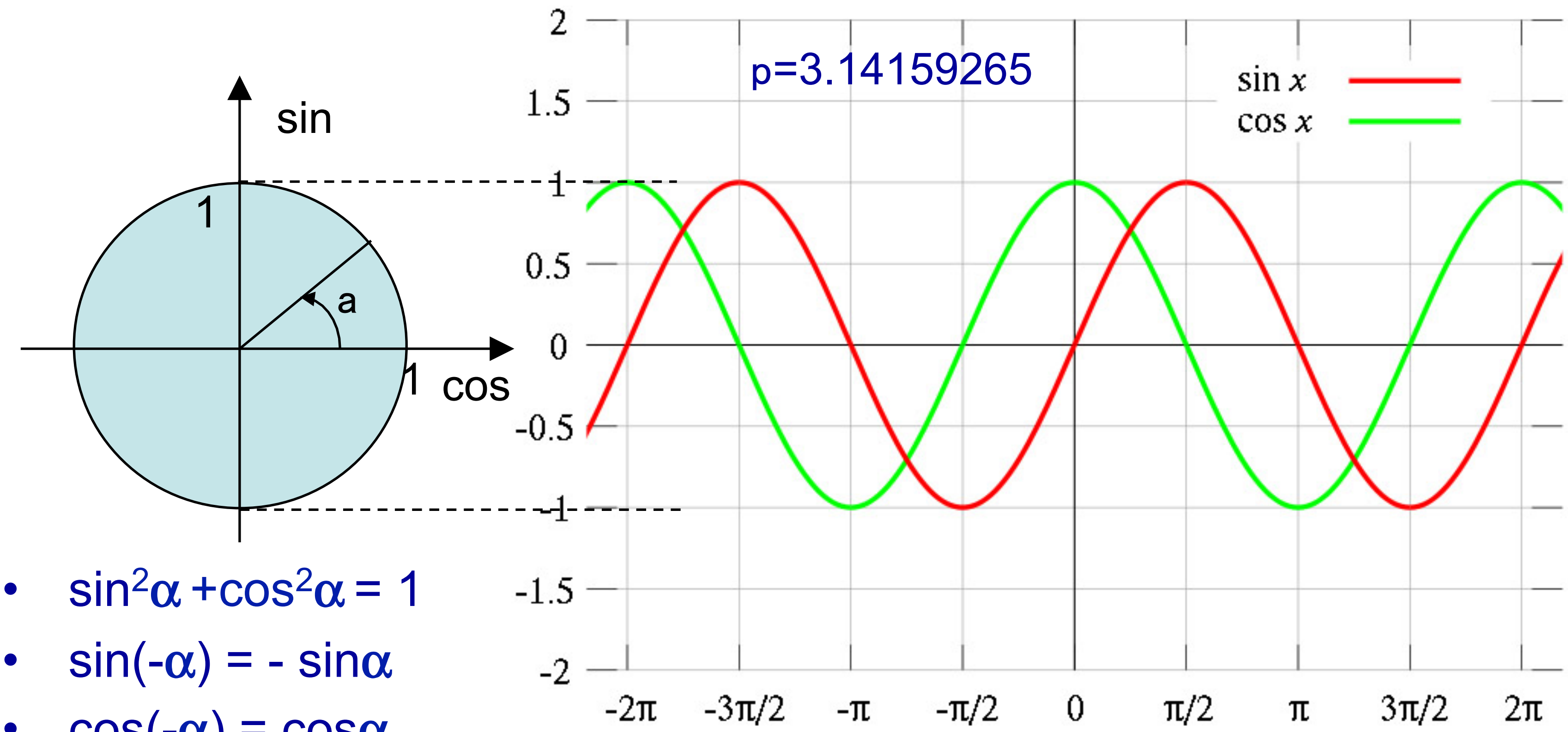
- Distance d between two points with coordinates $(x_1\ y_1)$ and $(x_2\ y_2)$ as a consequence of the theorem



$$d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2}$$

$$d=\sqrt{(x_2-x_1)^2+(y_2-y_1)^2+(z_2-z_1)^2}$$

Trigonometry



- $\sin^2\alpha + \cos^2\alpha = 1$
- $\sin(-\alpha) = -\sin\alpha$
- $\cos(-\alpha) = \cos\alpha$
- $\text{Sin}(\alpha+\beta) = \sin\alpha\cos\beta + \cos\alpha\sin\beta$
- $\text{Cos}(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta$

Any oscillation like in bouncing motion, surface height, change of color, etc.

Matrices

- A **matrix** (plural **matrices**) is a rectangular array of numbers denoted as:

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

- Basic matrix operations:
  - Addition $\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} + \begin{bmatrix} g & h & i \\ j & k & l \end{bmatrix} = \begin{bmatrix} a+g & b+h & c+i \\ d+j & e+k & f+l \end{bmatrix}$
  - Scalar multiplication $r \cdot \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} = \begin{bmatrix} r \cdot a & r \cdot b & r \cdot c \\ r \cdot d & r \cdot e & r \cdot f \end{bmatrix}$
  - Transpose $\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}^T = \begin{bmatrix} a & d \\ b & e \\ c & f \end{bmatrix}$

Matrices

- Matrix multiplication

$$\begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix} \begin{bmatrix} g & h \\ i & j \\ k & l \end{bmatrix} = \begin{bmatrix} ag+bi+ck & ah+bj+cl \\ dg+ei+fk & dh+ej+fl \end{bmatrix}$$

Multiplication of two matrices is defined only if the number of columns of the left matrix is the same as the number of rows of the right matrix.

Matrices

- Determinant

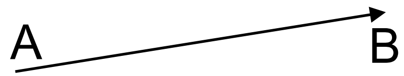
$$\det\begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - cb$$

$$\det\begin{bmatrix} a & b & c \\ d & e & f \\ g & h & k \end{bmatrix} = a\det\begin{bmatrix} e & f \\ h & k \end{bmatrix} - b\det\begin{bmatrix} d & f \\ g & k \end{bmatrix} + c\det\begin{bmatrix} d & e \\ g & h \end{bmatrix}$$

Sarrus' rule

Vectors

Vector is a geometric object that has both a magnitude (or length) and direction. A vector is visually represented by an arrow, connecting an **initial point** *A* with a **terminal point** *B,* and denoted by $\overrightarrow{AB}$ or **AB.**



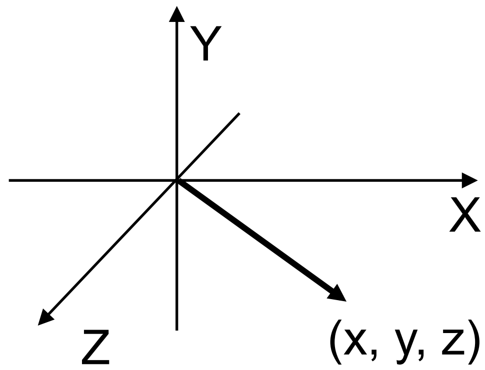
A vector is what is needed to "carry" the point *A* to the point *B.*

A vector can be represented by identifying the coordinates of its initial and terminal point. For instance, the points A = (a,b,c) and B = (d,e,f).

Vector coordinates

- To calculate with vectors, they are defined by the coordinates of their endpoints assuming that the tail of the vector coincides with the origin. The endpoint coordinates are arranged into column or raw vectors, particularly when dealing with matrices

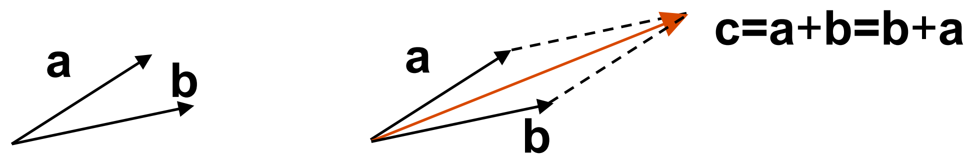
$$\mathbf{a} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \mathbf{a} = \begin{bmatrix} x & y & z \end{bmatrix}$$



Calculations with Vectors

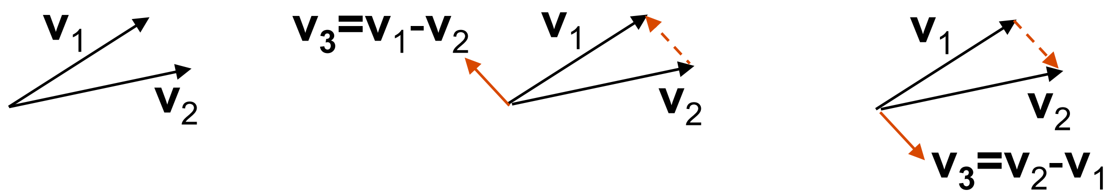
- Two vectors are said to be equal if they have the same magnitude and direction. Equivalently they will be equal if their coordinates are equal.
- Sum of vectors: $\mathbf{a}=[a_1\ a_2\ a_3]$, $\mathbf{b}=[b_1\ b_2\ b_3]$

  $\mathbf{c}=\mathbf{a}+\mathbf{b}=[a_1+b_1\ \ a_2+b_2\ \ a_3+b_3]$ $\mathbf{c}=\mathbf{b}+\mathbf{a}=[a_1+b_1\ \ a_2+b_2\ \ a_3+b_3]$



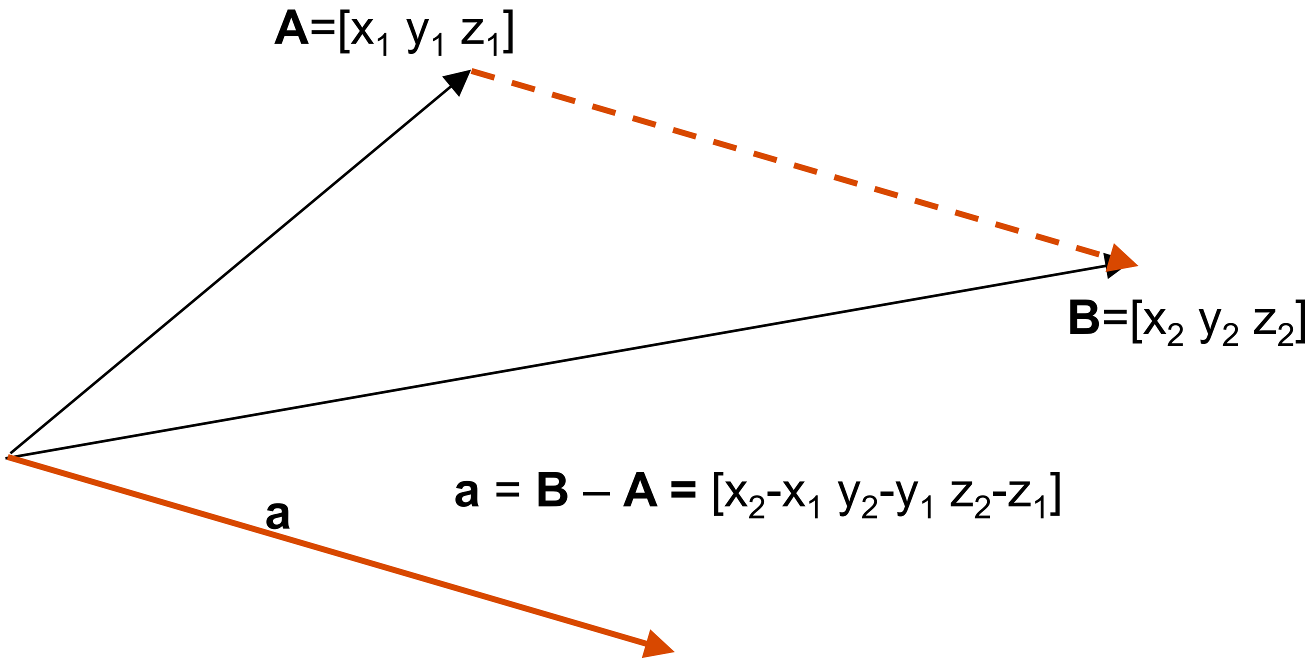
- Subtraction of vectors $\mathbf{a}=[a_1\ a_2\ a_3]$, $\mathbf{b}=[b_1\ b_2\ b_3]$

  $\mathbf{c}=\mathbf{a}-\mathbf{b}=[a_1-b_1\ \ a_2-b_2\ \ a_3-b_3]$ $\mathbf{c}=\mathbf{b}-\mathbf{a}=[b_1-a_1\ \ b_2-a_2\ \ b_3-a_3]$



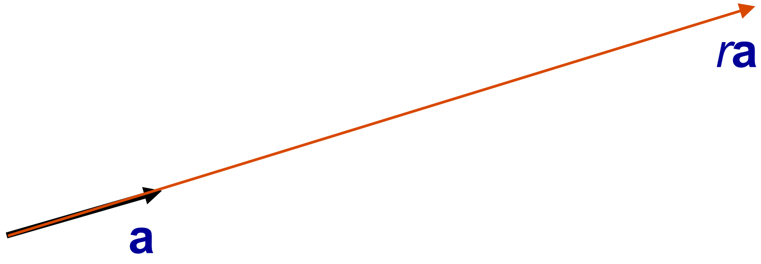
Calculations with Vectors

- Definition of a vector **a** from point A with coordinates (x_1, y_1, z_1) to point B with coordinates (x_2, y_2, z_2)



Calculations with Vectors

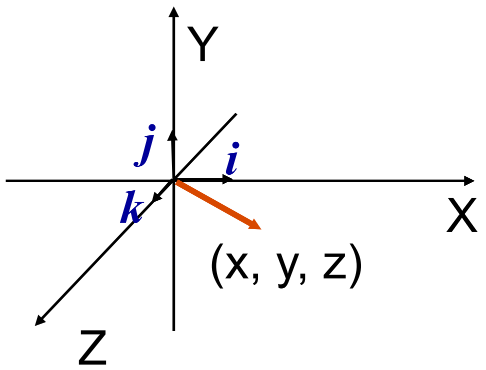
- A vector may also be multiplied, or re-*scaled*, by a real number r
  $\mathbf{a}=[a_1\ a_2\ a_3]$ $\quad r\mathbf{a} = [ra_1\ ra_2\ ra_3]$

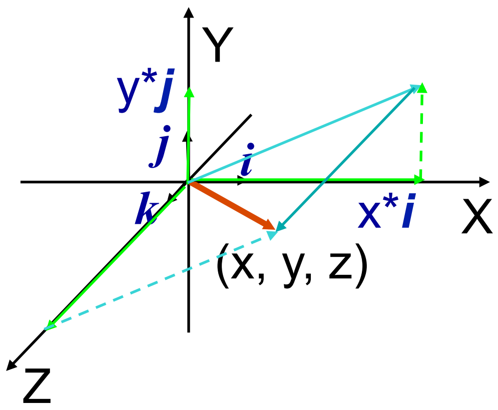


Calculations with Vectors

- Another way to represent a vector is using unit length vectors defining X, Y, and Z coordinate axes and coordinates x, y, z

  a = x****i*** + y****j*** + z****k***

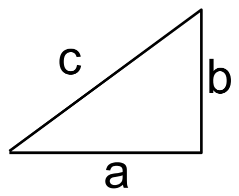




- Vector magnitude calculation:

  a=[x y z] ||a||= $\sqrt{x^2 + y^2 + z^2}$

  which is a consequence of the by Pythagorean formula $c^2=a^2+b^2$



Calculations with Vectors

- A **unit vector** is any vector with a length of one; normally unit vectors are used simply to indicate direction. A vector of arbitrary length can be divided by its length to create a unit vector. This is known as **normalizing** a vector.

  a=[x y z] ||a||= $\sqrt{x^2 + y^2 + z^2}$

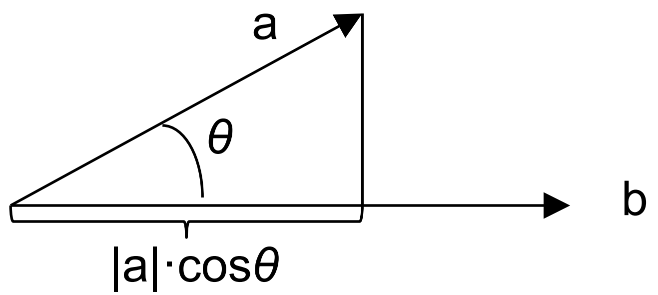
  normalized $\mathbf{a}_n$= [x/||a|| y/||a|| z/||a||]

Calculations with Vectors

- *Dot product* or *scalar product* of two vectors:

  $\mathbf{a}=[a_1\ a_2\ a_3]$, $\mathbf{b}=[b_1\ b_2\ b_3]$ $\mathbf{a}\cdot\mathbf{b} = |a|\cdot|b|\cdot\cos\theta$ where θ is the measure of the angle between **a** and **b**

  Projection of one vector onto another followed by scaling



  The dot product can also be defined as the sum of the products of the components of each vector as

  $\mathbf{a}\cdot\mathbf{b} = a_1\cdot b_1 + a_2\cdot b_2 + a_3\cdot b_3$

  The result of dot product is a **number** or scalar.

  For unit vectors, the dot product is a *cos* of the angle between them

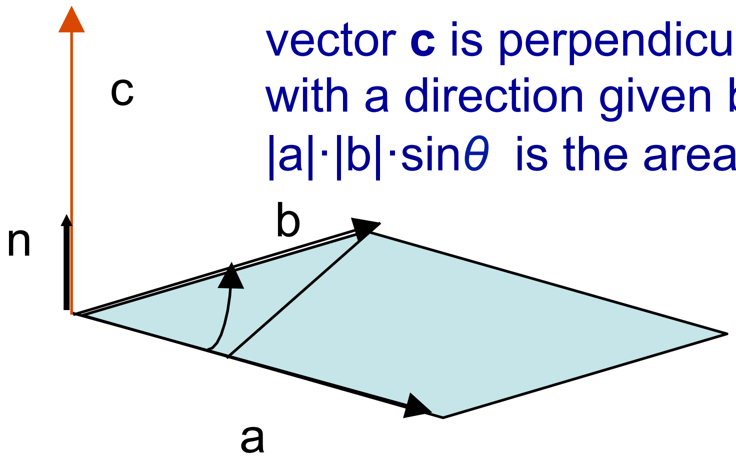
Calculations with Vectors

- *Vector product* or *cross product* of 2 vectors

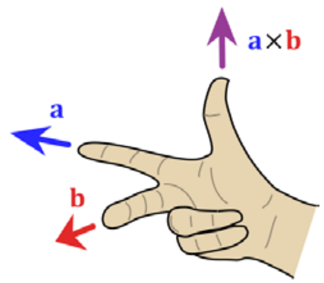
  a=[a_1 a_2 a_3], **b**=[b_1 b_2 b_3]   **a** × **b** = (|a|·|b|·sinθ) **n**

  c=**a**×**b** = [a_2b_3-a_3b_2 a_3b_1-a_1b_3 a_1b_2-a_2b_1]   which is actually by Sarrus' rule:

  c=**a**×**b** = $\det\begin{bmatrix} \boldsymbol{i} & \boldsymbol{j} & \boldsymbol{k} \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{bmatrix} = \boldsymbol{i}\det\begin{bmatrix} a_2 & a_3 \\ b_2 & b_3 \end{bmatrix} - \boldsymbol{j}\det\begin{bmatrix} a_1 & a_3 \\ b_1 & b_3 \end{bmatrix} + \boldsymbol{k}\det\begin{bmatrix} a_1 & a_2 \\ b_1 & b_2 \end{bmatrix}$



vector **c** is perpendicular to both **a** and **b**
with a direction given by the right-hand rule
|a|·|b|·sinθ is the area of the parallelogram



Counter-clockwise rotation from **a** to **b**

Summary

- Visualization requires to define some mathematical model to be rendered into images
- Shapes consist of geometry, colors, image textures, and geometrical textures
- All shape components can be defined in their own coordinate systems and merged together into one object
- Shapes can be further transformed and eventually grouped into one application coordinate system
- Viewer and light sources have to be defined to render the scene
- Vector and matrix algebra is used intensively in computer graphics to make it dimension independent and computationally efficient