Web Based Graphics & Virtual Reality Systems

Introduction

Lecturer

Lecturer: Dr. Pang Wai Man, Raymond

- Email: wmpang@ieee.org

Tutor: Dr. Kin-Chung Kwan

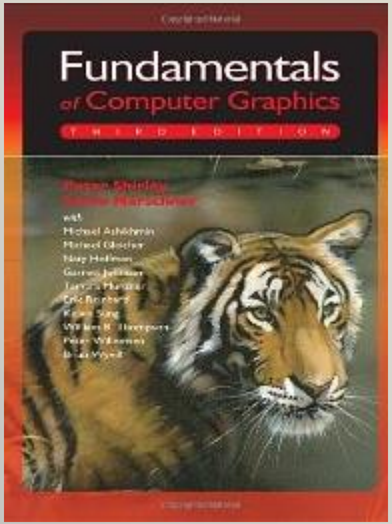
- Email: kckwan@cse.cuhk.edu.hk

Course Web Page:

http://www.cse.cuhk.edu.hk/~cmsc5716

Textbooks

- Fundamentals of Computer Graphics, 3rd edition, Peter Shirley, Steve Marschner, A K Peters, 2009.



- **WebGL: Up and Running:** Building 3D Graphics for the Web, Tony Parisi, O'Reilly Media, 2012

Assessment Scheme

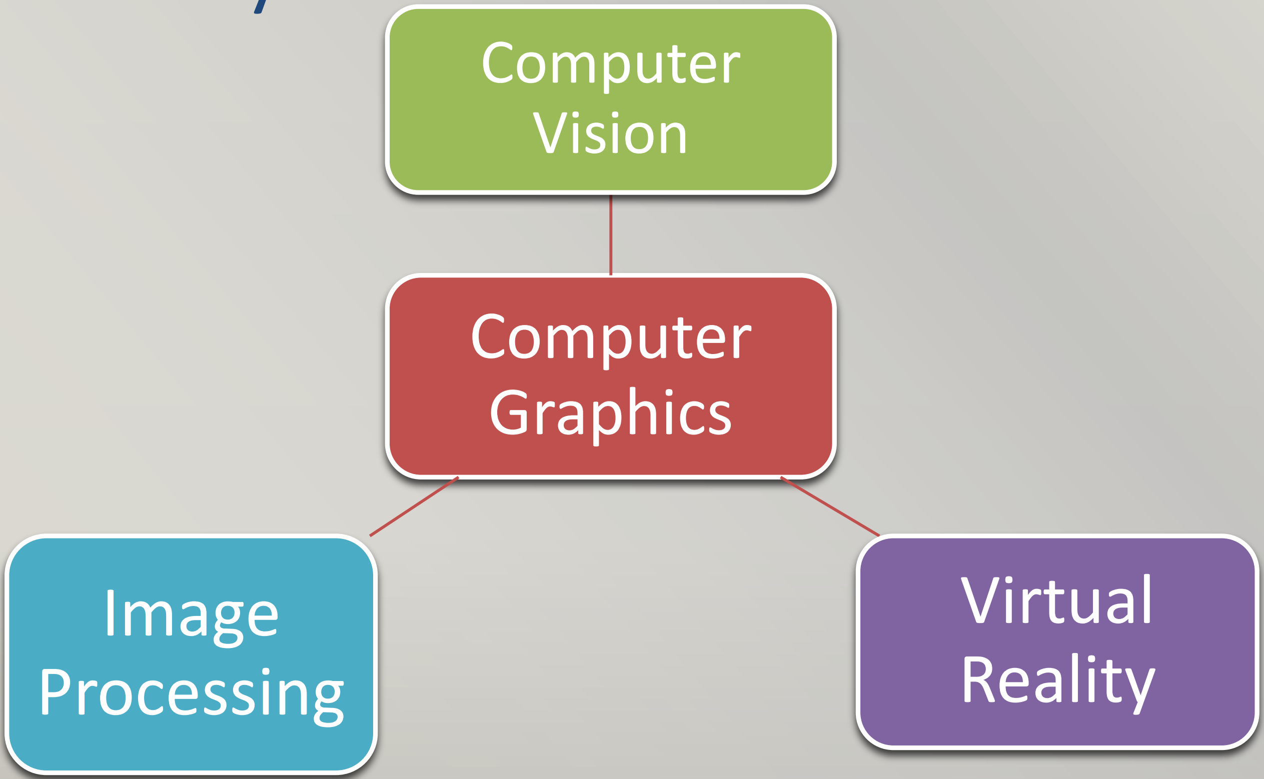
- Continuous Assessments (60%)
  - Written Assignment (15%)
  - Project (45%)
    - Proposal
    - Report
    - Presentation and Prototype
- Final Examination (40%)

Course topics and schedule

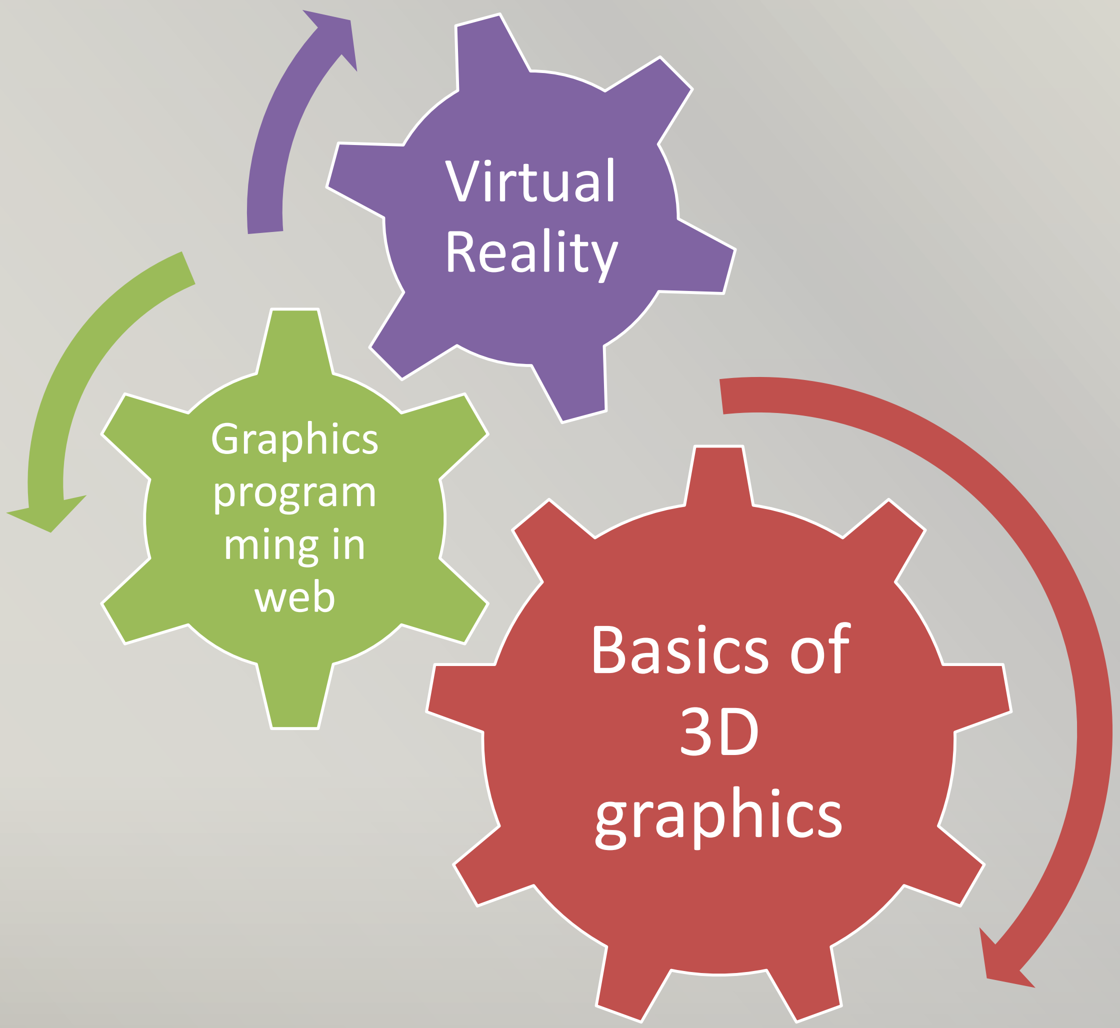
Session	**Date**	***Topics***
1	9Jan	Overview of Computer Graphics, Basics of 2D Graphics
2	16Jan	3D Graphics: Representation, Vector, Matrix, and Geometric Transformation
3	23Jan	Camera and Projection
4	6Feb	Illumination and Texture Mapping
5	13Feb	Transparency, Sampling and Antialiasing
6	20Feb	Rendering Pipeline, Surface Mesh Modeling and Scene Graph
7	27Feb	Virtual Reality
8	6Mar	Simple Animation, Spline Interpolation and Particle System
9	13Mar	Programmable Shaders and Ray-tracing
10	20Mar	Web-based Graphics and Augmented Reality
11	3Apr	Project Presentation and Revision
12	10Apr	Final Exam

What do you expected to learn in this course ?

- Web-based Graphics?
- Virtual Reality?



Major Areas to be included



What do you expect to learn in this course ?

- How do you understand “3D Computer Graphics” ?

Pixar—*Toy Story*

Electronic Arts—*NBA Live 07* (screenshot: gamespy.com)

Engineering Perspective of CG

- This course is **NOT** about graphics design
- This course is **NOT** about particular graphics tools and software
- This course is **NOT** about creating and designing 3D models

Engineering Perspective of CG

- We will try to learn many important **concepts and terms** in 3D graphics
- We are trying to learn **how things work** in 3D graphics
  - **how** 3D objects are **represented** and **drawn** on the screen in computers
- Try to **apply** things learnt to some small scale 3D applications

What is going on in 3D Graphics?

- 2D v.s 3D
- In 2D, we have
  - Image, or
  - vector graphics
- In 3D, what we have are
  - 3D model with material,
  - Lighting, and
  - Camera
  - Rendering process

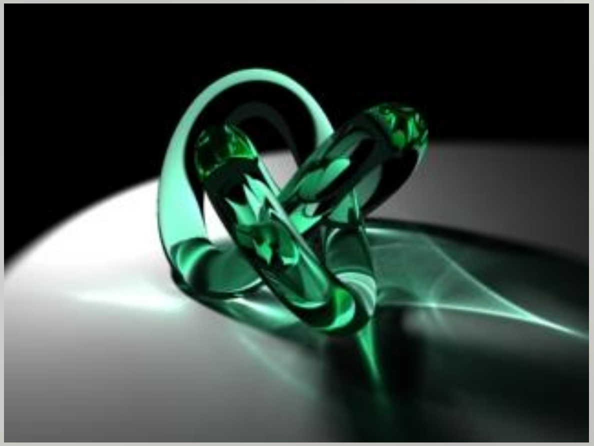
Represent what the real world had

Simulate what happened to the light
In the real world and inside the camera

Real-time v.s. Offline Rendering

- Real-time Rendering
  - Require fast update
    - Around 30 fps
  - Usually use less complex scene / objects and lower screen resolution
  - Z-buffer rendering approach
- Offline Rendering
  - Require high rendering quality
  - Usually involve more complex lighting effects and higher screen resolution
  - Ray-tracing rendering approach

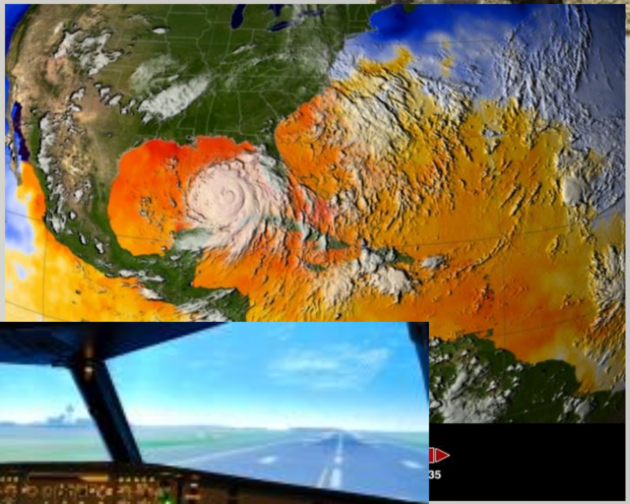
Sega-Virtual Racing (1992)


Caustics 101 by Keith Sereby (2003)

Application of R.T. Rendering Techniques

- Game
- Various kind of Visualization
  - Medical
  - Scientific
  - Engineering
- Training and Simulation

Call of Duty



Airbus A320 flight simulator

Chinese Visible Human (CUHK)

Applications of Offline Rendering

- Movie
- Advertisement

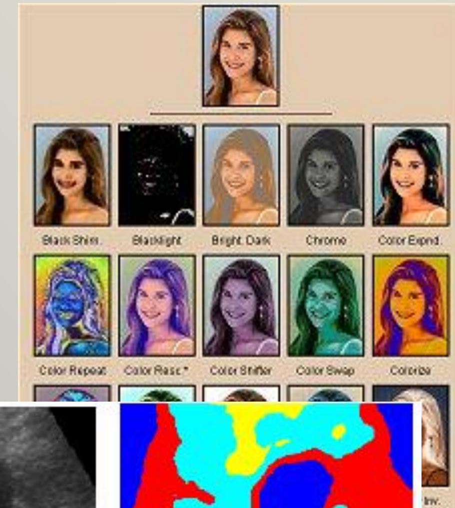
Pixar—*Ratatouille (2007)*

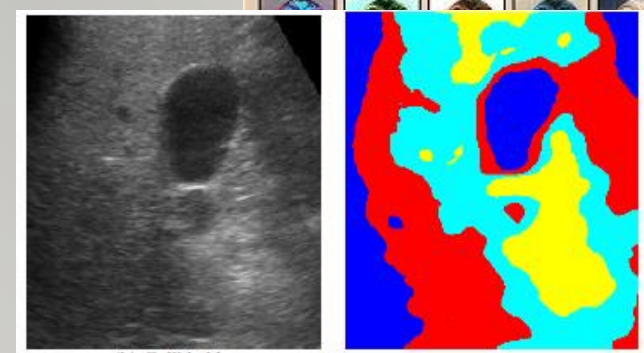
TOYOTA MARK X

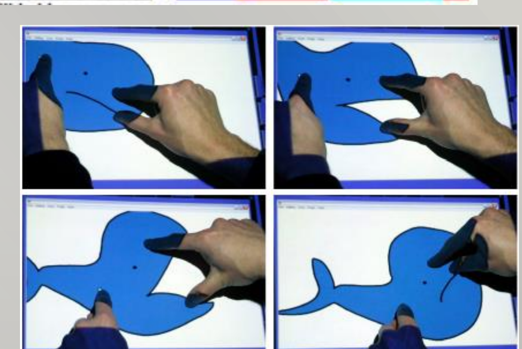
King Kong (Universal Pictures, 2005)—visual effects: WETA Digital

Areas related to Graphics

- 2D Imaging
  - Digital filtering and effects
  - Labeling and segmentation
  - Color transformation
- 2D Drawing
  - Vector graphics manipulation
  - Illustration and drafting tools







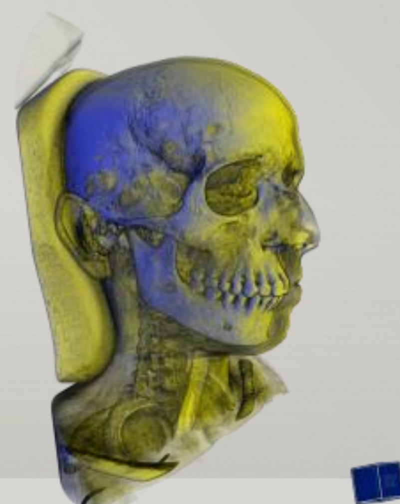
Igarashi et.al. 2005

Areas related to Graphics

- 3D Modeling
  - Representing 3D shapes with curved surfaces, polygons and etc.
  - 3D Reconstruction
  - Manipulating 3D shapes
  - Volume representation

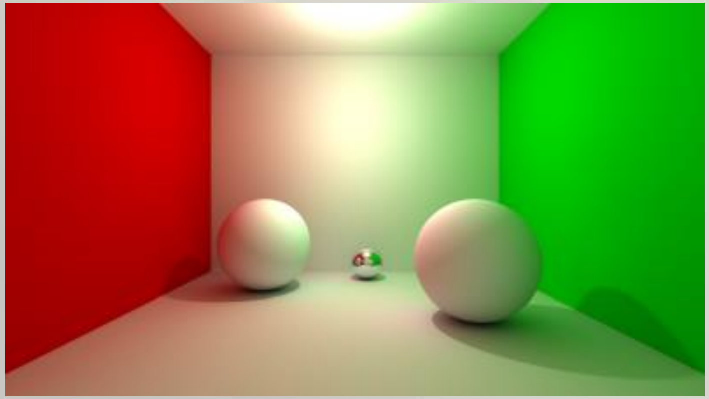
Mario Botsch et. al. 2006

RealView 3D



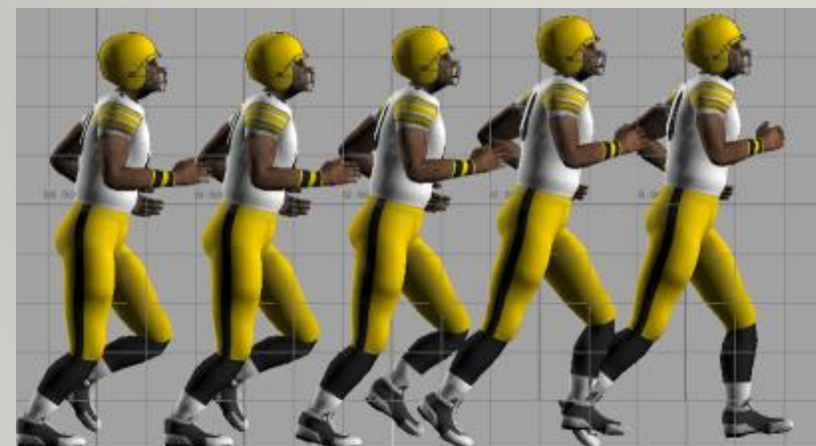
Areas related to Graphics

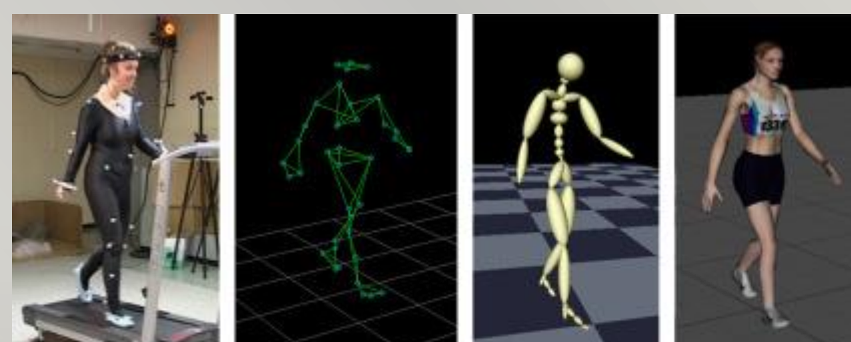
- 3D Rendering
  - Global illumination
  - Lighting simulation in complex material
  - Volume rendering
  - Toon-shading



Areas related to Graphics

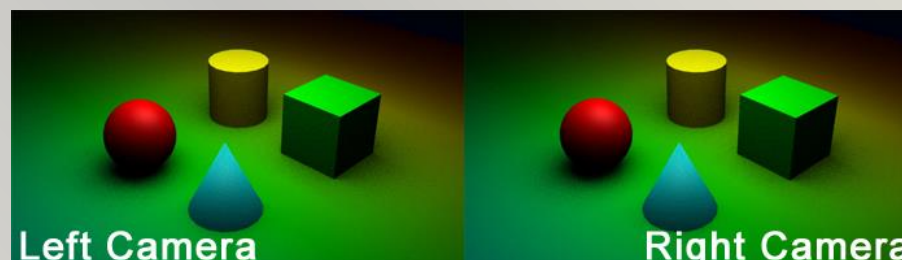
- Animation
  - Keyframe animation
  - Motion capture
  - Physical simulation





Areas related to Graphics

- Virtual Reality
  - Stereoscopy
    - Head Mount Display (HMD)
    - Eyewear



  - CAVE
  - Other VR devices
    - Cybergloves, VR hand controller

Areas related to Graphics

- Web-based Graphics
  - Plugin to browser
    - Flash, Silverlight, Java2D, SVG
    - Java3D, Unity
  - WebGL / WebVR
    - HTML5 Canvas
- Mobile Graphics
  - OpenGL ES

What Skills You will learn?

- Mathematics
- Geometry
- Physical simulation
- Virtual Reality
- Skills in developing 3D applications in web environment

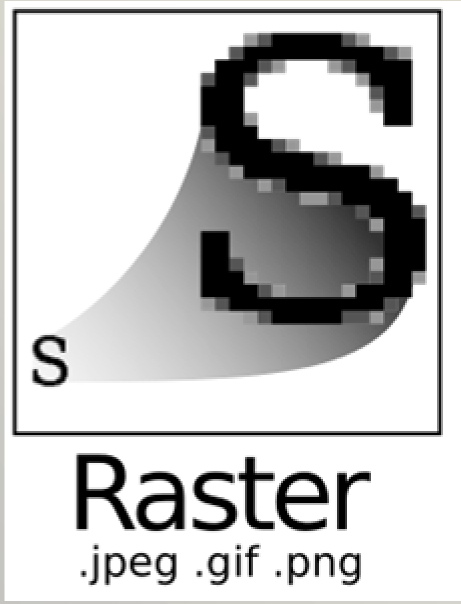
Basics of 2D Graphics

2D before 3D

- Although everything in real world is 3D, when we have to display something, everything reduce to a 2D image / screen space
- Drawing in 2D is therefore a basic needs
- Some concepts and maths are easier to understand in 2D before going into 3D

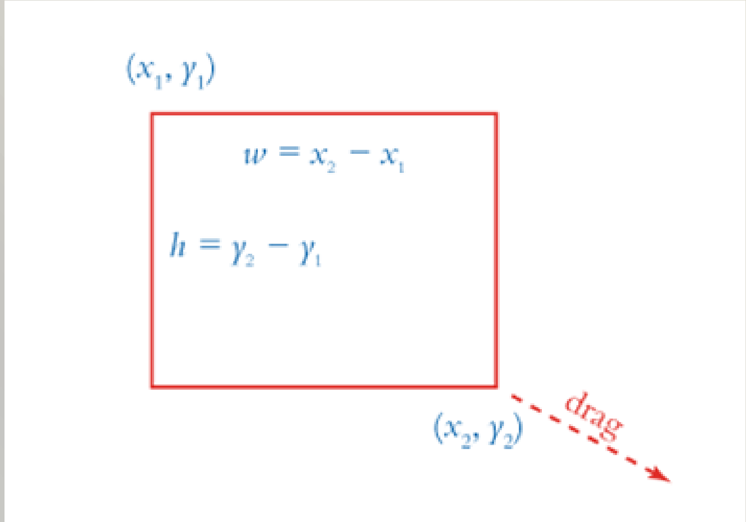
2D Graphics on the Web

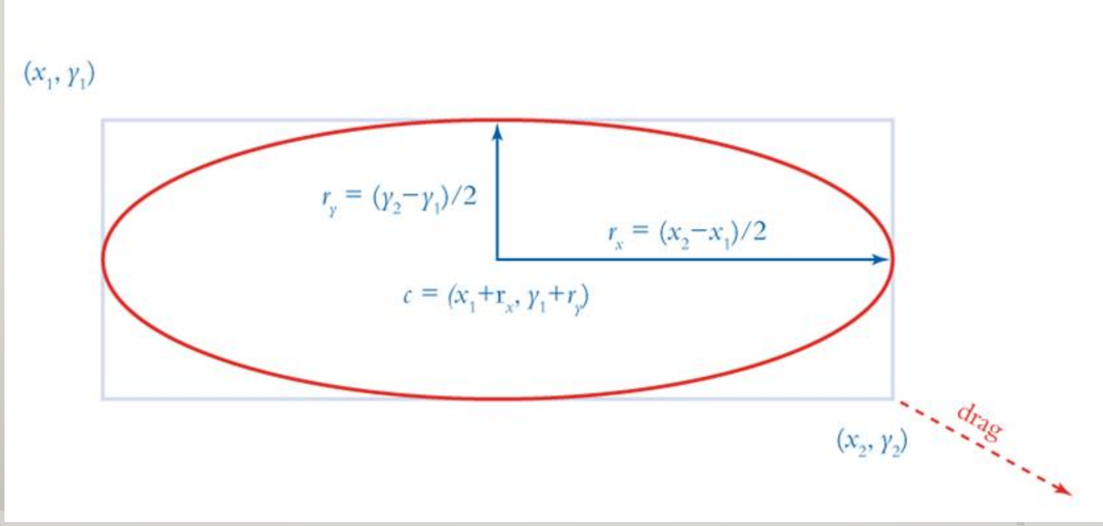
- Raster graphics
  - Or images (e.g. jpeg, gif, png)
  - Set of pixels ordered in rectangular / matrix
- Vector graphics
  - E.g. Flash, Sliverlight, SVG
  - Defined with vertices, line, curve and shape



Vector graphics

- Lines / curves
- Predefined shapes
  - Rectangle or quadrilateral
  - Circle or ellipse
- Scale by changing corner's coordinates





Vector graphics: lines

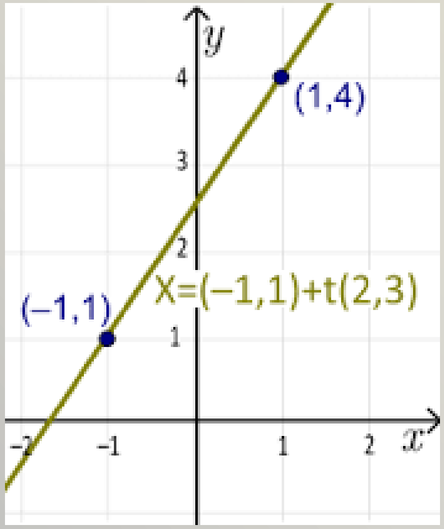
- Represent line with equation like

  y = ax +b

- An alternative will be parametric form

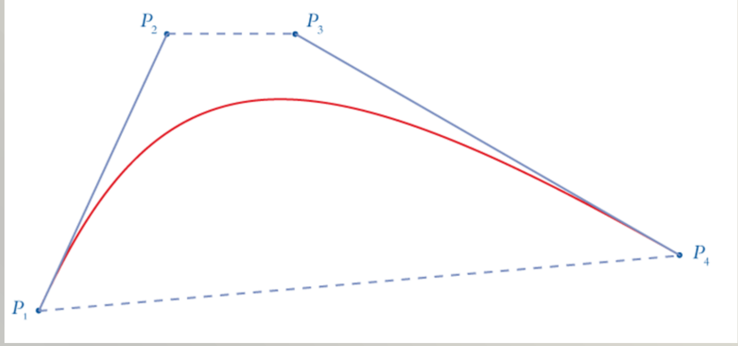
  (x,y) = (c,d) +t (e, f)

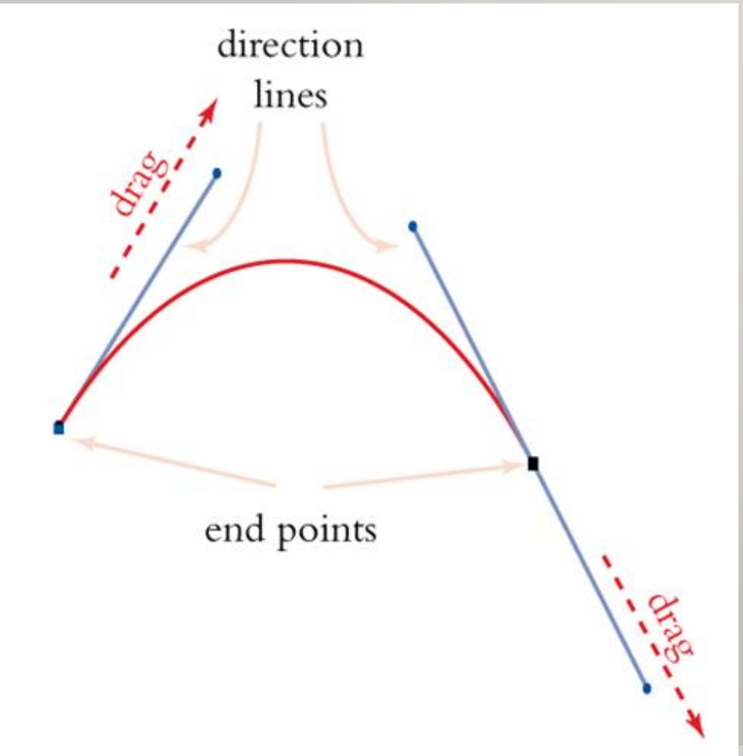
  A parameter t is used. It is more easy to get a point on the line by filling with any value of t



Vector graphics: curves

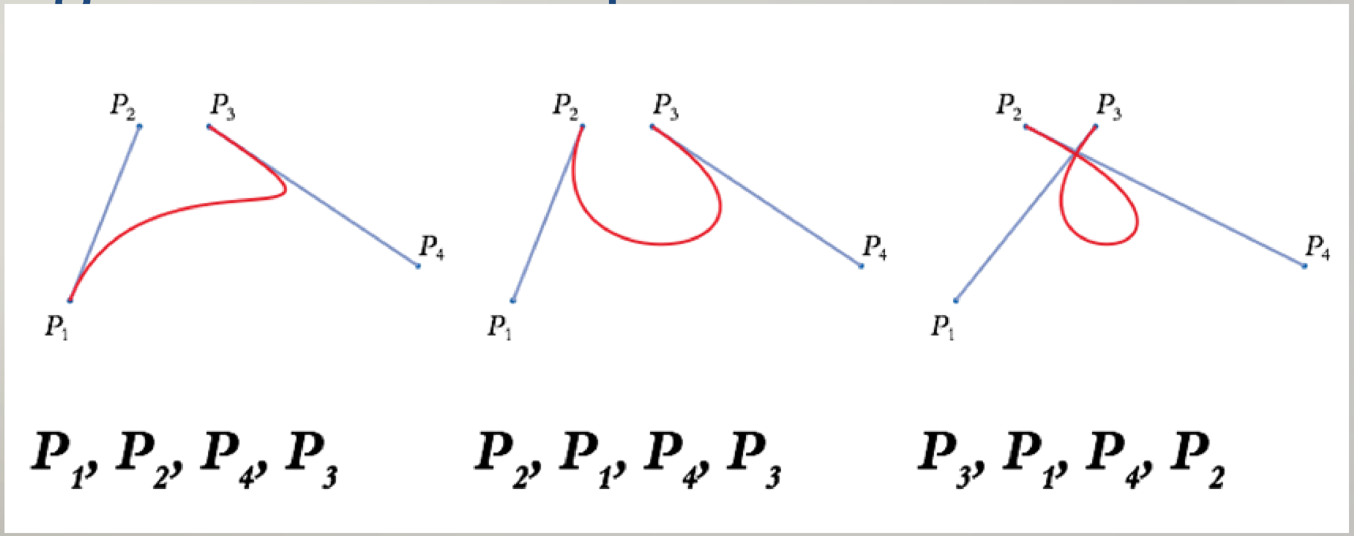
- Beizer curve is one of the most commonly used representation
  - Flash (SWF)
  - PDF
  - SVG
- Defined by control points
  - or by End points & Direction lines





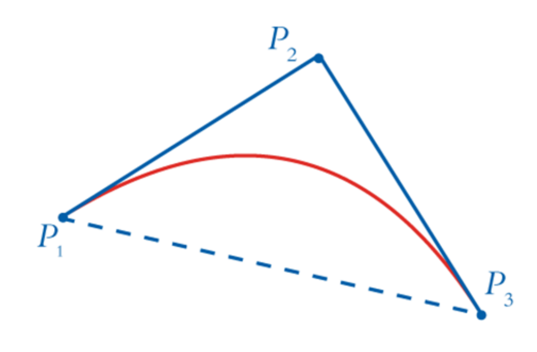
Vector graphics: curves

- The control points help to define the actual curve
- Only the 2 end points are on the curve
- The other 2 control points define the tangent at the end points

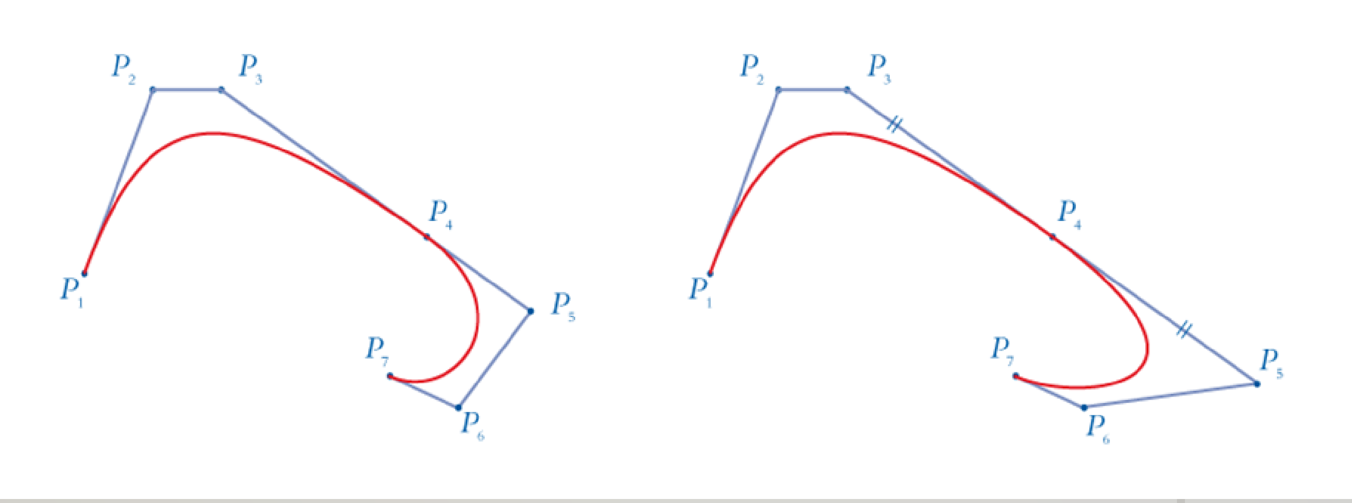


Vector graphics: curves

- Degree of freedom
  - Cubic (PDF and SVG)
  - Quadratic (SWF, PDF and SVG)
- Spline
  - Connecting multiple cubic or quadratic curves

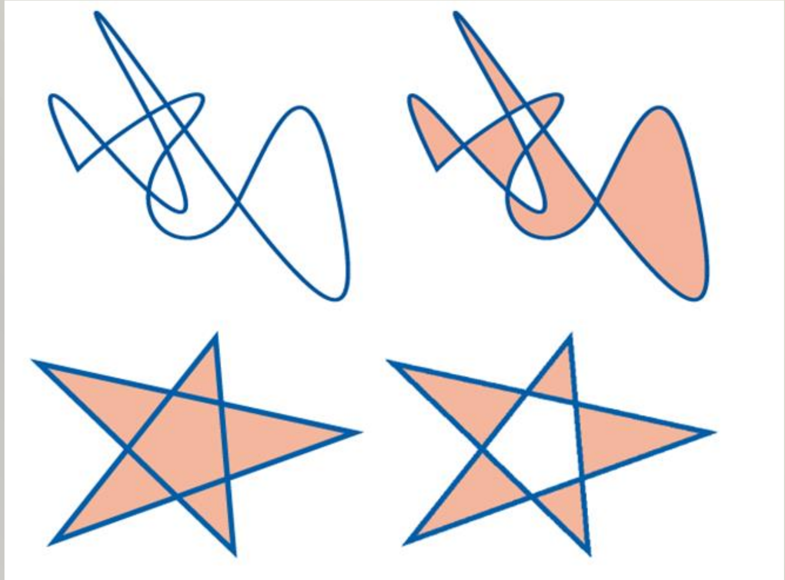


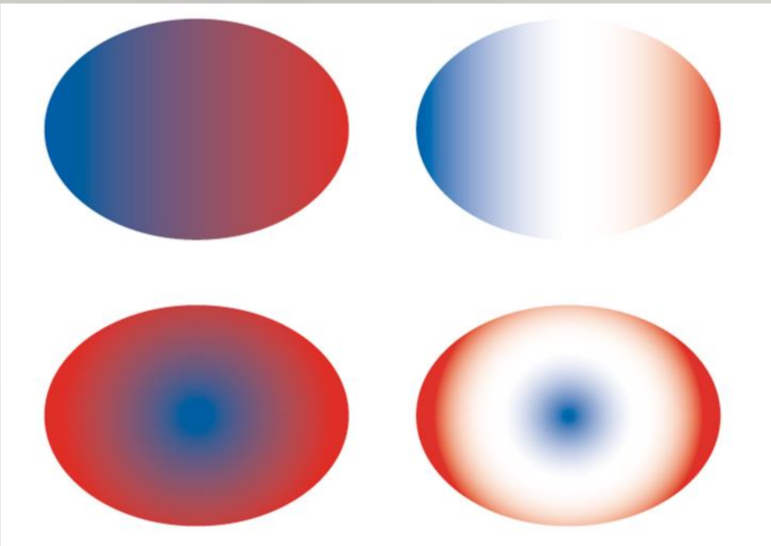
A quadratic Bézier curve



Color filling

- Define color within <u>closed</u> curves
- Simple single color
- Gradient fills
  - Color change from one to another (Interpolation)
  - Linear
  - Radial

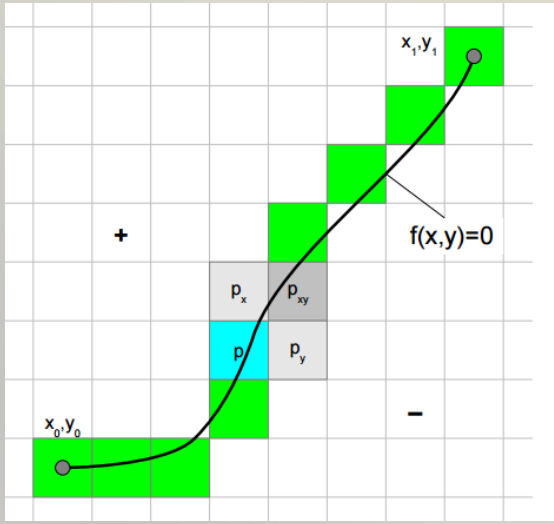


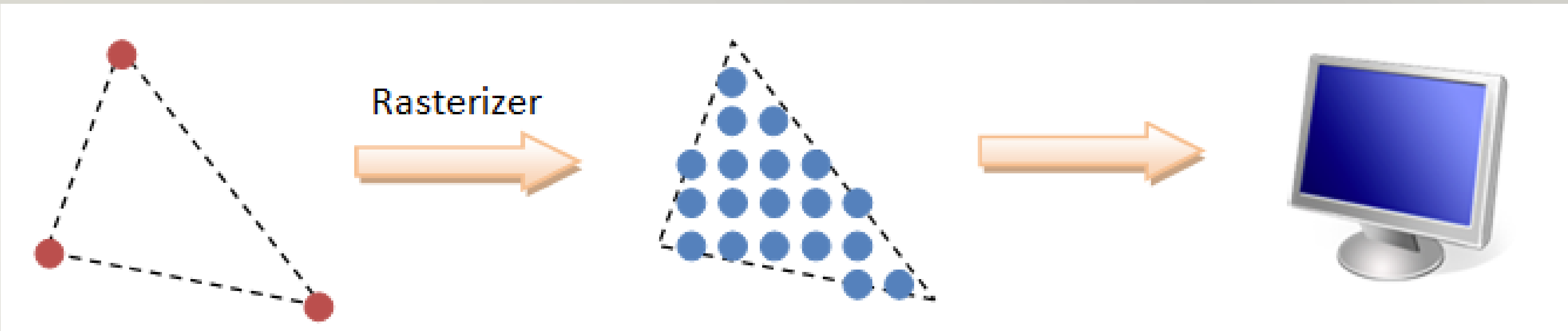


Linear (top) and radial (bottom) gradient fills

Rasterization

- All monitors are raster display
- Convert vector graphics to array of pixels (raster image)





SVG (Scalable Vector Graphics)

- Defined in XML format
- Shape, dimension, color and etc.
- Example

```
<?xml version="1.0"?>
<svg width="400" height="110">
  <rect width="300" height="100"
style="fill:rgb(0,0,255);stroke-width:3;stroke:rgb(0,0,0)" />
</svg>
```

Pros and Cons of vector graphics

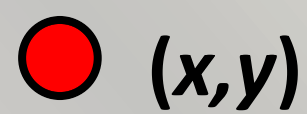
- Pros
  - Usually define in a precise way, smaller in size to raster image counterpart
  - Resolution independent
- Cons
  - Require rasterization before display
  - Support only simple color filling, texture is usually not included

Common Basic Shapes in 2D

- Point / Vertex
- Line
- Triangle
- Circle

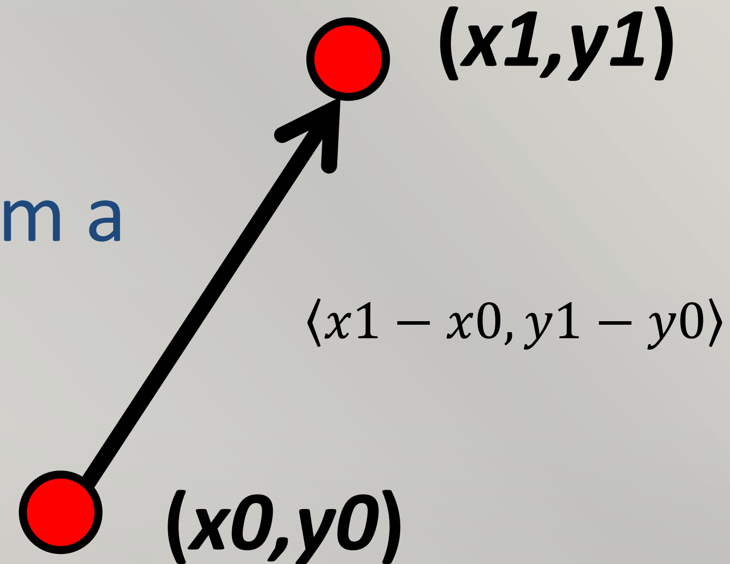
Vertex and vector

- In 2D, the most basic object is a point (or vertex)
- We can always represent a vertex with its coordinate **(x,y)**
- For a 3D point, we will have
  - *(x,y,z)*



Vertex and vector

- A vector can be formed between 2 vertices
- A vector is directional (from a vertex to another)
- To form a vector *v* from (*x0, y0*) to (*x1,y1*), we will have
  - $V = \langle v_x, v_y \rangle = \langle x1 - x0, y1 - y0 \rangle$
- So, every vertex (x,y) can form a vector $\langle x, y \rangle$ which originate from the origin (0,0)



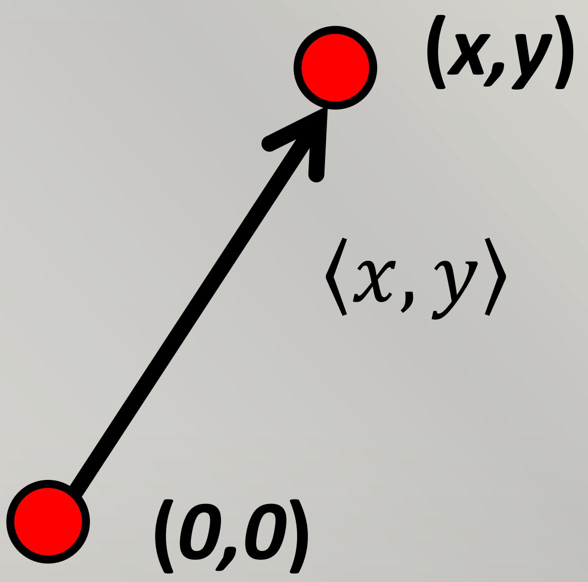
Vertex and vector

- So, every vertex (x,y) can form a vector $\langle x, y \rangle$ which originates from the origin (0,0)

$\langle x - 0, y - 0 \rangle$

= $\langle x, y \rangle$

- Here, we see that a vertex coordinate and vector are sometimes interchangeable



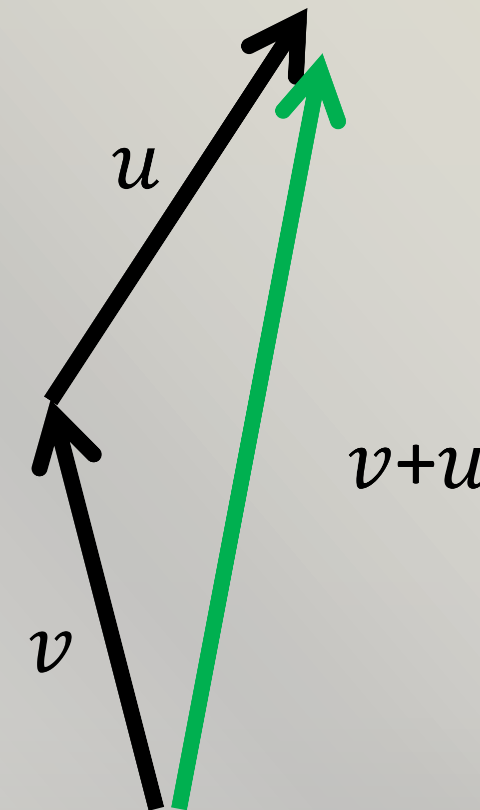
Basic Vector Arithmetic

- Addition
  - Adding two vector will create another new vector
- E.g. a vector v and a vector u

$$v+u$$

$$= \langle v_x, v_y \rangle + \langle u_x, u_y \rangle$$

$$= \langle v_x+u_x, v_y+u_y \rangle$$



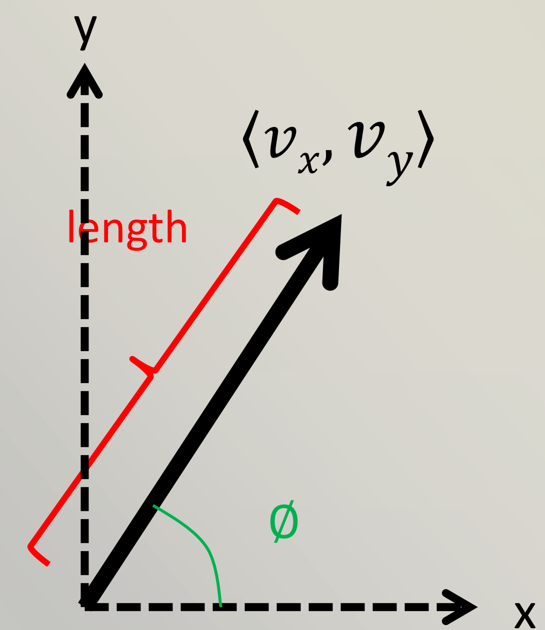
Basic Vector Arithmetic

- Length of vector v

$$|v| = \sqrt{(v_x \times v_x) + (v_y \times v_y)}$$

- Angle to x-axis,Ø

$$\frac{v_x}{v_y} = \tan(\emptyset)\ ,\ \emptyset = \arctan\left(\frac{v_x}{v_y}\right)$$



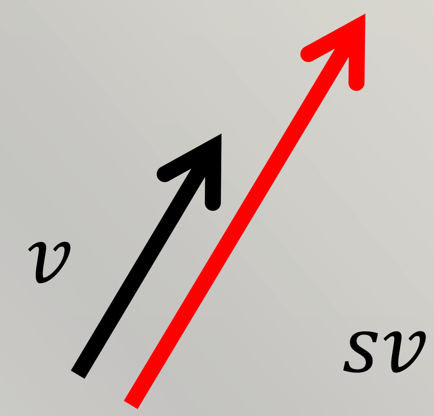
Basic Vector Arithmetic

- Scalar Multiplication
  - Multiply a scalar value (simple single value), it lengthen or shorten the original vector
- E.g. *s* = 3 , *v* = <4, 3>

sv = 3 x < 4, 3> = <12, 9>

Length of *v = 5*

Length of *sv = 15*

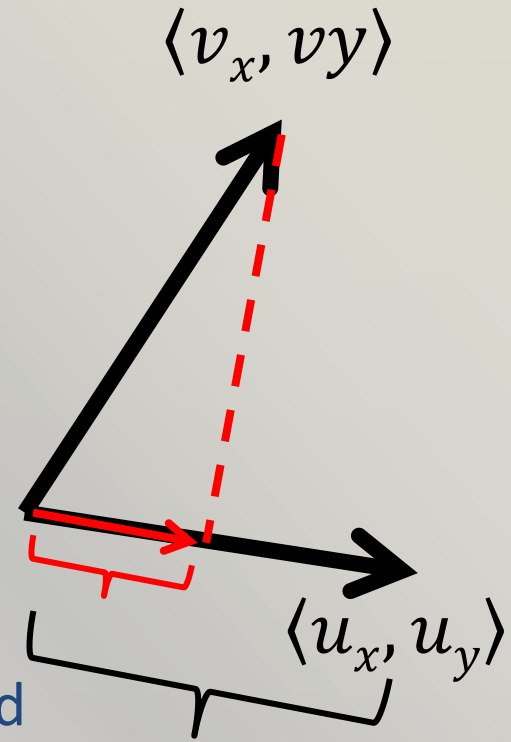


Basic Vector Arithmetic

- The dot product (·)

  $$v \cdot u = (v_x \times u_x)+(v_y \times u_y)$$

  - Notice the result is a scalar (single value) but not vector
  - One physical meaning of dot product is the length of the *projection* of v onto u multiplied by the length of u

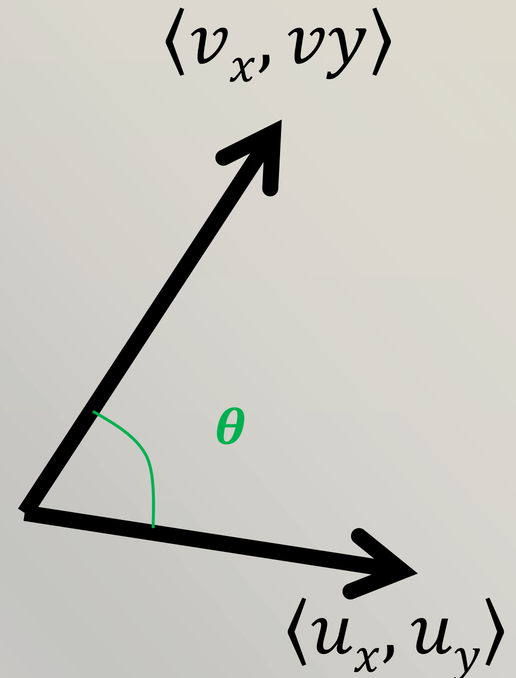


Basic Vector Arithmetic

- Angle between two vectors

$$\theta = arccos(\frac{v \cdot u}{|v| \times |u|})$$

  - arccos is the inverse of cosine
  - Notice that this angle will always be positive and being the smaller angle between the vectors

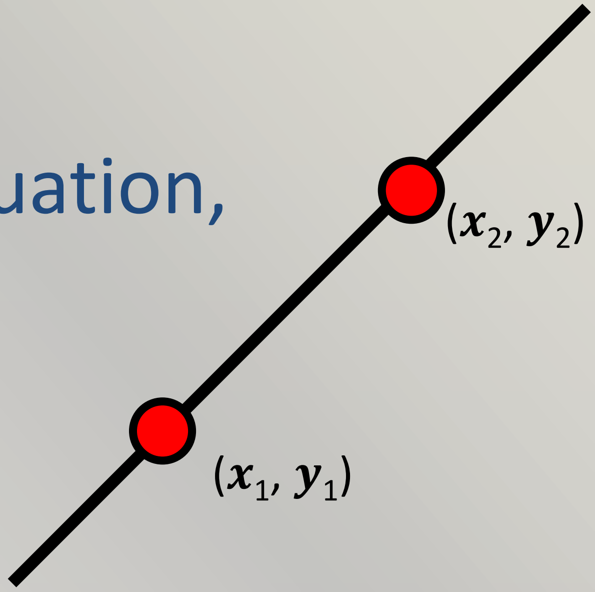


Representing a Line

- The implicit form of representing a line

  y = ax + b

- A point ($\boldsymbol{x}$, $\boldsymbol{y}$) if fulfilling this equation, this point is lying on this line



- Using 2 points, we can from the equation as :

- $$\frac{(y - y_2)}{(x - x_2)} = \frac{(y_1 - y_2)}{(x_1 - x_2)}$$

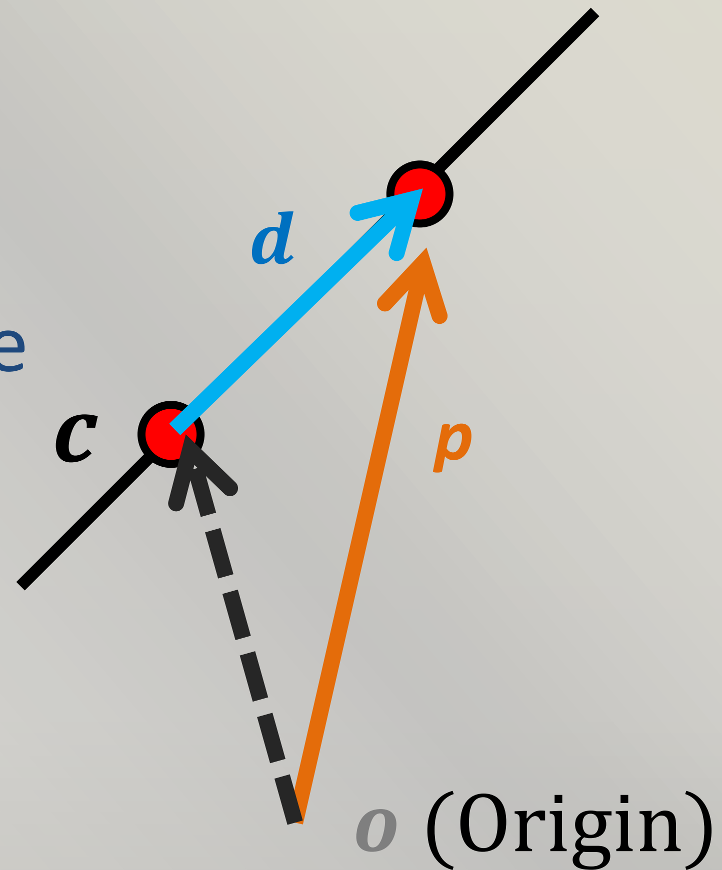
Representing a Line

- In vector form, we can use

  $$\boldsymbol{p} = \boldsymbol{c} + t\,\boldsymbol{d}$$

  - notice t is a scalar

- So, vertex $\boldsymbol{p}$ is always on the line



Determining which side a point is on

- It is simple to use the implicit form to check if a point is on which side of a line

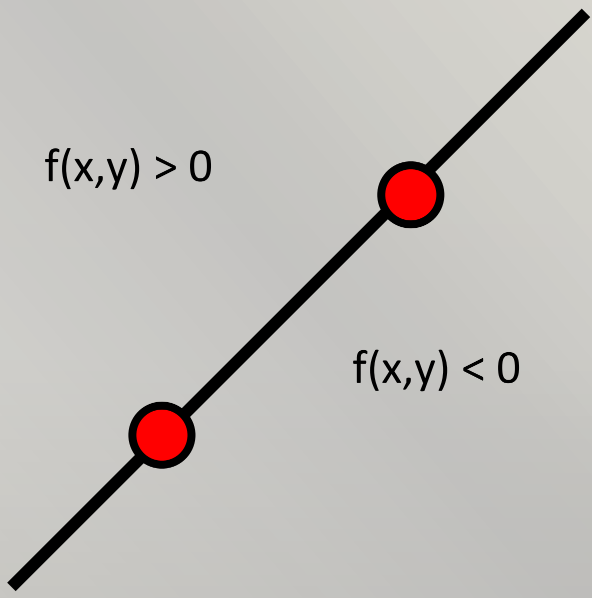
Given a line : $y = ax + b$, we can rewrite as a function $f(x,y) = y - ax - b$

Then,

If $f(x,y) = 0$, it is on the line

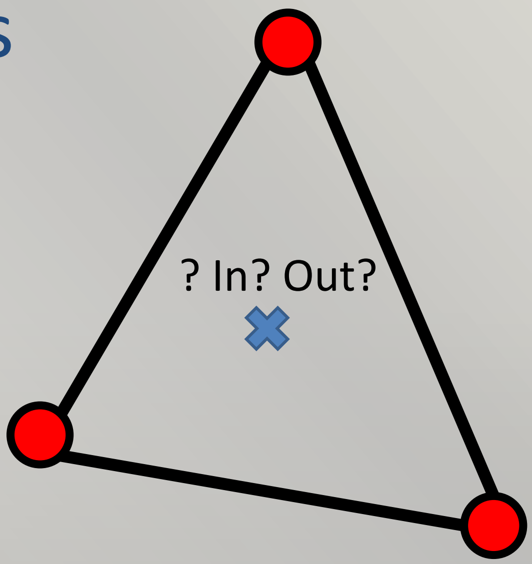
If $f(x,y) > 0$, it is on left of the line

If $f(x,y) < 0$, it is on right of the line



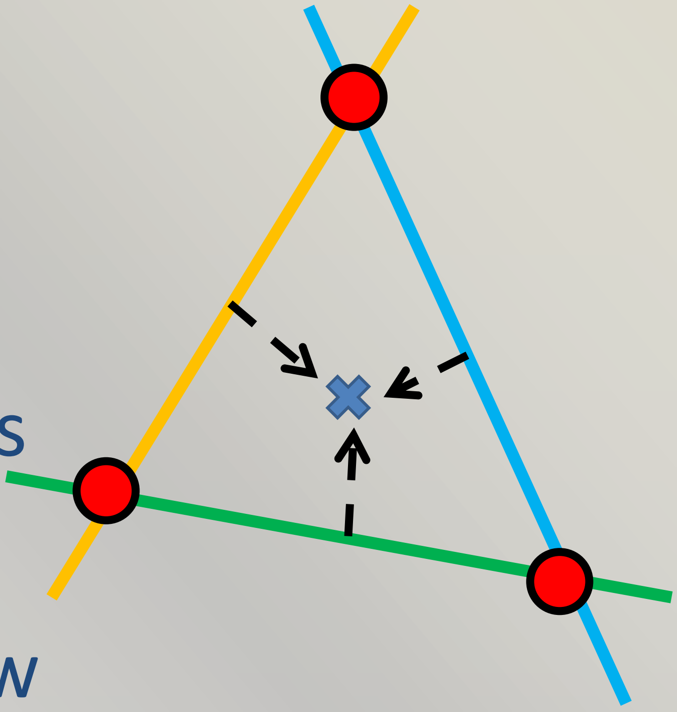
Representing a Triangle

- Triangle is the most basic unit to represent shapes in CG
- It is usually defined by its 3 vertices
  - They can be ordered in clockwise or anticlockwise
- For shapes, our common concerns is to know the region inside or outside the triangle
  - Whether an arbitrary point is inside?



Determining point inside a triangle

- There are many methods to do
- One way to think about the problem is that a triangle is formed by interception of 3 lines
- So, if a point are on the correct side of all 3 lines, then, we know it is inside the triangle

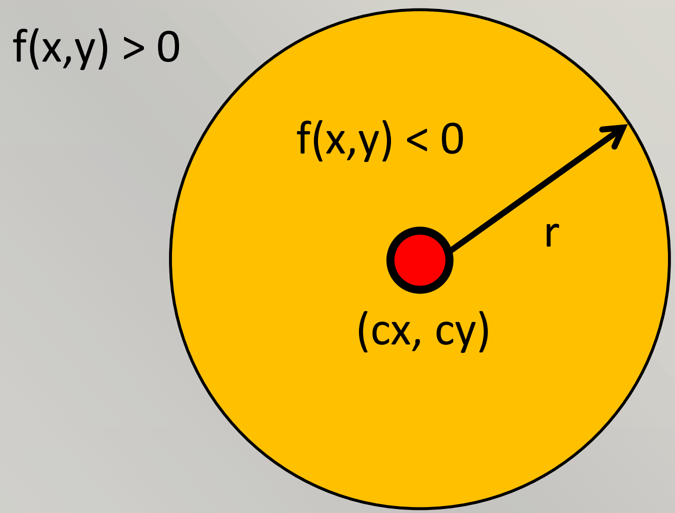


Representing a Circle

- Circle is one of the most commonly used shape in Graphics, as its representation is simple
  - Center (cx, cy)
  - Radius r
- The formula:

  $(x-cx)^2 + (y-cy)^2 = r^2$

- $f(x,y) = (x-cx)^2 + (y-cy)^2 - r^2$
- If $f(x,y) > 0$, outside of the circle
- If $f(x,y) < 0$, inside of the circle



Transformation in 2D

- Common operations on 2D shapes include
  - Scaling
  - Rotation
  - Translation

Using Matrix in Transformation

- In general, a matrix is a rectangular array of elements (usually numbers or values)
  - It can be any size

$$A = \begin{bmatrix} a_0 & a_{-1} & a_{-2} & \cdots & \cdots & a_{-n+1} \\ a_1 & a_0 & a_{-1} & \ddots & & \vdots \\ a_2 & a_1 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{-1} & a_{-2} \\ \vdots & & \ddots & a_1 & a_0 & a_{-1} \\ a_{m-1} & \cdots & \cdots & a_2 & a_1 & a_0 \end{bmatrix}$$

- Vector in Matrix Form
  - As a special case that a 2D vector <*x,y*> is put into a matrix of 2 x 1 like

$$\begin{bmatrix} x \\ y \end{bmatrix}$$

Common Matrix Operations in Graphics

- Multiplication
  - **A * *v***

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a_{11}x + a_{12}y \\ a_{21}x + a_{22}y \end{bmatrix}$$

Common Matrix Operations in Graphics

- Multiplication the general form

$$\begin{bmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & & \vdots \\ \boxed{a_{i1} \quad \cdots \quad a_{im}} \\ \vdots & & \vdots \\ a_{r1} & \cdots & a_{rm} \end{bmatrix} \begin{bmatrix} b_{11} & \cdots & \boxed{\begin{matrix} b_{1j} \\ \vdots \\ b_{mj} \end{matrix}} & \cdots & b_{1c} \\ \vdots & & & & \vdots \\ b_{m1} & \cdots & & \cdots & b_{mc} \end{bmatrix} = \begin{bmatrix} p_{11} & \cdots & p_{1j} & \cdots & p_{1c} \\ \vdots & & \vdots & & \vdots \\ p_{i1} & \cdots & \boxed{p_{ij}} & \cdots & p_{ic} \\ \vdots & & \vdots & & \vdots \\ p_{r1} & \cdots & p_{rj} & \cdots & p_{rc} \end{bmatrix}$$

$$p_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{im}b_{mj}.$$

- Another example

$$\begin{bmatrix} 0 & 1 \\ 2 & 3 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} 6 & 7 & 8 & 9 \\ 0 & 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 12 & 17 & 22 & 27 \\ 24 & 33 & 42 & 51 \end{bmatrix}$$

Common Matrix Operations in Graphics

- Identity Matrix

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

  - A special matrix which have no effect in matrix multiplication

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} 3 \\ 4 \end{pmatrix}$$

- Transpose
  - Change row to column, column to row

    E.g. here matrix M^T is the transpose of matrix M

$$M = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \xrightarrow{\text{Transpose}} M^T = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}$$

2D Scaling

- Scale Matrix, sx and sy are the scaling factor in x and y directions

$$\text{scale}(s_x, s_y) = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

- When multiplying with a vertex

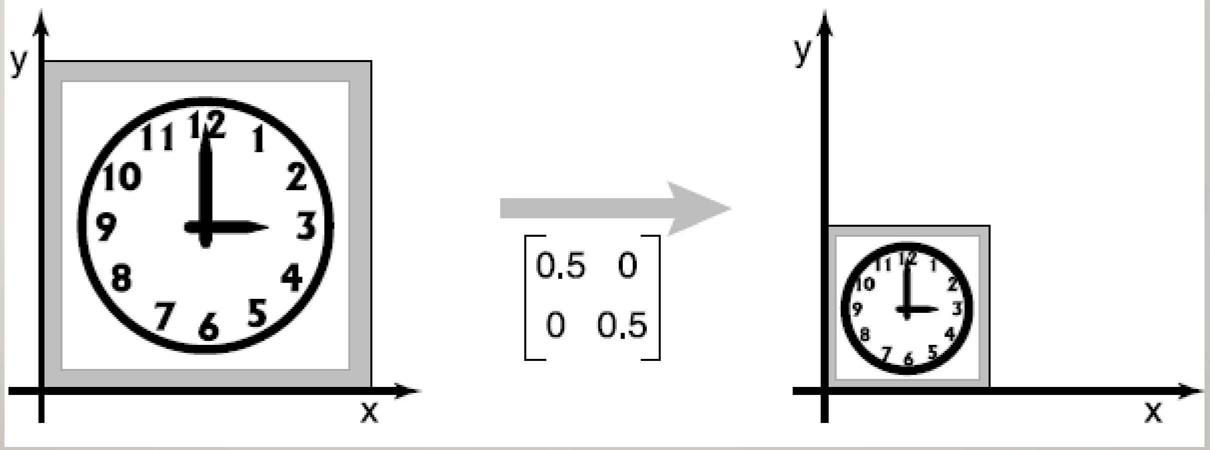
$$\begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} s_x x \\ s_y y \end{bmatrix}$$

2D Scaling

- An example to shrink half the size:

$$\text{scale}(0.5, 0.5) = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix}$$

- So, every vertices are multiplying this matrix



2D Rotation

- 2D Rotation commonly involves
  - Reference point
  - Angle of rotation
- Usually, it is easiest to use the **origin (0,0)** as the reference point
- The related equations:

$$x_b = x_a \cos\phi - y_a \sin\phi,$$

$$y_b = y_a \cos\phi + x_a \sin\phi.$$

anti-clockwise

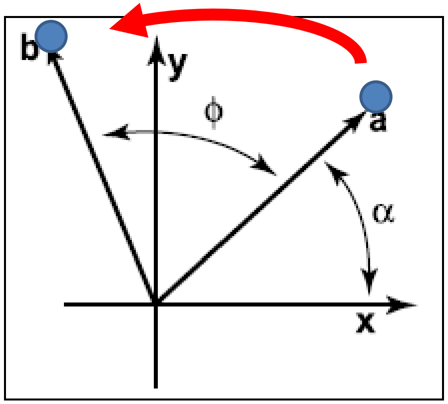


Figure 6.5. The geometry for Equation (6.1).

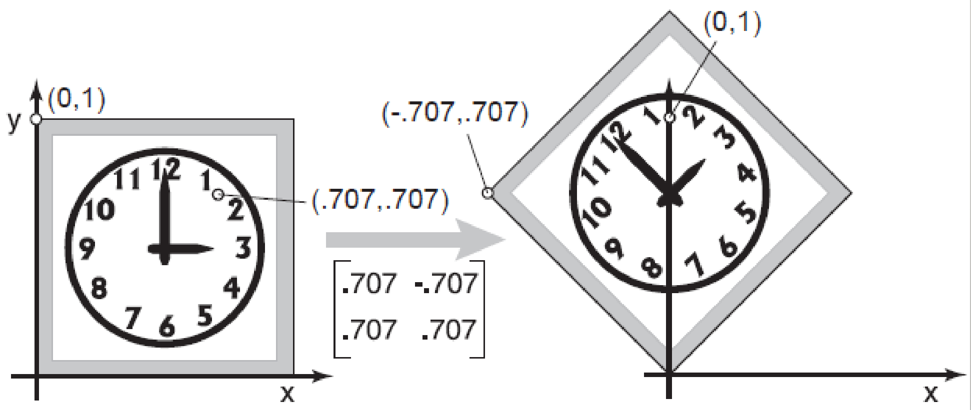
2D Rotation

- So, the Rotation Matrix

$$\text{rotate}(\phi) = \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix}$$

- A rotation with 45 degrees (PI/4) in anticlockwise

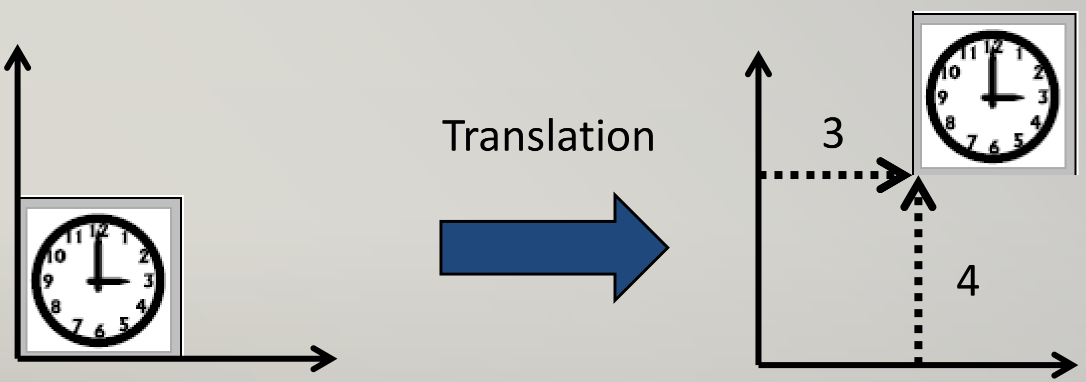
$$\begin{bmatrix} \cos\frac{\pi}{4} & -\sin\frac{\pi}{4} \\ \sin\frac{\pi}{4} & \cos\frac{\pi}{4} \end{bmatrix} = \begin{bmatrix} 0.707 & -0.707 \\ 0.707 & 0.707 \end{bmatrix}$$



2D Translation

- Move horizontally (in x axis) or vertically (in y axis)
  - E.g. move 3 units to right in x, 4 units up in y

$$\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} 3 \\ 4 \end{pmatrix} = \begin{pmatrix} x+3 \\ y+4 \end{pmatrix}$$



Homogeneous Coordinate

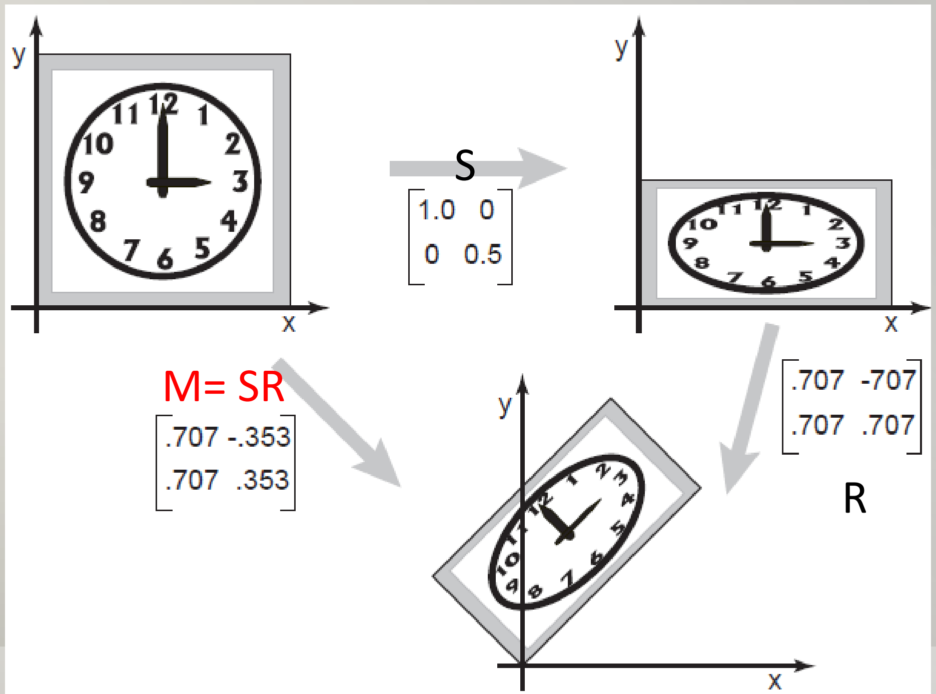
- If only 2x2 matrix is used, it is not possible to represent translation with a Matrix multiplication similar to what rotation and scaling does
- To let also translation to be done using only Matrix multiplication, we will talk about Homogeneous coordinate in the next lesson

Multiple 2D Transformations

- How about the case we want to do more than one transformation the same time?
- For example,
  - First, shrink in Y direction for 0.5 (Matrix S)
  - Then, rotate 45 degree in anticlockwise (Matrix R)
- If we represent in Matrix form, the above complex operation can also be represented in a matrix M

Multiple 2D Transformations

- This Matrix M is formed by multiplying Matrices S and R, because
  - Applying transform matrices in sequence is the same as applying the product of those matrices

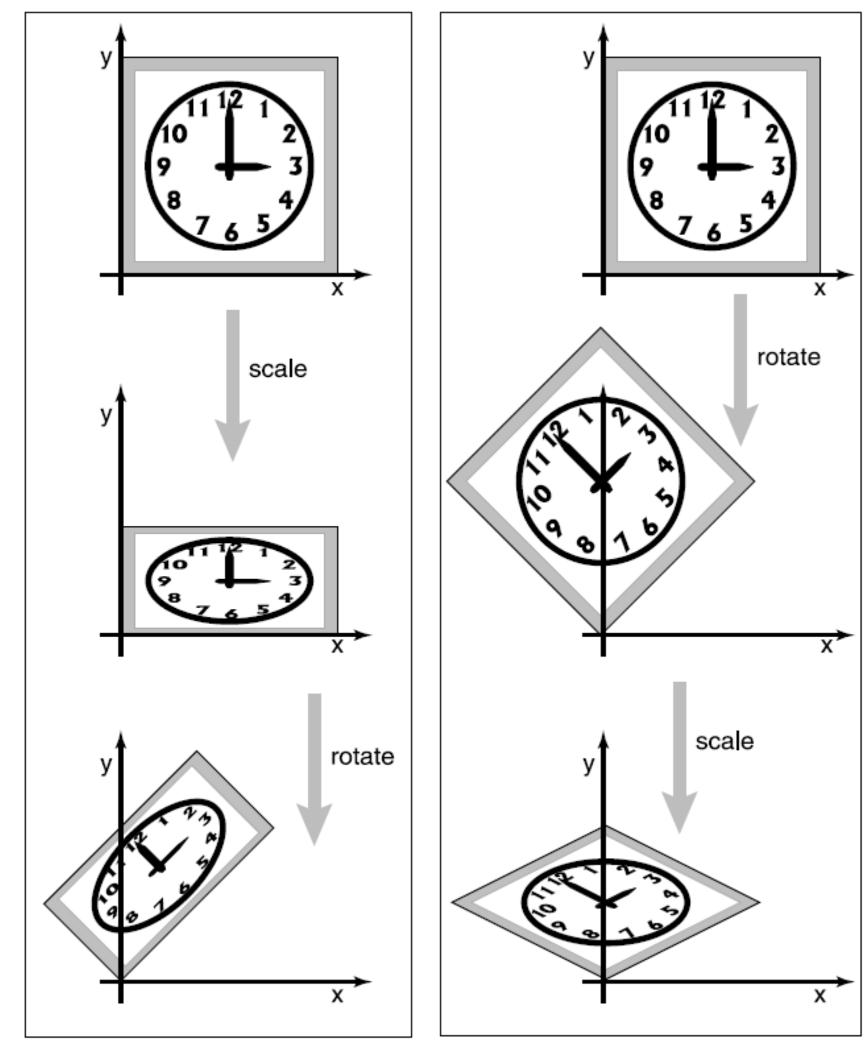


Order of Applying Transformations

- Another question, does
  M = SR = RS ?
- The order of multiplication a matter?
- From the example on right, we know that

  SR ≠ RS

- So be-careful, Order **DOES** Matter!!



Order of Applying Transformations

- We can also show the related Matrices formed:
  - RS = M1

$$\begin{bmatrix} 0.707 & -0.707 \\ 0.707 & 0.707 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.707 & -0.353 \\ 0.707 & 0.353 \end{bmatrix}$$

  - SR = M2

They are different!!!

$$\begin{bmatrix} 1 & 0 \\ 0 & 0.5 \end{bmatrix} \begin{bmatrix} 0.707 & -0.707 \\ 0.707 & 0.707 \end{bmatrix} = \begin{bmatrix} 0.707 & -0.707 \\ 0.353 & 0.353 \end{bmatrix}$$

Does Matrix really necessary?

- The answer is Yes or No
- No: You can do computation in sets of equations
- Yes: It sometimes make things more clear and easy
- Yes: We can think any kind of transformation (including S,R,T or etc) to be represented with matrix and performed by a multiplication in matrix
- Yes: It is already a standard and common language in Graphics

Summary

- An overview of computer graphics topics
- Basic 2D geometry: Vertex, Line, Triangle and etc.
- Mathematics of Vector, Matrix using 2D examples
- 2D Transformations: scaling, rotation and translation
- Relation between transformation and matrix multiplication