Virtual Worlds

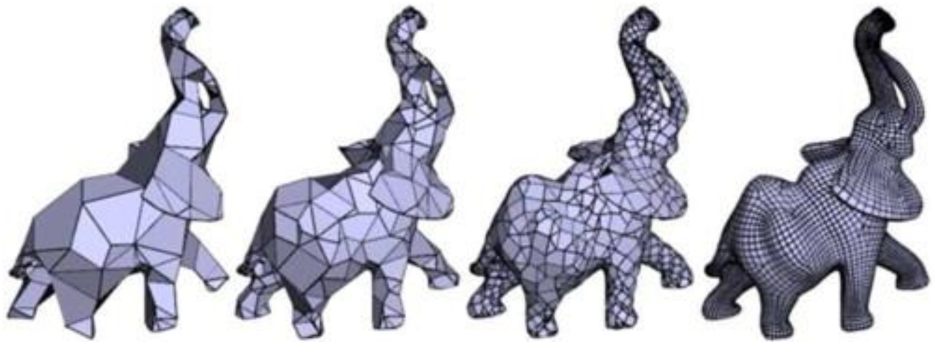
Lecture 02. Modeling

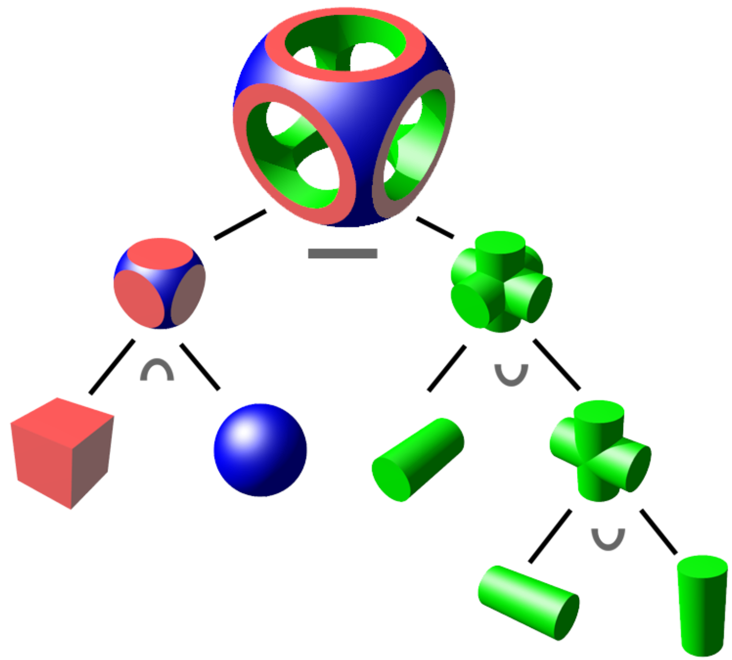
Kang Hoon Lee

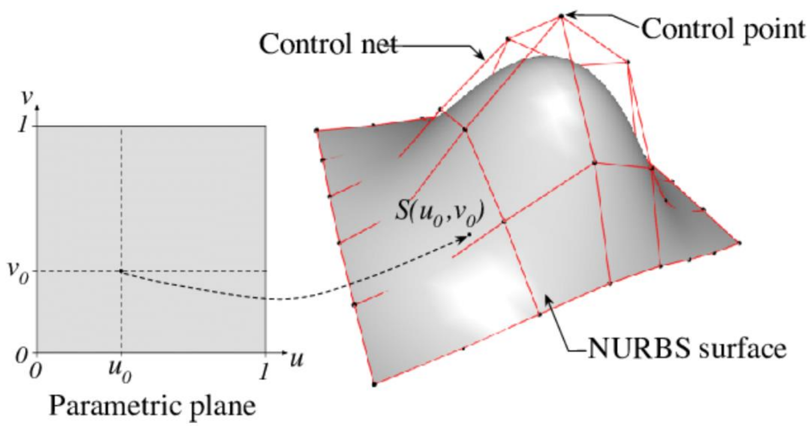
Department of Software
Kwangwoon University

3D Models

- Numerical representations of visual appearances of objects in the 3D space 어떤 객체들을 수치적으로 시각화 할 수만 있다면 3D 모델임



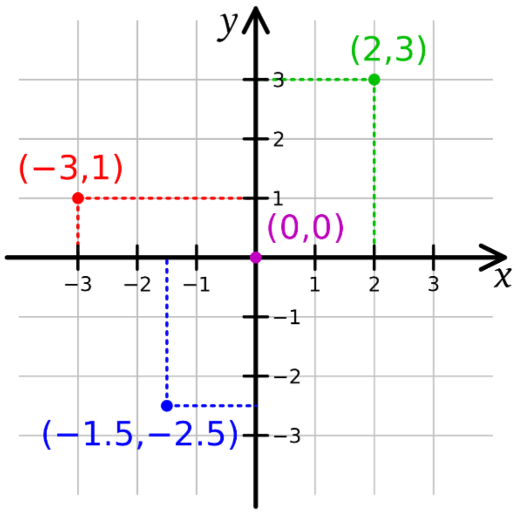


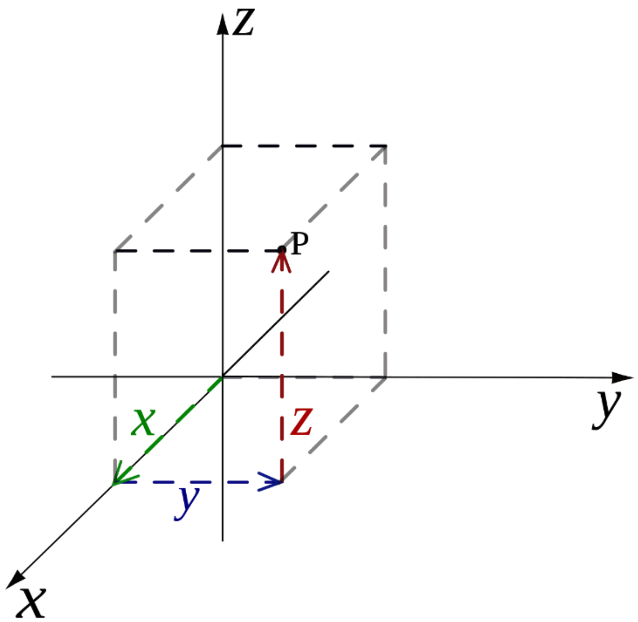


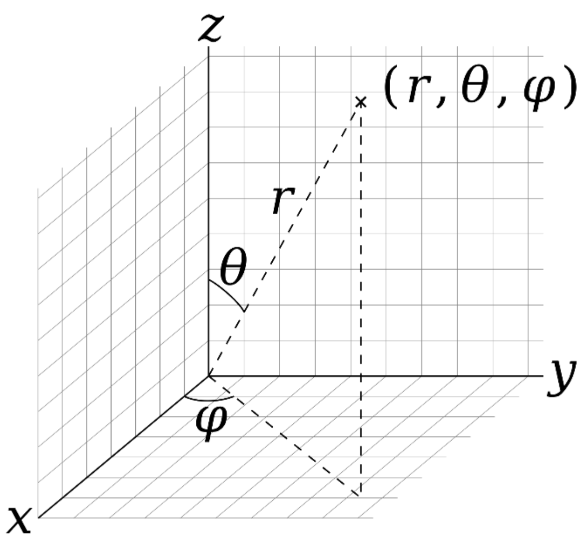
$$\mathbf{C}(t) = \frac{\sum_{i=0}^{n} N_{i,p}(t)\, w_i\, \mathbf{P}_i}{\sum_{i=0}^{n} N_{i,p}(t)\, w_i},$$

Coordinate Systems

- Use one or more numbers, or coordinates, to uniquely determine the positions of the points

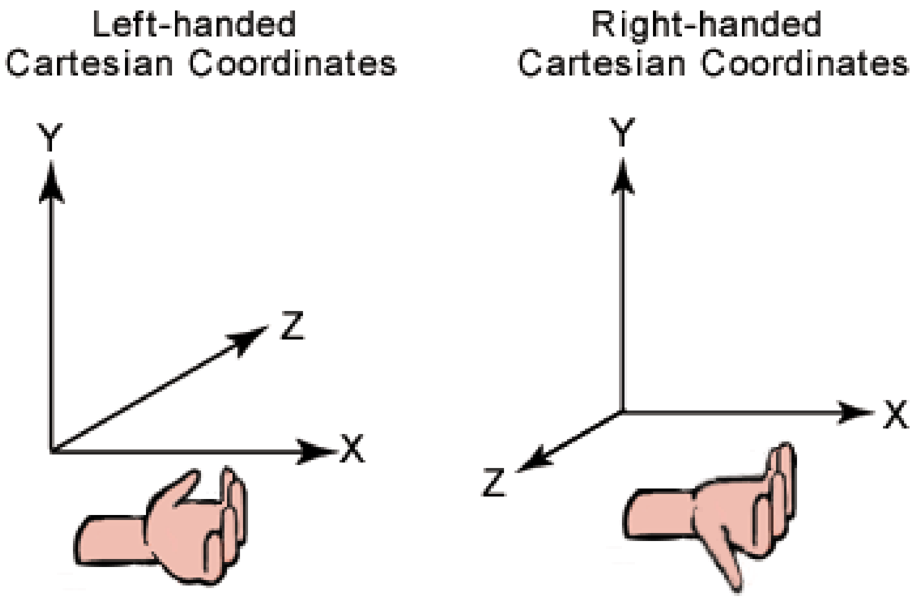


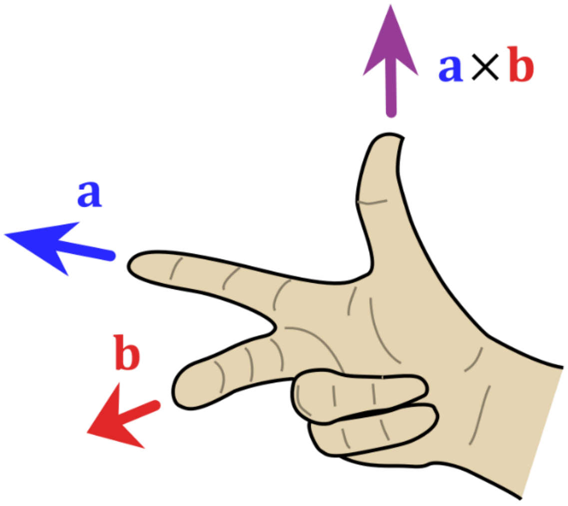




Coordinate Systems

- Left-handed vs. Right-handed cartesian coordinates

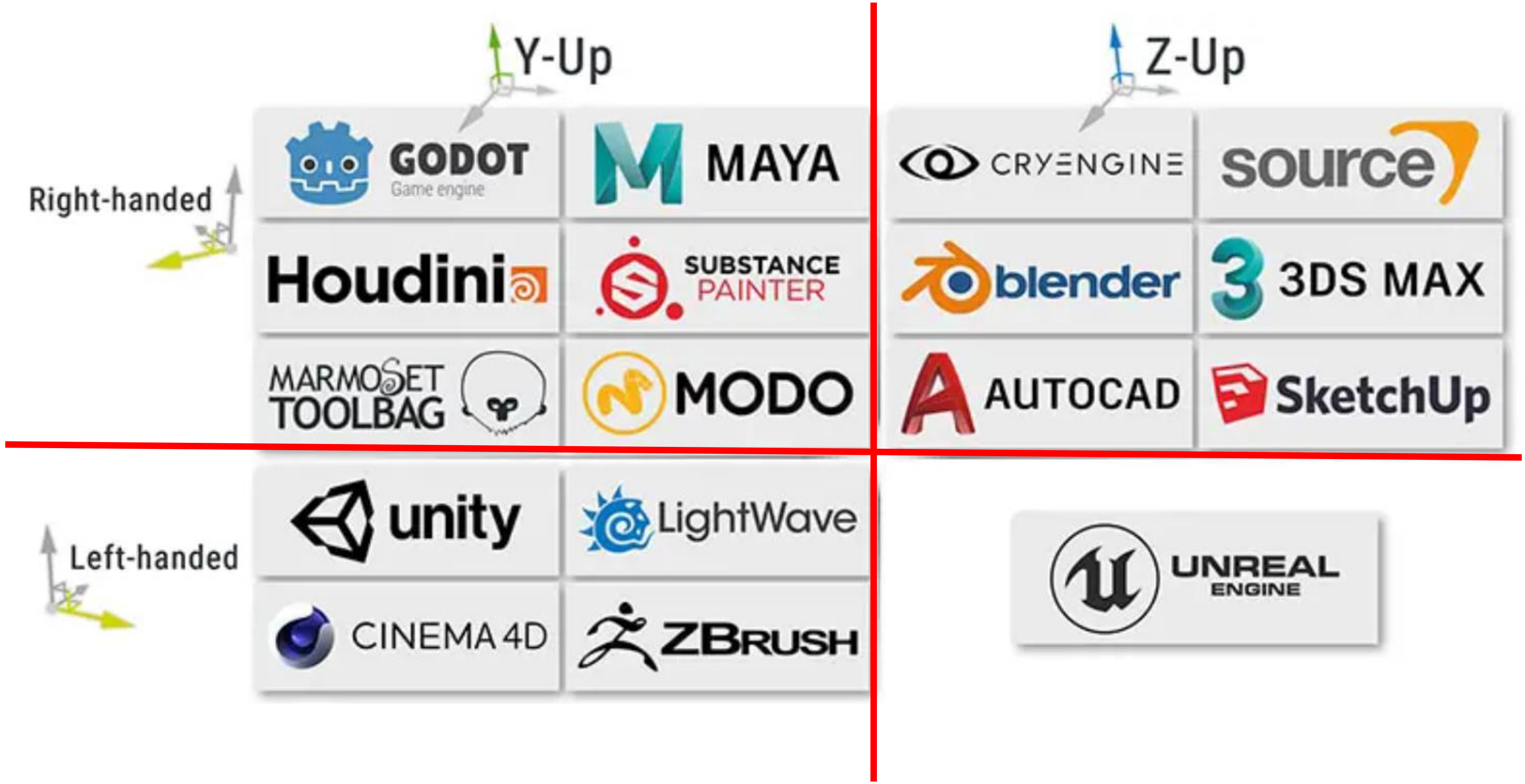




Coordinate Systems

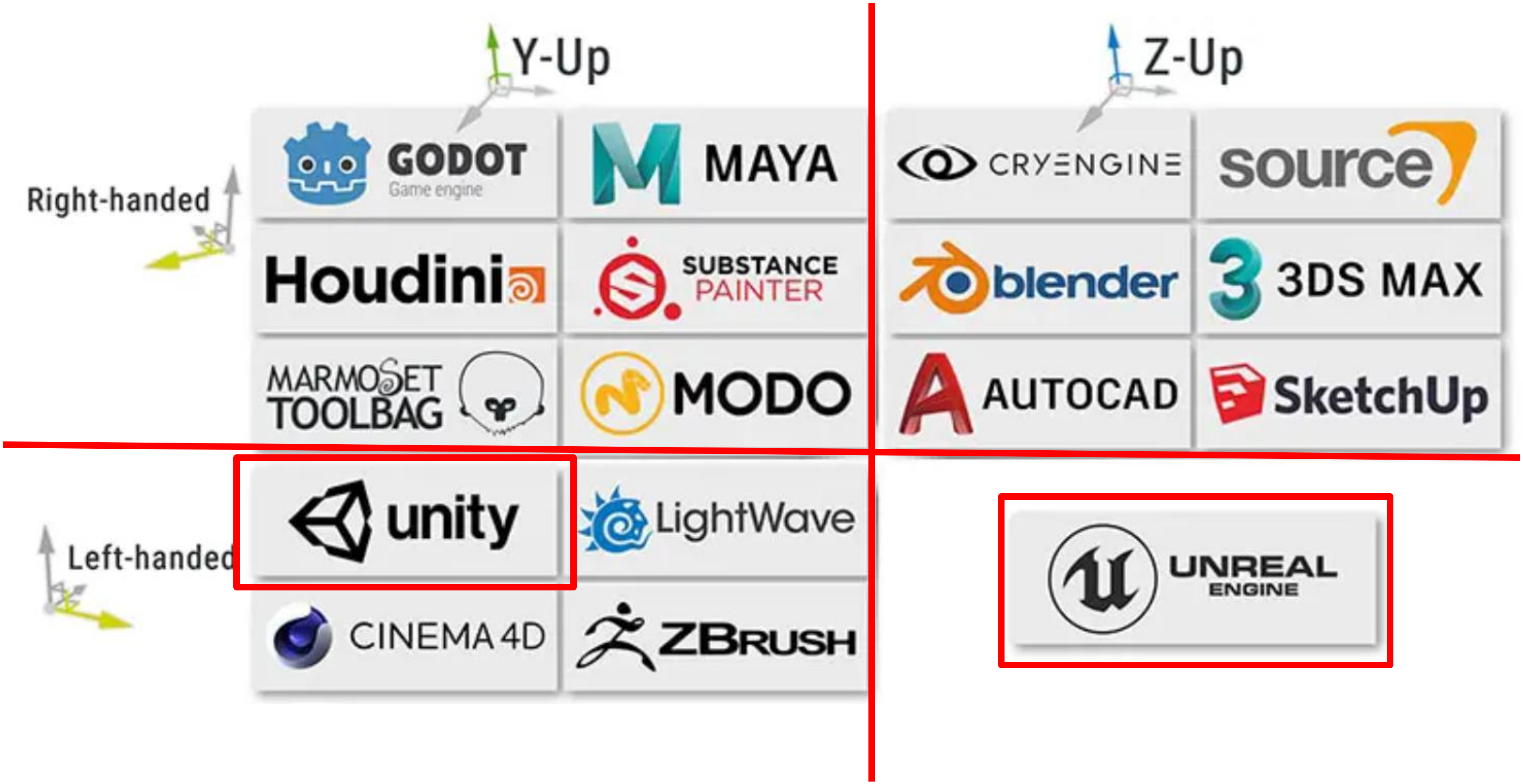
좌표값 호환이 안되니까 다른 프로그램에서 돌릴 때 문제 발생

- Coordinate systems used in popular game engines and 3D software packages including Unity and Unreal



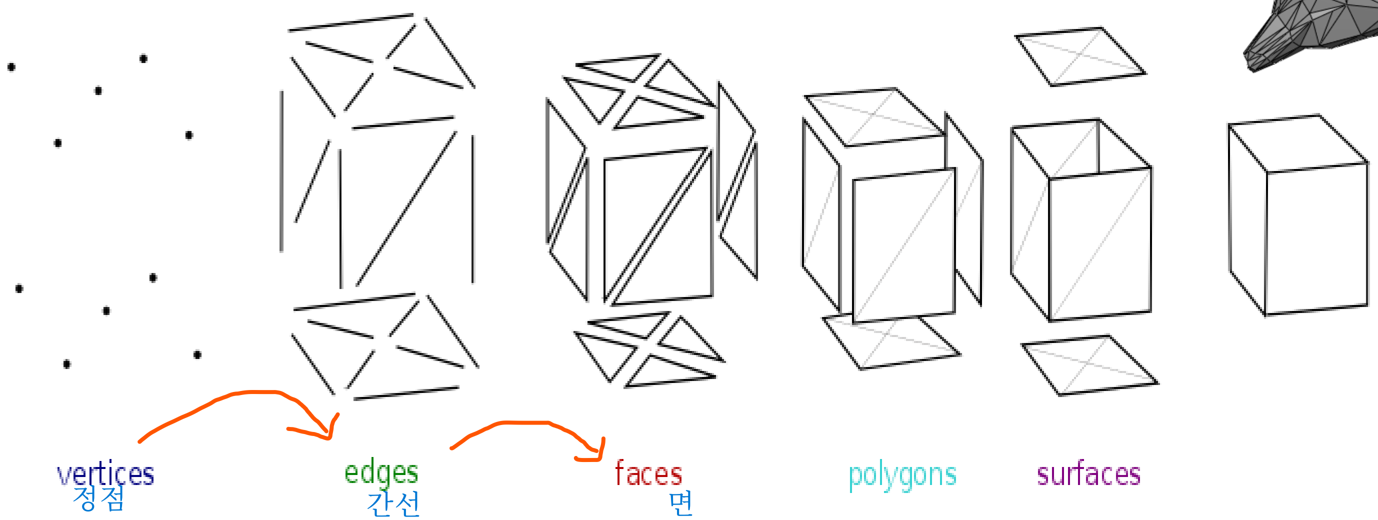
Coordinate Systems

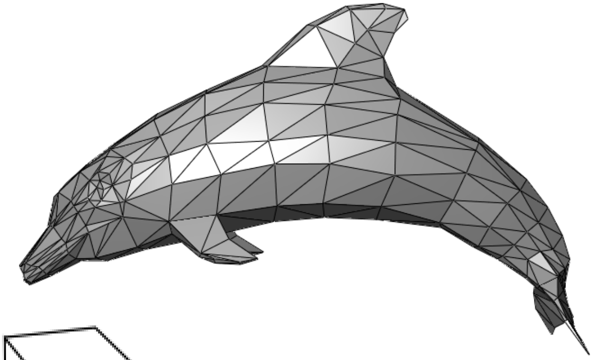
- Coordinate systems used in popular game engines and 3D software packages including Unity and Unreal



Polygonal Meshes

- A collection of vertices, edges and faces that defines the shape of a polyhedral object





이 용어들 신경써서 알아두면되겠구요

자주 사용되는

Data Structures

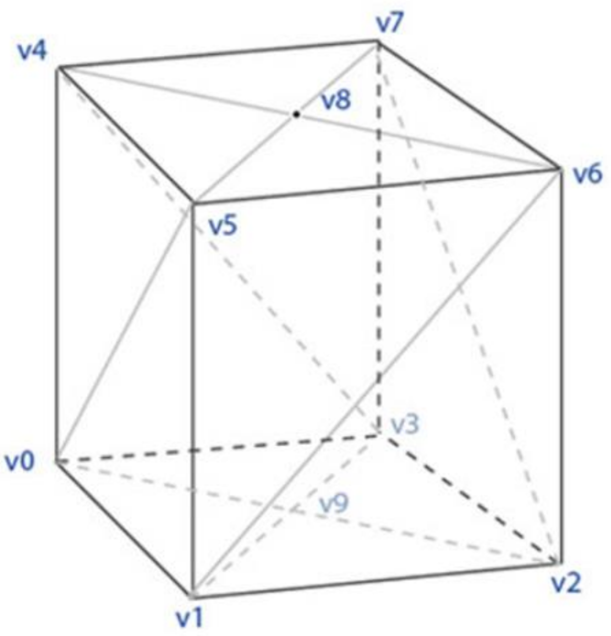
- **Vertex-vertex**
  - The simplest representation
- **Face-vertex**
  - The most widely used representation
- **Winged edge**
  - The most flexible representation

Vertex-Vertex Meshes

- **A set of vertices connected to other vertices**
  - The simplest representation, but not widely used since the face and edge information is implicit

Vertex List

v0	0,0,0	v1 v5 v4 v3 v9
v1	1,0,0	v2 v6 v5 v0 v9
v2	1,1,0	v3 v7 v6 v1 v9
v3	0,1,0	v2 v6 v7 v4 v9
v4	0,0,1	v5 v0 v3 v7 v8
v5	1,0,1	v6 v1 v0 v4 v8
v6	1,1,1	v7 v2 v1 v5 v8
v7	0,1,1	v4 v3 v2 v6 v8
v8	.5,.5,1	v4 v5 v6 v7
v9	.5,.5,0	v0 v1 v2 v3



왼쪽을 가지고 오른쪽이 나오긴하지만?
face 정보는 알 수 없어서
가공이 필요함

Face-Vertex

약간의 가공이 들어간 버전

A set of faces and a set of vertices

- Most widely used representation, being the input typically accepted by modern graphics hardware

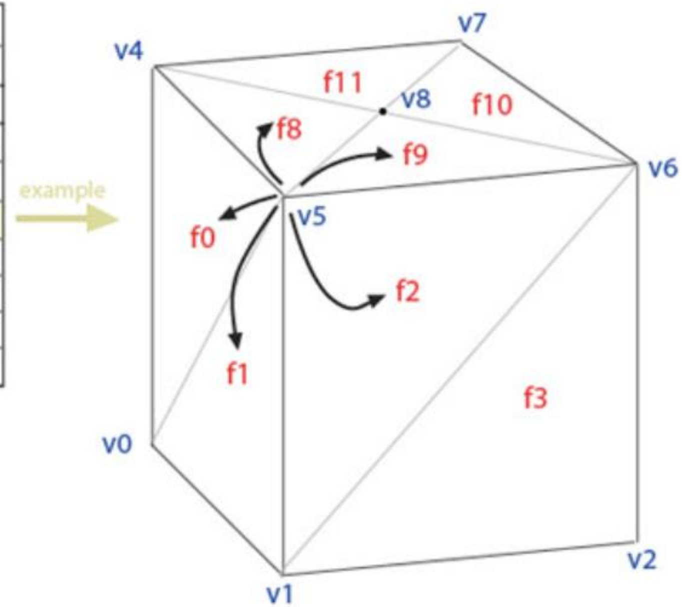
Face List

f0	v0 v4 v5
f1	v0 v5 v1
f2	v1 v5 v6
f3	v1 v6 v2
f4	v2 v6 v7
f5	v2 v7 v3
f6	v3 v7 v4
f7	v3 v4 v0
f8	v8 v5 v4
f9	v8 v6 v5
f10	v8 v7 v6
f11	v8 v4 v7
f12	v9 v5 v4
f13	v9 v6 v5
f14	v9 v7 v6
f15	v9 v4 v7

Vertex List

v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15

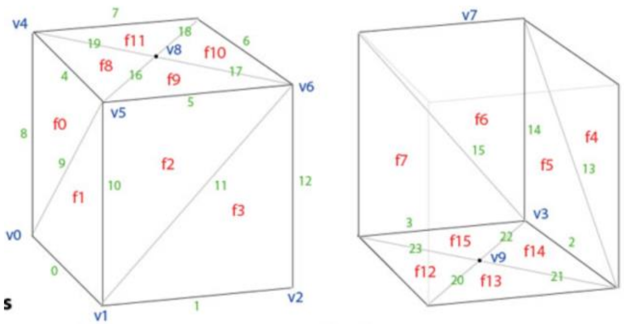
공유하는 vertex list`



face를 구성하는 vertex list
= face에 인접한 vertex list

Winged Edge

- **Explicitly represent the vertices, faces and edges**
  - Widely used in modeling programs to provide the greatest flexibility in dynamically changing the mesh geometry



Face List

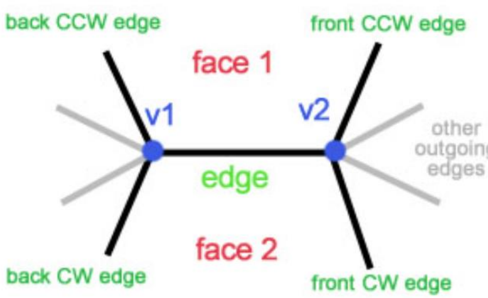
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



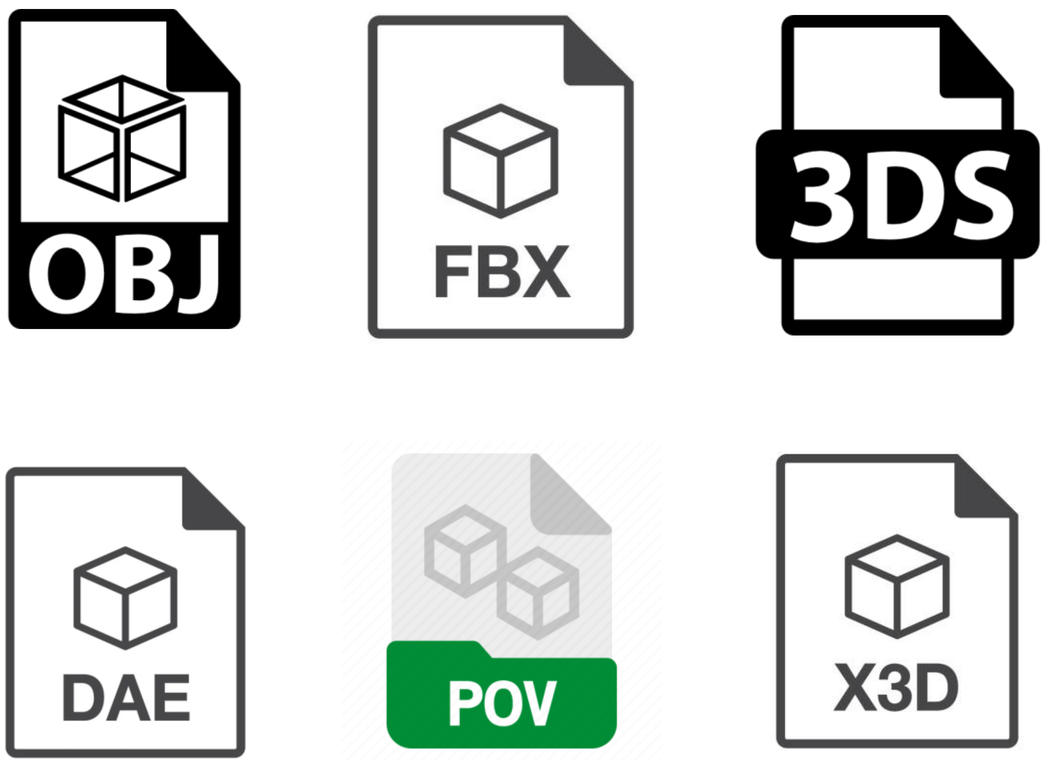
Winged Edge Structure

간선 정보 추가

아오...이게뭔ㅋㅋ

File Formats for Polygonal Meshes

- OBJ
- FBX
- STL
- DAE
- 3DS
- POV
- X3D
- VRML
- …



E.g. OBJ File Formats

- **Geometry definition file formats first developed by Wavefront Technologies**

- **Simple data format that represents 3D geometry alone**
  - The position of each vertex

    (이미지 좌표) : 렌더링 파트에서 설명
  - The texture coordinate of each vertex
  - The normal of each vertex 직교벡터 정보
  - The faces that make each polygon defined as a list of vertices, optionally with texture coordinates and normals

  텍스트 형식으로 되어 있는 파일 포맷임

E.g. OBJ File Formats

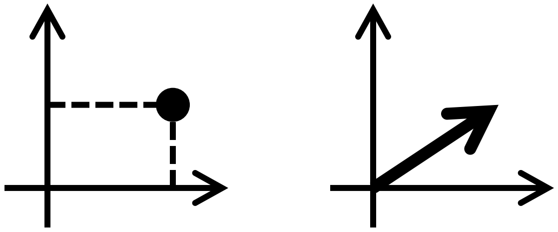
```
# List of geometric vertices, with (x, y, z, [w]) coordinates, w is optional and defaults to 1.0.
v 0.123 0.234 0.345 1.0
v ...
...
# List of texture coordinates, in (u, [v, w]) coordinates, these will vary between 0 and 1. v, w are optional and default to 0.
vt 0.500 1 [0]
vt ...
...
# List of vertex normals in (x,y,z) form; normals might not be unit vectors.
vn 0.707 0.000 0.707
vn ...
...
# Parameter space vertices in (u, [v, w]) form; free form geometry statement (see below)
vp 0.310000 3.210000 2.100000
vp ...
...
# Polygonal face element (see below)
f 1 2 3
f 3/1 4/2 5/3
f 6/4/1 3/5/3 7/6/5
f 7//1 8//2 9//3
f ...
...
# Line element (see below)
l 5 8 1 2 4 9
```

이런 파일 포맷이 있다 정도만 알아둡시당~

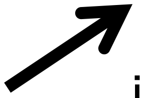
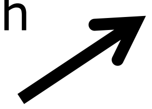
위치 방향 + 크기

Points vs. Vectors

- **Similarity**
  - Both can be numerically encoded as coordinates
    - p = (1.5, 1)
    - v = (1.5, 1)



- **Difference**
  - Points represent location in the space
  - Vectors represent direction and magnitude (without location)

 is same with 

- **Questions**
  - Points can be added? 좌표끼리 더하거나
  - Points can be multiplied with a scalar?
  - How about vectors? 스칼라 좌표값을 n배하는건 기하학적으로 의미가 없음

그런건 벡터에 하는거지 : 구분 잘 해서 써야함

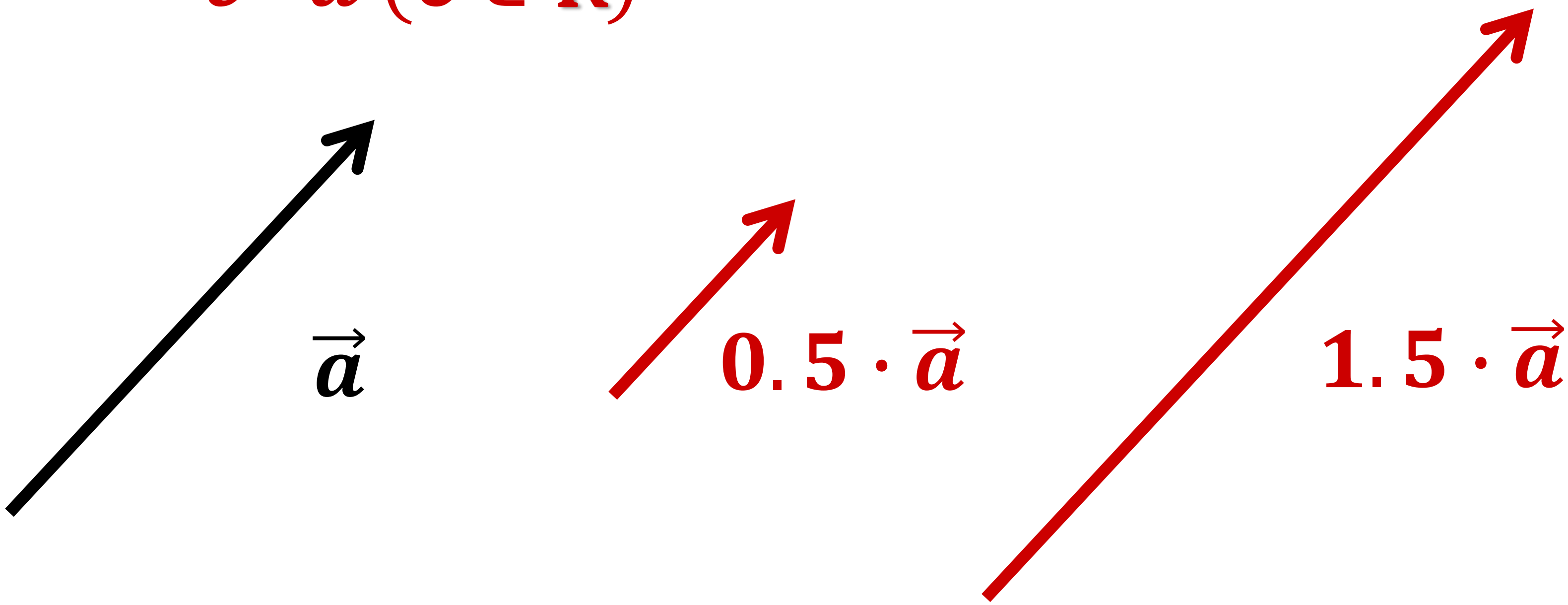
Vector Algebra 벡터에 대해서 리뷰를 좀 해봅시다

- Scalar Multiplication
- Addition
- Subtraction
- Dot Product
- Cross Product

Scalar Multiplication

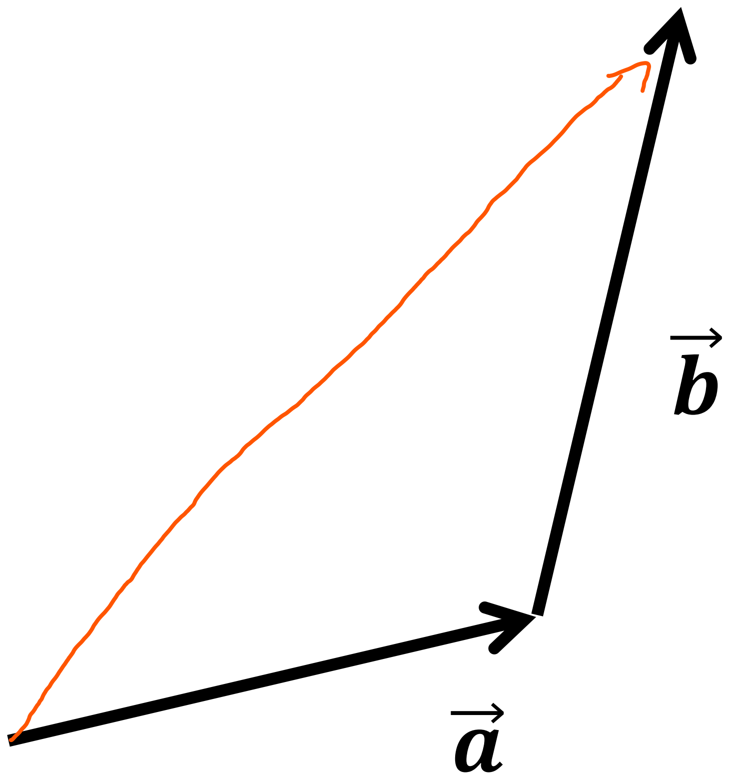
- Change only magnitude without modifying direction

$$c \cdot \vec{a}\ (c \in \mathbf{R})$$



Addition

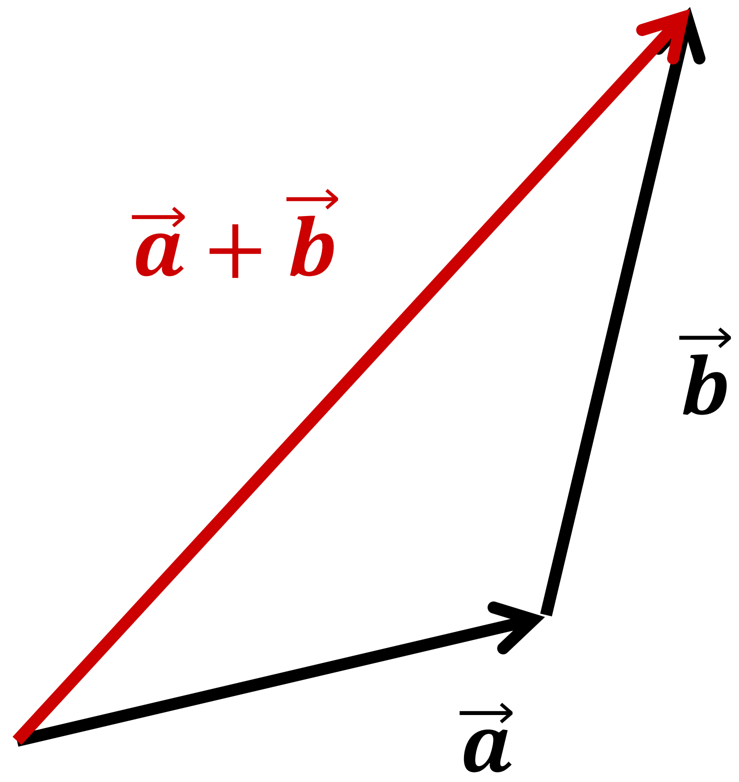
- Translate two vectors such that the head of one vector coincides with the tail of the other vector



Addition

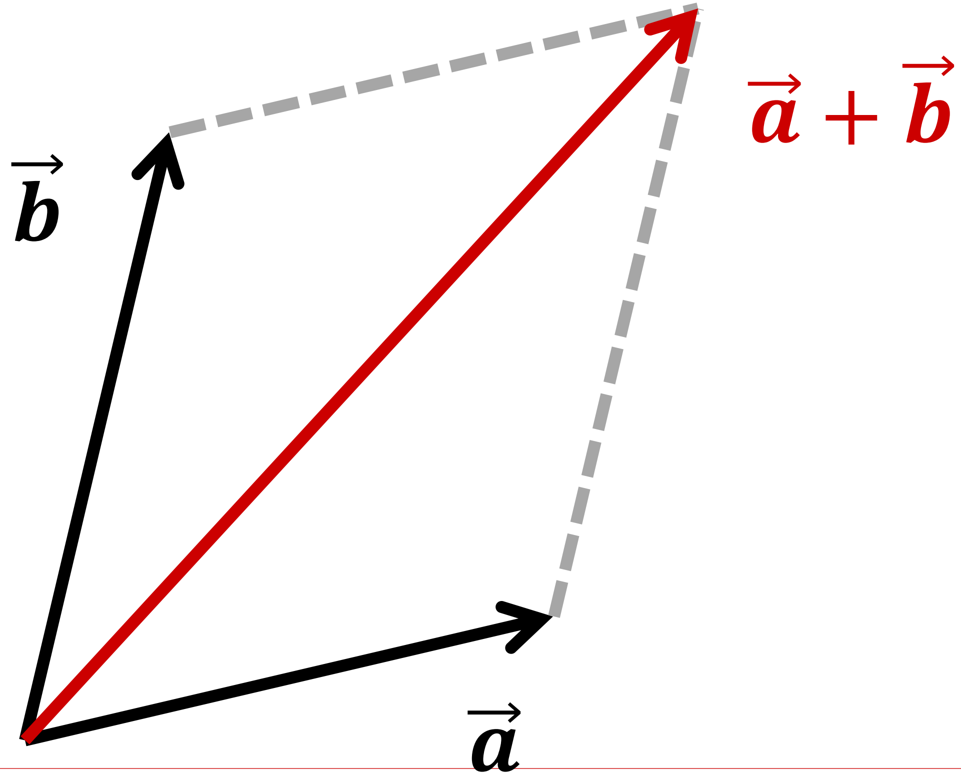
- Then, the addition corresponds to the vector from the tail of the former vector to the head of the latter one

구운~



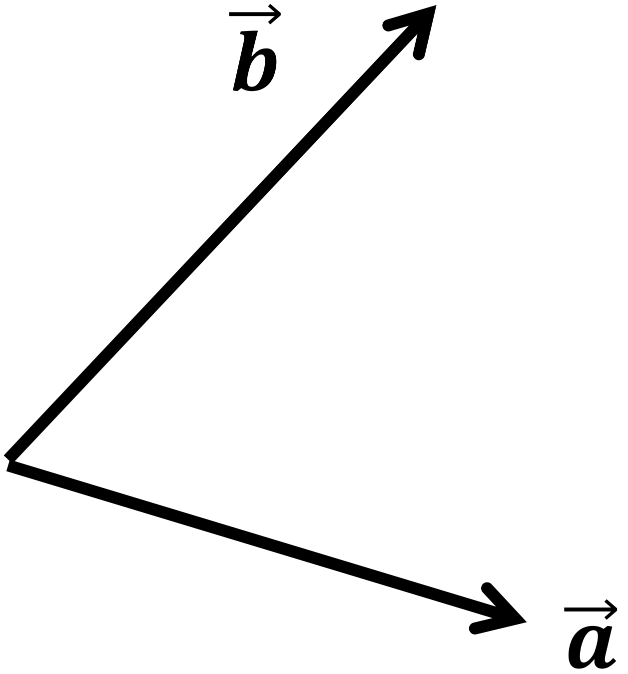
Addition

- Create a parallelogram such that given two vectors form its two adjacent sides, then the addition corresponds to the diagonal of the parallelogram



Subtraction

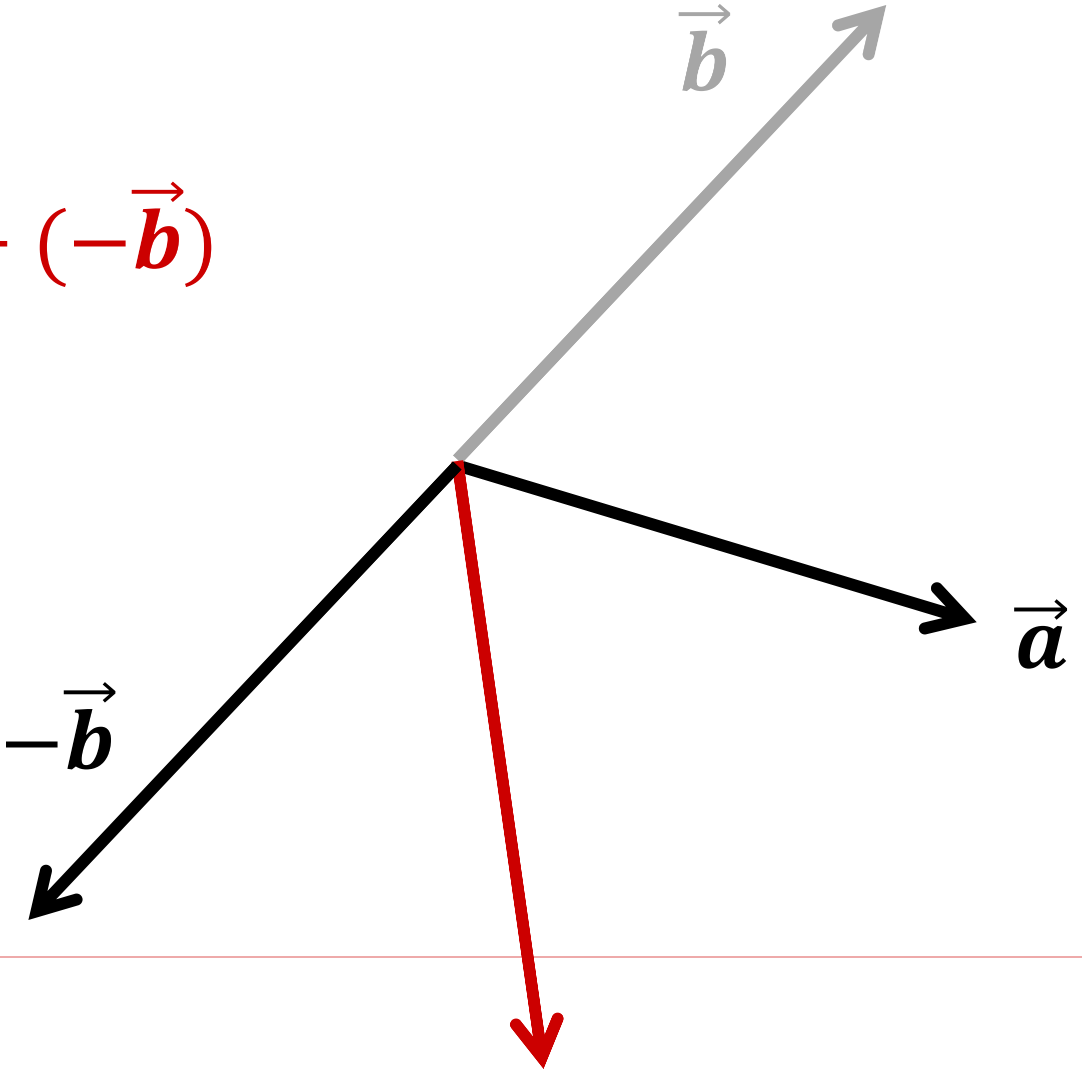
$\vec{a} - \vec{b}$



Subtraction

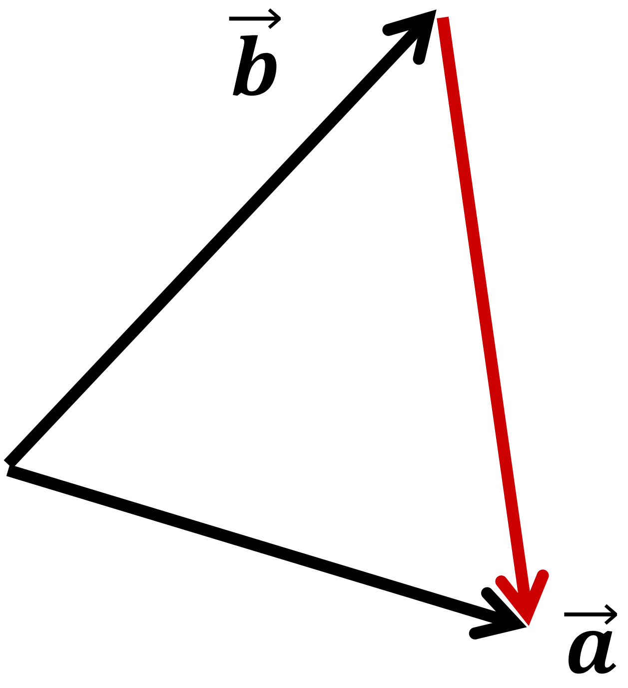
$$\vec{a} - \vec{b}$$

$$= \vec{a} + (-\vec{b})$$



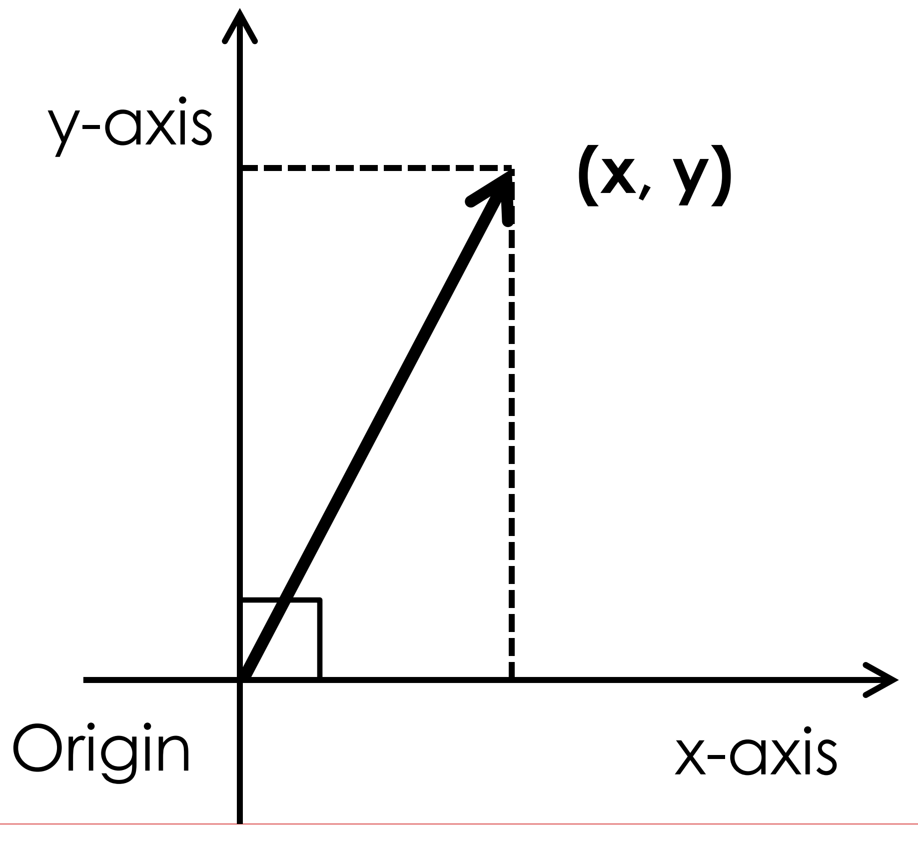
Subtraction

$\vec{a} - \vec{b}$



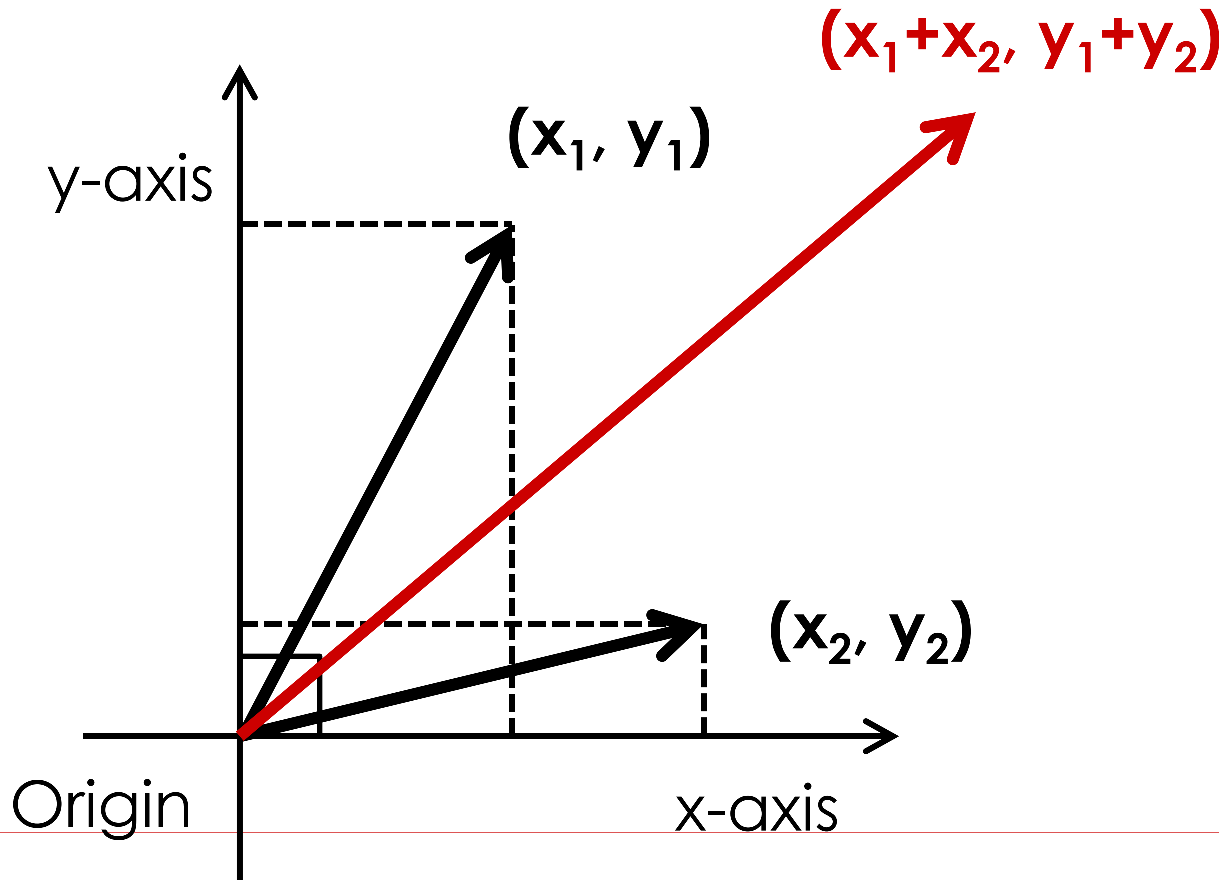
Coordinates of Vectors

- Location of the head of a vector when its tail coincides with the origin of the coordinate system



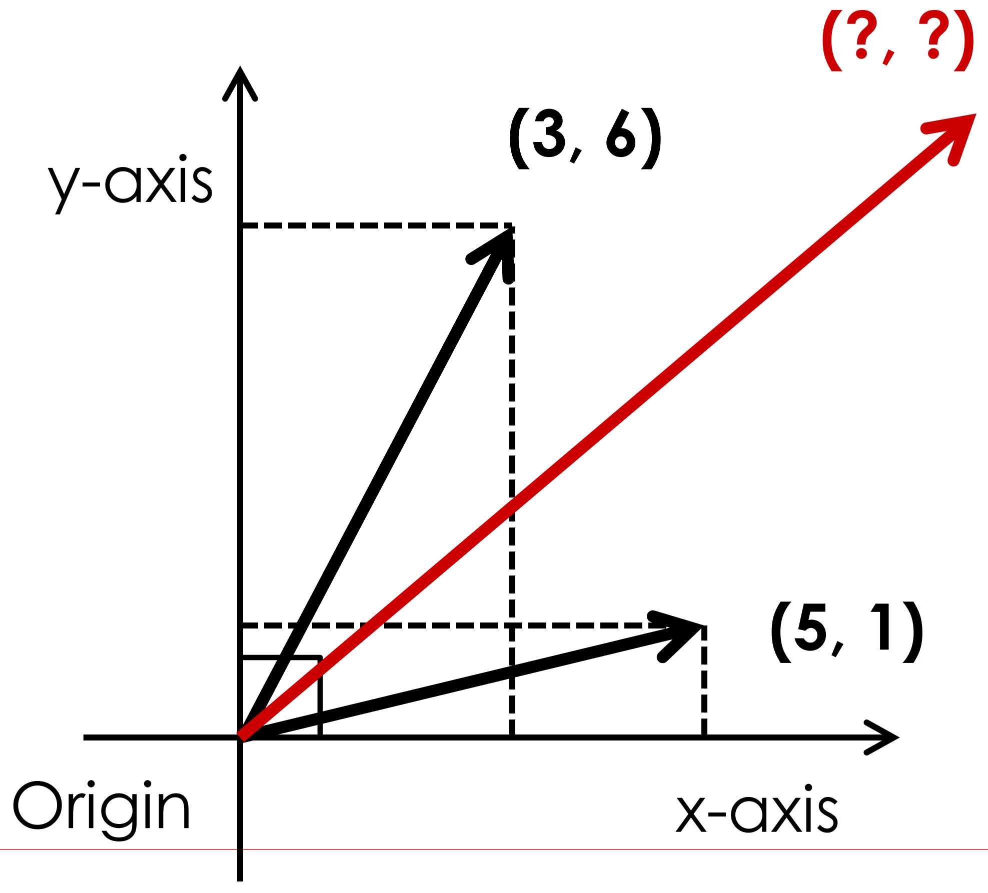
Numerical Computation

- Addition/subtraction of vectors can be obtained by adding/subtracting their associated coordinates



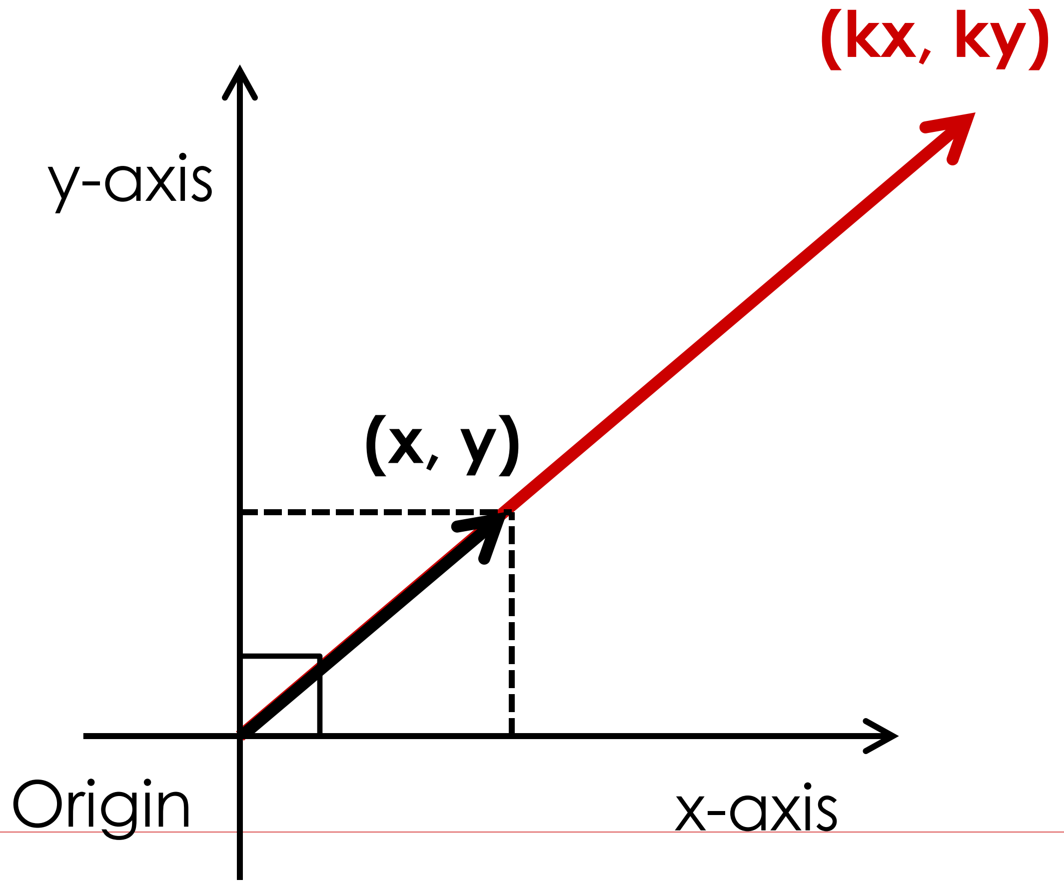
Numerical Computation

- Component-wise add/sub



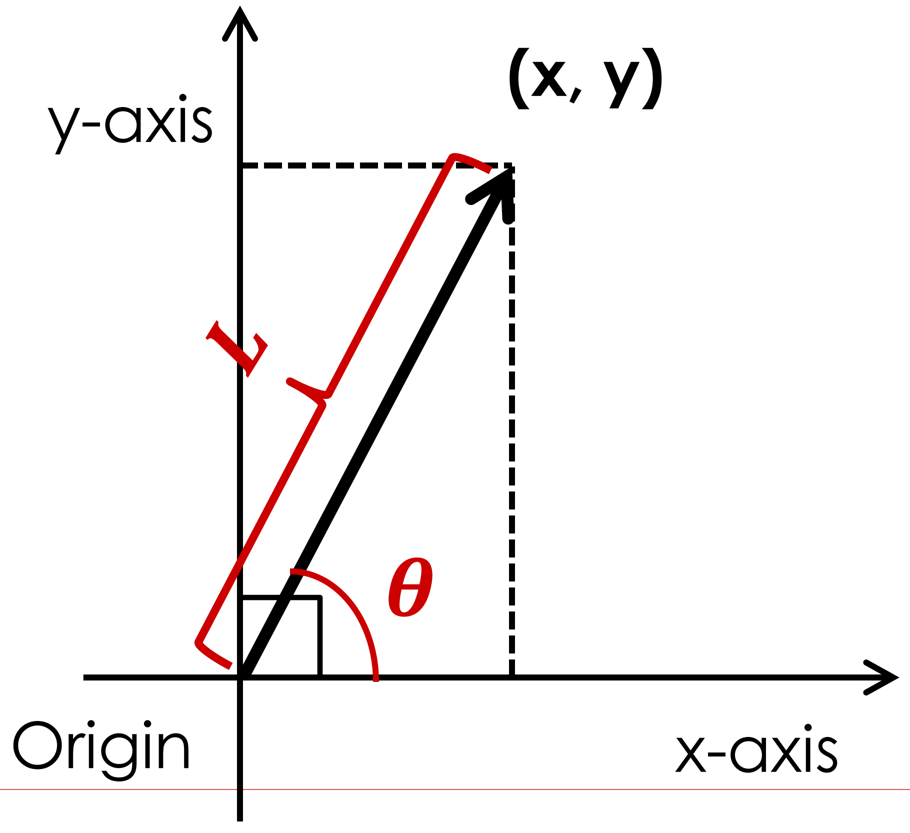
Numerical Computation

- Scalar multiplication of a vector can also be obtained by multiplying its associated coordinates



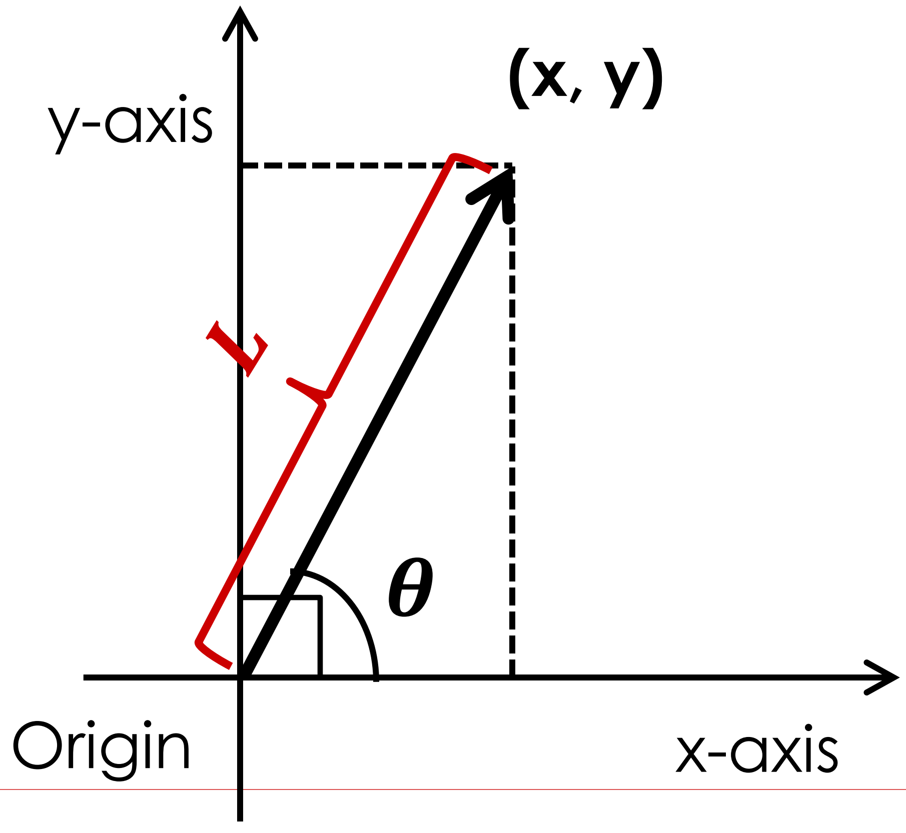
Length and Direction

- Given a vector, calculate its **length** and **direction**?



Length by Pythagorean Theorem

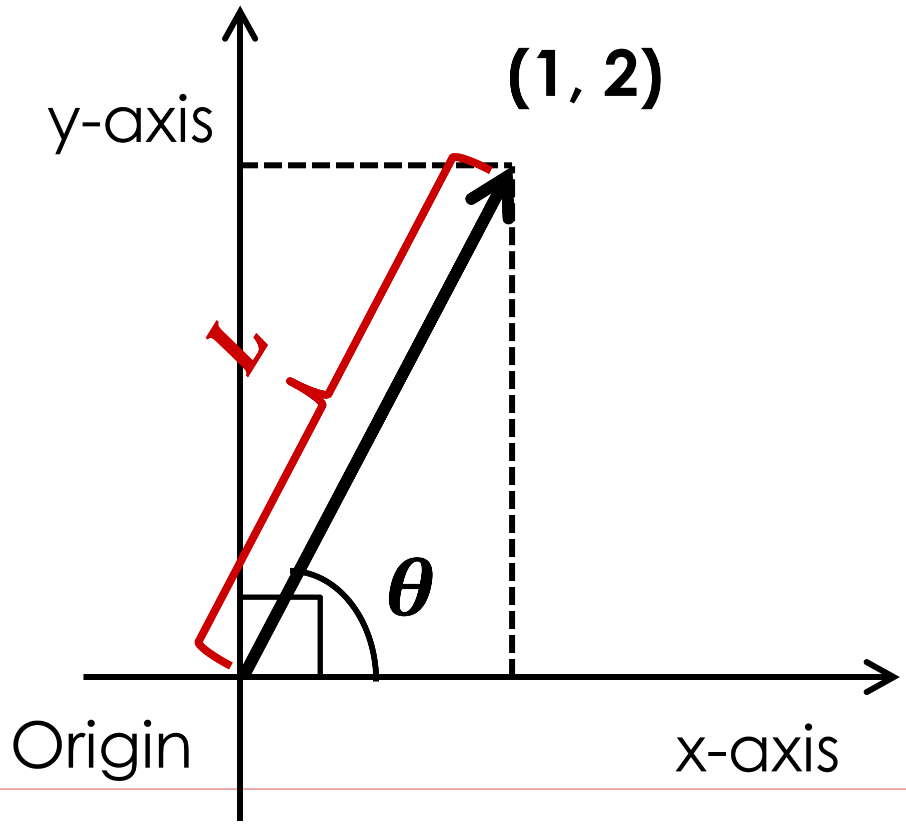
- $x^2 + y^2 = L^2$



Length by Pythagorean Theorem

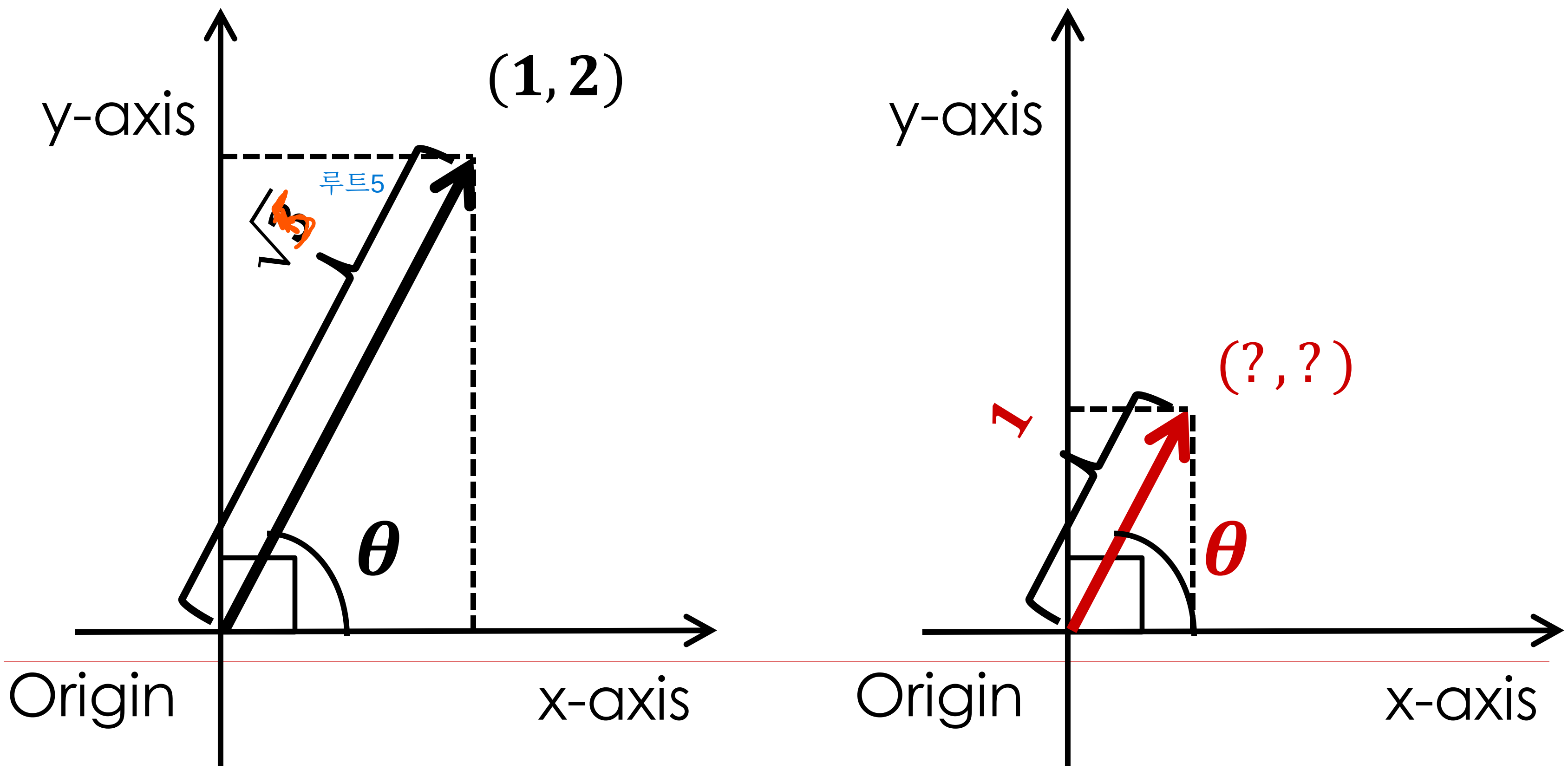
- $x^2 + y^2 = L^2$

Ex) length of $\vec{v} = (1, 2)$?



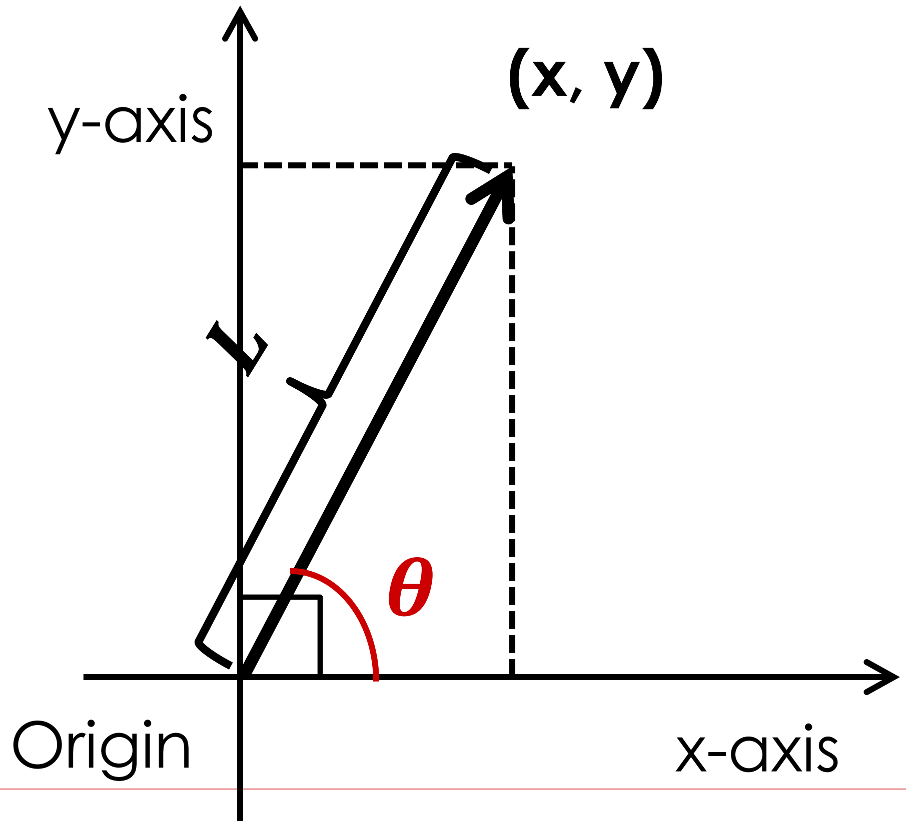
Normalization into Unit Vector

- Scale a vector such that its length becomes one while keeping its direction



Direction by Trigonometry

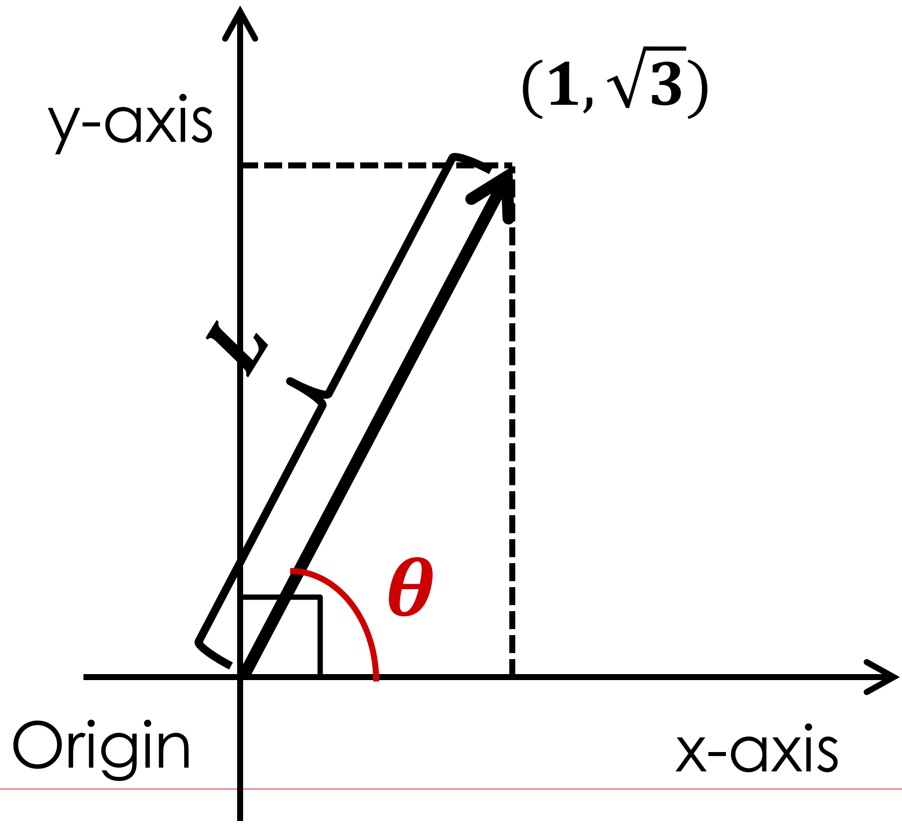
- $\tan\theta = \frac{y}{x}$



Direction by Trigonometry

- $\tan\theta = \frac{y}{x}$

Ex) length and angle of $\vec{v} = (\mathbf{1}, \sqrt{\mathbf{3}})$?



Linear Combination

선형 결합 ㅅㅂ

- Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ be vectors in R^n
- Let $c_1, c_2, \ldots, c_k$ be scalars

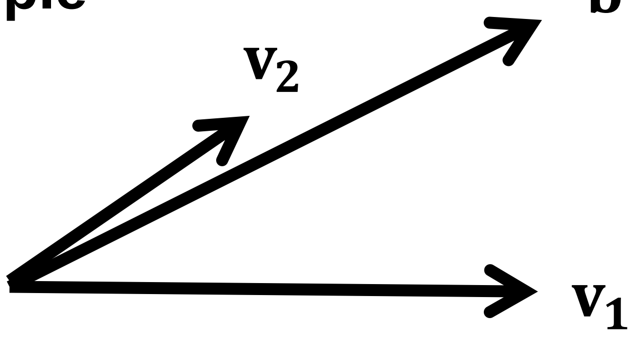
- Then, the vector $b = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k$ is called a linear combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$
  - $c_1, c_2, \ldots, c_k$ are commonly called the "weights"

벡터 상수배 가능 (가중치)

- Example

$$\mathbf{b} = 0.5\mathbf{v}_1 + 2\mathbf{v}_2$$

선형 결합 : 임의의 상수 k 들어갈때 나올수있는애들



Affine Combination

- Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ be vectors in R^n
- Let $c_1, c_2, \ldots, c_k$ be scalars such that

$$\sum_{i=1}^{k} c_i = 1$$

가중치의 합이 1인 경우 Affine 결합이라고함

- Then, the vector $b = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k$ is called an affine combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$
  - $c_1, c_2, \ldots, c_k$ are commonly called the "weights"

중점구할때 우째하노? 1/2 하고 더하죠?
1/2 1/2이 weight잖아요
합이 1이죠?
이게 일종의 affine 결합인거죠

Convex Combination

- Let $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ be vectors in $\mathbf{R}^n$
- Let $c_1, c_2, \ldots, c_k$ be scalars such that

$$c_i \geq 0 \text{ and } \sum_{i=1}^{k} c_i = 1$$

양수 조건 추가되면 Convex 결합

- Then, the vector $b = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_k\mathbf{v}_k$ is called an convex combination of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$
  - $c_1, c_2, \ldots, c_k$ are commonly called the “weights”

왜 소개하냐? 기하학적 의미를 가지니까요

간단히 넘어가네요 그래도

Vector Space 하...참고삼아 알아두세요 휴 ~

- For all vectors u, v, w and scalars a, b:
  - $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$
  - $\mathbf{u} + (\mathbf{v} + \mathbf{w}) = (\mathbf{u} + \mathbf{v}) + \mathbf{w}$
  - There exists a *zero vector* "$\mathbf{0}$" such that $\mathbf{v} + \mathbf{0} = \mathbf{0} + \mathbf{v} = \mathbf{v}$
  - For every $\mathbf{v}$ there is a vector "$-\mathbf{v}$" such that $\mathbf{v} + (-\mathbf{v}) = \mathbf{0}$
  - $1\mathbf{v} = \mathbf{v}$
  - $a(b\mathbf{v}) = (ab)\mathbf{v}$
  - $a(\mathbf{u} + \mathbf{v}) = a\mathbf{u} + a\mathbf{v}$
  - $(a + b)\mathbf{v} = a\mathbf{v} + b\mathbf{v}$ 연산에서 만족해야하는 성질들

이걸 만족하는 벡터들의 집합이 벡터공간 형성

- Any collection of objects satisfying all of this properties is a **vector space** (e.g. polynomials, functions, etc.)

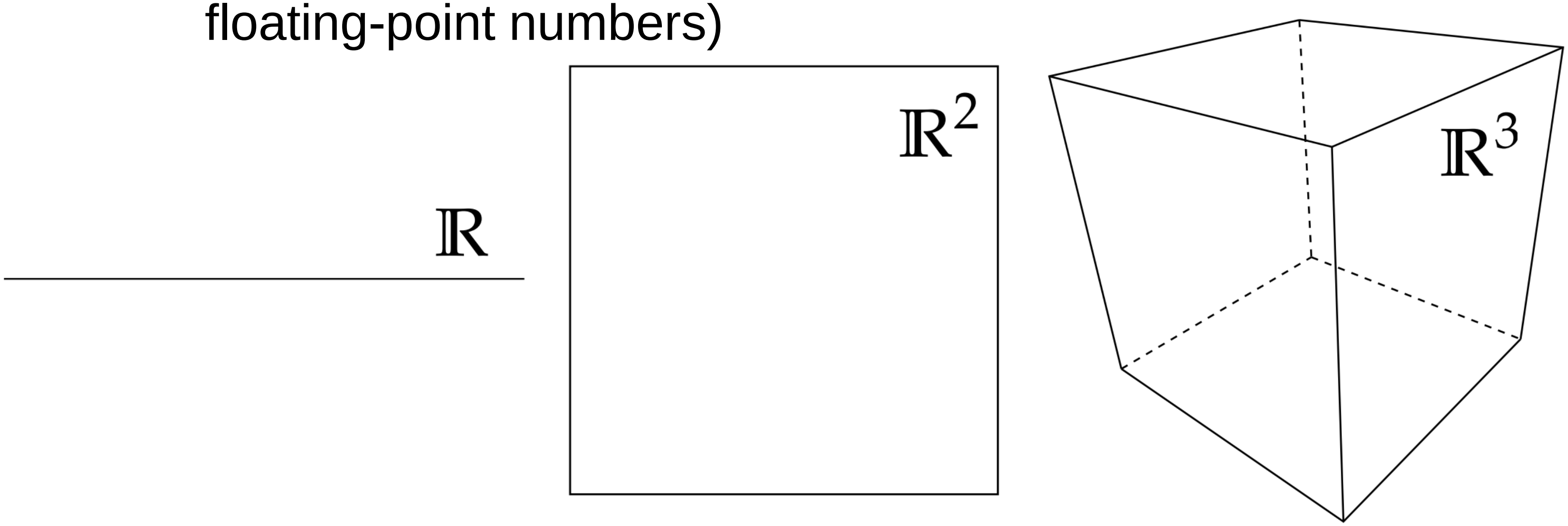
다항식들의 집합 / 함수들의 집합도
벡터공간으로 표현 가능
: 일반화할수있는 개념

인 것을 참고삼아 알아두세용~

유클리드 N차원 공간 : n개의 실수로 표현할 수 있는 이런 공간을 주로 다루죠 저희는

Euclidean Vector Space

- **Most common example: Euclidean n-dimensional space**
- **Typically denoted by $\mathbf{R}^n$, meaning “n real numbers”**
- **E.g., (1.23, 4.56, 1.57) is a point in $\mathbf{R}^3$**
- **Why such a common example?**
  - Looks a lot like the space we live in!
  - That’s what we can easily encode on a computer (a list of floating-point numbers)



Span 두 벡터의 선형 결합으로 만들 수 있는 모든 벡터의 집합

- Q: Geometrically, what is the *span* of two vectors $\mathbf{u}$, $\mathbf{v}$?

- A: The *span* is the set of all vectors that can be written as a linear combination of $\mathbf{u}$ and $\mathbf{v}$, i.e., vectors of the form:

$$a\mathbf{u} + b\mathbf{v}$$

  for any two numbers a, b

- More generally:

$$span(\mathbf{u}_1, \cdots, \mathbf{u}_k) = \left\{ \mathbf{x} \in V \,\middle|\, \mathbf{x} = \sum_{i=1}^{k} a_i \mathbf{u}_i \,, a_1, \cdots, a_k \in \mathbf{R} \right\}$$

~~간단히 넘어가네용?~~

Basis

기저 ㅅㅂ ㅠㅠㅠ

span을 만들 수 있는 최소환의 벡터

- Span is closely related to the idea of a basis
- In particular, if we have exactly n vectors $\mathbf{e}_1, \cdots, \mathbf{e}_n$ such that

$$span(\mathbf{e}_1, \cdots, \mathbf{e}_n) = \mathbf{R}^n$$

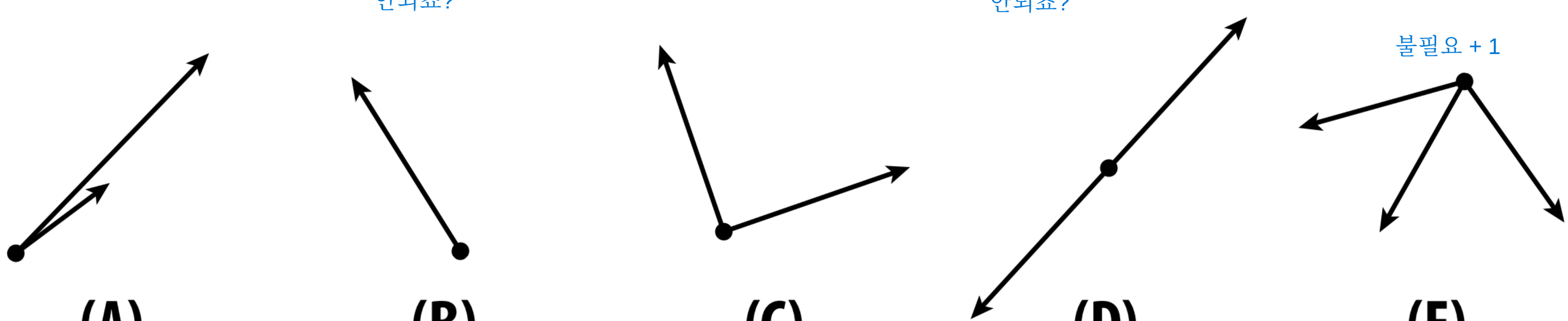
n차원 : n개의 기저 벡터 필요

- Then we say that these vectors are a basis for $\mathbf{R}^n$
- Note: many different choices of basis!
- Q: Which of the following are bases for the 2D plane (n=2)? 기저 벡터의 개수 = 차원 dim

안되죠?

안되죠?

불필요 + 1



(A) (B) (C) (D) (E)

보통 직교 벡터 쓰죠?

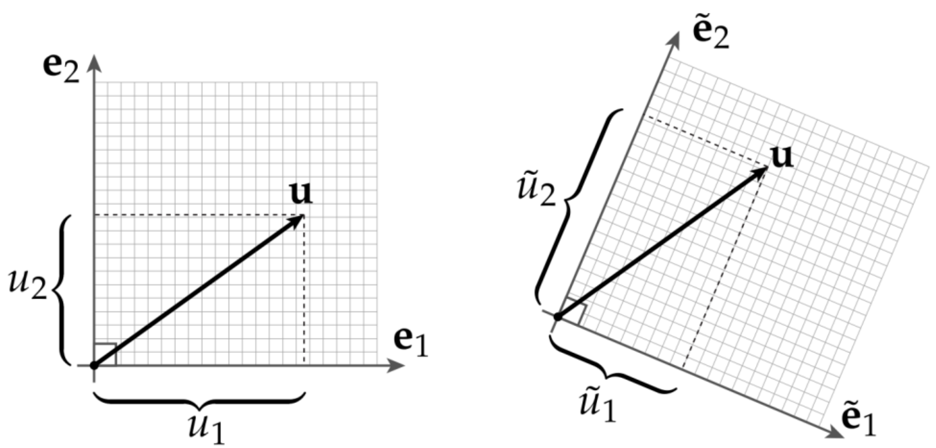
Orthonormal Basis

: 단위 벡터로 쓰면 => 직교 단위 벡터

- Most often, it is convenient to have basis vectors that are (i) unit length and (ii) mutually orthogonal.
- In other words, if $\mathbf{e}_1, \cdots, \mathbf{e}_n$ are our basis vectors then

$$\langle \mathbf{e}_i, \mathbf{e}_j \rangle = \begin{cases} 1, & i = j \\ 0, & \text{otherwise.} \end{cases}$$

- This way, the geometric meaning of the sum $\mathbf{u}_1^2 + \cdots + \mathbf{u}_n^2$ is maintained: it is the length of the vector **u**

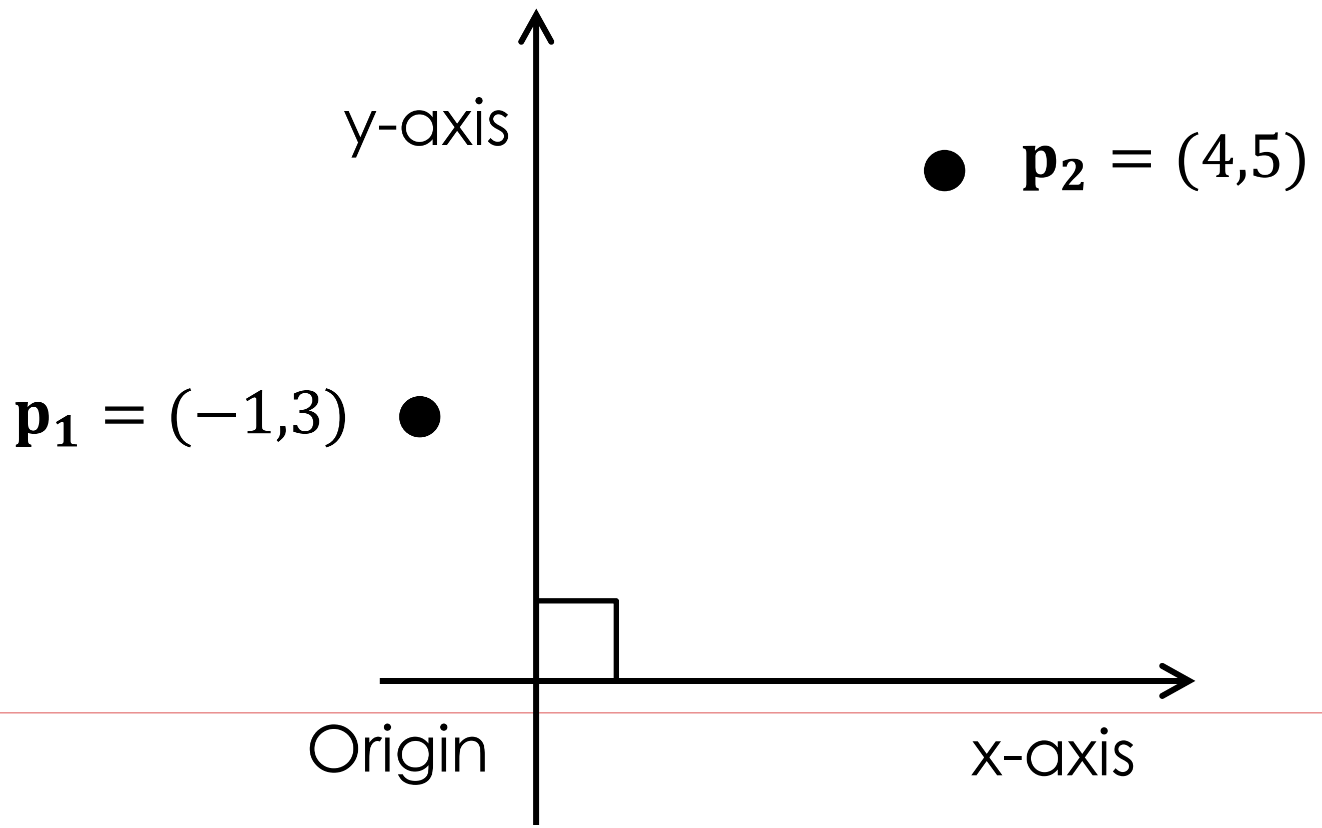


Revisiting Operations on Points

- Points can be added?
- Points can be subtracted?
- Points can be multiplied with scalars?

Point – Point = Vector

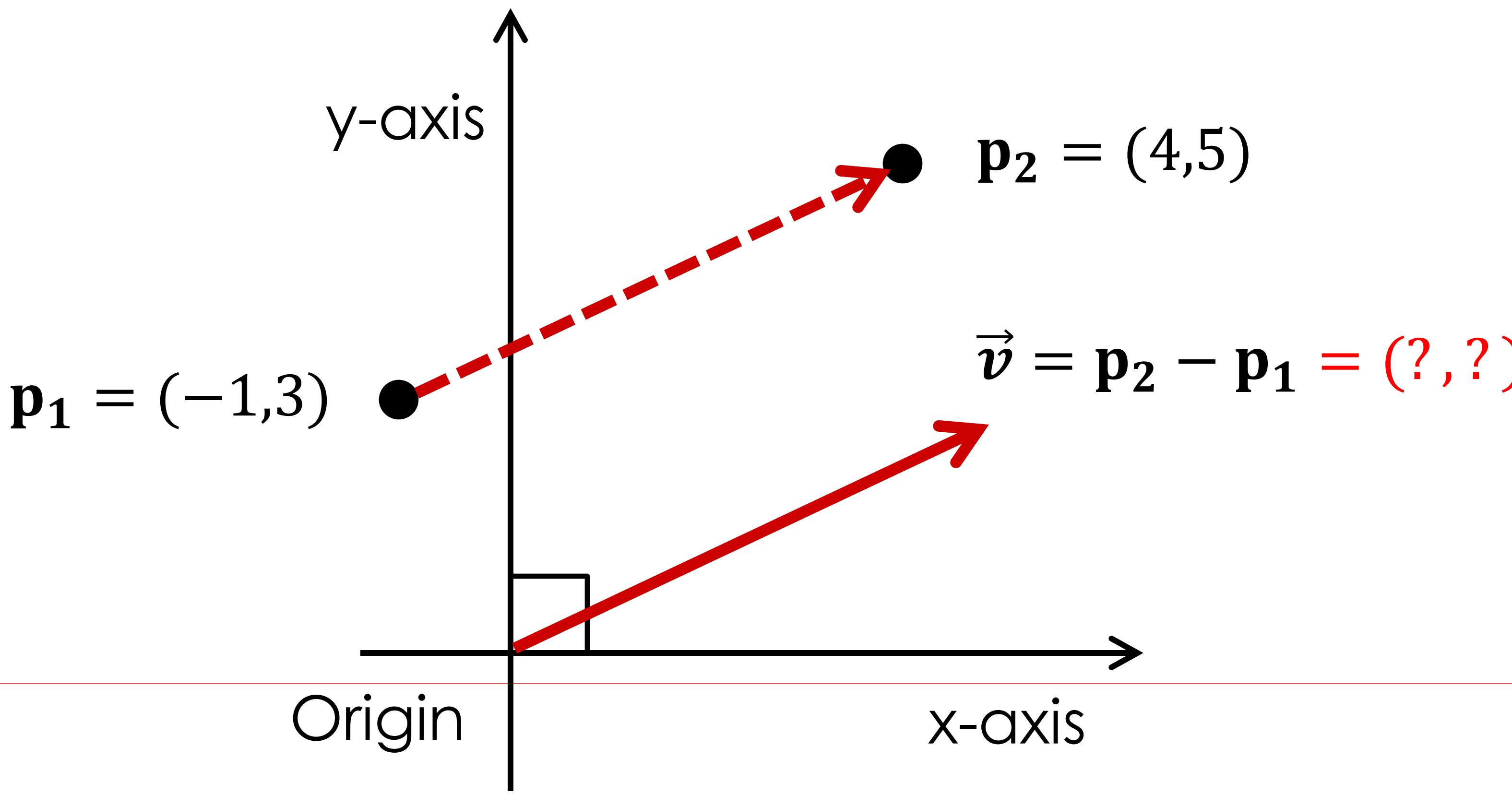
- Relative location of one point with respect to another point



Point – Point = Vector

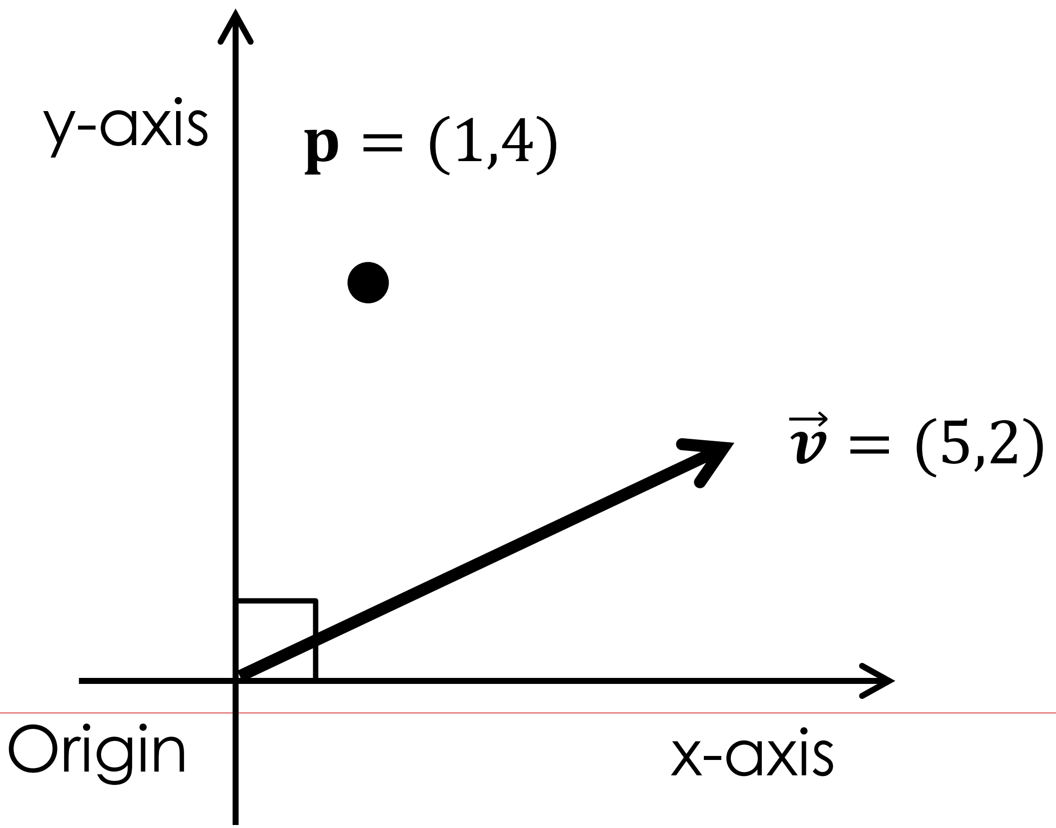
- Relative location of one point with respect to another point

: 특정 방향과 거리로 Point 가 이동한 효과



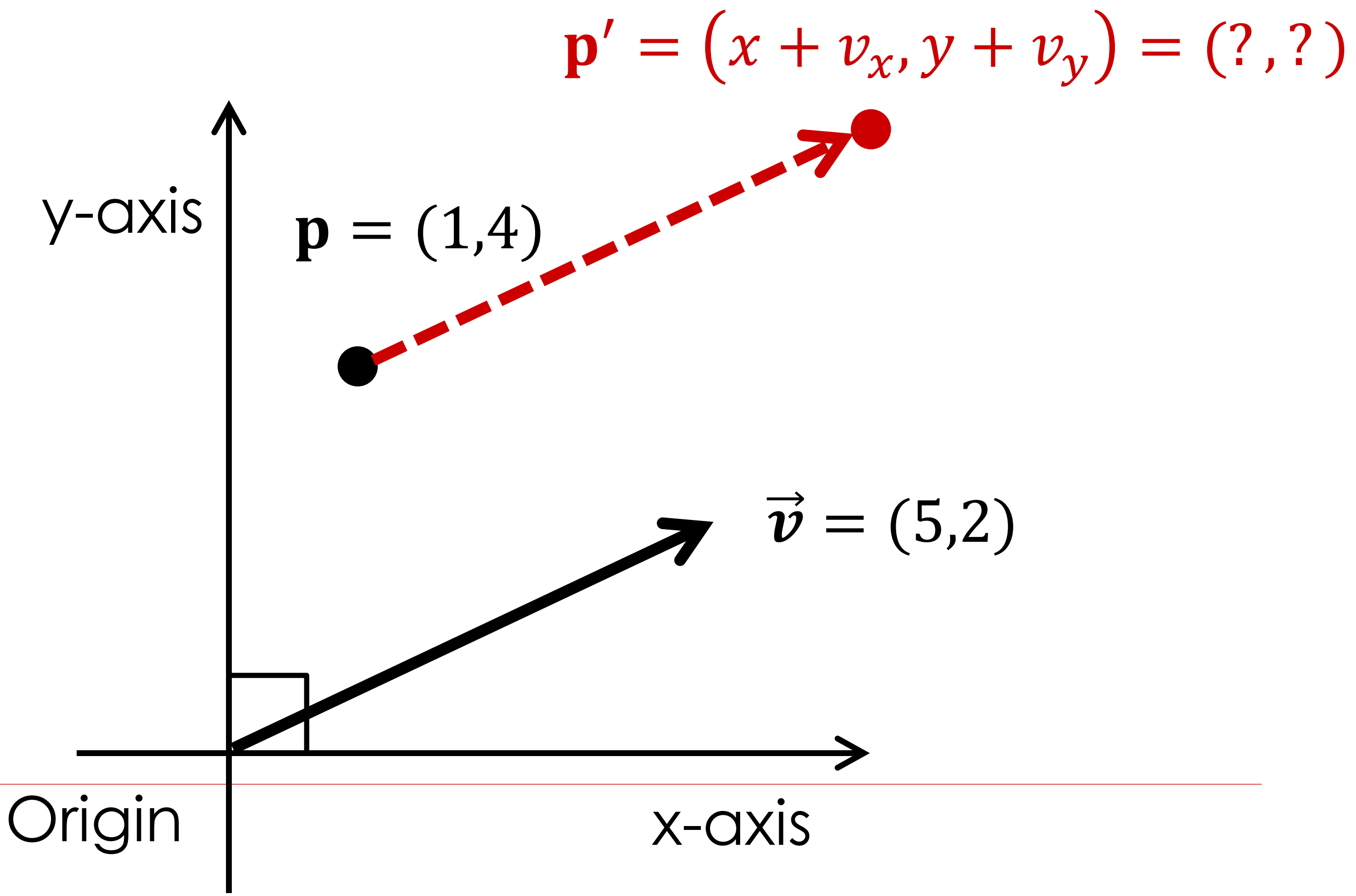
Point + Vector = Point

- Translate a point toward a direction by a magnitude



Point + Vector = Point

- Translate a point toward a direction by a magnitude

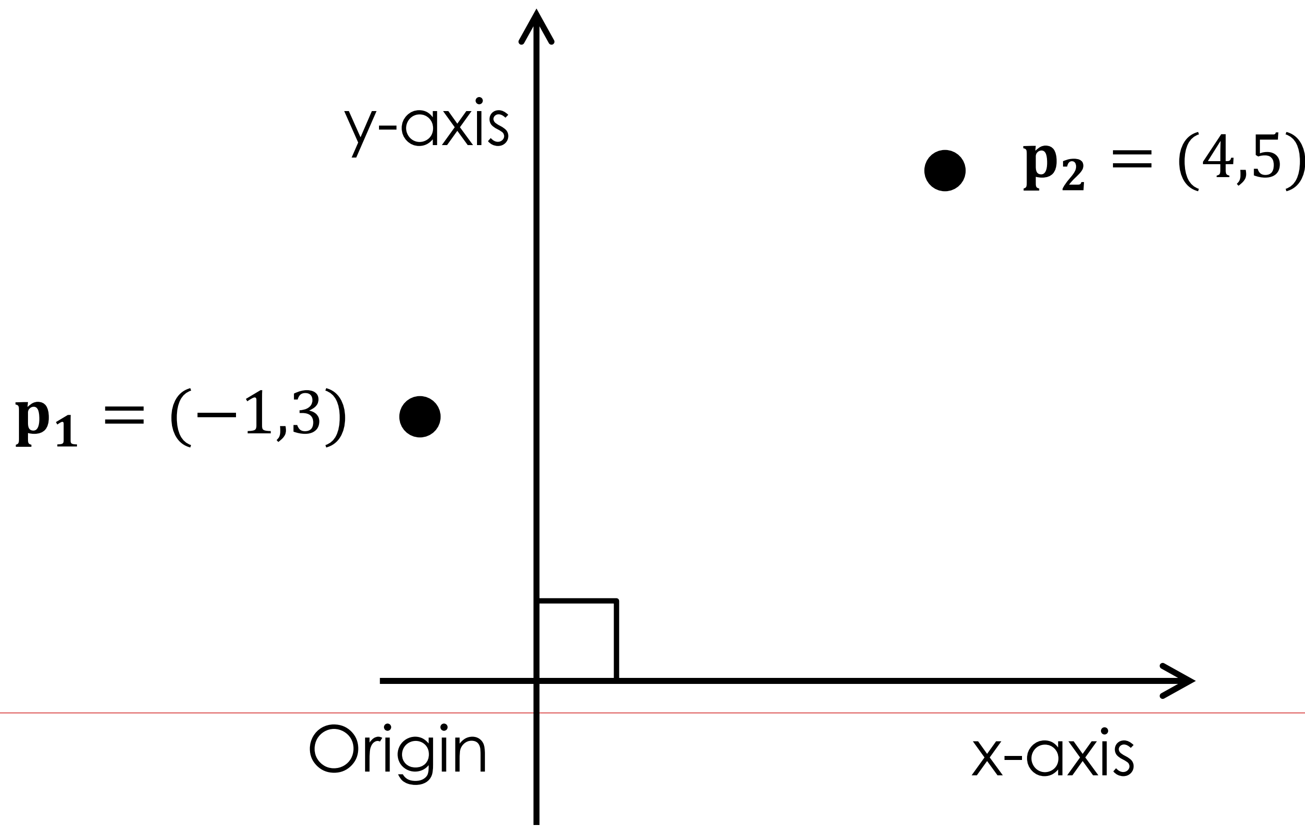


Point·t? Point + Point ?

- What is $10 \cdot \mathbf{p}_1$?
- What is $\mathbf{p}_1 + \mathbf{p}_2$?

이런 연산들은 애초에 허용하지 않아용

근데? 만약 허용한다면? 좌표계에 따라서 다른 결과가 일어나니까?
그냥 허용을 안하는거지



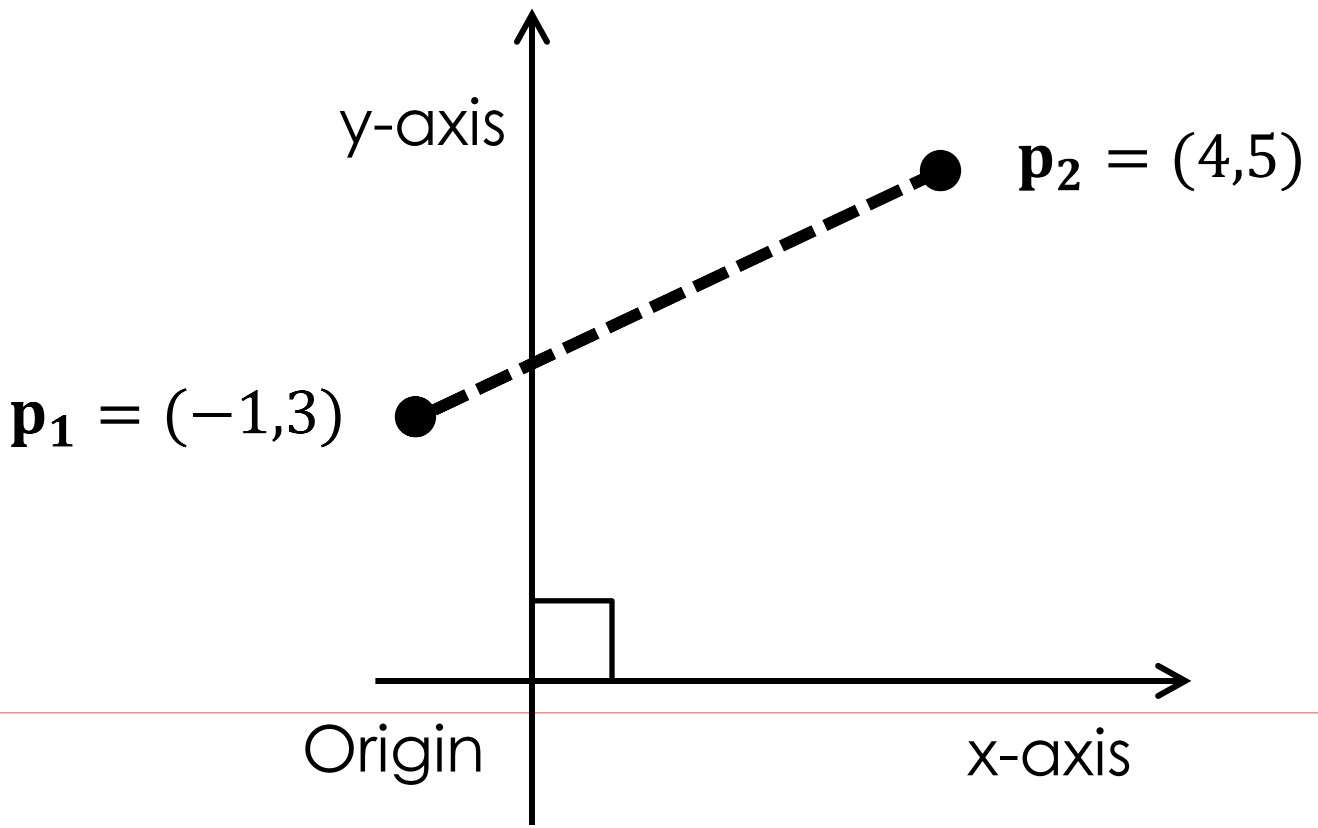
아주 예외적인 case인데 이건 허용합니다

Point·*t* + Point·*(1-t)* = Point

이게 뭔데?
선형 결합 중 affine combination 조건을
만족하면 허용합니다

왜냐?? 내분의 의미를 가지거든요
계수의 합이 1이 된다는 조건

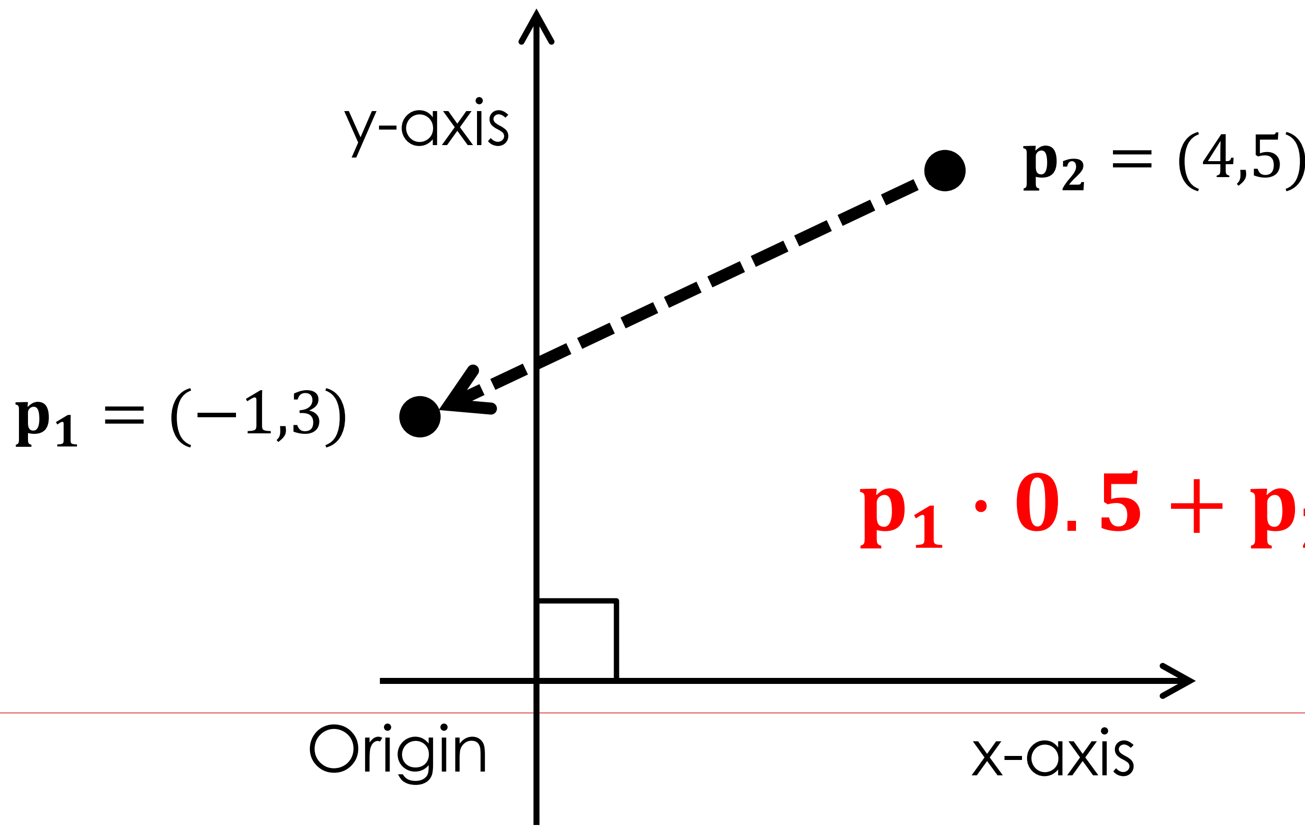
- What is $\mathbf{p}_1 \cdot 0.5 + \mathbf{p}_2 \cdot 0.5$?



Point·t + Point·$(1-t)$ = Point

왜 허용되느냐를 따져보려면? 전개한다음에 까보면 다음과 같고, 포인터에 벡터를 더하는 건 가능하니까 허용된다는거지

$$\mathbf{p_1} \cdot t + \mathbf{p_2} \cdot (1 - t) = \mathbf{p_2} + (\mathbf{p_1} - \mathbf{p_2}) \cdot t$$



$$\mathbf{p_1} \cdot \mathbf{0.5} + \mathbf{p_2} \cdot \mathbf{0.5} = ?$$

Additional Vector Operations

- **Dot product**

  내적이죠? N차원 가능

  - Also called **scalar** product

    왜냐? 결과가 스칼라고

- **Cross product**

  외적이죠?
  3차원에서만 정의가 되구요

  - Also called **vector** product

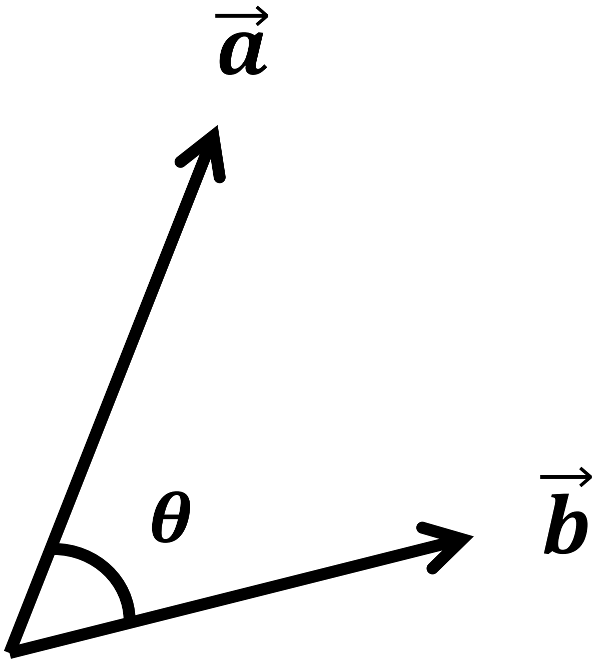
    결과가 벡터거든요

Dot Product

- $\vec{a} \cdot \vec{b} = \|\vec{a}\| \|\vec{b}\| cos\theta$

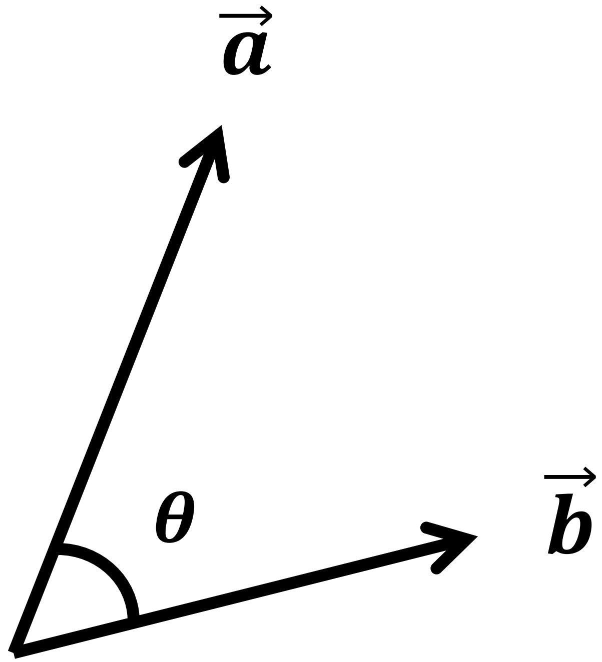
두 벡터의 길이 x cos(사이각)

= 스칼라 결과



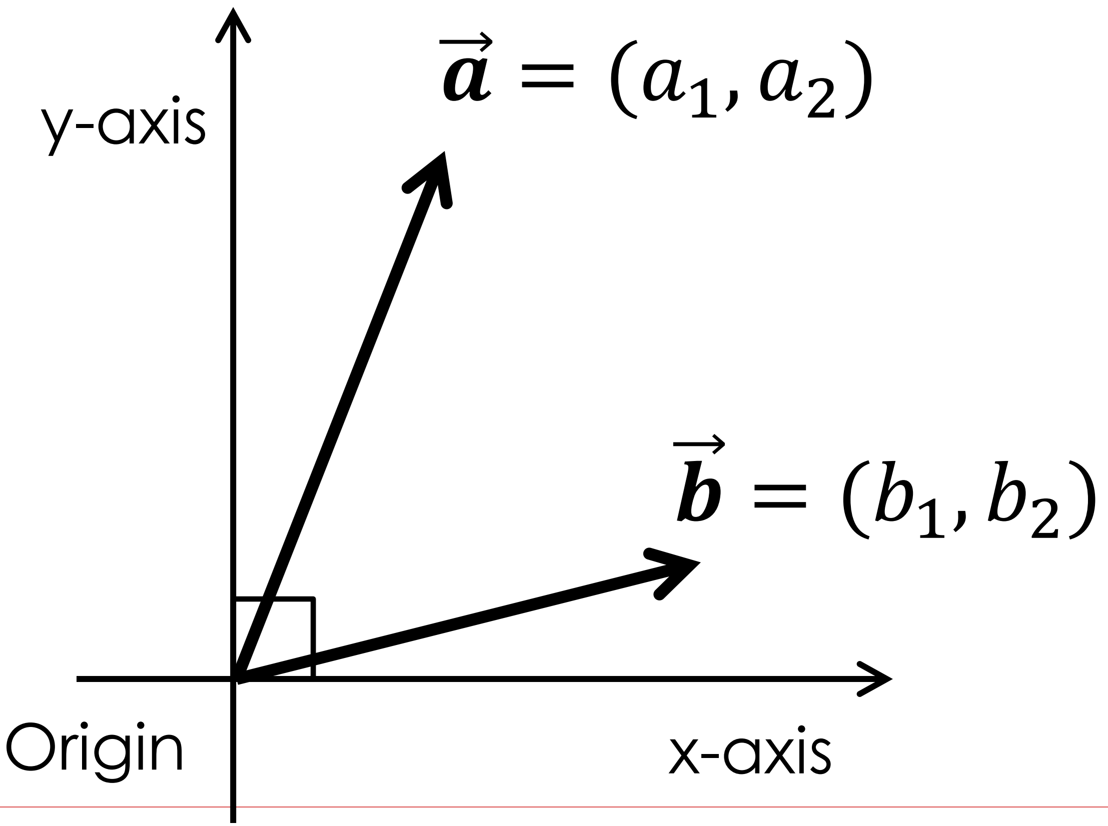
Dot Product

- $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = \|\vec{\boldsymbol{a}}\| \|\vec{\boldsymbol{b}}\| cos\theta$
  - $\|\vec{\boldsymbol{a}}\| = 4$
  - $\|\vec{\boldsymbol{b}}\| = 3$
  - $\theta = 45°$
  - $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = ?$



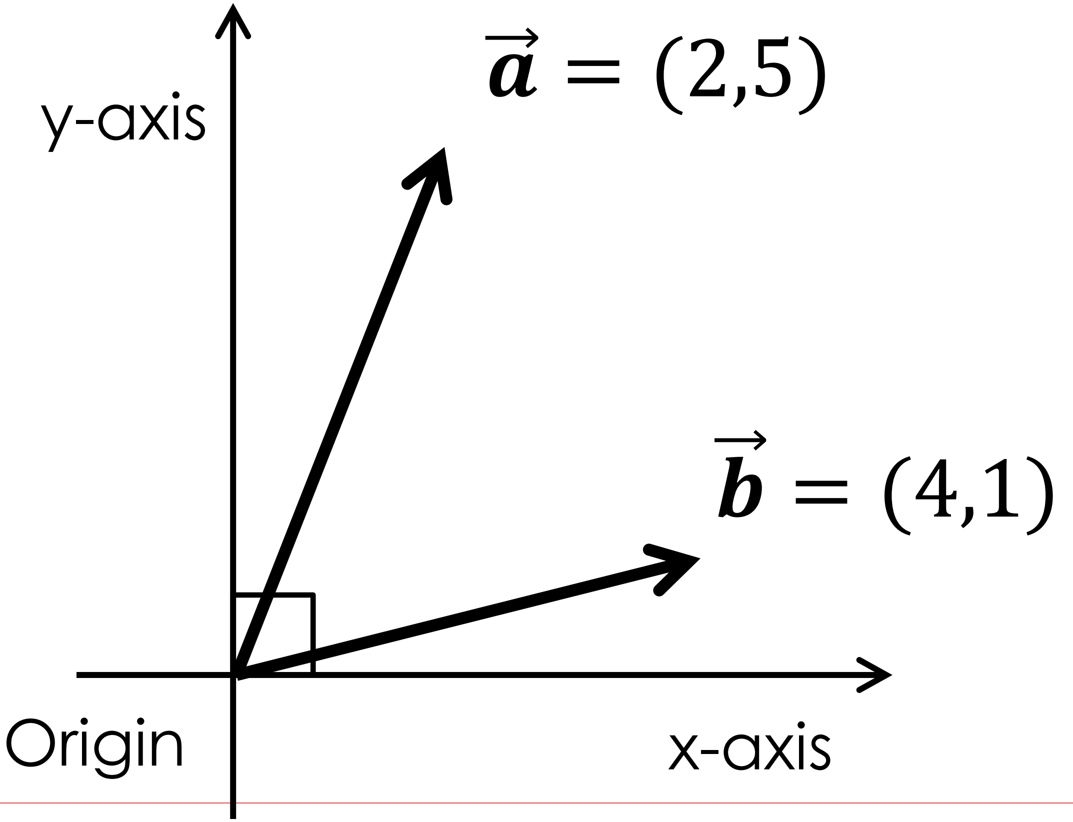
Dot Product

- $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = a_1 b_1 + a_2 b_2$



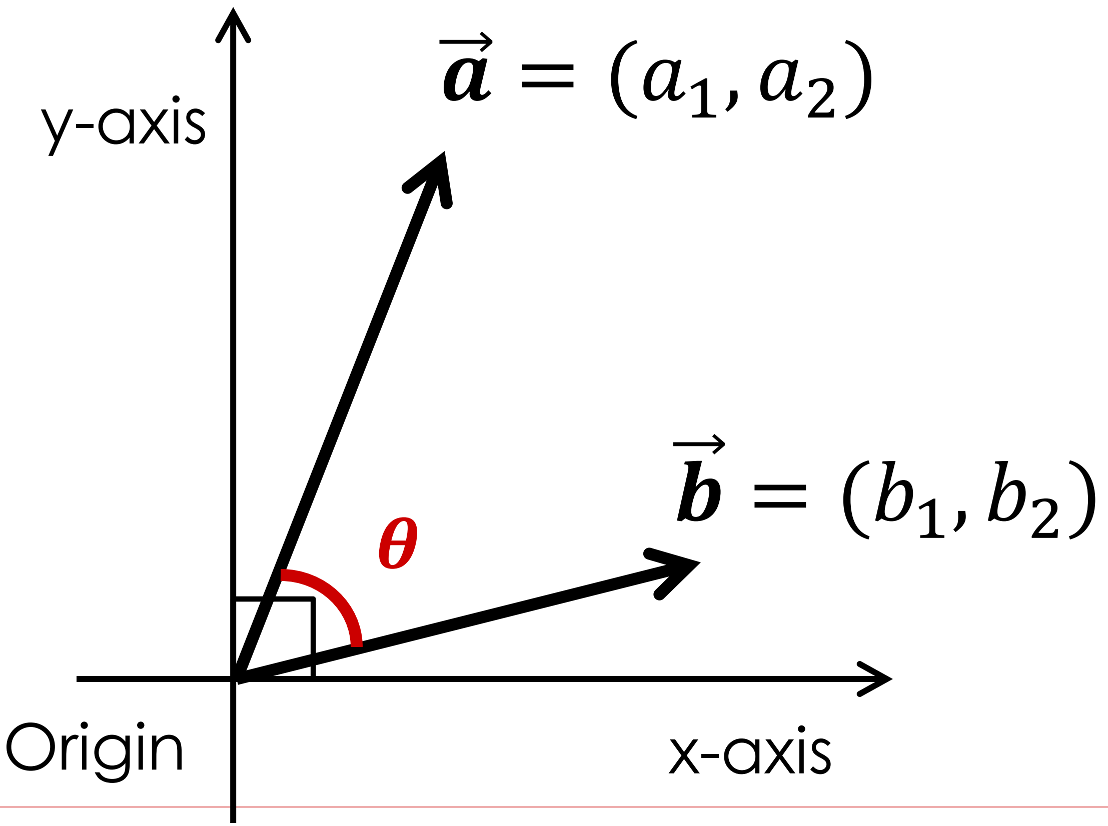
Dot Product

- $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = a_1 b_1 + a_2 b_2$
  - $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = ?$



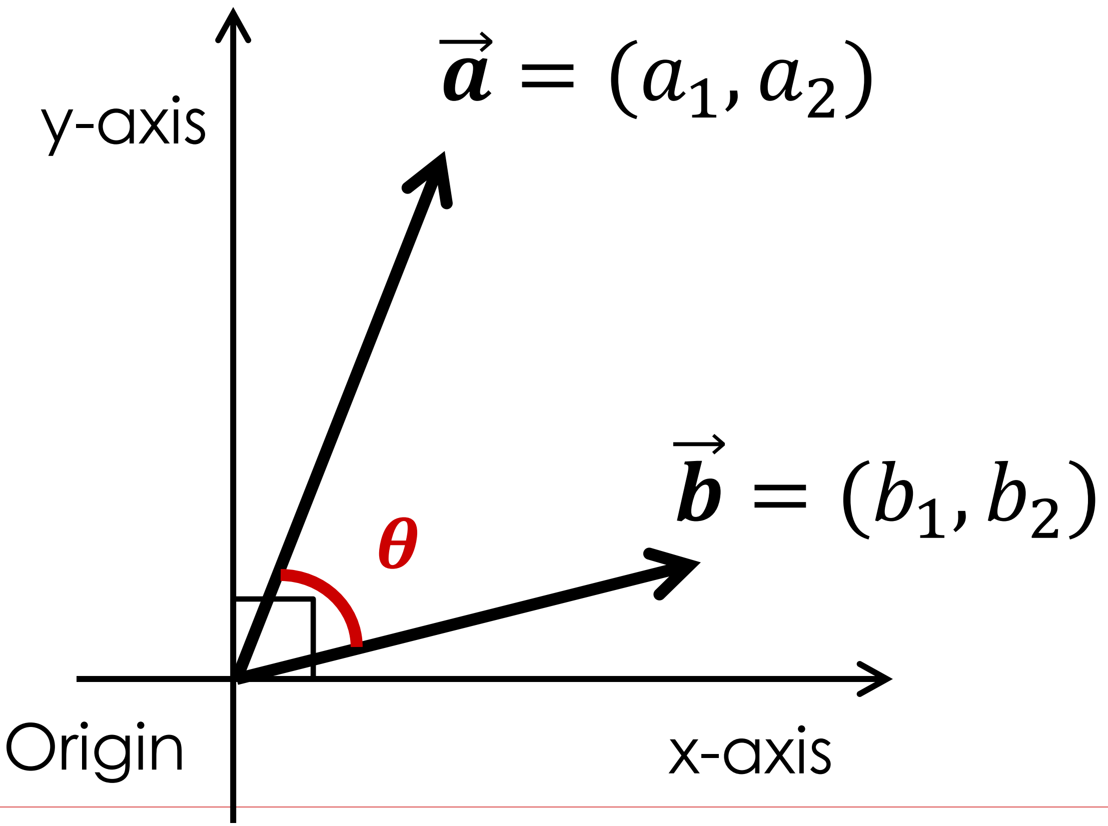
Dot Product

- How can we get the angle?



Angle between Two Vectors

- $\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}} = a_1 b_1 + a_2 b_2 = \|\vec{\boldsymbol{a}}\| \|\vec{\boldsymbol{b}}\| cos\theta$



Angle between Two Vectors

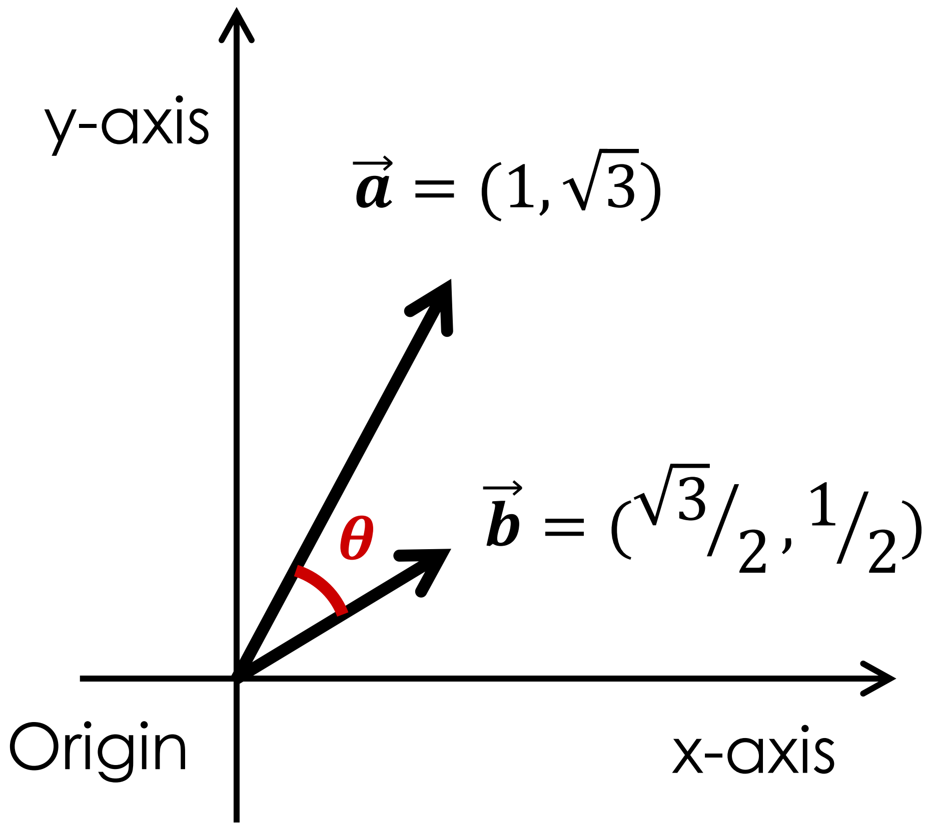
- $cos\theta = \frac{\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}}}{\|\vec{\boldsymbol{a}}\| \|\vec{\boldsymbol{b}}\|}$

- $\theta = arccos \frac{\vec{\boldsymbol{a}} \cdot \vec{\boldsymbol{b}}}{\|\vec{\boldsymbol{a}}\| \|\vec{\boldsymbol{b}}\|}$

Angle between Two Vectors

- $cos\theta = \frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|}$
- $\theta = arccos\frac{\vec{a} \cdot \vec{b}}{\|\vec{a}\|\|\vec{b}\|}$

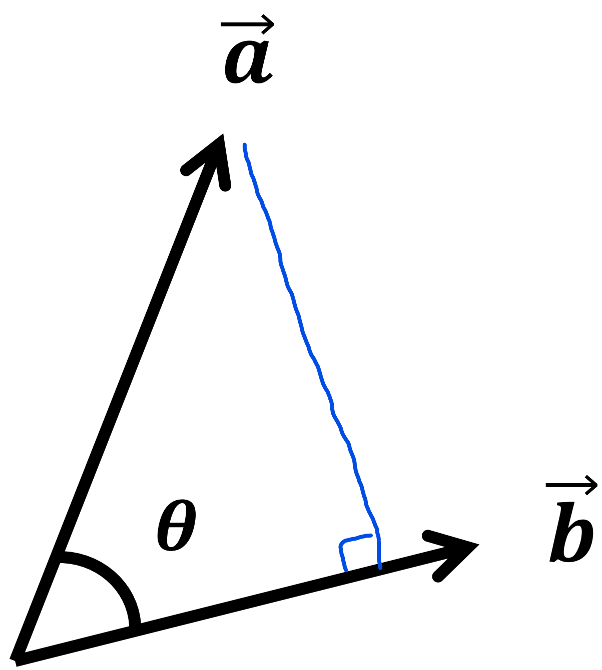
$\boldsymbol{\theta} = ?$



정사영이죠?

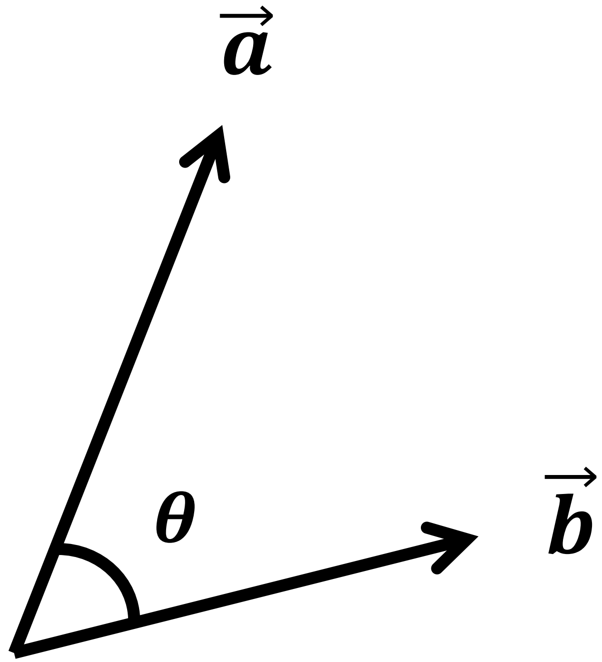
Project One Vector onto Another

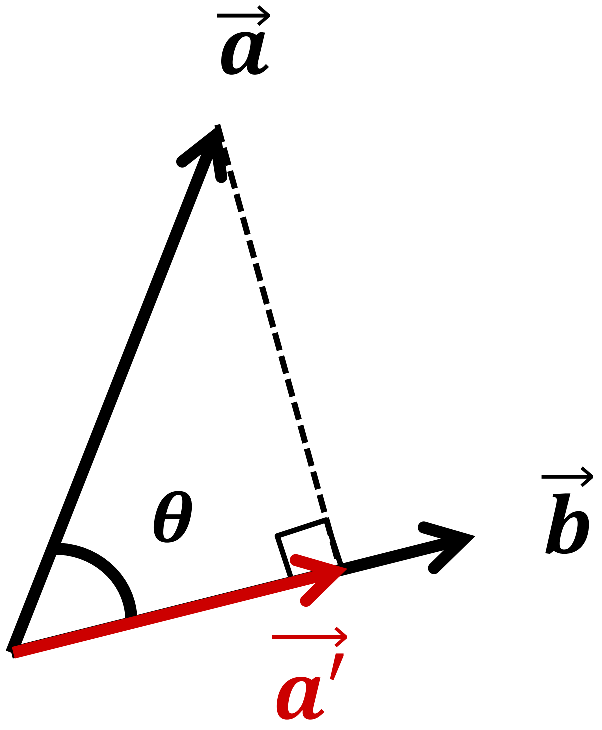
- $\overrightarrow{a'}$ = projection of $\vec{a}$ onto $\vec{b}$



Project One Vector onto Another

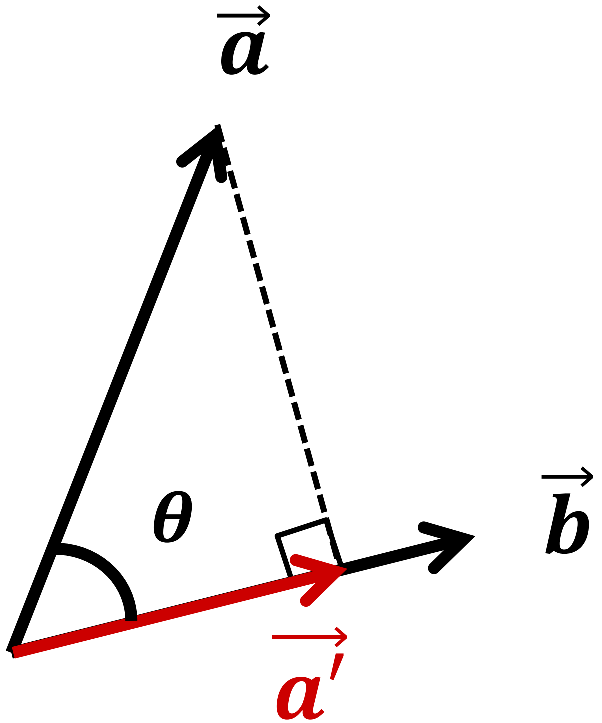
- $\overrightarrow{a'}$ = projection of $\vec{a}$ onto $\vec{b}$





Project One Vector onto Another

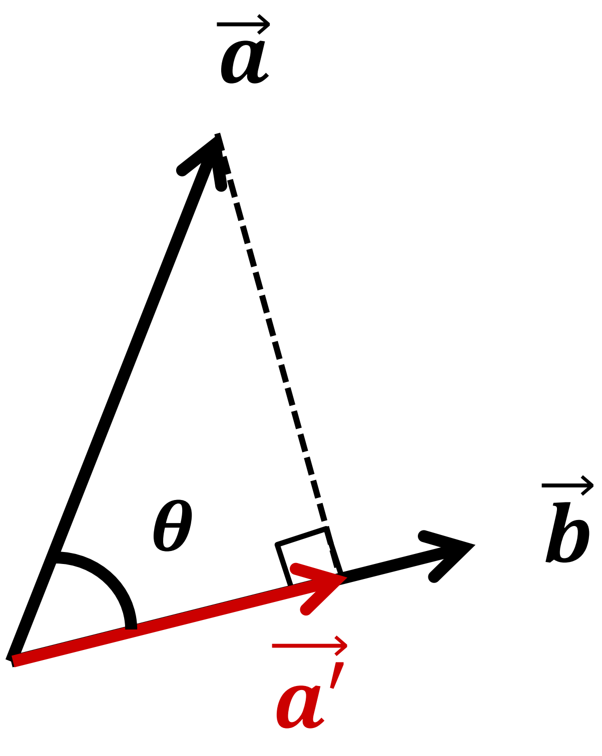
- $\overrightarrow{a'}$ = projection of $\vec{a}$ onto $\vec{b}$
  - **Length** of $\overrightarrow{a'}$ = $\|\vec{a}\| cos\theta$



Project One Vector onto Another

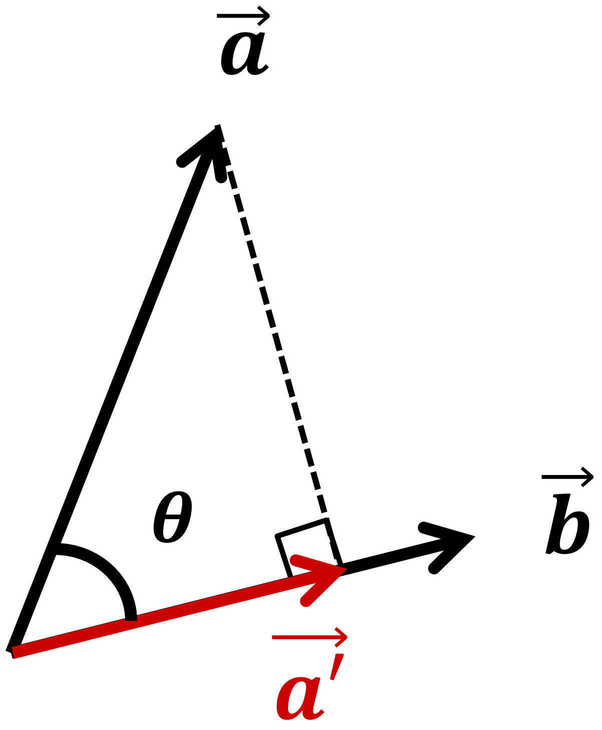
- $\overrightarrow{a'}$ = **projection of** $\vec{a}$ **onto** $\vec{b}$
  - **Length** of $\overrightarrow{a'} = \|\vec{a}\| cos\theta$

$$= \|\vec{a}\| \frac{\vec{a}\cdot\vec{b}}{\|\vec{a}\|\|\vec{b}\|} = \frac{\vec{a}\cdot\vec{b}}{\|\vec{b}\|}$$



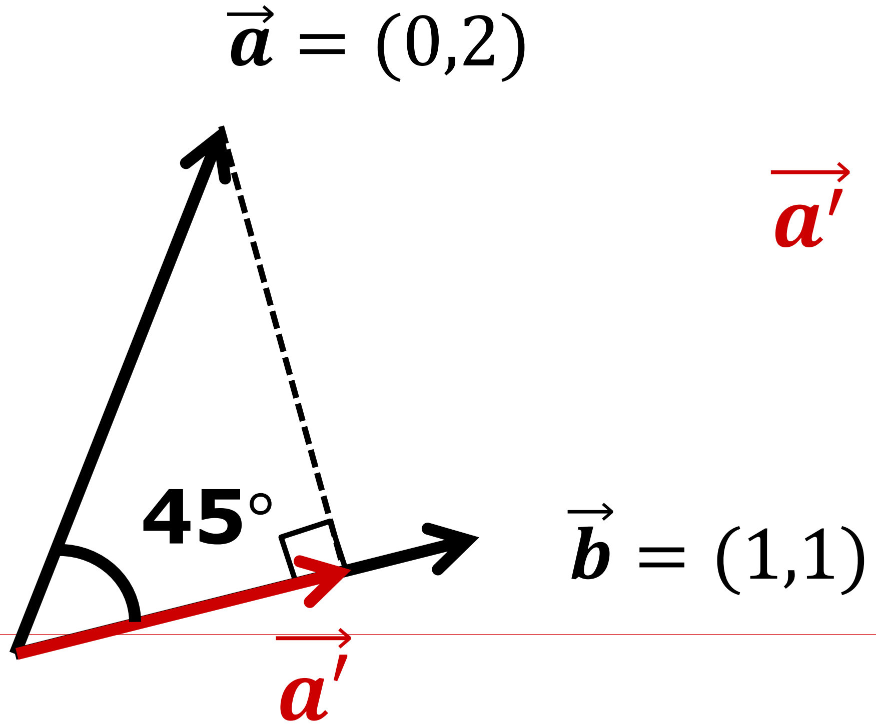
Project One Vector onto Another

- $\overrightarrow{a'}$ = projection of $\vec{a}$ onto $\vec{b}$
  - **Length** of $\overrightarrow{a'} = \frac{\vec{a} \cdot \vec{b}}{\|\vec{b}\|}$
  - **Direction** of $\overrightarrow{a'} = \frac{\vec{b}}{\|\vec{b}\|}$



Project One Vector onto Another

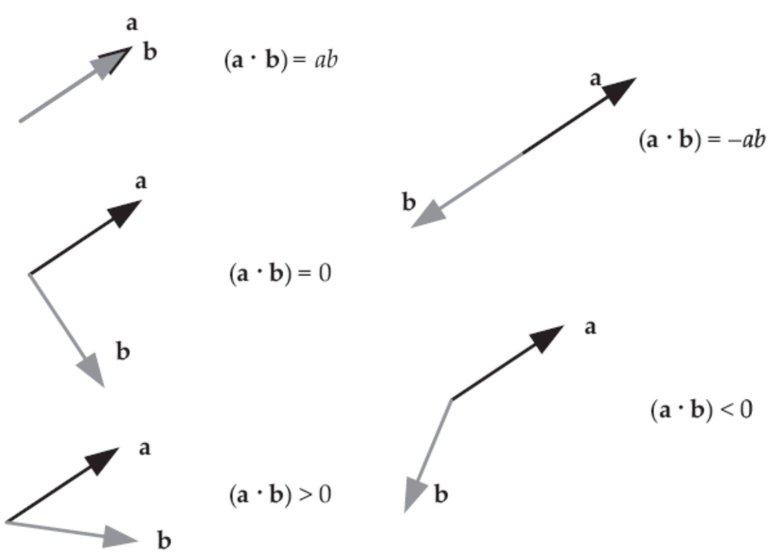
- $\overrightarrow{a'} = \frac{\vec{a}\cdot\vec{b}}{\|\vec{b}\|} \cdot \frac{\vec{b}}{\|\vec{b}\|} = \frac{\vec{a}\cdot\vec{b}}{\|\vec{b}\|^2} \cdot \vec{b} = \frac{\vec{a}\cdot\vec{b}}{\vec{b}\cdot\vec{b}} \cdot \vec{b}$



$\overrightarrow{a'} = (?\,,?\,)$

Dot Product in Practice

- **Given two unit vectors $\vec{a}$ and $\vec{b}$, they are:**
  - Collinear, if $\vec{a} \cdot \vec{b} = 1$
  - Collinear but opposite, if $\vec{a} \cdot \vec{b} = -1$
  - Perpendicular, if $\vec{a} \cdot \vec{b} = 0$
  - In the same direction, if $\vec{a} \cdot \vec{b} > 0$
  - In the opposite direction, if $\vec{a} \cdot \vec{b} < 0$



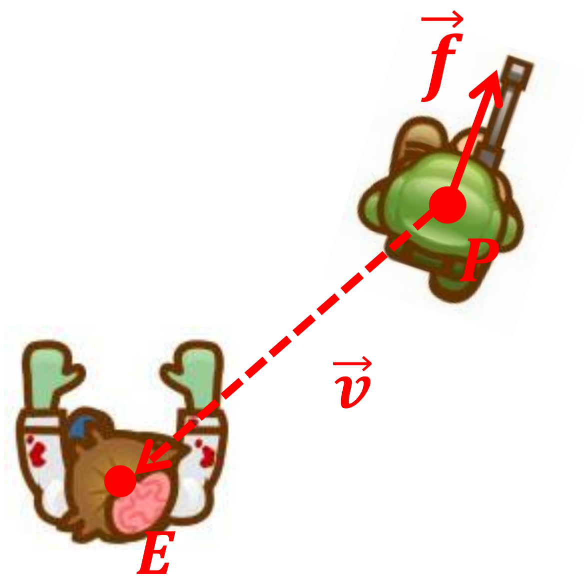
Dot Product in Practice

- Checking if an enemy at $\boldsymbol{E}$ is in front of or behind the player character at $\boldsymbol{P}$ facing in $\vec{f}$ direction:

Dot Product in Practice

- Checking if an enemy at E is in front of or behind the player character at P facing in $\vec{f}$ direction:

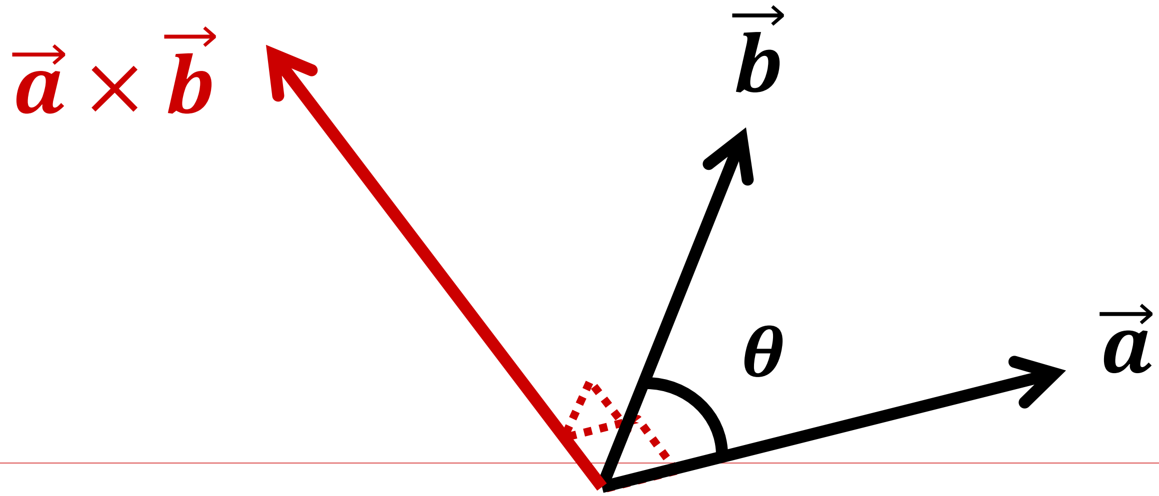
$$\mathrm{d} = \vec{v} \cdot \vec{f} \ (\vec{v} = E - P)$$



Cross Product

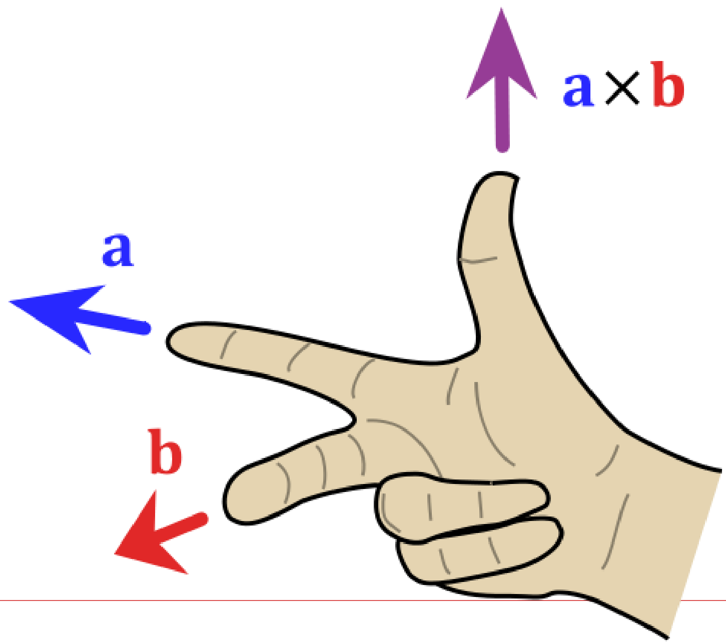
- $\vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}} = \|\vec{\boldsymbol{a}}\| \|\vec{\boldsymbol{b}}\| sin\theta \cdot \vec{\boldsymbol{n}}$
  - $\vec{\boldsymbol{n}}$ is a unit vector that is perpendicular to both $\vec{\boldsymbol{a}}$ and $\vec{\boldsymbol{b}}$

(Only defined in 3D Euclidean space)



Direction: Handedness

- Under right-hand rule, the direction of $\vec{a} \times \vec{b}$ is the direction of thumb of right hand when index and middle fingers coincide with $\vec{a}$ and $\vec{b}$

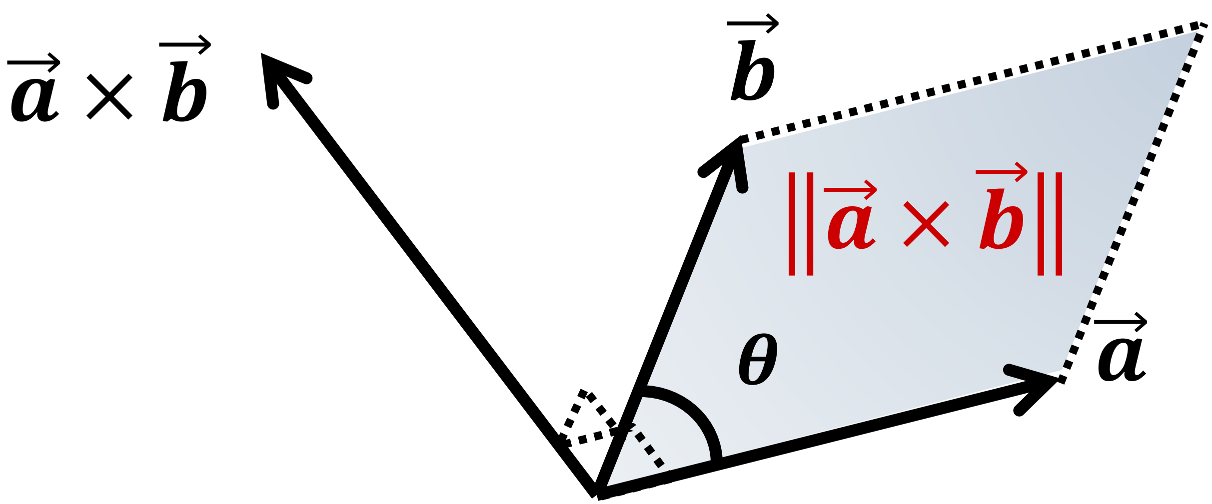


Magnitude:
Area of Parallelogram

- $\vec{a} \times \vec{b} = \|\vec{a}\|\|\vec{b}\| sin\theta \cdot \vec{n}$

느아? 두 벡터가 이루는 평행사변형의 넓이 = 외적

기하학적인 정의죠
좌표계를 상정하지 않고



Computation

- Coordinate notation

외적 많이 했었죠?

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}.$$

- Matrix determinant

Det (행렬식 많이 했었죠?) 선대 무려 B+

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \mathbf{k}$$

Computation

- **Ex 1)** $\vec{\boldsymbol{a}} = (1,0,0), \vec{\boldsymbol{b}} = (0,1,0)$, $\vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}} = ?$

이런건 기하학적인 의미를 알기 어려우니
아래처럼 결과를 직접 연산해보는거죠

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}.$$

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \mathbf{k}$$

Computation

- Ex 2) $\vec{\boldsymbol{a}} = (3,-3,1), \vec{\boldsymbol{b}} = (4,9,2)$, $\vec{\boldsymbol{a}} \times \vec{\boldsymbol{b}} = ?$

$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}.$$

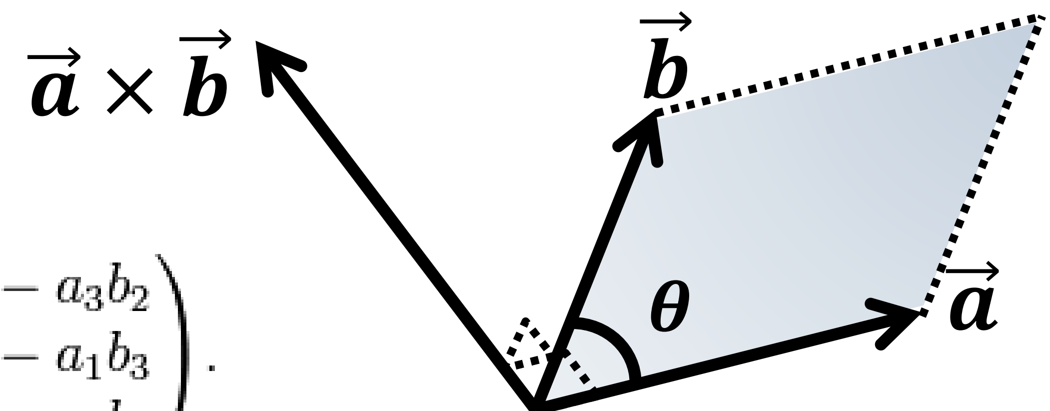
$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} a_2 & a_3 \\ b_2 & b_3 \end{vmatrix} \mathbf{i} - \begin{vmatrix} a_1 & a_3 \\ b_1 & b_3 \end{vmatrix} \mathbf{j} + \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \mathbf{k}$$

Computation

- **Ex 3) Given $\vec{a} = (3, -3, 1)$ and $\vec{b} = (4, 9, 2)$, calculate the area of the parallelogram spanned by the vectors $\vec{a}$ and $\vec{b}$?**

요런 문제같은게 나올수있겠죠?

2개의 벡터를 주고, 얘네가 이루는 평행사변형의 넓이를 구하시오
= 그냥 2개 외적하세요

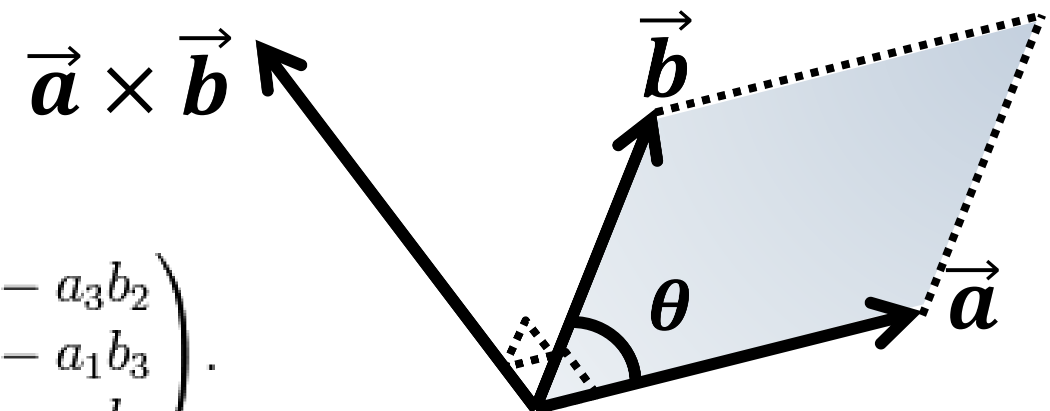


$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}.$$

Computation

- Ex 4) Given $\vec{a} = (3, -3, 1)$ and $\vec{b} = (-12, 12, -4)$, calculate the area of the parallelogram spanned by the vectors $\vec{a}$ and $\vec{b}$?

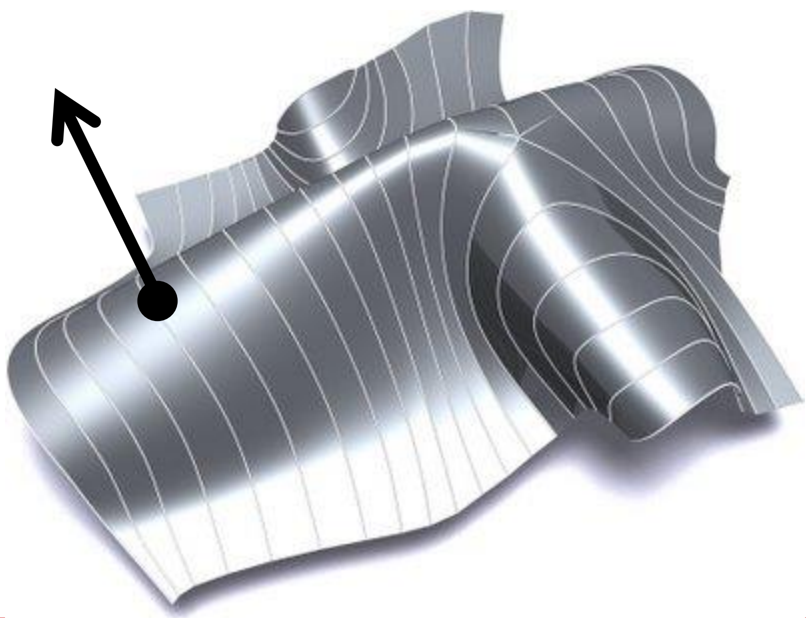
2개의 벡터가 평행하죠?
그럼 넓이가 0이 되죠
연산 안하고도 풀 수 있음



$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix}.$$

Normal Vector

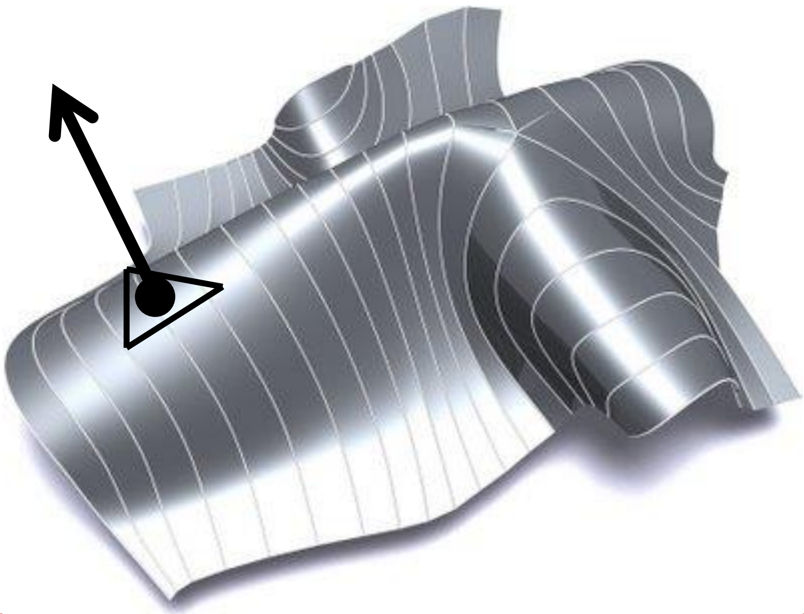
- In computer graphics, we often need to get a unit vector that is perpendicular to a curved surface at a point



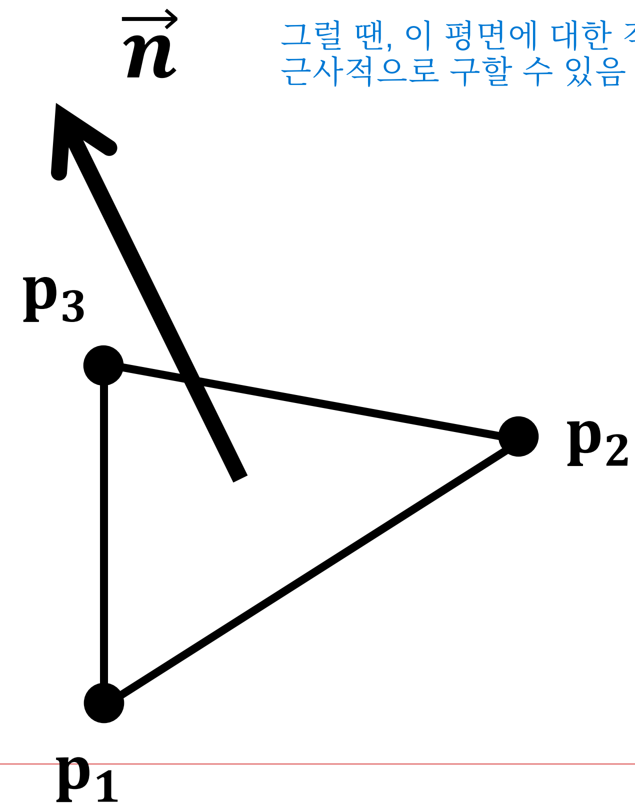
Normal Vector

- In polygonal model, it can be approximated as a plane normal

쉬운 케이스임
수학적으로 잘 정의된 곡면이 아니고?
폴리곤 메쉬일때 : 진정한 의미의 곡면이 아닐때 : 사실상 다면체?임

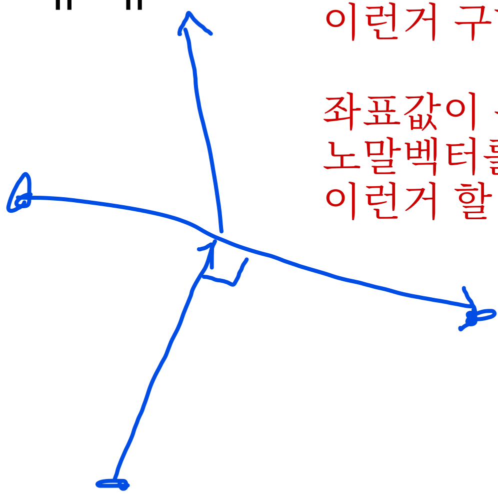


그럴 땐, 이 평면에 대한 직교 벡터로
근사적으로 구할 수 있음



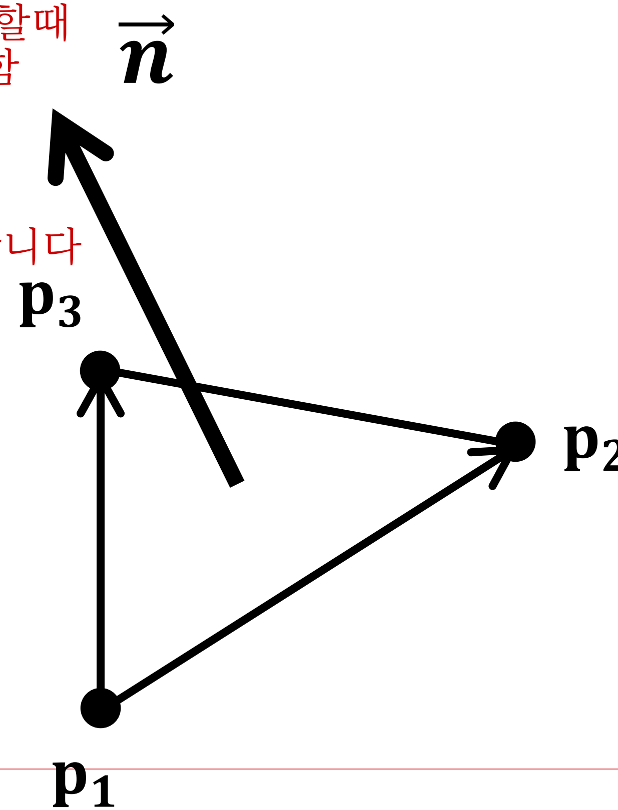
Normal Vector

- $\vec{N} = \overrightarrow{(\mathbf{p}_2 - \mathbf{p}_1)} \times \overrightarrow{(\mathbf{p}_3 - \mathbf{p}_1)}$
- $\vec{n} = {}^{\vec{N}}\!/_{\|\vec{N}\|}$



매쉬를 직접 만들어야할때
이런거 구할줄알아야함

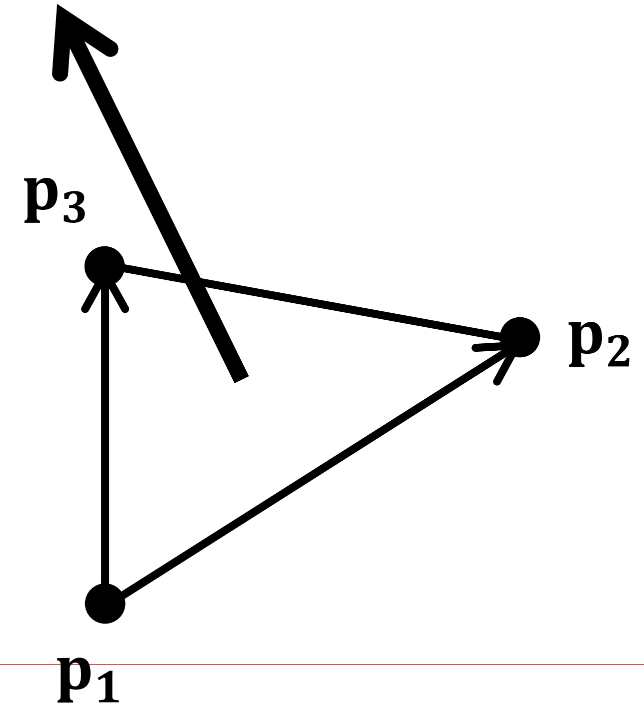
좌표값이 구해졌을때
노말벡터를 구해봐
이런거 할 줄 알아야합니다



Normal Vector

- $\vec{N} = \overrightarrow{(\mathbf{p}_2 - \mathbf{p}_1)} \times \overrightarrow{(\mathbf{p}_3 - \mathbf{p}_1)}$
- $\vec{n} = {\vec{N}}/{\|\vec{N}\|}$
  - $\mathbf{p}_1 = (\mathbf{0}, \mathbf{0}, \mathbf{0})$
  - $\mathbf{p}_2 = (\mathbf{1}, \mathbf{1}, \mathbf{1})$
  - $\mathbf{p}_3 = (\mathbf{0}, \mathbf{0}, \mathbf{1})$

$\vec{n} = ?$



$$\begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2b_3 - a_3b_2 \\ a_3b_1 - a_1b_3 \\ a_1b_2 - a_2b_1 \end{pmatrix}.$$