

게임 수학 강의 노트

강영민

동명대학교

소개

강의소개

강의개요

- 교재명: 게임프로그래머를 위한 수학과 OpenGL 프로그래밍
- 저자 - 강영민
- 출판사 - 도서출판 GS인터비전.
- 강의방식
 - 강의식 진행
 - 수시고사 10회
 - 프로그래밍 구현 실습

강의 목표

게임과 같은 3차원 콘텐츠를 설계하고 제작하는 데에 필요한 기본적인 수학 지식을 습득한다.

무엇을 다루나

주제	내용
벡터	벡터의 개념과 연산
행렬	행렬의 기학적 개념 이해와 응용
변환	변환의 개념 및 행렬 표현 이해
사원수	사원수를 이용한 변환 개념 이해
카메라	카메라 투영 행렬의 이해
조명과 재질	조명 계산 모델의 이해
충돌	기하 객체의 충돌을 감지하는 방법 이해

마음의 준비

- 수식에 겁 먹지 말자
 - $\frac{1}{n} \sum_{i=1}^n x_i$ 은 그냥 $\frac{x_1+x_2+x_3+x_4+\cdots+x_n}{n}$ 이다.
- 거의 매주 시험을 친다. 시험 문제는 언제나 미리 알려 준다.
 - 결석은 출석 점수를 못 받아서 무서운 것이 아니라, 시험 문제를 알지 못하게 되어 무서운 것이다.
- 출석인정에 대하여
 - 사정이 있으면 결석을 하라. 세상에는 수업보다 중요한 일이 많다.
 - 선택의 책임은 본인 것이다. 출석 인정은 없다. 세상은 그런 곳이다.
 - 세상에 안 되는 일은 없다. 필요한 경우 진지하게 상담요청을 하라.

벡터

벡터란 무엇인가?

벡터란 무엇인가?

- 벡터의 의미

- 벡터(vector)는 ‘나르다’라는 의미의 라틴어 동사 ‘vehere’에서 유래
- ‘무엇인가를 나르는 것’이라는 의미
- 벡터라는 것은 무엇인가를 옮겨 놓는 역할을 수행한다.

- 수학과 물리학에서의 개념

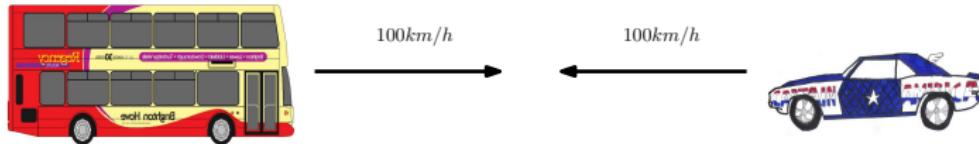
- 크기와 방향으로 결정되는 양(量, quantity)
- 방향량(方向量)이라고도 함.
- 예: 힘(force)은 크기만으로는 그 성질을 온전히 표현할 수 없고, 방향도 같이 고려해야 하므로 벡터로 표현된다.

벡터

수학자들은 자연 현상의 많은 것들이 수로 표현될 수 있음을 알았는데, 하나의 숫자로 충분히 표현할 수 있지만 하나 이상의 수가 필요한 경우도 있다. 이러한 양을 벡터(vector)라고 부른다.

벡터의 개념

- 물리적 현상 등을 표현할 때 - 대상을 양(量, quantity)으로 표현
- 이 양은 스칼라(scalar) 혹은 벡터(vector)
 - 스칼라 값은 오로지 크기만으로 완전히 그 양을 표현할 수 것으로서 물체의 질량, 소요된 시간, 길이, 열량 등이 해당
 - 벡터(vector)는 이와 달리 크기와 함께 방향도 같이 존재하는 양으로 힘, 속도, 변위와 같은 양이 바로 여기에 해당
- 속도(速度, velocity)와 속력(速力, speed)
- 속도는 벡터, 속력은 스칼라



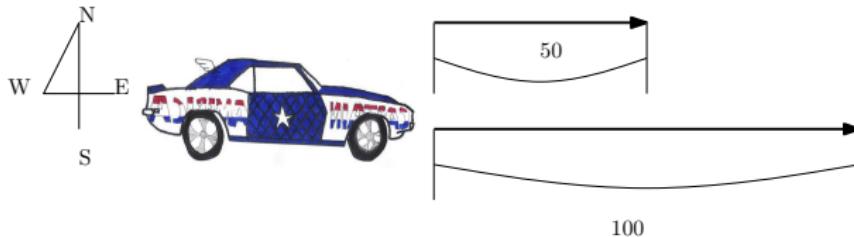
동일한 속력으로 서로 마주 보며 달리는 차량의 속도

화살표를 이용한 벡터 표현

- 벡터를 표현하는 가장 직관적인 방법은 화살표를 이용
- 화살표: 시점(始點)과 종점(終點)으로 구성
- 화살표의 방향은 벡터의 방향을 시각적으로 표현하고, 화살표의 길이는 벡터의 크기를 시각적으로 표현



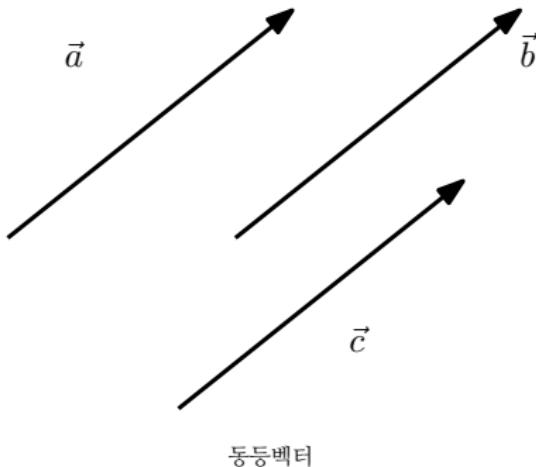
벡터의 시각적 표현과 달리는 자동차 속도 표현의 예



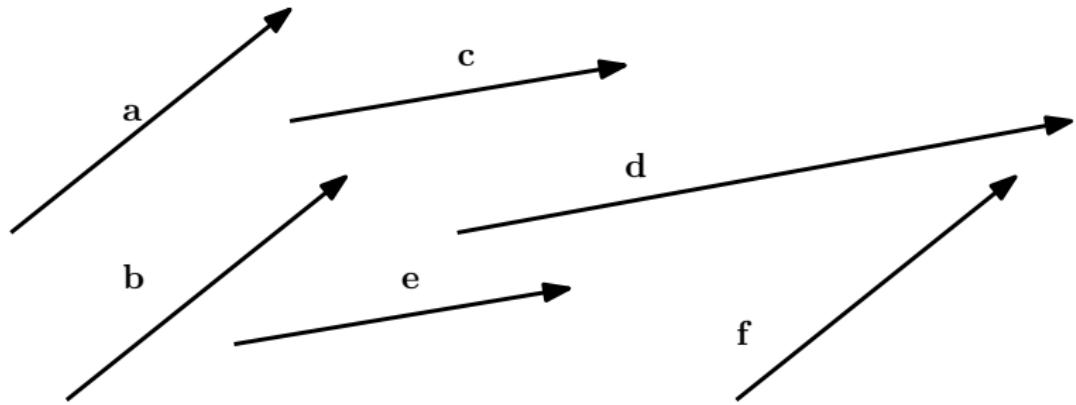
속력이 두 배로 늘어난 자동차의 속도

동등벡터(equivalent vector)

- 벡터의 표기법
 - \vec{a}, \mathbf{a}
- 동등벡터
 - 크기와 방향이 같으면 모두 동등한 벡터로 간주



동등벡터 찾기

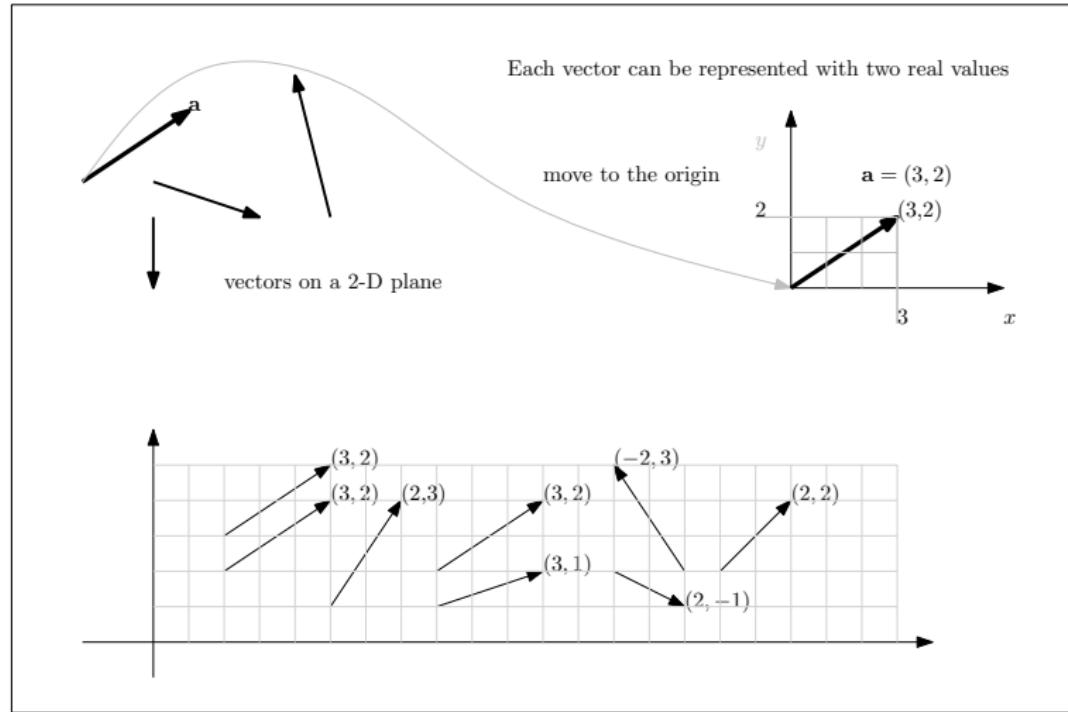


동등벡터 찾기

벡터의 수학적 표현

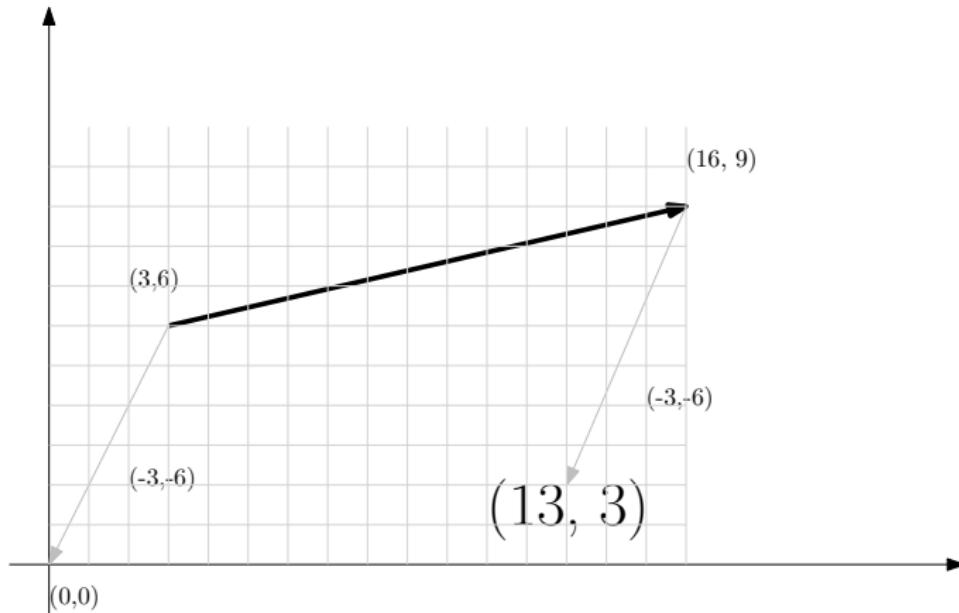
- 벡터: 화살표가 그려지는 공간의 차원(次元,dimension)에 따라 결정되는 개수의 성분
 - n -튜플(tuple)
 - $\mathbf{v} = (v_1, v_2, v_3, \dots, v_n)$
 - n 개의 차원을 가진 공간에서 그려지는 화살표 = n 차원 벡터

2차원 벡터



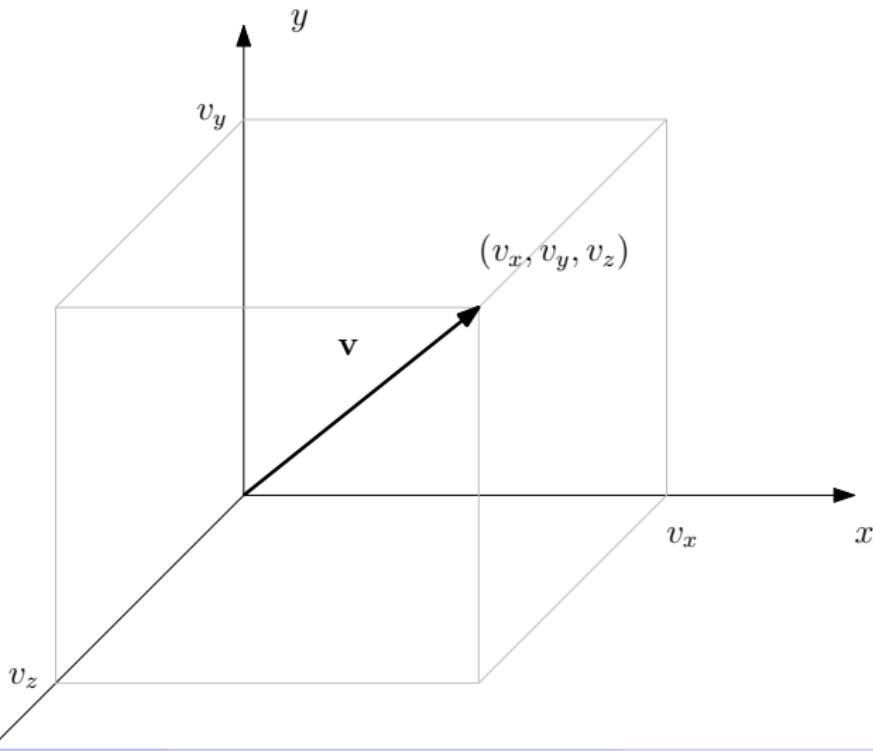
시점을 원점으로 옮기기

시점이 원점이 아닌 벡터는 시점을 원점으로 끌어 오면 된다. 벡터의 시점은 $(3,6)$ 의 좌표에 놓여있고, 끝점은 $(16,9)$ 이다. 시작점을 원점으로 옮기는 것은 $(-3,-6)$ 만큼의 이동을 하는 것이다. 따라서 끝점은 $(13,3)$ 의 위치로 이동하게 된다. 그러므로 이 벡터는 $(13,3)$ 으로 표현된다.

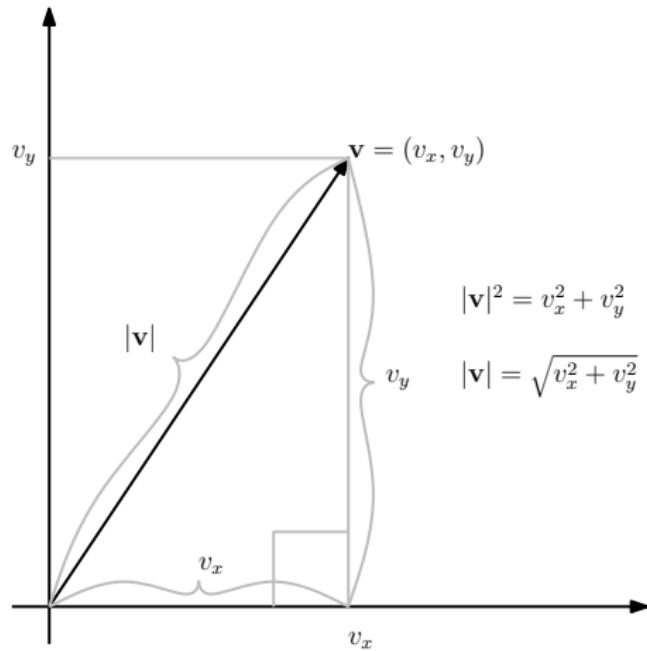


3차원 벡터

3차원 벡터는 지금까지 살펴본 2차원 벡터에 축(軸, axis)을 하나 더하기만 하면 된다.



벡터의 크기

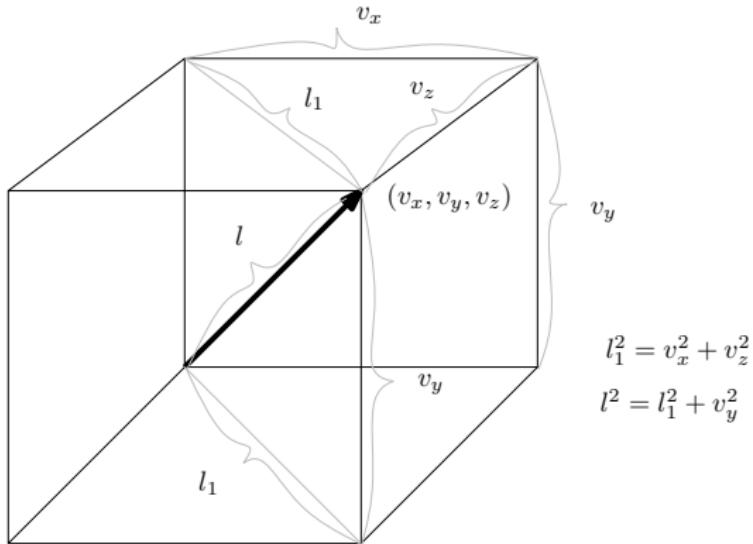


어떤 벡터 \mathbf{v} 가 (v_x, v_y) 로 표현될 때 이 벡터의 크기는 v_x 와 v_y 로 어떻게 구할 수 있을까? 그 값은 '길이'에 대한 상식적 정의에 따라, 스칼라 값이며 양(陽, positive)의 값이 된다. 이렇게 양의 길이(positive length)를 벡터에 할당하는 것을 놈(norm)이라 하며, 어떤 벡터 \mathbf{v} 의 놈은 $\|\mathbf{v}\|$ 로 표현한다. 유클리드 기하에 의해 얻어지는 길이는 Euclidean Norm이라고 한다. 이는 피타고라스의 정리를 이용하여 쉽게 구할 수 있다.

3차원 벡터의 크기

$$\mathbf{v} = (v_x, v_y, v_z)$$

$$\|\mathbf{v}\| = \sqrt{v_x^2 + v_y^2 + v_z^2}$$



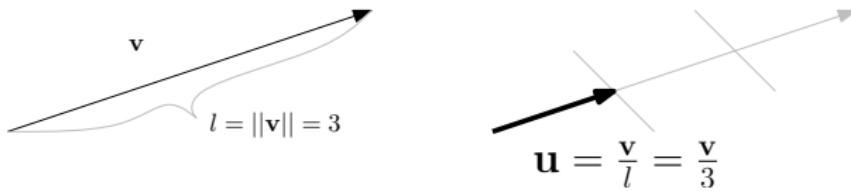
$$l_1^2 = v_x^2 + v_z^2$$

$$l^2 = l_1^2 + v_y^2$$

벡터의 정규화

- 정규화

- 단위 벡터는 길이가 1인 벡터.
- 어떤 벡터의 방향과 일치하는 단위벡터를 구하는 작업은 종종 많은 응용에서 필요.
- 이러한 작업은 벡터의 길이를 1로 만드는 것과 같다.
- 이를 정규화(normalization)이라고 한다.

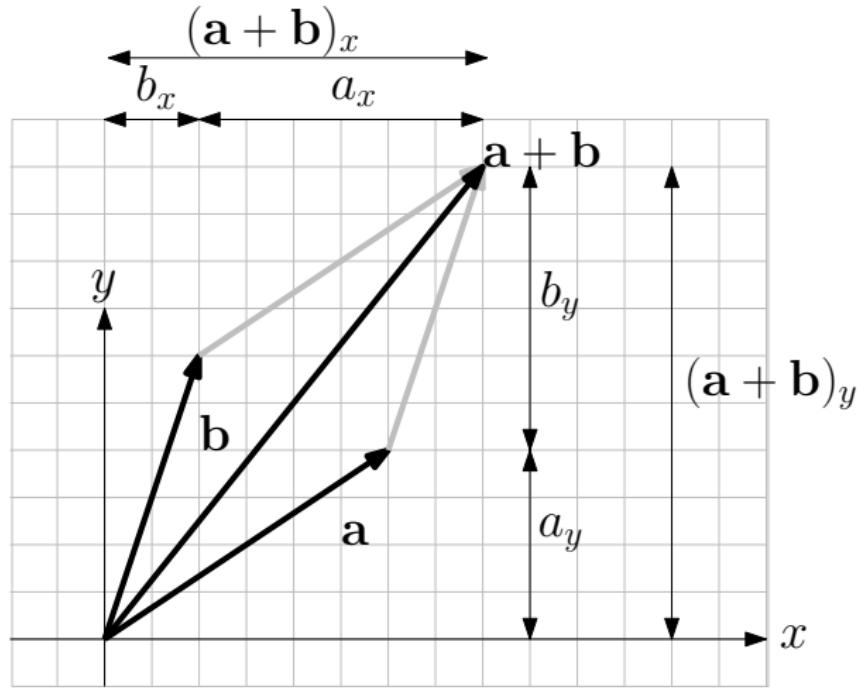


벡터

벡터 연산 심화

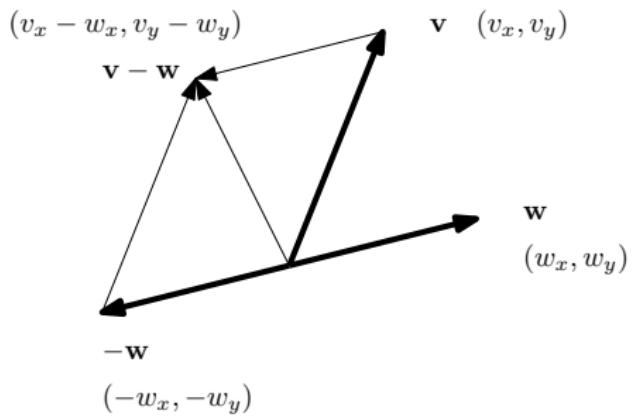
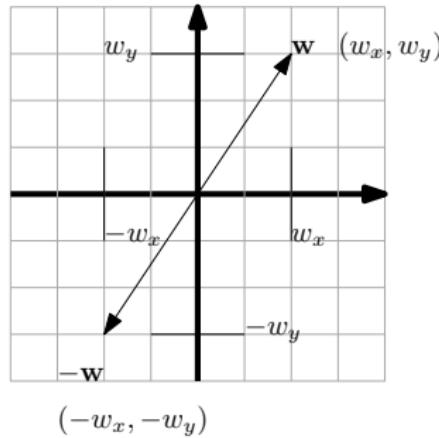
벡터의 덧셈

$$\mathbf{v} + \mathbf{w} = (v_x + w_x, v_y + w_y, v_z + w_z)$$



벡터의 뺄셈

$$\mathbf{v} - \mathbf{w} = (v_x - w_x, v_y - w_y, v_z - w_z)$$



벡터에 스칼라 곱하기

벡터는 크기만을 가진 스칼라와 곱할 수 있다. 어떤 스칼라 값 s 가 있다고 하자, 이 스칼라 값과 벡터 $\mathbf{v} = (v_x, v_y, v_z)$ 를 곱한 $s\mathbf{v}$ 는 다음과 같다.

$$s\mathbf{v} = (sv_x, sv_y, sv_z)$$

벡터의 기본적인 연산 규칙

$$\mathbf{a} + \mathbf{b} = \mathbf{b} + \mathbf{a}$$

$$(\mathbf{a} + \mathbf{b}) + \mathbf{c} = \mathbf{a} + (\mathbf{b} + \mathbf{c})$$

$$\mathbf{a} + \vec{0} = \vec{0} + \mathbf{a} = \mathbf{a}$$

$$\mathbf{a} + (-\mathbf{a}) = \mathbf{a} - \mathbf{a} = \vec{0}$$

$$(k + l)\mathbf{a} = k\mathbf{a} + l\mathbf{a}$$

$$(kl)\mathbf{a} = k(l\mathbf{a})$$

$$1\mathbf{a} = \mathbf{a}$$

$$0\mathbf{a} = \vec{0}$$

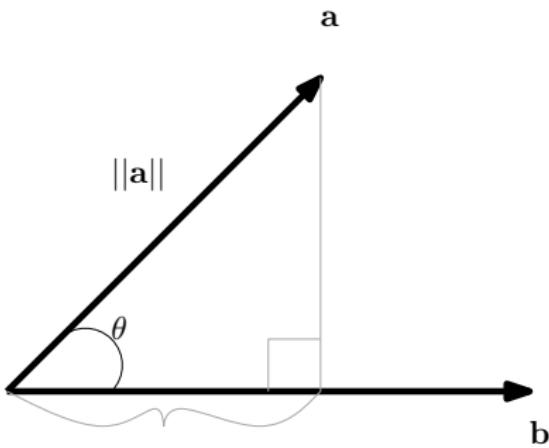
$$(-1)\mathbf{a} = -\mathbf{a}$$

벡터의 스칼라 곱, 혹은 내적(dot product)

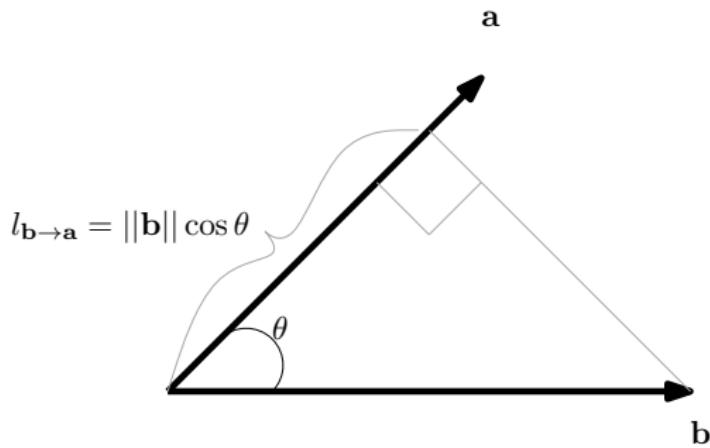
- 내적

- 스칼라 곱(scalar product)라고도 부름
- 두 개의 벡터를 피연산자(operand)로 하는 이항 연산(binary operator)로서 그 결과가 스칼라 값
- 두 벡터 \mathbf{a} 와 \mathbf{b} 의 내적은 $\mathbf{a} \cdot \mathbf{b}$ 로 표현
- 두 벡터가 이루는 사잇각이 θ 라고 하며, 내적의 크기는 다음과 같다.
 - $\mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \cos \theta$
- 실제 계산 방법
 - $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$
 - $\mathbf{a} \cdot \mathbf{b} = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n = \sum_{i=1}^n a_i b_i$

벡터 내적의 의미



$$l_{\mathbf{a} \rightarrow \mathbf{b}} = ||\mathbf{a}|| \cos \theta$$



$$l_{\mathbf{a} \rightarrow \mathbf{b}} = (||\mathbf{a}|| ||\mathbf{b}|| \cos \theta) / ||\mathbf{b}||$$

$$l_{\mathbf{a} \rightarrow \mathbf{b}} = \mathbf{a} \cdot \mathbf{b} / ||\mathbf{b}||$$

$$l_{\mathbf{b} \rightarrow \mathbf{a}} = (||\mathbf{a}|| ||\mathbf{b}|| \cos \theta) / ||\mathbf{a}||$$

$$l_{\mathbf{b} \rightarrow \mathbf{a}} = \mathbf{a} \cdot \mathbf{b} / ||\mathbf{a}||$$

벡터 내적의 활용

- 코사인 함수의 특성을 통해 간단히 얻어지는 사실
 - $\theta = 0 \Rightarrow \cos \theta = 1, \mathbf{a} \cdot \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\|$
 - $\theta = \pi/2 \Rightarrow \cos \theta = 0, \mathbf{a} \cdot \mathbf{b} = 0$
 - $\mathbf{a} \cdot \mathbf{a} = \|\mathbf{a}\|^2$
- 벡터를 이용하여 각도를 계산하거나 투영을 계산하는 데에 널리 사용
 - $\mathbf{v} \cdot \mathbf{w} = \|\mathbf{v}\| \|\mathbf{w}\| \cos \theta$
 - $\cos \theta = \frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\| \|\mathbf{w}\|}$
 - $\theta = \cos^{-1} \frac{\mathbf{v} \cdot \mathbf{w}}{\|\mathbf{v}\| \|\mathbf{w}\|}$
 - $\theta = \cos^{-1} \frac{v_x w_x + v_y w_y + v_z w_z}{\sqrt{v_x^2 + v_y^2 + v_z^2} \sqrt{w_x^2 + w_y^2 + w_z^2}}$

벡터 내적의 활용

예제

어떤 두 벡터가 각각 $(3,2)$ 와 $(4,1)$ 이라고 하자. 두 벡터가 이루는 각도를 구하라.

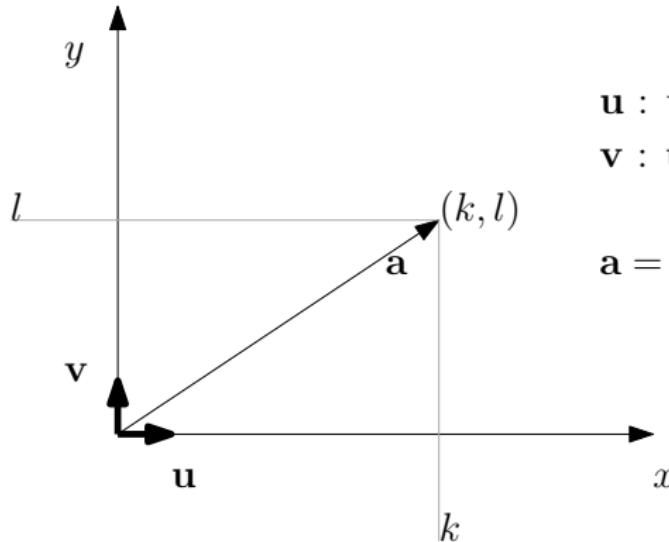
정답

두 벡터를 각각 \mathbf{v} 와 \mathbf{w} 로 표현하자. 두 벡터의 내적 $\mathbf{v} \cdot \mathbf{w}$ 는 $3 \cdot 4 + 2 \cdot 1$, 즉 14 이다. 각각의 길이는 $\|\mathbf{v}\| = \sqrt{9+4} = \sqrt{13}$ 과 $\|\mathbf{w}\| = \sqrt{16+1} = \sqrt{17}$ 이다. 따라서 두 벡터의 사이각은 다음과 같다.

$$\theta = \cos^{-1} \frac{14}{\sqrt{13}\sqrt{17}} = \cos^{-1} \frac{14}{\sqrt{221}} \simeq \cos^{-1} 0.94174191159484 \simeq 19.65^\circ$$

좌표축과 좌표 - 1/3 기본 의미

- $\mathbf{a} = (k, l)$ 로 표현된다는 것은 xy 좌표계에서 기저벡터가 되는 x 축 단위벡터를 \mathbf{u} 와 y 축 단위벡터를 \mathbf{v} 를 다음과 같이 합성한 것



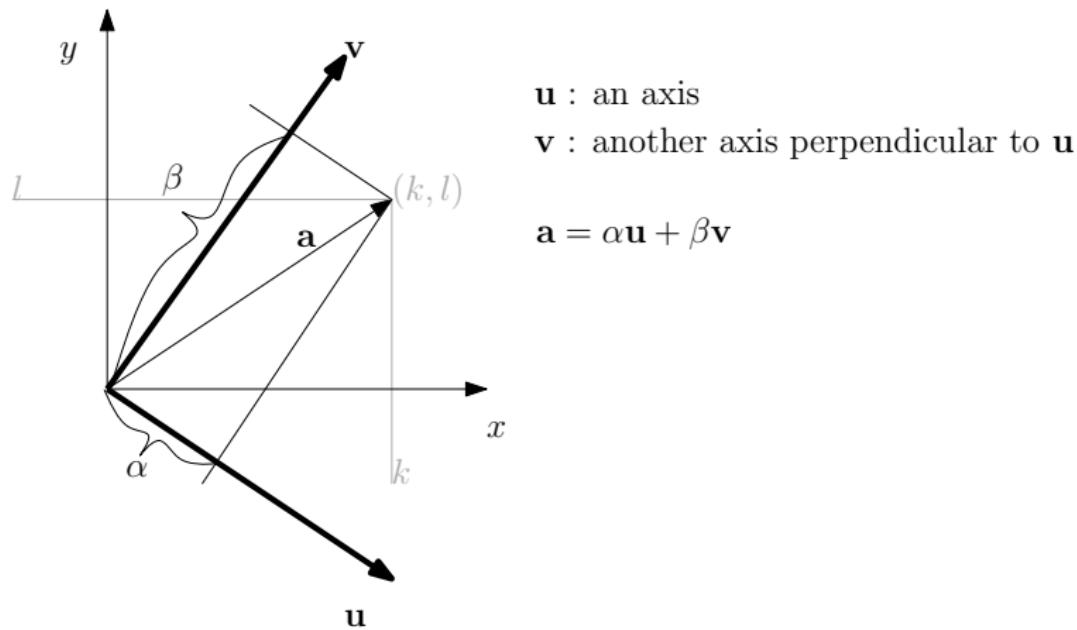
\mathbf{u} : unit vector along x axis

\mathbf{v} : unit vector along y axis

$$\mathbf{a} = k\mathbf{u} + l\mathbf{v}$$

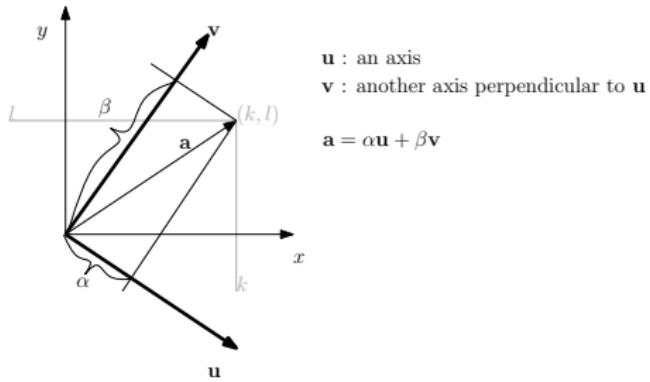
좌표축과 좌표 - 2/3 새로운 축의 정의

- 새로운 직교 좌표계를 고려해 보자. 여기서는 두 축이 \mathbf{u} 와 \mathbf{v}
- \mathbf{a} 의 \mathbf{u} 축 투영 길이 α , \mathbf{v} 축 투영 길이 β 계산
- 이 두 축을 기준으로 하는 좌표계에서는 \mathbf{a} 가 (α, β) 의 좌표로 표현됨



좌표축과 좌표 - 3/3 내적을 이용한 투영 길이 구하기

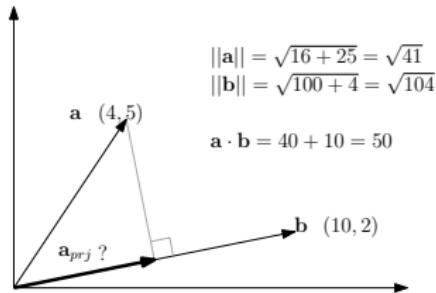
- \mathbf{a} 가 축 \mathbf{u} 와 \mathbf{v} 방향으로 가지는 길이 α 와 β 는 어떻게 구하나
 - 내적을 이용
 - α 는 \mathbf{a} 를 \mathbf{u} 방향으로 투영한 그림자의 길이 = $\mathbf{a} \cdot \mathbf{u} / \|\mathbf{u}\|$
 - 축은 단위 벡터로 표현하므로 $\|\mathbf{u}\| = 1$. 따라서 $\alpha = \mathbf{a} \cdot \mathbf{u}$
 - 비슷한 방법으로 $\beta = \mathbf{a} \cdot \mathbf{v}$
 - \mathbf{u} 와 \mathbf{v} 를 축으로 하는 좌표계에서 \mathbf{a} 는 $(\mathbf{a} \cdot \mathbf{u}, \mathbf{a} \cdot \mathbf{v})$



새로운 좌표축의 정의

예제

그림처럼 어떤 벡터 \mathbf{a} 가 $(4,5)$ 이고, 다른 벡터 \mathbf{b} 는 $(10,2)$ 라고 하자. 이때 벡터 \mathbf{a} 를 \mathbf{b} 위에 수직방향으로 내린 그림자가 되는 벡터 \mathbf{a}_{prj} 을 구하라.



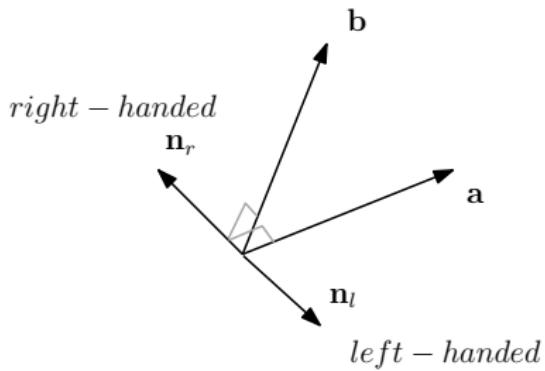
정답

$$l = \mathbf{a} \cdot \mathbf{b} / \|\mathbf{b}\|$$

$$\mathbf{a}_{prj} = l \tilde{\mathbf{b}} = \frac{50}{104} (10, 2) \simeq (4.8, 0.96)$$

벡터의 외적(外積) - 의미

- 벡터의 외적(cross product)
 - 벡터 곱(vector product): 두 벡터를 피연산자로 하는 이항연산으로 그 결과가 벡터
 - 벡터를 곱해 행렬을 얻는 외적(outer product)과 용어의 혼동이 있음.
여기서는 결과가 벡터인 곱
- 표현
 - 두 벡터 \mathbf{a} 와 \mathbf{b} 의 외적은 $\mathbf{a} \times \mathbf{b}$ 로 표현
 - 그 결과는 벡터이므로 $k\mathbf{n}$ (\mathbf{n} 은 \mathbf{a} 와 \mathbf{b} 에 동시에 수직인 단위벡터)
 - 동시에 수직인 벡터는 두 개가 존재. 좌표계에 의해 결정됨.

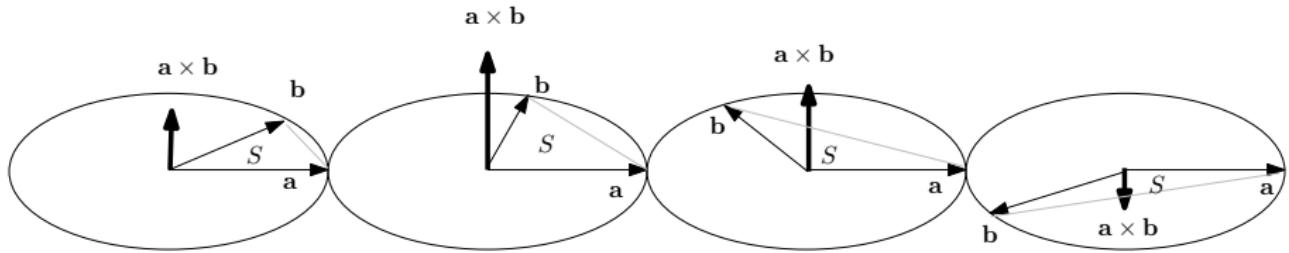


벡터의 외적(外積) 의미의 수학적 표현

- 두 벡터의 외적
 - 외적의 크기 k
 - 두 벡터의 크기와 사잇각의 사인(sine) 값에 비례
 - $k = \|\mathbf{a}\| \|\mathbf{b}\| \sin \theta$
 - 외적의 방향 \mathbf{n}
 - \mathbf{n} : 두 벡터에 수직인 방향 벡터
- 따라서 두 벡터의 외적은 다음과 같이 표현할 수 있다.
 - $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \sin \theta \mathbf{n}$

외적이 가진 의미

- 외적을 표현하는 식 $\mathbf{a} \times \mathbf{b} = \|\mathbf{a}\| \|\mathbf{b}\| \sin \theta \mathbf{n}$ 의 의미
 - 외적은 두 벡터에 동시에 수직한 벡터
 - 크기는 두 벡터가 수직일 때에 최대, 같은 방향이나 반대방향일 때 최소
 - 외적의 크기는 벡터 \mathbf{a} 와 \mathbf{b} 의 끝점을 연결한 삼각형의 넓이 S 에 비례
 - $\|\mathbf{a} \times \mathbf{b}\| = 2S$



외적의 계산

- 3차원 벡터의 외적을 구하기
 - $\mathbf{a} = (a_1, a_2, a_3), \mathbf{b} = (b_1, b_2, b_3)$
 - 두 벡터의 외적: $\mathbf{a} \times \mathbf{b} = (a_2b_3 - a_3b_2, a_3b_1 - a_1b_3, a_1b_2 - a_2b_1)$
- “행렬”을 이용한 곱셈으로 구하기
 - “반대칭(skew-symmetric 혹은 antisymmetric)” 행렬 이용

$$\mathbf{A}^* = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix}$$

$$\mathbf{A}^*\mathbf{b} = \begin{pmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{pmatrix} \mathbf{b} = (a_2b_3 - a_3b_2, a_3b_1 - a_1b_3, a_1b_2 - a_2b_1)$$

외적의 연산 법칙들

$$\mathbf{a} \times \mathbf{b} = -\mathbf{b} \times \mathbf{a}$$

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b}) \times \mathbf{c}$$

$$(\mathbf{a} + \mathbf{b}) \times \mathbf{c} = (\mathbf{a} \times \mathbf{c}) + (\mathbf{b} \times \mathbf{c})$$

$$k(\mathbf{a} \times \mathbf{b}) = k\mathbf{a} \times \mathbf{b} = \mathbf{a} \times k\mathbf{b}$$

$$\mathbf{a} \times \vec{0} = \vec{0} \times \mathbf{a} = \vec{0}$$

$$\mathbf{a} \times \mathbf{a} = \vec{0}$$

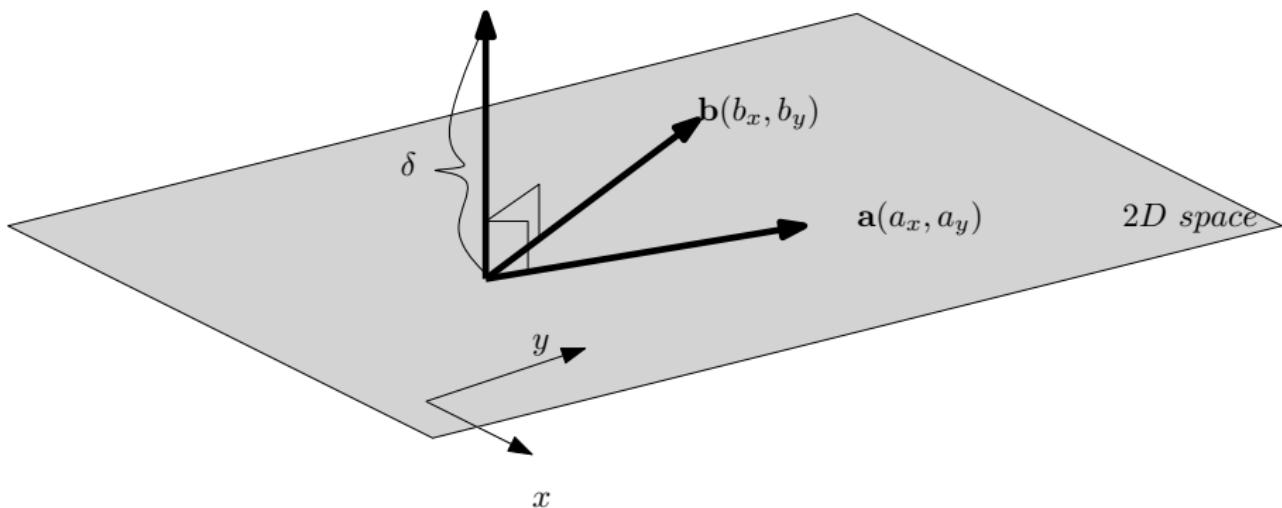
$$\mathbf{a} \cdot (\mathbf{a} \times \mathbf{b}) = \vec{0}$$

$$\mathbf{b} \cdot (\mathbf{a} \times \mathbf{b}) = \vec{0}$$

2차원 공간에서의 외적과 외적의 응용

- 2차원 공간의 두 벡터 $\mathbf{a} = (a_x, a_y)$ 와 $\mathbf{b} = (b_x, b_y)$ 의 외적은?
 - 그림의 회색 면은 2차원 공간의 일부
 - 두 벡터 \mathbf{a} 와 \mathbf{b} 의 외적은 2차원 공간 밖에서 정의
 - 축 z 가 필요하며, 이 z 축 성분으로만 표현

$$\mathbf{a} \times \mathbf{b} = (0, 0, \delta)$$



2차원 공간에서의 외적과 외적의 응용

- 2차원 벡터의 외적이 2차원 공간 밖에 정의가 되고, 이것은 3차원 벡터라고 볼 수 있다.
- 2차원 벡터 $\mathbf{a} = (a_x, a_y)$ 와 $\mathbf{b} = (b_x, b_y)$ 를 3차원 벡터로 가정
 - $\mathbf{a} = (a_x, a_y, 0)$
 - $\mathbf{b} = (b_x, b_y, 0)$
 - $\mathbf{a} \times \mathbf{b} = (0, 0, a_x b_y - a_y b_x)$
- z 성분의 값으로 알 수 있는 것들 (이를 δ 라고 하자)
 - $\delta > 0$ 인 경우는 \mathbf{b} 가 \mathbf{a} 의 진행 방향을 기준으로 왼쪽에 있음
 - $\delta < 0$ 인 경우는 오른쪽
 - 절대값은 두 벡터 사이에 만들어지는 삼각형의 크기에 비례

삼각형 넓이 구하기

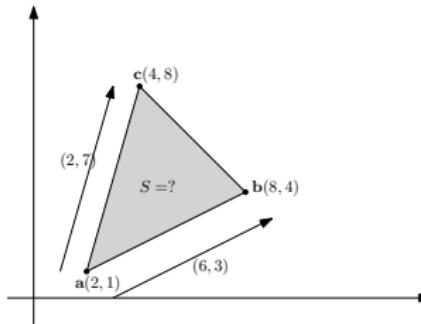
예제

꼭지점 좌표가 $(2,1)$, $(8,4)$, $(4,8)$ 인 삼각형의 넓이 S 를 구하라.

정답

꼭지점들을 각각 \mathbf{a} , \mathbf{b} , \mathbf{c} 로 표현하자. 우리는 \mathbf{a} 에서 \mathbf{b} 로 가는 벡터를 구할 수 있고, $\mathbf{u} = (6, 3)$ 가 된다. 비슷한 방식으로 \mathbf{a} 에서 \mathbf{c} 로 가는 벡터는 $\mathbf{v} = (2, 7)$ 이다. 삼각형의 넓이는 이 두 벡터의 외적이 가지는 크기의 반이다.

$$S = \frac{1}{2} \|\mathbf{a} \times \mathbf{b}\| = \frac{6 \cdot 7 - 3 \cdot 2}{2} = 18$$



외적과 평면

- 외적의 또 다른 응용

- 평면 표현

- 평면은 그 평면 위의 삼각형으로 표현 가능 = 3 개의 점 = 9 개의 원소
 - 좀 더 효율적인 방법 = 법선 벡터를 이용하기
 - 법선벡터 = 평면이 바라보는 방향을 나타내는 벡터
 - 법선벡터가 나타내는 것은 하나의 평면이 아니라 동일한 방향을
쳐다보는 모든 평면
 - 평면의 표현 = (법선벡터, 평면이 지나는 점) : 6 개의 원소로 표현 가능
 - 법선벡터 구하기: 벡터의 외적을 이용

행렬

행렬은 무엇이며, 무엇을 하는가?

행렬

• 행렬의 역사

- 1차 방정식의 풀이에 아주 오래 전부터 사용
- 그 특성이 정확히 파악되지 않고 1800년대까지는 배열(array)이라는 이름으로 알려짐
- 기원전 10세기에서 기원전 2세기 사이에 여러 세대에 걸쳐 쓰여진 중국의 구장산술(九章算術)에 연립 방정식을 풀기 위해 소개
- 판별식의 개념 등장
- 1545년에야 이탈리아 수학자 지롤라모 카르다노(Girolamo Cardano)가 그의 저서 "위대한 기술(Ars Magna)"를 통해 유럽에 전함
- 오랜 기간 동안 많은 수학자들이 이 행렬을 다루며 다양한 성질을 발견
- 행렬은 공간을 나누는 데에 필요한 유용한 도구
- 공간 내의 점들을 어떤 위치에서 다른 위치로 옮겨 놓는 다양한 변환이 행렬을 이용하여 표현됨

행렬이란 무엇인가

- 행렬은 2차원으로 배열된 수
- 가로 줄을 행(row), 세로 줄을 열(column)
- m 개의 행과 n 개의 열로 이루어진 행렬은 \mathbf{A} 는 다음의 모양

$$\bullet \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1j} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2j} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{i1} & a_{i2} & \cdots & a_{ij} & \cdots & a_{in} \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mj} & \cdots & a_{mn} \end{bmatrix}$$

- $m \times n$ 행렬이라고 하며 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 로 표현
- 각 행은 n 개의 원소를 가진 행벡터(row vector)
- 각 열은 m 개의 원소를 가진 열벡터(column vector)

행벡터와 열벡터

\mathbf{A} 를 행벡터 \mathbf{v}_i^r 로 표현하면 다음과 같다. $\mathbf{A} =$

$$\begin{bmatrix} \mathbf{v}_1^r \\ \mathbf{v}_2^r \\ \vdots \\ \mathbf{v}_m^r \end{bmatrix}$$

\mathbf{A} 를 열벡터 \mathbf{v}_j^c 로 표현하면 다음과 같다. $\mathbf{A} = [\mathbf{v}_1^c \quad \mathbf{v}_2^c \quad \cdots \quad \mathbf{v}_n^c]$

$$\mathbf{A} \in \mathcal{R}^{m \times n}$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{v}_1^r & \boxed{} \\ \mathbf{v}_2^r & \boxed{} \\ \mathbf{v}_3^r & \boxed{} \\ \vdots & \boxed{} \\ \mathbf{v}_m^r & \boxed{} \end{bmatrix}$$

m row vectors $\mathbf{v}_i^r \in \mathcal{R}^n$

$$\mathbf{A} = \boxed{} \begin{bmatrix} \mathbf{v}_1^c & \mathbf{v}_2^c & \mathbf{v}_3^c & \cdots & \mathbf{v}_n^c \end{bmatrix} \boxed{}$$

n column vectors $\mathbf{v}_j^c \in \mathcal{R}^m$

정사각 행렬 - square matrix

- 정방행렬, 혹은 정사각형 행렬은 행과 열의 수가 동일한 행렬.
 - $\mathbf{A} \in \mathbb{R}^{n \times n}$ 인 행렬
 - 정방행렬은 물리 문제에서 운동을 다루거나 그래픽스에서 변환을 다룰 때에 빈번히 나타남
- 다음 행렬 \mathbf{A} 는 3×3 의 정사각 행렬이다.

$$\bullet \mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

전치 행렬 - transpose

- 어떤 행렬 \mathbf{A} 의 전치행렬은 \mathbf{A}^T
 - $\mathbf{B} = \mathbf{A}^T \Rightarrow b_{ij} = a_{ji}$
 - 따라서, $m \times n$ 행렬의 전치는 $n \times m$ 행렬
 - $\mathbf{A} \in \mathbb{R}^{m \times n} \Rightarrow \mathbf{B} \in \mathbb{R}^{n \times m}$
- 전치행렬의 성질
 - $(\mathbf{A}^T)^T = \mathbf{A}$
 - $(\mathbf{A} + \mathbf{B})^T = \mathbf{A}^T + \mathbf{B}^T$
 - $(k\mathbf{A})^T = k\mathbf{A}^T$
 - $(\mathbf{AB})^T = \mathbf{B}^T \mathbf{A}^T$
- 그래픽스(graphics) 분야에서 매우 유용한 사용 방법은 회전 변환 행렬의 역행렬을 구할 때
- 3차원 공간의 회전 변환은 정규직교(orthonormal) 특성
- 정규직교 행렬의 역행렬은 그 행렬의 전치임이 알려져 있음

대각 행렬 - diagonal matrix

- 대각성분은 어떤 행렬 \mathbf{A} 의 i 행 j 열 성분을 a_{ij} 라고 표현할 때, $i = j$ 인 성분
- 대각행렬은 대각성분을 제외한 다른 모든 성분의 값이 0인 행렬이다

- $\mathbf{A} = \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix}$

- 대각행렬을 다른 행렬과 곱했을 때의 성질

- 어떤 3×3 대각행렬을 \mathbf{D} , 1,2,3 행 대각성분은 d_1, d_2, d_3
- 다른 어떤 3×3 행렬 \mathbf{A} 와 \mathbf{D} 의 곱

- $\mathbf{DA} = \begin{bmatrix} d_1 a_{11} & d_1 a_{12} & d_1 a_{13} \\ d_2 a_{21} & d_2 b_{22} & d_2 b_{23} \\ d_3 a_{31} & d_3 b_{32} & d_3 b_{33} \end{bmatrix}$

- $\mathbf{AD} = \begin{bmatrix} d_1 a_{11} & d_2 a_{12} & d_3 a_{13} \\ d_1 a_{21} & d_2 b_{22} & d_3 b_{23} \\ d_1 a_{31} & d_2 b_{32} & d_3 b_{33} \end{bmatrix}$

대각 행렬과 벡터의 곱 (1/2)

대각행렬 \mathbf{D} 와 열벡터 $\mathbf{v} \in \mathbb{R}^3$ 인 $(v_1, v_2, v_3)^T$ 의 곱

$$\mathbf{D}\mathbf{v} = \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} d_1 v_1 \\ d_2 v_2 \\ d_3 v_3 \end{bmatrix}$$

대각 행렬과 벡터의 곱 (2/2)

행벡터 \mathbf{v}^T 와 대각행렬의 곱

$$\mathbf{v}^T \mathbf{D} = \begin{bmatrix} v_1 & v_2 & v_3 \end{bmatrix} \begin{bmatrix} a_{11} & 0 & 0 \\ 0 & a_{22} & 0 \\ 0 & 0 & a_{33} \end{bmatrix} = \begin{bmatrix} d_1 v_1 & d_2 v_2 & d_3 v_3 \end{bmatrix}$$

행렬의 덧셈과 뺄셈

두 행렬의 덧셈과 뺄셈은 동일한 크기의 행렬 사이에 정의됨

$$\mathbf{A} \in \mathbb{R}^{m \times n} \wedge \mathbf{A} + \mathbf{B} = \mathbf{C} \Rightarrow \mathbf{B}, \mathbf{C} \in \mathbb{R}^{m \times n}$$

$$\mathbf{A} \in \mathbb{R}^{m \times n} \wedge \mathbf{A} - \mathbf{B} = \mathbf{D} \Rightarrow \mathbf{B}, \mathbf{D} \in \mathbb{R}^{m \times n}$$

행렬과 덧셈과 뺄셈은 동일한 행과 열에 있는 성분을 서로 더하고, 빼서 원래의 자리에 기록

$$\mathbf{A} + \mathbf{B} = \mathbf{C} \Rightarrow c_{ij} = a_{ij} + b_{ij}$$

$$\mathbf{A} - \mathbf{B} = \mathbf{D} \Rightarrow d_{ij} = a_{ij} - b_{ij}$$

행렬의 곱셈

- $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times x}$
 - $\mathbf{AB} = \mathbf{C} \in \mathbb{R}^{m \times x}$
- \mathbf{A} 의 앞에 \mathbf{B} 를 곱할 경우
 - $\mathbf{B} \in \mathbb{R}^{x \times m}$ 이어야 함
 - 그 결과는 $\mathbf{BA} = \mathbf{D} \in \mathbb{R}^{x \times n}$
- $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times x}$
- $\mathbf{AB} = \mathbf{C} \in \mathbb{R}^{m \times x}$
 - $c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj}$
 - $c_{ij} = \sum_{k=1}^n a_{ik}b_{kj}$

행렬 \mathbf{A} 의 i 번째 행 벡터를 $\mathbf{A}_{i,*}$ 라고 하고, j 번째 열 벡터를 $\mathbf{B}_{*,j}$ 라고 하면, 위의 식은 다음과 같이 다시 쓸 수 있다.

$$c_{ij} = \mathbf{A}_{i,*}^T \cdot \mathbf{B}_{*,j} \quad (1)$$

행렬과 스칼라의 곱

행렬에 스칼라를 곱하는 연산은 해당 스칼라 값을 행렬의 모든 원소에 곱하면 된다.

$$k\mathbf{A} = \mathbf{B} \Rightarrow b_{ij} = ka_{ij}$$

행렬 덧셈과 뺄셈의 연산 법칙

행렬은 다음과 같은 연산 법칙을 가진다.

$$\mathbf{A} + \mathbf{B} = \mathbf{B} + \mathbf{A}$$

$$(\mathbf{A} + \mathbf{B}) + \mathbf{C} = \mathbf{A} + (\mathbf{B} + \mathbf{C})$$

$$\mathbf{A} + 0 = 0 + \mathbf{A} = \mathbf{A}$$

$$\mathbf{A} + (-\mathbf{A}) = 0$$

이때, 0은 \mathbf{A} 와 같은 차원의 행렬로 모든 원소가 0인 행렬이다.

행렬 스칼라 곱의 연산 법칙

$$(k + l)\mathbf{A} = k\mathbf{A} + l\mathbf{A}$$

$$k(\mathbf{A} + \mathbf{B}) = k\mathbf{A} + k\mathbf{B}$$

$$(kl)\mathbf{A} = k(l\mathbf{A})$$

$$(-1)\mathbf{A} = -\mathbf{A}$$

$$0\mathbf{A} = 0$$

행렬 곱셈의 연산 법칙

$$\mathbf{AB} \neq \mathbf{BA}$$

$$\mathbf{A}(\mathbf{BC}) = (\mathbf{AB})\mathbf{C}$$

$$\mathbf{A}(\mathbf{B} + \mathbf{C}) = \mathbf{AB} + \mathbf{AC}$$

$$(\mathbf{A} + \mathbf{B})\mathbf{C} = \mathbf{AC} + \mathbf{BC}$$

$$\mathbf{AI} = \mathbf{IA} = \mathbf{A}$$

$$k\mathbf{AB} = (k\mathbf{A})\mathbf{B} = \mathbf{A}k\mathbf{B}$$

I는 항등행렬

행렬식 1/2

- 행렬식은 정방행렬에서 정의된다.
- 어떤 행렬 \mathbf{A} 의 행렬식은 $\det \mathbf{A}$, $\det(\mathbf{A})$, 또는 $|\mathbf{A}|$ 로 표현
- 행렬식을 계산하기 위해 필요한 개념
 - 소행렬식(minor)
 - 여인자(cofactor)
- 소행렬식
 - $\mathbf{A} \in \mathbb{R}^{m \times n}$: 이 행렬은 $m \times n$ 개의 소행렬식(minor) M_{ij} 를 가짐
 - 각 M_{ij} 는 \mathbf{A} 행렬의 i 행 벡터 전체와 j 열 벡터 전체를 제거하고 얻어지는 행렬($\in \mathbb{R}^{m-1 \times n-1}$)의 행렬식
- 여인자
 - 행렬 \mathbf{A} 의 여인자는 소행렬식이 구해지는 위치마다 결정
 - 다음과 같이 정의되는 $m \times n$ 개의 여인자 C_{ij} 가 존재
 - $C_{ij} = (-1)^{i+j} M_{ij}$

행렬식 2/2

- 행렬 $\mathbf{A} \in \mathbb{R}^{m \times n}$ 의 i 행, j 열 여인자 C_{ij} 를 이용한 행렬식 계산

$$\begin{aligned} \det \mathbf{A} = |\mathbf{A}| &= \sum_{j=1}^n A_{1j} C_{1j} = \sum_{j=1}^n A_{2j} C_{2j} = \cdots = \sum_{j=1}^n A_{mj} C_{mj} \quad (2) \\ &= \sum_{i=1}^m A_{i1} C_{i1} = \sum_{i=1}^m A_{i2} C_{i2} = \cdots = \sum_{i=1}^n A_{in} C_{in} \end{aligned}$$

- \mathbf{A} 의 임의의 행 벡터 $\mathbf{A}_{i,*}$ 와 \mathbf{C} 의 동일 위치 행 벡터 $\mathbf{C}_{i,*}$ 의 내적
- \mathbf{A} 의 임의의 열 벡터 $\mathbf{A}_{*,j}$ 와 \mathbf{C} 의 동일 위치 열 벡터 $\mathbf{C}_{*,j}$ 의 내적

$$\det \mathbf{A} = |\mathbf{A}| = \mathbf{A}_{i,*} \mathbf{C}_{i,*}^T = \mathbf{A}_{*,j}^T \mathbf{C}_{*,j} \quad (3)$$

예제

예제

다음 행렬의 행렬식을 구하라. $\begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$

정답

여인자 M_{ij} 를 구한다.

$$\mathbf{M} = \begin{bmatrix} M_{11} & M_{12} \\ M_{21} & M_{22} \end{bmatrix} = \begin{bmatrix} \det[A_{22}] & \det[A_{21}] \\ \det[A_{12}] & \det[A_{11}] \end{bmatrix} = \begin{bmatrix} A_{22} & A_{21} \\ A_{12} & A_{11} \end{bmatrix}$$

여인자의 정의에 따라, 여인자로 구성된 행렬 \mathbf{C} 는 다음과 같다.

$$\mathbf{C} = \begin{bmatrix} (-1)^{1+1}A_{22} & (-1)^{1+2}A_{21} \\ (-1)^{2+1}A_{12} & (-1)^{2+2}A_{11} \end{bmatrix} = \begin{bmatrix} A_{22} & -A_{21} \\ -A_{12} & A_{11} \end{bmatrix}$$

\mathbf{A} 의 임의의 행 벡터를 선택할 수 있으므로 우선 1행을 가지고 오자.
그리고 여인자 행렬 \mathbf{C} 의 1행을 가지고 와서, 두 행 벡터의 내적을 구하면
다음과 같다.

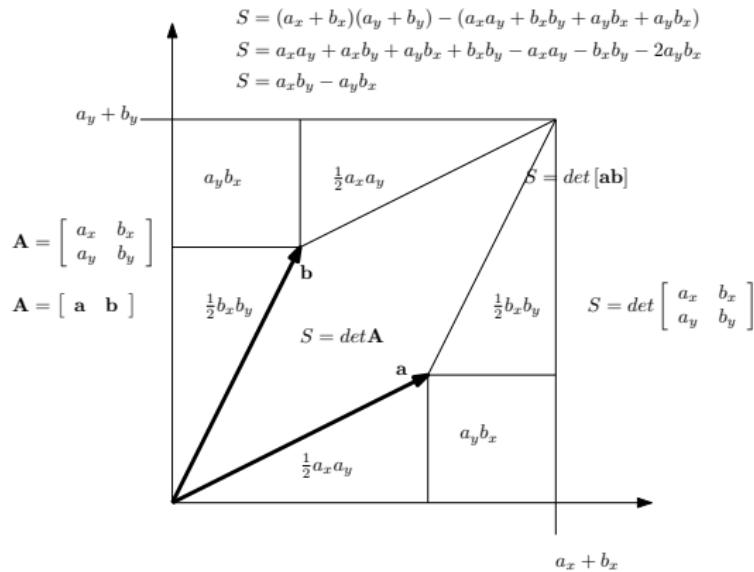
$$\det \mathbf{A} = \mathbf{A}_{1,*} \mathbf{C}_{1,*}^T = A_{11}A_{22} + A_{12}(-A_{21})$$

행렬식의 기하적 의미

두 열 벡터 $\mathbf{a} = (a_x a_y)^T$ 와 $\mathbf{b} = (b_x, b_y)^T$ 를 열로 하는 행렬 \mathbf{A}

$$\mathbf{A} = \begin{bmatrix} a_x & b_x \\ a_y & b_y \end{bmatrix}$$

이 두 벡터를 두 개의 변으로 하는 평행사변형의 넓이가 행렬 \mathbf{A} 의 행렬식

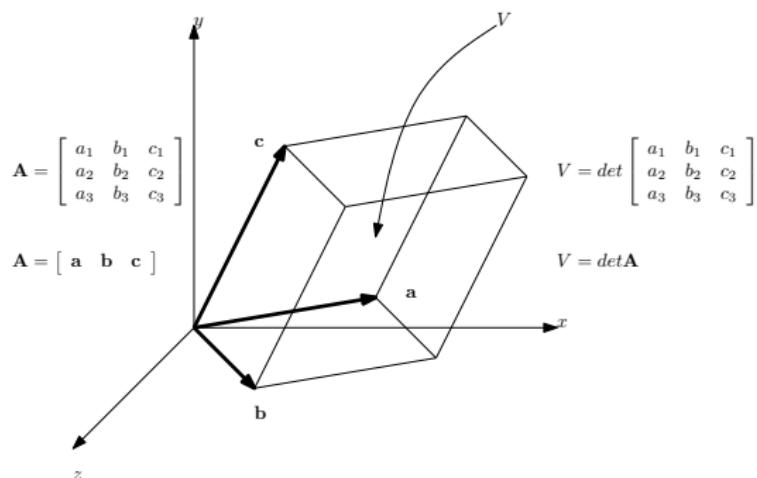


3×3 행렬의 행렬식 의미

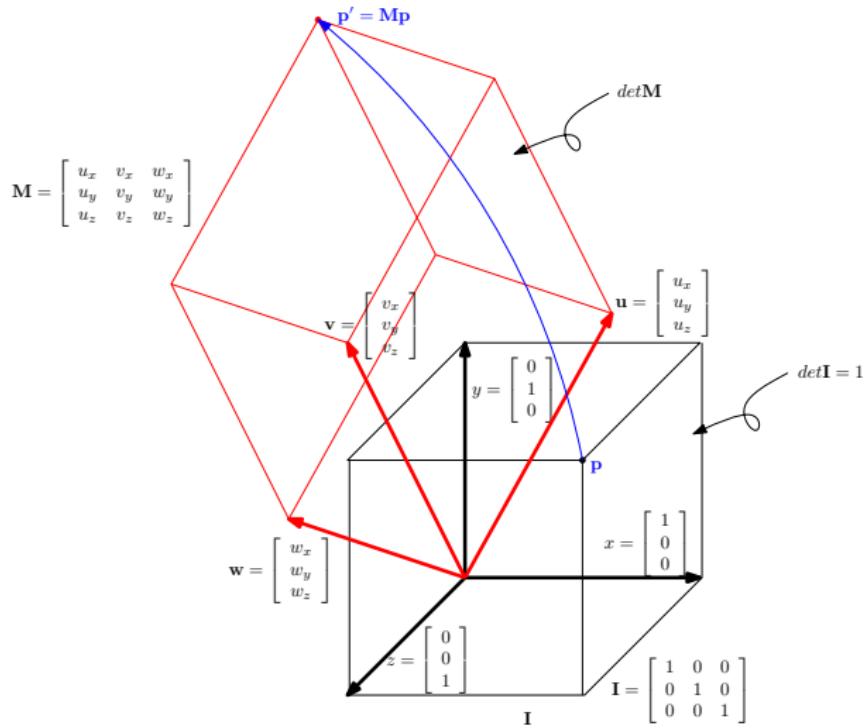
$\mathbf{A} \in \mathbb{R}^{3 \times 3}$ 는 세 개의 벡터 $\mathbf{a}, \mathbf{b}, \mathbf{c}$ 를 포함

$$\mathbf{A} = \begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} = [\mathbf{a} \ \mathbf{b} \ \mathbf{c}]$$

이 세 개의 벡터들이 만드는 평행육면체의 크기가 세 개의 벡터들로 구성된 행렬의 행렬식



행렬과 행렬식의 기하적 의미



행렬식의 특성

몇 가지 기억해 둘 행렬식의 특성은 다음과 같다.

$$|\mathbf{A}| = |\mathbf{A}^T|$$

$$\mathbf{A} \in \mathbb{R}^{n \times n} \Rightarrow |k\mathbf{A}| = k^n |\mathbf{A}|$$

$$|\mathbf{AB}| = |\mathbf{A}||\mathbf{B}|$$

역행렬

- 역행렬은 정방행렬에만 존재
- \mathbf{A} 의 역행렬이 존재한다면, 이 역행렬을 \mathbf{A}^{-1} 로 표현
- 역행렬 \mathbf{A}^{-1} 은 다음과 같은 조건을 만족
 - $\mathbf{A}\mathbf{A}^{-1} = \mathbf{I}$
 - $\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}$
- 역행렬이 존재하는 행렬을 가역행렬(invertible matrix)
- 역행렬이 존재하지 않는 행렬은 특이행렬(singular matrix)
- 의사 역행렬(pseudo-inverse)
 - 행렬 \mathbf{A} 가 정방행렬이 아니고 $\mathbb{R}^{m \times n}$ 에 속한다고 하자. 다른 어떤 행렬 \mathbf{B} 가 $\mathbb{R}^{n \times m}$ 에 속하면, 두 행렬의 곱 \mathbf{AB} 는 $\mathbb{R}^{m \times m}$ 에 속하는 정방행렬이 된다. 만약 $\mathbf{AB} = \mathbf{I} \in \mathbb{R}^{n \times n}$ 이라면, \mathbf{B} 를 \mathbf{A} 의 의사 역행렬(pseudo-inverse)라고 한다.

역행렬의 계산

- 역행렬의 계산은 수반행렬(adjoint matrix)를 이용하여 쉽게 정의
 - 행렬 \mathbf{A} 의 수반행렬: 여인자 C_{ij} 를 성분으로 하는 행렬 \mathbf{C} 의 전치(transpose)

$$adj \mathbf{A} = \begin{pmatrix} C_{11} & C_{21} & \cdots & C_{n1} \\ C_{12} & C_{22} & \cdots & C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{1n} & C_{2n} & \cdots & C_{nn} \end{pmatrix} = \mathbf{C}^T$$

- 수반행렬을 행렬의 행렬식으로 나누면 역행렬이 된다.

$$\mathbf{A}^{-1} = \frac{adj \mathbf{A}}{|\mathbf{A}|} = \frac{1}{|\mathbf{A}|} \begin{pmatrix} C_{11} & C_{21} & \cdots & C_{n1} \\ C_{21} & C_{22} & \cdots & C_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ C_{n1} & C_{2n} & \cdots & C_{nn} \end{pmatrix} = \mathbf{C}^T$$

식은 간단하지만, 여인자를 구하는 재귀호출이 매우 많은 계산을 요구

변환

변환의 개념과 행렬로 표현되는 변환

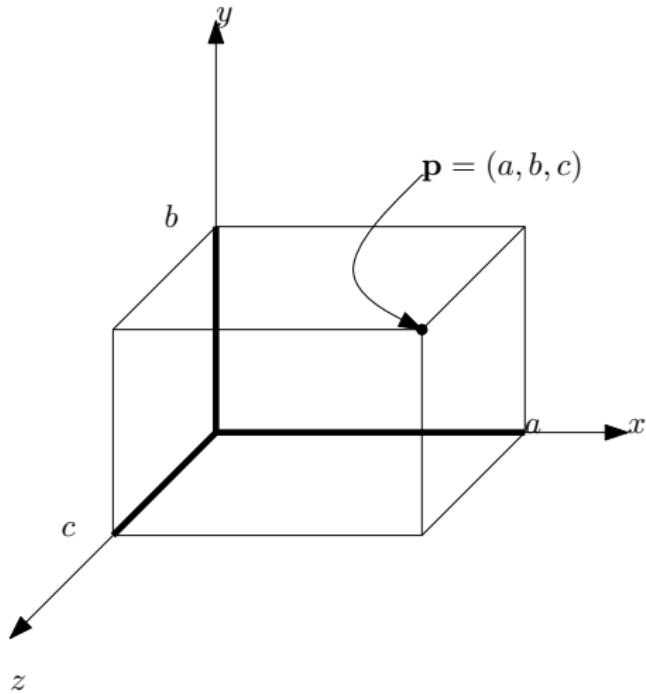
변환이란

수학적 의미에서 변환(transformation)

- 어떤 집합 S 를 다른 어떤 집합 S' 로 대응시키는 함수
- 공간과 점, 그리고 벡터의 문제로 이해할 때, 변환이란 공간 상의 벡터나 점을 다른 벡터나 점으로 바꾸는 연산
- 변환 행렬
 - 어떤 벡터 a 가 \mathbb{R}^n 에 속한다고 할 때, 이 벡터에 행렬 $A \in \mathbb{R}^{n \times n}$ 을 곱하면 a 와 같은 차원의 벡터 b 를 얻는다.
 - $b = Aa$ ($a, b \in \mathbb{R}^n, A \in \mathbb{R}^{n \times n}$).
 - 어떤 벡터를 동일한 차원의 다른 벡터로 옮기는 행렬 $A \in \mathbb{R}^{n \times n}$ 을 변환행렬(transform matrix)라고 한다.

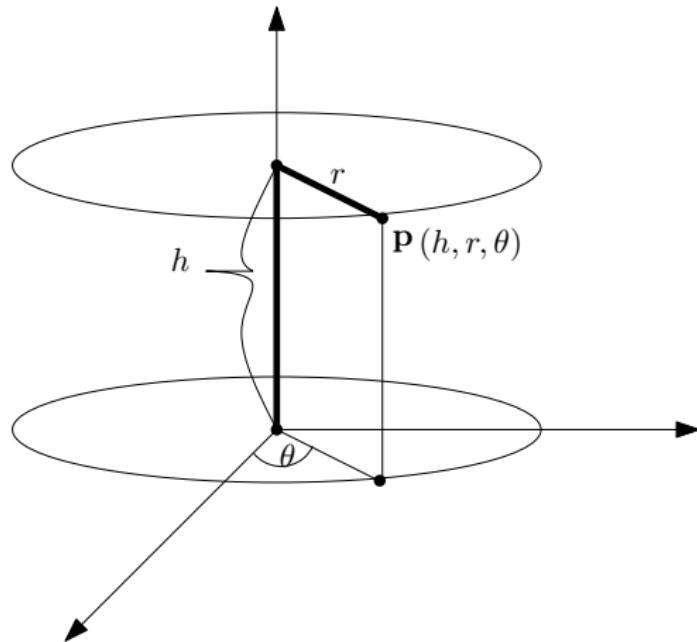
좌표계 - 직교 좌표계

- 일반적으로 가장 익숙한 좌표계
- 데카르트 좌표계(Cartesian coordinate system)



좌표계 - 원기둥 좌표계

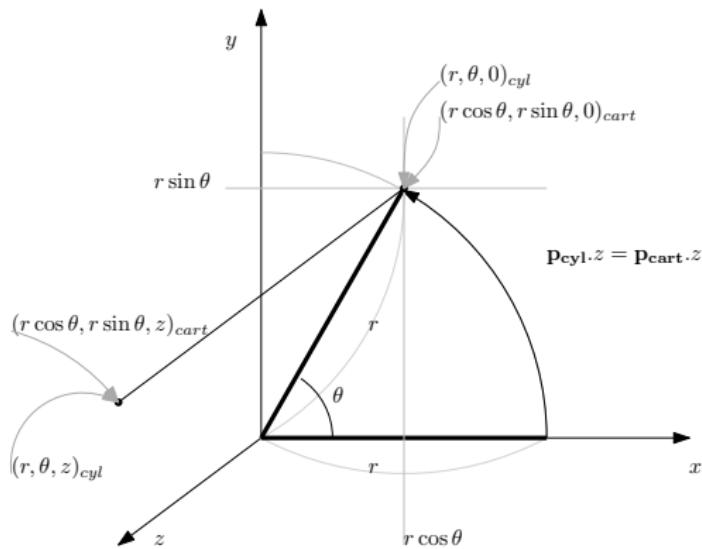
- \mathbf{p} 는 이러한 높이 h 와 반지름 r 을 가진 원기둥의 윗쪽 원주에 놓임.
- 원주에서 특정한 위치는 각도 θ 로 표현
- 원기둥 좌표: (r, θ, h)



원기둥 좌표를 직교 좌표로 옮기기

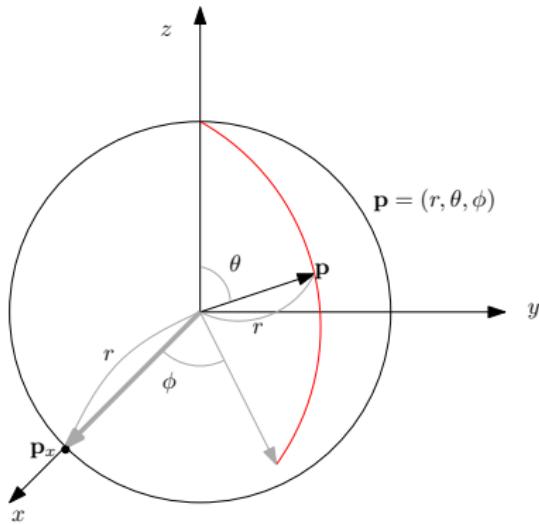
원기둥 좌표계의 좌표를 \mathbf{p}_{cyl} , 직교 좌표계의 좌표를 \mathbf{p}_{cart} 으로 표현하면

$$(r, \theta, h)_{cyl} = (r \cos \theta, r \sin \theta, h)_{cart}$$



좌표계 - 구면 좌표계

- \mathbf{p} 를 지나며 중심이 원점인 구면의 반지름을 r
- 반지름 r 인 점 가운데 x 축 위에 있는 점을 \mathbf{p}_x
- \mathbf{p}_x 을 xy 평면 위에서 \mathbf{p} 와 같은 경도선에 놓는 각도가 ϕ
- 이를 들어 올려 점 \mathbf{p} 를 지나도록 하는 데에 필요한 각도를 θ
- 구면 좌표 (r, θ, ϕ)



구면 좌표와 직교 좌표

구면 좌표는 일반적으로 다음과 같은 제한을 갖는다.

$$r \geq 0$$

$$0 \leq \theta \leq \pi$$

$$0 \leq \phi \leq 2\pi$$

직교 좌표계의 좌표 (x, y, z) 를 구면 좌표계로 옮기기

$$r = \sqrt{x^2 + y^2 + z^2}$$

$$\theta = \arccos \left(\frac{z}{\sqrt{x^2+y^2+z^2}} \right)$$

$$\phi = \arctan \left(\frac{y}{x} \right)$$

구면 좌표계의 좌표를 직교 좌표로 옮기기

$$x = r \sin \theta \cos \phi$$

$$y = r \sin \theta \sin \phi$$

$$z = r \cos \theta$$

무슨 좌표계를 사용할 것인가

- 공간에 존재하는 점을 다룰 때에는 어떠한 좌표계를 사용해도 무방
- 컴퓨터 그래픽스 분야에서 가장 많이 사용되는 좌표계는 직교 좌표계
- 우리는 직교 좌표계에서 변환에 대해 다룰 예정
- 직교 좌표계를 기본적인 좌표계로 삼고 변환과 관련된 행렬 연산을 살필 것

어파인(affine) 변환

게임을 구현하기 위한 3차원 그래픽스에서 흔히 사용되는 변환

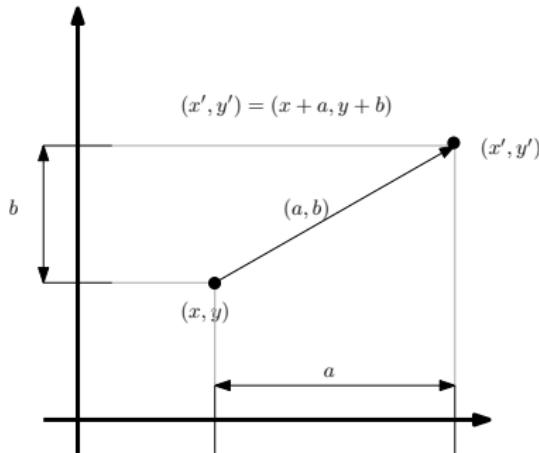
- 이동변환(translation): 주어진 벡터만큼 좌표를 동일하게 옮김
- 회전변환(rotation): 2차원은 기준점, 3차원은 기준축을 중심으로 돌림
- 크기변경=scaling): 각 축 방향으로 주어진 비율에 따라 좌표 값이 커지거나 줄어든다.

이러한 변환은 어파인 변환(affine transformation)의 일종

- 서로 연결되어 있음을 의미하는 라틴어 ‘affinis’에서 유래
- 직선 위의 점들을 직선을 유지한 상태로 변환하는 변환
- 직선 위에서의 점들 사이의 거리 비가 변환된 직선 위에서 그대로 유지
- 직선은 직선으로, 평행선은 평행선으로 유지
- 실시간 컴퓨터 그래픽스에서는 여러 가지 효율성의 이유로 어파인 변환을 사용

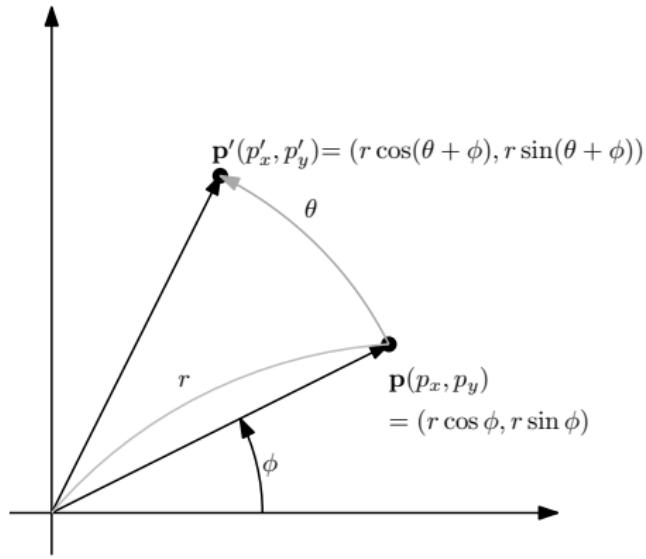
이동 변환(translation)

- 2차원: 좌표 (x, y) 를 x 축 방향으로 a , y 축 방향으로 b 만큼 옮기기
- $(x', y') = (x, y) + (a, b) = (x + a, y + b)$
- 모든 차원에 대해 어떤 벡터 \mathbf{a} 를 변위 벡터 \mathbf{d} 를 이용하여 \mathbf{x}' 로 옮기는 이동 변환을 다음과 같이 벡터 더하기로 정의할 수 있음
 - $\mathbf{a} \in \mathbb{R}^n, \mathbf{d} \in \mathbb{R}^n$
 - $\mathbf{x}' = \mathbf{a} + \mathbf{d} \quad \mathbf{x}' \in \mathbb{R}^n$

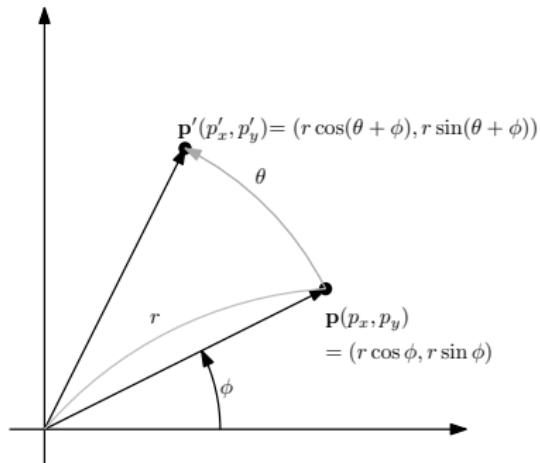


2차원 회전 변환(rotation) - 문제

- 2차원 회전의 중심: 피벗(pivot)
- 기본적인 회전: 피벗이 원점인 경우
 - \mathbf{p} 를 원점을 중심으로 θ 만큼 회전하여 놓이는 지점 \mathbf{p}' 를 구하는 문제
 - 원래 좌표 (p_x, p_y) 를 θ 만큼 회전하여 얻는 (p'_x, p'_y) 를 얻는 문제



2차원 회전 변환(rotation) - 좌표값에 대한 이해



- 원점에서 (p_x, p_y) 로 선분: 선분 길이 r 과 x 축과 이루는 각도 ϕ
- $(p_x, p_y) = (r \cos \phi, r \sin \phi)$
- 이 좌표를 θ 만큼 회전하여 얻는 (p'_x, p'_y)
 - $(p'_x, p'_y) = (r \cos(\theta + \phi), r \sin(\theta + \phi))$

2차원 회전 변환(rotation) - 회전결과

- ϕ 를 계산하지 않고 답을 얻어야 함
- 참조할 공식
 - $\cos(a + b) = \cos a \cos b - \sin a \sin b$
 - $\sin(a + b) = \sin a \cos b + \cos a \sin b$
- 회전하여 얻는 좌표는 다음과 같이 표현
 - $p'_x = (r \cos \phi) \cos \theta - (r \sin \phi) \sin \theta$
 - $p'_y = (r \cos \phi) \sin \theta + (r \sin \phi) \cos \theta$
- 원래의 좌표 (p_x, p_y) 를 이용하여 표현
 - $p'_x = p_x \cos \theta - p_y \sin \theta$
 - $p'_y = p_x \sin \theta + p_y \cos \theta$

2차원 회전 변환(rotation) - 행렬표현

이러한 변환은 다음과 같은 행렬과 벡터의 곱으로 표현할 수 있다.

$$\begin{bmatrix} p'_x \\ p'_y \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

2차원 공간에서 어떤 점 \mathbf{p} 를 원점 기준으로 θ 만큼 회전시켜 \mathbf{p}' 를 얻는 변환은 회전변환 행렬 $\mathbf{R}(\theta)$ 을 이용하여 $\mathbf{p}' = \mathbf{R}(\theta)\mathbf{p}$ 로 표현할 수 있다.

$$\mathbf{R}(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

3차원 회전 - z 축 회전

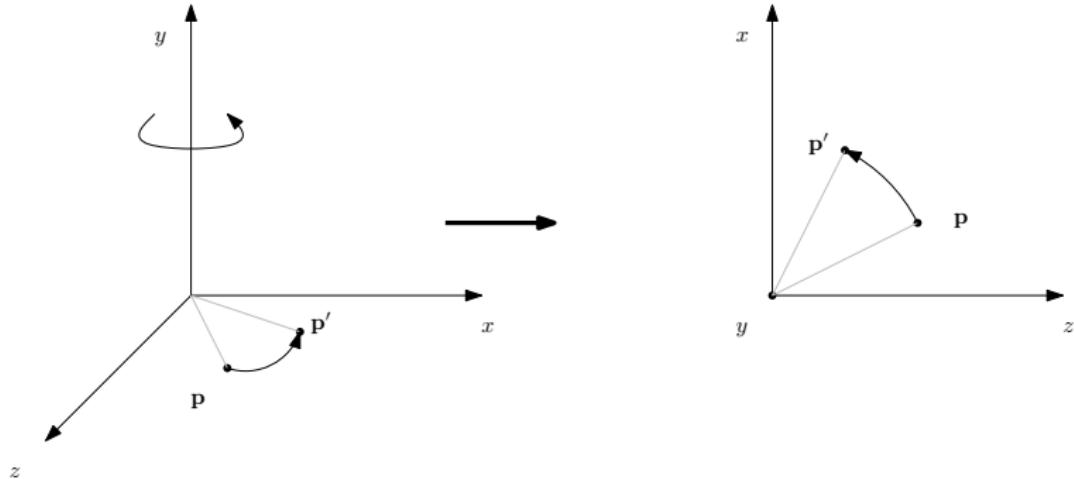
- 2차원 회전을 그대로 3차원에 적용
 - 3차원 좌표 $\mathbf{p} = (p_x, p_y, p_z)$ 을 z 기준으로 회전
- 이 변환은 2차원 변환에 z 성분만 추가
 - z 축 성분은 그대로 유지된다. ($p'_z = p_z$)
 - p_x, p_y 의 값은 2차원 회전과 동일하게 변환된다.

$$\begin{aligned} p'_x &= \cos \theta \cdot p_x & -\sin \theta \cdot p_y & +0 \cdot p_z \\ p'_y &= \sin \theta \cdot p_x & +\cos \theta \cdot p_y & +0 \cdot p_z \\ p'_z &= 0 \cdot p_x & +0 \cdot p_y & +1 \cdot p_z \end{aligned}$$

이것은 다음과 같은 행렬 표현으로 다시 쓸 수 있다.

$$\begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} \cos \theta & -\sin \theta & 0 \\ \sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

3차원 회전 - y 축 회전 (1/2)



- z 축은 2차원 회전의 x 축에 대응
- x 축은 2차원 회전의 y 축에 대응

$$\begin{aligned} p'_z &= \cos \theta \cdot p_z & -\sin \theta \cdot p_x & +0 \cdot p_y \\ p'_x &= \sin \theta \cdot p_z & +\cos \theta \cdot p_x & +0 \cdot p_y \\ p'_y &= 0 \cdot p_z & +0 \cdot p_x & +1 \cdot p_y \end{aligned}$$

3차원 회전 - y 축 회전 (1/2)

순서를 재배열하면 다음과 같은 식을 얻는다.

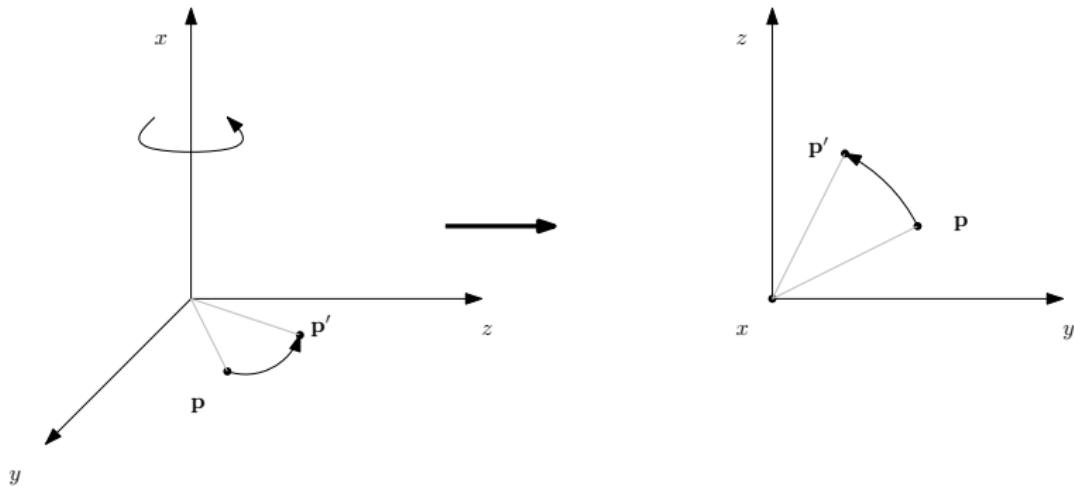
$$\begin{aligned} p'_x &= \cos \theta \cdot p_x + 0 \cdot p_y + \sin \theta \cdot p_z \\ p'_y &= 0 \cdot p_x + 1 \cdot p_y + 0 \cdot p_z \\ p'_z &= -\sin \theta \cdot p_x + 0 \cdot p_y + \cos \theta \cdot p_z \end{aligned}$$

이것도 역시 행렬 표현으로 다시 쓸 수 있다.

$$\begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

3차원 회전 - x 축 회전 (1/2)

- 2차원 회전에서 x, y 의 역할에 y 와 z 축이 각각 대응



$$\begin{aligned} p'_y &= \cos \theta \cdot p_y & -\sin \theta \cdot p_z & +0 \cdot p_x \\ p'_z &= \sin \theta \cdot p_y & +\cos \theta \cdot p_z & +0 \cdot p_x \\ p'_x &= 0 \cdot p_y & +0 \cdot p_z & +1 \cdot p_x \end{aligned}$$

3차원 회전 - x 축 회전 (2/2)

행렬과 벡터의 곱으로 표현하면 다음과 같다.

$$\begin{bmatrix} p'_x \\ p'_y \\ p'_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta \\ 0 & \sin \theta & \cos \theta \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

회전 행렬의 역행렬

- 회전행렬은 특별한 특징을 지님 (2차원 회전행렬을 보자)
 - 첫 열 벡터는 길이는 $\sqrt{\cos^2 \theta + \sin^2 \theta}$ 이므로 1
 - 두 번째 열의 길이 역시 $\sqrt{\sin^2 \theta + \cos^2 \theta}$ 로 1
 - 즉 두 벡터 모두 단위 벡터 (정규)
 - 이 두 벡터를 서로 내적하면 $\cos \theta(-\sin \theta) + \sin \theta \cos \theta$ 로 0
 - 두 벡터가 서로 수직 (직교)
- 모든 벡터는 단위 벡터이고 서로 직교 = 정규직교(orthonormal)
- 정규직교 행렬의 역행렬은 그 행렬의 전치(transpose)와 같음
- 3차원 회전행렬들도 정규직교임을 쉽게 확인 가능

$$\mathbf{R}_x^{-1}(\theta) = \mathbf{R}_x^T(\theta)$$

$$\mathbf{R}_y^{-1}(\theta) = \mathbf{R}_y^T(\theta)$$

$$\mathbf{R}_z^{-1}(\theta) = \mathbf{R}_z^T(\theta)$$

임의의 축에 대한 회전

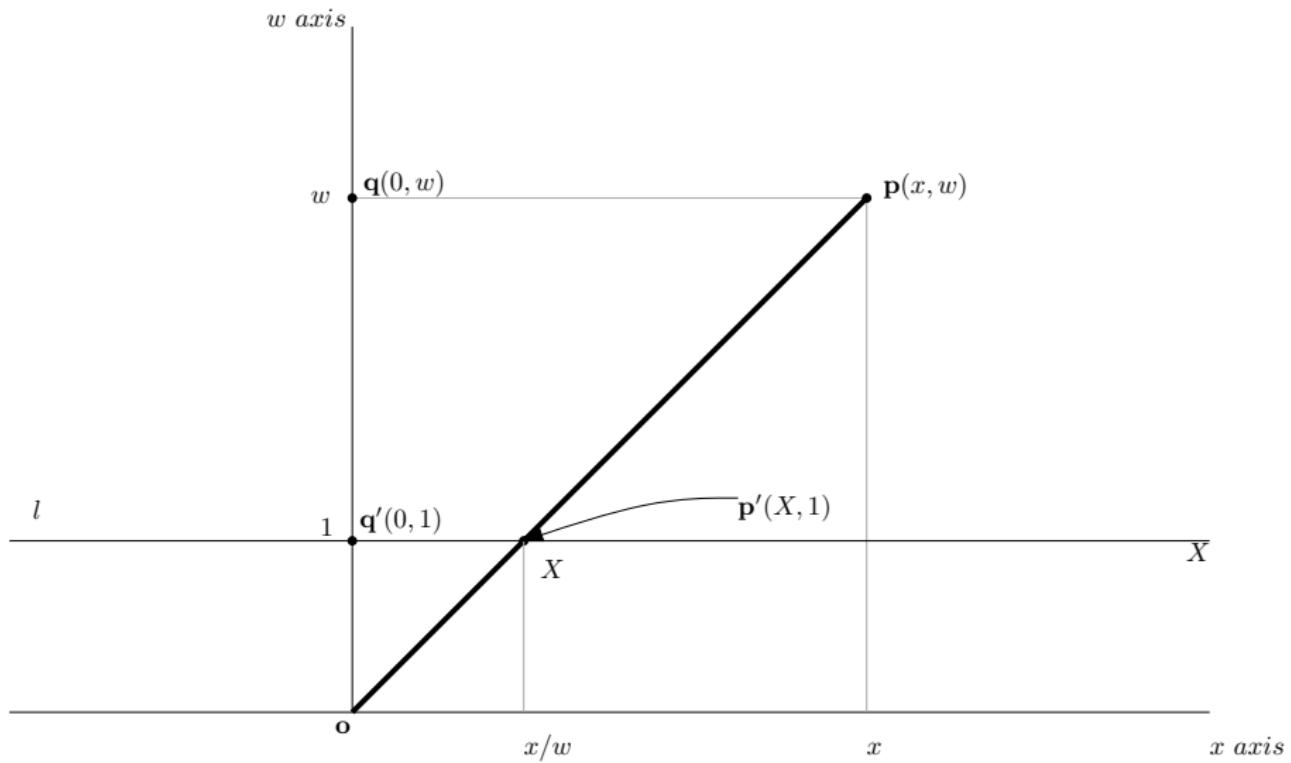
- 임의의 회전축을 기준으로 θ 만큼 회전하는 변환은 다음과 같은 절차를 따라 구현할 수 있다.
 - ① 회전축이 원점을 지나도록 이동변환 \mathbf{T} 를 적용한다.
 - ② 원점을 지나는 회전축이 xz 평면에 놓이도록 z 축 회전 \mathbf{R}_1 적용
 - ③ 이 회전축이 z 축과 동일한 방향이 되도록 y 축 회전 \mathbf{R}_2 를 적용한다.
 - ④ z 축을 기준으로 θ 만큼 회전하도록 $\mathbf{R}_z(\theta)$ 를 적용한다.
 - ⑤ \mathbf{R}_2^{-1} , 즉 \mathbf{R}_2^T 를 적용한다.
 - ⑥ \mathbf{R}_1^{-1} , 즉 \mathbf{R}_1^T 를 적용한다.
 - ⑦ \mathbf{T}^{-1} , 즉 $-\mathbf{T}$ 를 적용한다.
- 벡터 더하기(이동변환)와 행렬 곱하기(회전)가 혼재
- 동차좌표계(homogeneous coordinate)을 이용
 - 이동 변환과 회전 변환 모두 4×4 행렬
 - 위의 절차를 모두 누적한 하나의 행렬로 표현 가능
- $\mathbf{R}_{pivot}(\theta) = -\mathbf{T}\mathbf{R}_1^T\mathbf{R}_2^T\mathbf{R}_z(\theta)\mathbf{R}_2\mathbf{R}_1\mathbf{T}$

동차좌표계(homogeneous coordinate)

- 사영기하학(射影幾何學, projective geometry)에서 사용되는 좌표계
- n 차원의 사영공간을 $n + 1$ 의 좌표로 표현
- 그래픽 API나 라이브러리들은 동차좌표계를 표준적인 좌표계로 채용
 - 3차원 그래픽스에서 정의된 3차원 가상 공간 객체의 2차원 투영 이미지를 얻는 일과 유관
 - 3차원 공간의 어파인(affine) 변환들을 모두 4×4 의 행렬로 표현
- 3차원 공간의 좌표를 표현하는 벡터 $[x, y, z]^T$ 는 동차좌표계에서 $[x, y, z, 1]^T$ 로 표현할 수 있음
- 보다 일반적인 형태는 마지막 숫자를 1이 아닌 다른 값에 하는 것
- 2차원: $[x, y, w]^T$, 3차원: $[x, y, z, w]^T$

동차좌표계의 시각적 이해 (1/2)

동차좌표계 이해를 위해 Ammeraal이 사용한 그림



동차좌표계의 시각적 이해 (2/2)

- 두개의 축: x 축이고, 다른 하나는 w 축
 - 동차 좌표계에서 마지막 원소 w 를 제외한 모든 성분은 이 x 축 값
 - 마지막 원소는 이 w 축 값
- x 축 위에 있지 않는 점 \mathbf{p} 는 중심 사영(central projection) \mathbf{p}' 를 가짐
 - $w = 1$ 의 직선 l 과 원점 \mathbf{o} 에서 \mathbf{p} 를 연결한 직선의 교차점
 - 이때 원점은 사영중심(center of projection)
- w 축과 x 축을 모두 포함한 차원의 공간을 이보다 한 차원 낮은 x 축 공간으로 떨어뜨리는 것
- 선분 $\overline{\mathbf{op}}$ 를 지나는 직선 위의 모든 점들이 이 \mathbf{p}' 로 사영
- (x, w) 에 해당하는 $\mathbf{p}'(X, 1)$ 구하기
 - 닳은 삼각형 \mathbf{opq} 와 $\mathbf{op}'\mathbf{q}'$
 - 등비 관계를 이용하여 구한다
 - $X = \frac{x}{1} = \frac{|\mathbf{p}'\mathbf{q}'|}{|\mathbf{oq}'|}$
- 이 식은 다음과 같이 바뀐다.
- $X = \frac{x}{1} = \frac{|\mathbf{oq}'|}{|\mathbf{p}'\mathbf{q}'|} = \frac{|\mathbf{oq}|}{|\mathbf{pq}|} = \frac{x}{w}$

동차좌표계와 데카르트 좌표계의 관계

w 좌표의 의미

사영기하에서 **op**를 지나는 직선 위의 모든 점들은 (x, w) 형태의 좌표로 표현할 수 있고, 이 모든 점들은 $w = 1$ 인 평면으로 중심사영을 수행했을 때, w 좌표는 무의미해지면서 (x/w) 의 좌표로 바뀌게 된다. 즉, 3차원 공간의 좌표를 표현하기 위해 동차좌표계를 사용한다면 $[x, y, z, w]^T$ 의 형태가 되며, 이것은 위의 그림에서 w 축을 포함한 공간이 된다. 이를 3차원 데카르트 좌표로 바꾸는 것은 중심사영이 이루어지는 $w = 1$ 평면으로 옮겨 놓는 것이고 이때의 좌표는 $[x/w, y/w, z/w]^T$ 가 되는 것이다. 그리고 3차원 공간의 측면에서 보면, **op**를 지나는 직선 위의 모든 점들이 동일한 점으로 간주되는 것이다.

동차 좌표계 사용의 이점(利點)

- 3차원 좌표 $[x, y, z]^T$ 를 동차좌표계 좌표로 바꾸는 간단한 방법은 $w = 1$ 평면에서의 좌표인 $[x, y, z, 1]^T$
- 동차좌표계를 쓰면 좋은 점
 - 좌표와 벡터의 구분이 가능
 - $[x, y, z]^T$ 가 3차원 좌표라면 이 좌표로 표현되는 지점은 3차원 공간내 유일
 - 벡터로 해석된다면 그것은 수 많은 동등 벡터를 표현하게 되며, 공간 내의 특별한 지점을 가리키지 않음
 - 이 둘은 분명히 다르지만 단순한 좌표 표현 방식으로는 구분이 불가능
- 동차좌표계에서 좌표와 벡터의 구분
 - 좌표는 $w \neq 0$ 인 $[x, y, z, w]^T$
 - 벡터는 $w = 0$ 인 $[x, y, z, 0]^T$
 - $[x, y, z, 0]^T$ 는 위치를 가진 좌표 $[x, y, z]^T$ 가 아니라 위치가 없는 벡터 $[x, y, z]^T$
- 또 다른 이점
 - 이동변환과 회전변환을 모두 같은 차원의 행렬로 표현

동차 좌표계에서 이동 변환

동차좌표계에서의 이동

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} + \begin{bmatrix} d_x \\ d_y \\ d_z \\ 0 \end{bmatrix} = \begin{bmatrix} x + d_x \\ y + d_y \\ z + d_z \\ 1 \end{bmatrix}$$

행렬 표현

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + d_x \\ y + d_y \\ z + d_z \\ 1 \end{bmatrix}$$

이동 변환 행렬의 역행렬

이제 이동변환을 행렬로 표현할 수 있게 되었다. 변위 벡터 $\mathbf{d}(d_x, d_y, d_z)$ 만큼의 이동을 수행하는 변환행렬을 \mathbf{T}_d 라고 하면 이동 변환은 다음과 같이 표현할 수 있다.

$$\mathbf{p}' = \mathbf{T}_d \mathbf{p}, \quad \mathbf{T}_d \in \mathbb{R}^{4 \times 4}$$

이동변환 행렬 \mathbf{T}_d 의 역행렬은 어떻게 될까? 역행렬은 이 행렬이 일으킨 변환을 원래대로 되돌려 놓는 것이므로 \mathbf{T}_{-d} 가 됨을 알 수 있다.

$$\begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & 0 & 0 & -d_x \\ 0 & 1 & 0 & -d_y \\ 0 & 0 & 1 & -d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

동차좌표계에서의 회전 행렬

- 3차원 공간에서 정의되었던 회전 변환을 \mathbf{R}_{33}
- $\mathbb{R}^{4 \times 4}$ 에 속하는 동차좌표계 회전 행렬은 \mathbf{R}_{44}
- 원소가 모두 0인 3차원 열벡터를 \mathbf{O}_3^{col} , 행벡터를 \mathbf{O}_3^{row}

$$\mathbf{R}_{44} = \begin{bmatrix} \mathbf{R}_{33} & \mathbf{O}_3^{col} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}$$

$$\mathbf{R}_{44}^x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & -\sin \theta & 0 \\ 0 & \sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R}_{44}^y = \begin{bmatrix} \cos \theta & 0 & \sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{44}^z = \begin{bmatrix} \cos \theta & -\sin \theta & 0 & 0 \\ \sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

복합변환

좌표를 \mathbf{R}_{44} 를 이용하여 회전하고, 이를 \mathbf{T}_d 만큼 이동하는 변환

$$\mathbf{p}' = \mathbf{T}_d \mathbf{R}_{44} \mathbf{p}$$

두 변환을 모두 수행하는 하나의 행렬 구할 수 있음

$$\begin{aligned}\mathbf{p}' &= \begin{bmatrix} \mathbf{I}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{33} & \mathbf{O}_3^{col} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p} \\ &= \begin{bmatrix} \mathbf{R}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p}\end{aligned}$$

- 행렬의 역행렬은?

$$\begin{bmatrix} \mathbf{R}_{33}^T & \mathbf{O}_3^{col} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{33} & -\mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{33}^T & -\mathbf{R}_{33}^T \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}$$

복합변환

좌표를 \mathbf{R}_{44} 를 이용하여 회전하고, 이를 \mathbf{T}_d 만큼 이동하는 변환

$$\mathbf{p}' = \mathbf{T}_d \mathbf{R}_{44} \mathbf{p}$$

두 변환을 모두 수행하는 하나의 행렬 구할 수 있음

$$\begin{aligned}\mathbf{p}' &= \begin{bmatrix} \mathbf{I}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{R}_{33} & \mathbf{O}_3^{col} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p} \\ &= \begin{bmatrix} \mathbf{R}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p}\end{aligned}$$

- 행렬의 역행렬은?

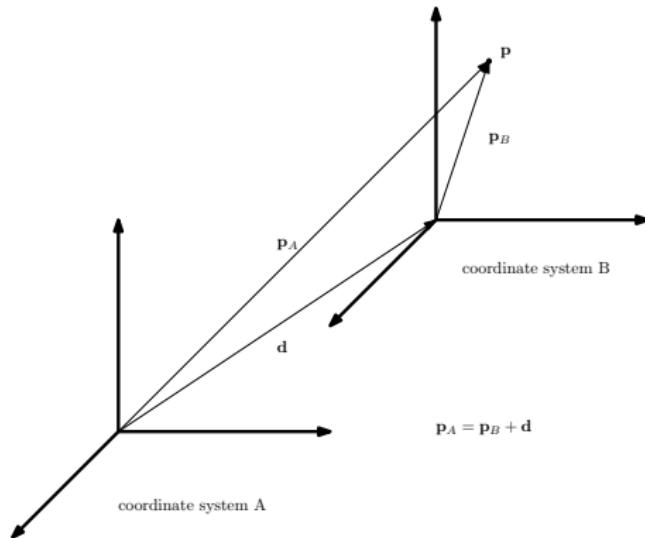
$$\begin{bmatrix} \mathbf{R}_{33}^T & \mathbf{O}_3^{col} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I}_{33} & -\mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R}_{33}^T & -\mathbf{R}_{33}^T \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}$$

좌표계의 변환

- 어떤 점이 좌표계 A 에 의해 \mathbf{p}_A 라 표현된다고 가정
- 다른 좌표계 B 의 입장에서 보면 다른 좌표 \mathbf{p}_B
- 이렇게 좌표계가 달라질 때 바뀐 좌표계에 따라 새로운 좌표를 계산하는 일은 그래픽스에서 매우 빈번히 나타나는 작업
 - 가상 공간 내의 모든 객체의 위치를 하나의 기준으로 정의하는 데에 필요한 전역좌표계(global coordinate system)과 개별 객체 내에 정의된 지역좌표계(local coordinate system)

좌표계의 이동

$$\mathbf{p}_A = \mathbf{p}_B + \mathbf{d}$$

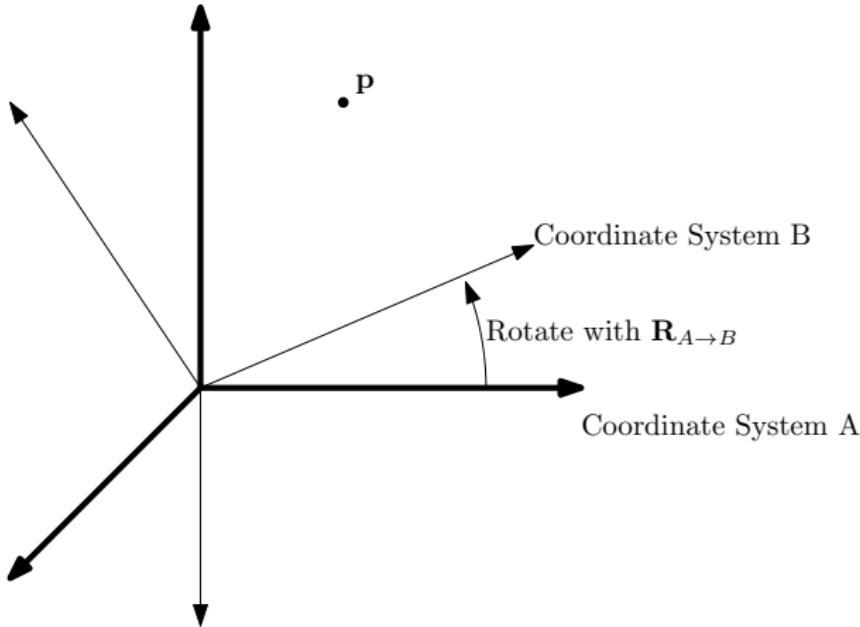


$$\mathbf{T}_d = \begin{bmatrix} 1 & 0 & 0 & d_x \\ 0 & 1 & 0 & d_y \\ 0 & 0 & 1 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad \mathbf{p}_A = \mathbf{T}_d \mathbf{p}_B, \quad \mathbf{p}_B = \mathbf{T}_d^{-1} \mathbf{p}_A$$

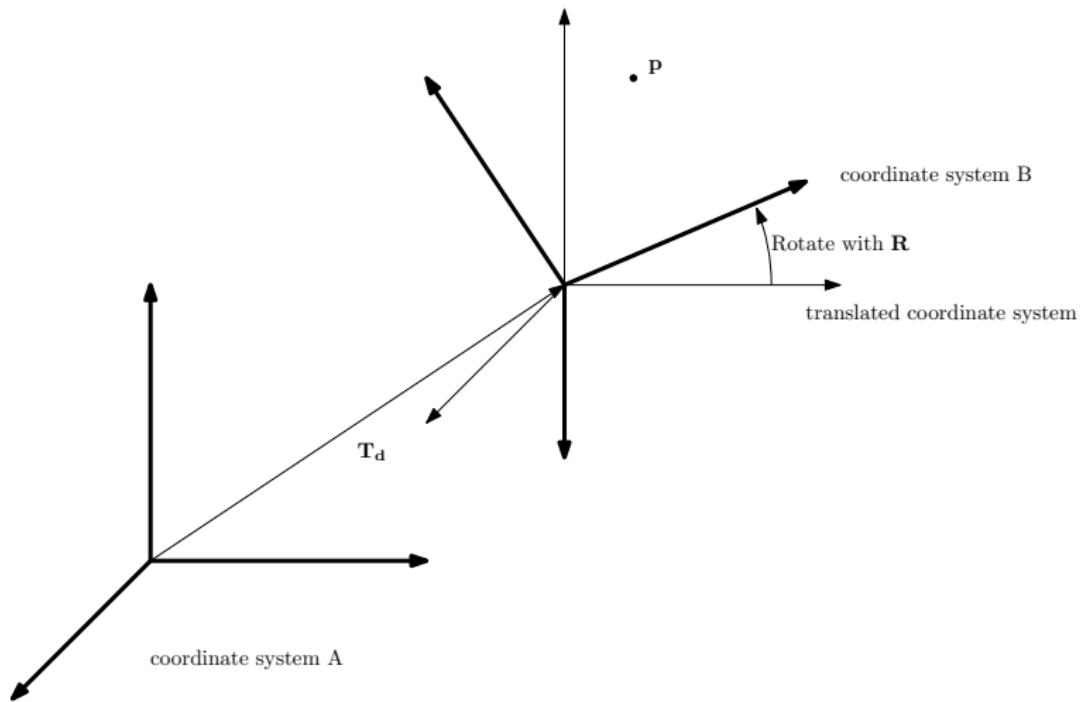
좌표계의 회전

$$\mathbf{p}_A = \mathbf{R}_{A \rightarrow B} \mathbf{p}_B$$

$$\mathbf{p}_B = \mathbf{R}_{A \rightarrow B}^{-1} \mathbf{p}_A = \mathbf{R}_{A \rightarrow B}^T \mathbf{p}_A$$



회전과 이동이 함께 이루어진 좌표계 변환 (1/2)



회전과 이동이 함께 이루어진 좌표계 변환 (1/2)

$$\mathbf{R} = \begin{bmatrix} \mathbf{R}_{33} & \mathbf{0} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}, \quad \mathbf{T_d} = \begin{bmatrix} \mathbf{I}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}, \quad \mathbf{T_d R} = \begin{bmatrix} \mathbf{R}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix}$$

$$\mathbf{p}_A = \begin{bmatrix} \mathbf{R}_{33} & \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p}_B$$

$$\mathbf{p}_B = \begin{bmatrix} \mathbf{R}_{33}^T & \mathbf{R}_{33}^T \mathbf{d} \\ \mathbf{O}_3^{row} & 1 \end{bmatrix} \mathbf{p}_A$$

사원수

사원수의 이해와 회전 변환에서의 활용

사원수

- 사원수(quaternion)는 스칼라(scalar) 값과 3차원 벡터를 묶어 구성한 복소수(complex number)
- 3차원 벡터 $\mathbf{v} = (a, b, c)$ 를 각각의 축 방향 단위 벡터인 기저 $\mathbf{i}, \mathbf{j}, \mathbf{k}$ 로 표현하면 $a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$
- 사원수는 여기에 스칼라 값 d 가 추가된 $d + a\mathbf{i} + b\mathbf{j} + c\mathbf{k}$
- 기저를 제외하고 표현하면 $(d, (a, b, c))$ 와 같이 표현
- 벡터 기호로 표현하면 (d, \mathbf{v})
- 스칼라 값은 기저가 실수의 단위 값인 1이라고 이해하면 사원수는 $1, \mathbf{i}, \mathbf{j}, \mathbf{k}$ 를 기저로 하는 벡터
- 로보틱스와 컴퓨터 그래픽스 분야에서 회전을 다루는 데에 빈번히 이용

사원수의 연산 - 덧셈과 뺄셈

- 사원수 덧셈: 성분별로 더하면 된다.
- 두 개의 사원수 \hat{p} 와 \hat{q} 가 각각 (s_p, \mathbf{v}_p) 와 (s_q, \mathbf{v}_q) 일 때, 덧셈은
 - $\hat{p} + \hat{q} = (s_p + s_q, \mathbf{v}_p + \mathbf{v}_q)$
 - $\hat{p} = (a_p, b_p, c_p, d_p)$ 와 $\hat{q} = (a_q, b_q, c_q, d_q)$
 - $\hat{p} + \hat{q} = (a_p + a_q, b_p + b_q, c_p + c_q, d_p + d_q)$
- 뺄셈
 - $\hat{p} - \hat{q} = (a_p - a_q, b_p - b_q, c_p - c_q, d_p - d_q)$
 - 스칼라와 벡터로 나누어 표현하면 다음과 같다.
 - $\hat{p} = (s_p, \mathbf{v}_p), \quad \hat{q} = (s_q, \mathbf{v}_q)$
 - $\hat{p} + \hat{q} = (s_p - s_q, \mathbf{v}_p - \mathbf{v}_q)$

사원수의 연산 - 스칼라 곱셈

사원수와 어떤 스칼라 λ 를 곱하는 것은 모든 성분에 이 스칼라 값을 곱하는 것이다.

$$\lambda \hat{p} = (\lambda s_p, \lambda \mathbf{v}_p) = (\lambda a_p, \lambda b_p, \lambda c_p, \lambda d_p)$$

사원수의 연산 - 사원수 곱셈 1/4

- 두 사원수 \hat{p} 와 \hat{q} 를 곱하려면 어떻게 해야 할까?
- 사원수는 서로 다른 허수 i, j, k 를 가진 벡터와 스칼라의 합인 복소수
- 허수들 사이의 곱
 - $i^2 = j^2 = k^2 = -1$
 - $ij = k, jk = i, ki = j$
 - $ji = -k, kj = -i, ik = -j$
- $\hat{p}\hat{q}$ 는 다음과 같이 표현 가능
 - $\hat{p} = d_p + a_p\mathbf{i} + b_p\mathbf{j} + c_p\mathbf{k}$
 - $\hat{q} = d_q + a_q\mathbf{i} + b_q\mathbf{j} + c_q\mathbf{k}$

$$\begin{aligned}\hat{p}\hat{q} = & d_p d_q + d_p a_q \mathbf{i} + d_p b_q \mathbf{j} + d_p c_q \mathbf{k} + \\& a_p \mathbf{i} d_q + a_p \mathbf{i} a_q \mathbf{i} + a_p \mathbf{i} b_q \mathbf{j} + a_p \mathbf{i} c_q \mathbf{k} + \\& b_p \mathbf{j} d_q + b_p \mathbf{j} a_q \mathbf{i} + b_p \mathbf{j} b_q \mathbf{j} + b_p \mathbf{j} c_q \mathbf{k} + \\& c_p \mathbf{k} d_q + c_p \mathbf{k} a_q \mathbf{i} + c_p \mathbf{k} b_q \mathbf{j} + c_p \mathbf{k} c_q \mathbf{k}\end{aligned}$$

사원수의 연산 - 사원수 곱셈 2/4

정리하면 다음과 같다.

$$\begin{aligned}\hat{p}\hat{q} = & d_p d_q + d_p a_q \mathbf{i} + d_p b_q \mathbf{j} + d_p c_q \mathbf{k} + \\ & a_p d_q \mathbf{i} + a_p a_q \mathbf{i}^2 + a_p b_q \mathbf{ij} + a_p c_q \mathbf{ik} + \\ & b_p d_q \mathbf{j} + b_p a_q \mathbf{ji} + b_p b_q \mathbf{j}^2 + b_p c_q \mathbf{jk} + \\ & c_p d_q \mathbf{k} + c_p a_q \mathbf{ki} + c_p b_q \mathbf{kj} + c_p c_q \mathbf{k}^2\end{aligned}\tag{4}$$

허수의 곱이 나타나는 부분을 정리하면,

$$\begin{aligned}\hat{p}\hat{q} = & d_p d_q + d_p a_q \mathbf{i} + d_p b_q \mathbf{j} + d_p c_q \mathbf{k} + \\ & a_p d_q \mathbf{i} - a_p a_q + a_p b_q \mathbf{k} - a_p c_q \mathbf{j} + \\ & b_p d_q \mathbf{j} - b_p a_q \mathbf{k} - b_p b_q + b_p c_q \mathbf{i} + \\ & c_p d_q \mathbf{k} + c_p a_q \mathbf{j} - c_p b_q \mathbf{i} - c_p c_q\end{aligned}\tag{5}$$

사원수의 연산 - 사원수 곱셈 3/4

허수별로 정리하면 다음과 같다.

$$\begin{aligned}\hat{p}\hat{q} = & d_p d_q - a_p a_q - b_p b_q - c_p c_q \\ & + d_p a_q \mathbf{i} + a_p d_q \mathbf{i} + (b_p c_q - c_p b_q) \mathbf{i} \\ & + d_p b_q \mathbf{j} + b_p d_q \mathbf{j} + (c_p a_q - a_p c_q) \mathbf{j} \\ & + d_p c_q \mathbf{k} + c_p d_q \mathbf{k} + (a_p b_q - b_p a_q) \mathbf{k}\end{aligned}$$

계산의 의미는 다음과 같다.

$$\begin{aligned}\hat{p}\hat{q} = & d_p d_q - (a_p a_q + b_p b_q + c_p c_q) \\ & + d_p (a_q \mathbf{i} + b_q \mathbf{j} + c_q \mathbf{k}) \\ & + d_q (a_p \mathbf{i} + b_p \mathbf{j} + c_p \mathbf{k}) \\ & + (b_p c_q - c_p b_q) \mathbf{i} + (c_p a_q - a_p c_q) \mathbf{j} + (a_p b_q - b_p a_q) \mathbf{k}\end{aligned}$$

사원수의 연산 - 사원수 곱셈 4/4

벡터의 내적과 외적을 이용하여 설명하면,

$$\begin{aligned}\hat{pq} = & d_p d_q - (\mathbf{v}_p \cdot \mathbf{v}_q) \\ & + d_p \mathbf{v}_q + d_q \mathbf{v}_p + \mathbf{v}_p \times \mathbf{v}_q\end{aligned}$$

사원수를 스칼라 값과 벡터 표현인 (d, \mathbf{v}) 로 표현하면,

$$\begin{aligned}\hat{pq} &= (d_p, \mathbf{v}_p)(d_q, \mathbf{v}_q) \\ &= (d_p d_q - \mathbf{v}_p \cdot \mathbf{v}_q, \quad d_p \mathbf{v}_q + d_q \mathbf{v}_p + \mathbf{v}_p \times \mathbf{v}_q)\end{aligned}$$

- 스칼라: 두 사원수가 가진 스칼라 값의 곱에서 두 사원수가 가진 벡터 내적을 뺀 것
- 벡터: 각 사원수가 가진 스칼라 값을 상대편의 벡터 부분에 곱한 결과 두 개를 더하고, 두 사원수가 가진 벡터를 서로 외적하여 얻는 벡터를 추가로 더하여 얻음

사원수의 연산 규칙

$$\hat{p} + \hat{q} = \hat{q} + \hat{p}$$

$$(\hat{p} + \hat{q}) + \hat{r} = \hat{p} + (\hat{q} + \hat{r})$$

$$\lambda\hat{p} = \hat{p}\lambda$$

$$-\lambda\hat{p} = \lambda(-\hat{p})$$

$$\hat{p}\hat{q} \neq \hat{q}\hat{p}$$

켤레 사원수 1/2

- 켤레 사원수(공액 사원수, conjugate)
 - 어떤 사원수 $\hat{p} = (d_p, \mathbf{v}_p)$ 의 켤레 사원수를 \hat{p}^* 라고 표현
 - 이 켤레 이 사원수는 $(d_p, -\mathbf{v}_p)$ 의 값을 가짐

$$\hat{p} = (d_p, \mathbf{v}_p) \Rightarrow \hat{p}^* = (d_p, -\mathbf{v}_p)$$

- 사원수의 크기는 벡터의 크기와 같은 방식으로 구한다.

$$\begin{aligned} |\hat{q}| &= \sqrt{d_q d_q + a_q a_q + b_q b_q + c_q c_q} \\ &= \sqrt{d_q^2 + a_q^2 + b_q^2 + c_q^2} \\ &= \sqrt{d_q^2 + \mathbf{v}^T \mathbf{v}} \\ &= \sqrt{d_q^2 + \mathbf{v} \cdot \mathbf{v}} \\ &= \sqrt{\hat{q} \hat{q}^*} \end{aligned}$$

켤레 사원수 2/2

- 사원수의 항등원은 \hat{i} 는 $(1, 0, 0, 0)$
- 사원수 \hat{q} 의 역원 \hat{q}^{-1} 은 $\hat{q}^*/|\hat{q}|$

$$\hat{q}\hat{i} = \hat{i}\hat{q} = \hat{q}$$

$$\hat{q}\hat{q}^{-1} = \hat{q}^{-1}\hat{q} = \hat{q}\hat{q}^*/|q| = \hat{i}$$

- 켤레 사원수의 크기는 서로 동일하다.

$$|\hat{q}| = |\hat{q}^*|$$

- 다음과 같은 연산 규칙도 중요

$$(\hat{q} + \hat{r})^* = \hat{q}^* + \hat{r}^*$$

$$(\hat{q}\hat{r})^* = \hat{r}^*\hat{q}^*$$

사원수 연산 법칙 정리

$$\hat{p} + \hat{q} = \hat{q} + \hat{p}$$

$$(\hat{p} + \hat{q}) + \hat{r} = \hat{p} + (\hat{q} + \hat{r})$$

$$\lambda\hat{p} = \hat{p}\lambda$$

$$-\lambda\hat{p} = \lambda(-\hat{p})$$

$$\hat{p}\hat{q} \neq \hat{q}\hat{p}$$

$$\hat{p} = (d_p, \mathbf{v}_p) \implies \hat{p}^* = (d_p, -\mathbf{v}_p)$$

$$|\hat{q}| = \sqrt{\hat{q}\hat{q}^*}$$

$$\hat{q}\hat{i} = \hat{q} \implies \hat{i} = (1, 0, 0, 0)$$

$$\hat{q}\hat{p} = \hat{i} \implies \hat{p} = \hat{q}^{-1} = \hat{q}^*/|\hat{q}|$$

$$\hat{q}\hat{q}^{-1} = \hat{q}^{-1}\hat{q} = \hat{q}\hat{q}^*/|q| = \hat{i}$$

$$|\hat{q}| = |\hat{q}^*|$$

$$(\hat{q} + \hat{r})^* = \hat{q}^* + \hat{r}^*$$

$$(\hat{q}\hat{r})^* = \hat{r}^*\hat{q}^*$$

사원수와 회전: 곱셈

- 행렬로 표현했던 회전은 사원수를 이용하여 표현 가능
- 어떤 좌표 $\mathbf{p}(x, y, z)$ 는 사원수 표현으로는 $\hat{p} = (0, (x, y, z)) = (0, \mathbf{p})$
- 이 좌표에 다음과 같은 사원수 \hat{q} 를 곱하면 어떻게 되는지 보자.

$$\hat{p} = (0, \mathbf{p})$$

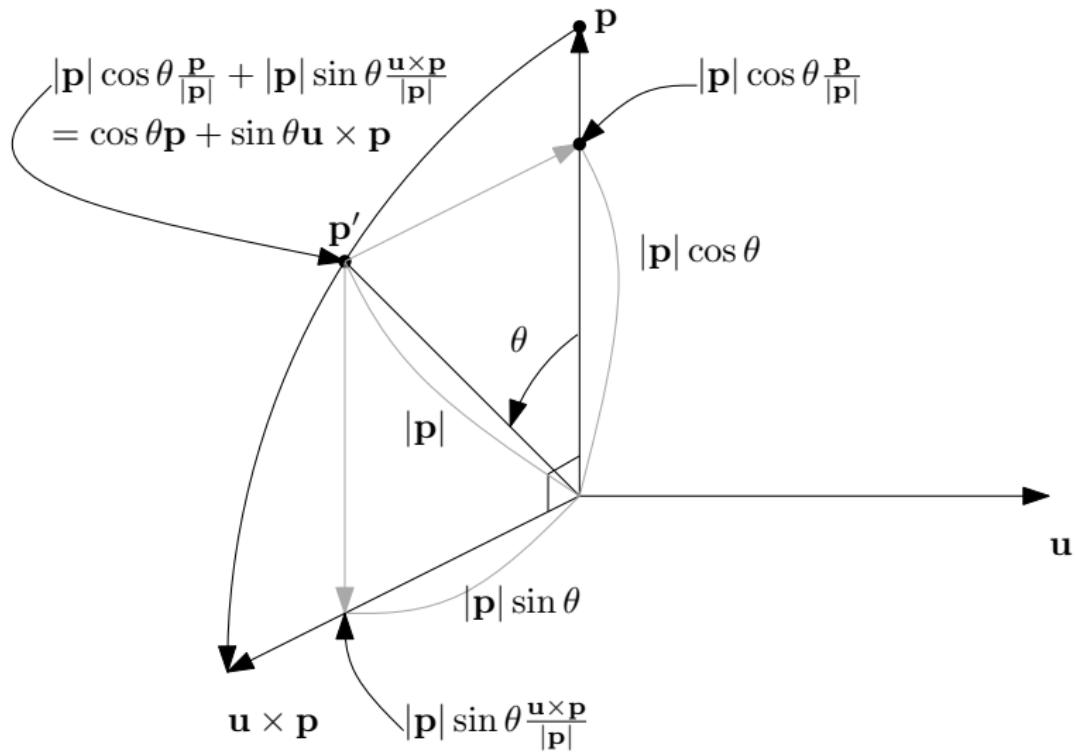
$$\hat{q} = (\cos \theta, \sin \theta \mathbf{u}), |\mathbf{u}| = 1, |\hat{q}| = 1$$

- 벡터 \mathbf{u} 는 단위벡터 (\mathbf{u} 가 어떤 방향이나 축을 표현)

$$\hat{p}' = (d_{p'}, \mathbf{p}') = \hat{q}\hat{p} = (-\sin \theta \mathbf{u} \cdot \mathbf{p}, \cos \theta \mathbf{p} + \sin \theta \mathbf{u} \times \mathbf{p})$$

사원수와 회전: 사원수의 벡터가 수직인 경우 1/2

- \mathbf{p} 와 \mathbf{u} 가 서로 직교하는 경우



사원수와 회전: 사원수의 벡터가 수직인 경우 2/2

- 어떤 단위 벡터 \mathbf{u} 와 임의의 벡터 \mathbf{p} 는 서로 직교하다고 가정
- 두 벡터를 외적한 $\mathbf{u} \times \mathbf{p}$ 는 \mathbf{p} 를 \mathbf{u} 축을 중심으로 90도 회전한 것
- $\mathbf{u}, \mathbf{p}, \mathbf{u} \times \mathbf{p}$ 는 직교 좌표계의 세 축 위
- 다음과 같은 세 벡터가 직교 좌표축
 - $\mathbf{u}, \frac{\mathbf{p}}{|\mathbf{p}|}, \frac{\mathbf{u} \times \mathbf{p}}{|\mathbf{p}|}$
- \mathbf{p} 를 \mathbf{u} 를 중심축으로 θ 만큼 회전시킨 점 \mathbf{p}'
 - 이 점은 $\frac{\mathbf{p}}{|\mathbf{p}|}$ 축 방향으로의 길이 α 와 $\frac{\mathbf{u} \times \mathbf{p}}{|\mathbf{p}|}$ 축 방향으로의 길이 β 를
안다면 $\alpha \frac{\mathbf{p}}{|\mathbf{p}|} + \beta \frac{\mathbf{u} \times \mathbf{p}}{|\mathbf{p}|}$ 로 표현 가능
 - $\alpha = |\mathbf{p}| \cos \theta$
 - $\beta = |\mathbf{p}| \sin \theta$
- 회전된 좌표 \mathbf{p}' 는 $\cos \theta \mathbf{p} + \sin \theta \mathbf{u} \times \mathbf{p}$
- 두 사원수의 곱으로 얻은 사원수의 벡터 부분과 동일
- 스칼라 부분은 $\mathbf{u} \perp \mathbf{p}$ 의 경우라면 0

사원수와 회전: 일반적 경우 1/7

- \mathbf{p} 와 \mathbf{u} 가 서로 직교하지 않는 일반적 경우
- 스칼라 부분 $-\sin \theta \mathbf{u} \cdot \mathbf{p}$ 이 0이 아님
- 스칼라 값이 0이 될 수 있도록 사원수 곱하기를 두 번 수행
- 하나의 사원수 \hat{q} 를 곱하는 것이 아니라 그 역원 \hat{q}^{-1} 도 같이 곱함

$$\hat{p}' = \hat{q}\hat{p}\hat{p}^* = (\cos \theta, \sin \theta \mathbf{u})(0, \mathbf{p})(\cos \theta, -\sin \theta \mathbf{u})$$

$$\hat{q}\hat{p}\hat{p}^* = (-\sin \theta \mathbf{u} \cdot \mathbf{p}, \cos \theta \mathbf{p} + \sin \theta \mathbf{u} \times \mathbf{p})(\cos \theta, -\sin \theta \mathbf{u})$$

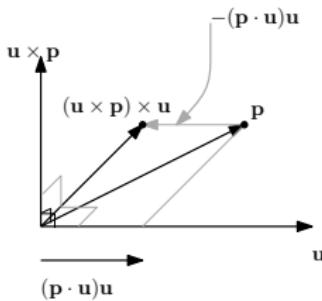
사원수와 회전: 일반적 경우 2/7

사원수 곱셈 연산법에 따라 계산하면 다음을 얻는다.

$$\begin{aligned}\hat{q}\hat{p}\hat{q}^* &= (s, \mathbf{v}) \\ s &= -\sin \theta \cos \theta \mathbf{u} \cdot \mathbf{p} + \sin \theta \cos \theta \mathbf{u} \cdot \mathbf{p} + \sin^2 \theta (\mathbf{u} \times \mathbf{p}) \cdot \mathbf{u} \\ \mathbf{v} &= \cos^2 \theta \mathbf{p} \\ &\quad + \sin \theta \cos \theta \mathbf{u} \times \mathbf{p} \\ &\quad + (\sin^2 \theta \mathbf{u} \cdot \mathbf{p}) \mathbf{u} \\ &\quad - \sin \theta \cos \theta \mathbf{p} \times \mathbf{u} \\ &\quad - \sin^2 \theta \mathbf{u} \times \mathbf{p} \times \mathbf{u}\end{aligned}$$

$\mathbf{u} \times \mathbf{p}$ 과 \mathbf{u} 는 서로 수직이므로, 이 둘의 내적 $(\mathbf{u} \times \mathbf{p}) \cdot \mathbf{u}$ 가 0이다. 따라서 스칼라 부분인 s 가 0.

사원수와 회전: 일반적 경우 3/7



- $\mathbf{u} \times \mathbf{p} \times \mathbf{u}$ 는 $\mathbf{p} - (\mathbf{u} \cdot \mathbf{p})\mathbf{u}$
- $\mathbf{u} \times \mathbf{p} \times \mathbf{u}$ 는 \mathbf{u} 와 $\mathbf{u} \times \mathbf{p}$ 에 동시에 수직인 직교축
- 길이는 \mathbf{p} 와 \mathbf{u} 의 내적을 통해 알 수 있고, 이를 \mathbf{u} 축의 음의 방향으로 떨어트리면 됨
- $\mathbf{u} \times \mathbf{p} \times \mathbf{u} = \mathbf{p} - (\mathbf{p} \cdot \mathbf{u})\mathbf{u}$

$$\hat{q}\hat{p}\hat{q}^* = (0, (\cos^2 \theta - \sin^2 \theta)\mathbf{p} + 2 \sin \theta \cos \theta \mathbf{u} \times \mathbf{p} + (2 \sin^2 \theta \mathbf{u} \cdot \mathbf{p})\mathbf{u})$$

사원수와 회전: 일반적 경우 4/7

$$\begin{aligned}\cos 2\theta &= \cos^2 \theta - \sin^2 \theta \\ \sin 2\theta &= 2 \sin \theta \cos \theta\end{aligned}$$

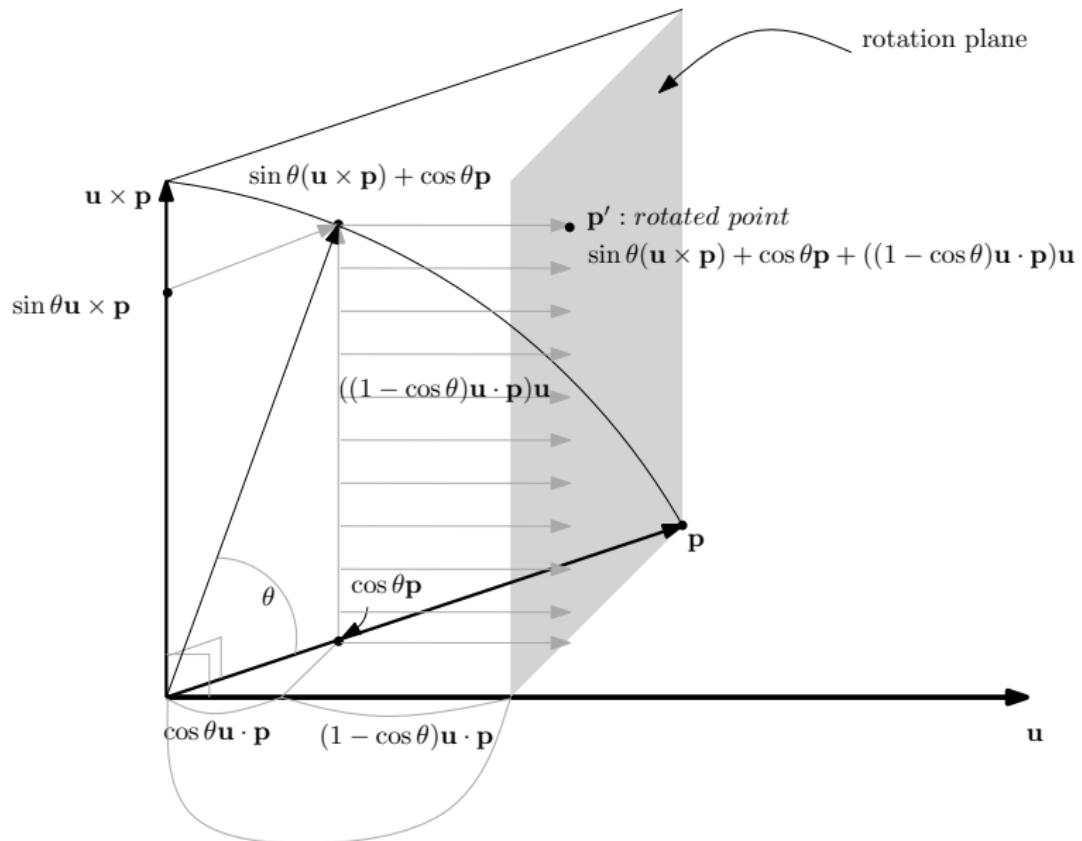
이 항등식을 적용하면 다음을 얻을 수 있다.

$$\hat{q}\hat{p}\hat{q}^* = (0, (\cos 2\theta \mathbf{p} + \sin 2\theta (\mathbf{u} \times \mathbf{p}) + (2 \sin^2 \theta \mathbf{u} \cdot \mathbf{p}) \mathbf{u}))$$

- $1 = \cos^2 \theta + \sin^2 \theta$ 이므로 $\sin^2 \theta = 1 - \cos^2 \theta$
- $2 \sin^2 \theta$ 는 $\sin^2 \theta + \sin^2 \theta = \sin^2 \theta + 1 - \cos^2 \theta$
- $1 - (\cos^2 \theta - \sin^2 \theta)$ 이므로 다음 성립
 - $2 \sin^2 \theta = 1 - \cos 2\theta$

$$\hat{q}\hat{p}\hat{q}^* = (0, (\cos 2\theta \mathbf{p} + \sin 2\theta (\mathbf{u} \times \mathbf{p}) + ((1 - \cos 2\theta) \mathbf{u} \cdot \mathbf{p}) \mathbf{u}))$$

사원수와 회전: 일반적 경우 5/7



사원수와 회전: 일반적 경우 6/7

- 점 \mathbf{p} 가 축 \mathbf{u} 를 기준으로 회전
- 회전 과정에 지나는 곡선이 놓인 회색 평면 = 회전 평면
- \mathbf{p} 와 $\mathbf{u} \times \mathbf{p}$ 가 만드는 평면의 원점을 기준을 θ 만큼 회전하여 얻는 점
 - 이 점은 \mathbf{p} 축으로의 길이는 $|\mathbf{p}| \cos \theta$
 - 이 점의 $\mathbf{u} \times \mathbf{p}$ 축 방향으로의 길이는 $|\mathbf{p}| \sin \theta$
 - $\sin \theta(\mathbf{u} \times \mathbf{p}) + \cos \theta \mathbf{p}$
 - 이 점을 회전 평면 위로 옮기면 원하는 좌표
- 회전 평면으로 옮기는 데에 필요한 길이는 $(1 - \cos \theta)\mathbf{u} \cdot \mathbf{p}$
- 이 길이만큼 \mathbf{u} 축으로 옮겨 놓는 벡터는 $((1 - \cos \theta)\mathbf{u} \cdot \mathbf{p})\mathbf{u}$

회전의 결과 좌표는

$$\mathbf{p}' = \sin \theta(\mathbf{u} \times \mathbf{p}) + \cos \theta \mathbf{p} + ((1 - \cos \theta)\mathbf{u} \cdot \mathbf{p})\mathbf{u}$$

사원수와 회전: 일반적 경우 7/7

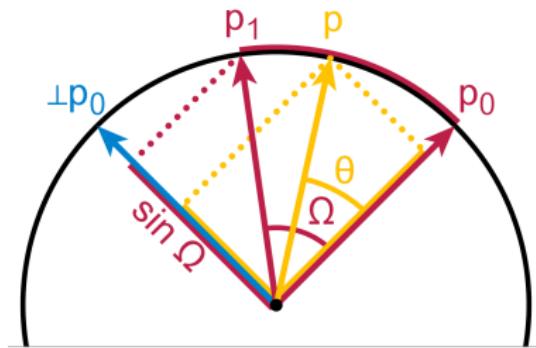
$$\mathbf{p}' = \sin \theta (\mathbf{u} \times \mathbf{p}) + \cos \theta \mathbf{p} + ((1 - \cos \theta) \mathbf{u} \cdot \mathbf{p}) \mathbf{u}$$

어떤 점 \mathbf{p} 를 \mathbf{u} 축을 중심으로 θ 만큼 회전하여 \mathbf{p}' 를 얻고 싶을 때

- $\hat{p} = (0, \mathbf{p})$
- $\hat{q} = (\cos \frac{\theta}{2}, \sin \frac{\theta}{2} \mathbf{u})$
- $\hat{p}' = (0, \mathbf{p}') = \hat{q} \hat{p} \hat{q}^*$

사원수의 보간

- 사원수가 회전을 의미한다면, 그래픽스에서 두 사원수를 보간하는 일은 빈번
- 보간: $t = 0$ 에서 \hat{q}_0 이고, $t = 1$ 에서 \hat{q}_1 인 사원수 \hat{q}_t 구하기
- 사원수는 행렬에 비해 보간이 쉽다는 장점

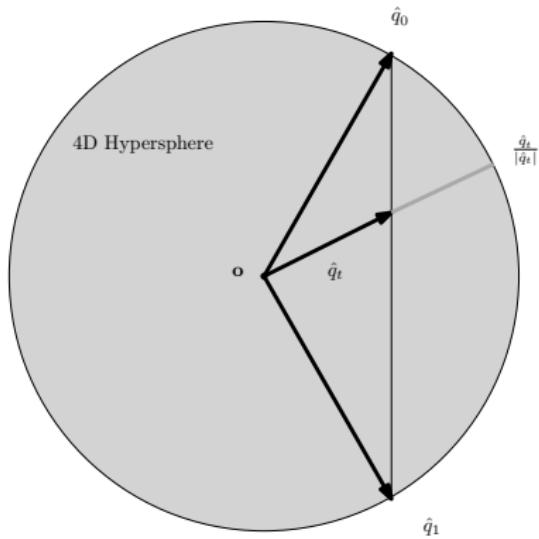


$$\hat{q}_0 = (s_0, u_0, v_0, w_0)$$

$$\hat{q}_1 = (s_1, u_1, v_1, w_1)$$

사원수의 보간: 단순한 선형보간

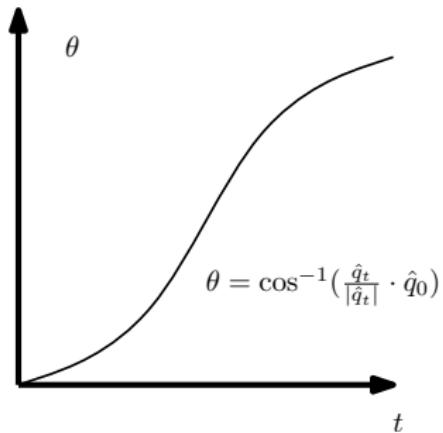
- 간단한 선형보간
 - $\hat{q}_t = (ts_1 + (1-t)s_0, tu_1 + (1-t)u_0, tv_1 + (1-t)v_0, tw_1 + (1-t)w_0)$
 - 즉, $\hat{q}_t = t\hat{q}_1 + (1-t)\hat{q}_0$
- 매우 간단하고 빠르다는 장점
- 보간이 적용되는 동안 사원수의 길이가 길이가 변하는 단점



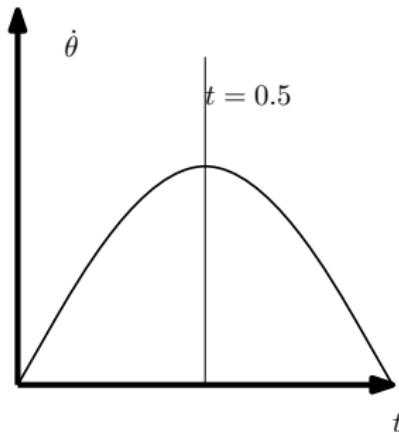
사원수의 보간: 간단한 개선 방법

- 사원수가 항상 초구면의 표면에 있도록 그 길이를 조정
- $\hat{q}_t / |\hat{q}_t|$ 로 정규화
- 길이는 유지되지만 회전속도는 일정하지 않음
 - 극단적인 상황: \hat{q}_0 와 \hat{q}_1 이 서로 반대 방향
 - 선형 보간하여 얻는 사원수는 $t = 0.5$ 가 될 때까지는 초기의 회전각
 - $t = 0.5$ 시점을 지나면 바로 다음 회전각으로 전환

사원수의 단순한 선형보간으로 얻어지는 각도의 변화와 각속도의 변화



(a) 각도의 변화

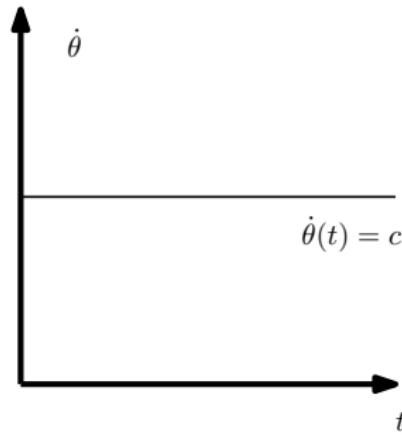
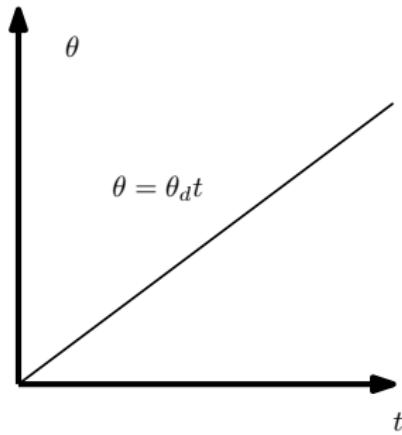


(b) 각속도의 변화

사원수의 보간: 구면보간(球面補間) 혹은 Slerp

- 사원수의 보간은 각도 θ 가 선형으로 보간되어야 함

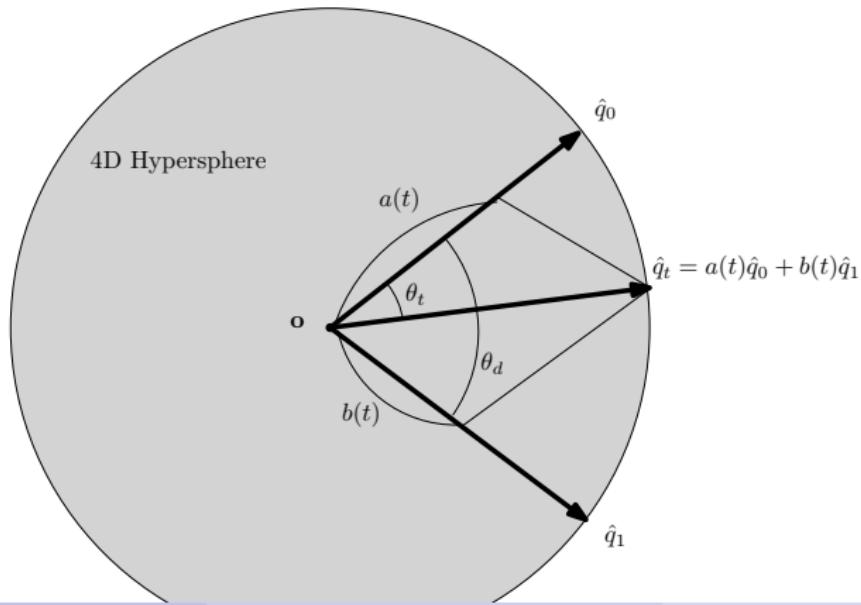
구면 보간을 통해 얻어야 하는 사원수 보간의 각도의 변화와 각속도의 변화



구면보간(球面補間): 등각속도 구면보간의 개념

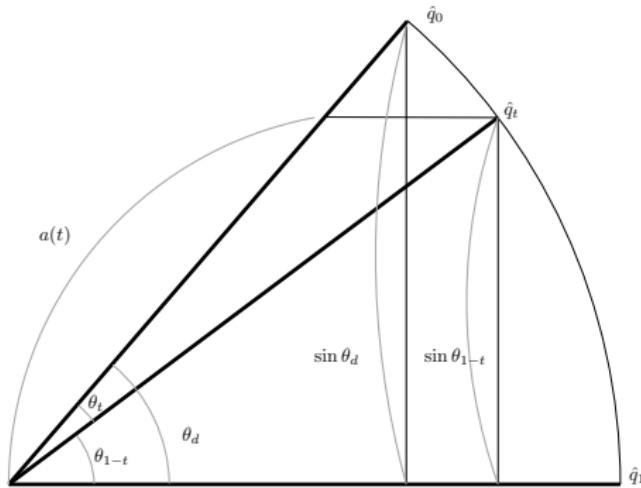
- 구면보간

- 시작과 끝 회전을 표현하는 사원수 \hat{q}_0 와 \hat{q}_1 에 적절한 가중치 $a(t)$ 와 $b(t)$ 적용
- 가중치가 적용된 두 사원수를 합성
- 현재 시간 t 에 제한조건을 만족하는 $a(t)$ 와 $b(t)$ 를 찾는 작업



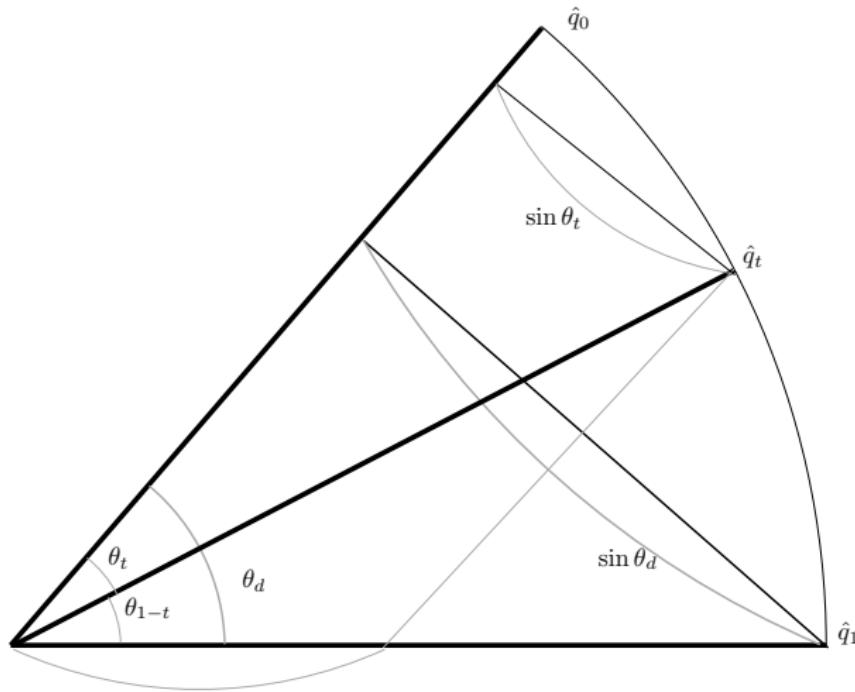
구면보간(球面補間): $a(t)$ 구하기

- \hat{q}_0 에서 \hat{q}_1 까지의 회전각 전체가 θ_d
- 시간 t 에서 보간된 사원수 \hat{q}_t 와 \hat{q}_0 가 이루는 각이 θ_t
- θ_d 에서 θ_t 를 뺀 각도를 θ_{1-t}
- $a(t)$ 와 $|\hat{q}_0|$ 의 비(比)는 결국 $|\hat{q}_t| \sin \theta_{1-t}$ 과 $|\hat{q}_0| \sin \theta_d$ 의 비
- $a(t) = \frac{\sin \theta_{1-t}}{\sin \theta_d}$



구면보간(球面補間): $b(t)$ 구하기

- $b(t)$ 와 $|\hat{q}_1|$ 의 비는 $|\hat{q}_t| \sin \theta_t$ 의 값과 $|\hat{q}_1| \sin \theta_d$ 가 이루는 비
- $b(t) = \frac{\sin \theta_t}{\sin \theta_d}$



구면보간(球面補間) 계산법

- $a(t)$ 와 $b(t)$ 를 구하고 나면, 보간된 사원수 \hat{q}_t
- $\hat{q}_t = a(t)\hat{q}_0 + b(t)\hat{q}_1$
- 사원수가 동일한 각속도로 부드럽게 보간된다. 이러한 보간 방법은 구면보간
- “slerp”이라는 이름으로 종종 부름
- 이때 사원수의 크기는 언제나 1

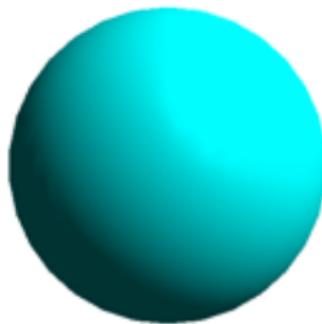
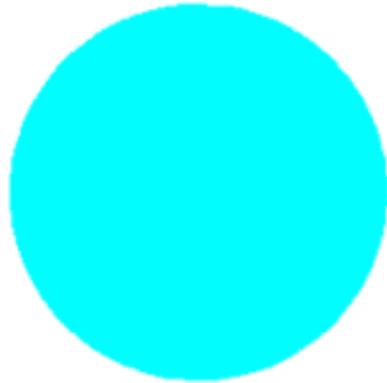
$$\begin{aligned}\theta_d &= \cos^{-1}(\hat{q}_0 \cdot \hat{q}_1) \\ s &= \sin \theta_d = \sqrt{1 - (\hat{q}_0 \cdot \hat{q}_1)^2} \\ \hat{q}_t &= \frac{\sin \theta_{1-t}}{s} \hat{q}_0 + \frac{\sin \theta_t}{s} \hat{q}_1\end{aligned}$$

조명

조명 모델의 이해

음영 계산의 필요성

- 음영(陰影) 계산, 혹은 셰이딩(shading)은 어떤 물체의 표면에서 어두운 부분과 밝은 부분을 서로 다른 밝기로 그려내는 것
- 모든 면을 동일한 색으로 그리면 입체감이 없다.



조명과 재질

음영 계산에 필요한 두 가지 핵심 요소

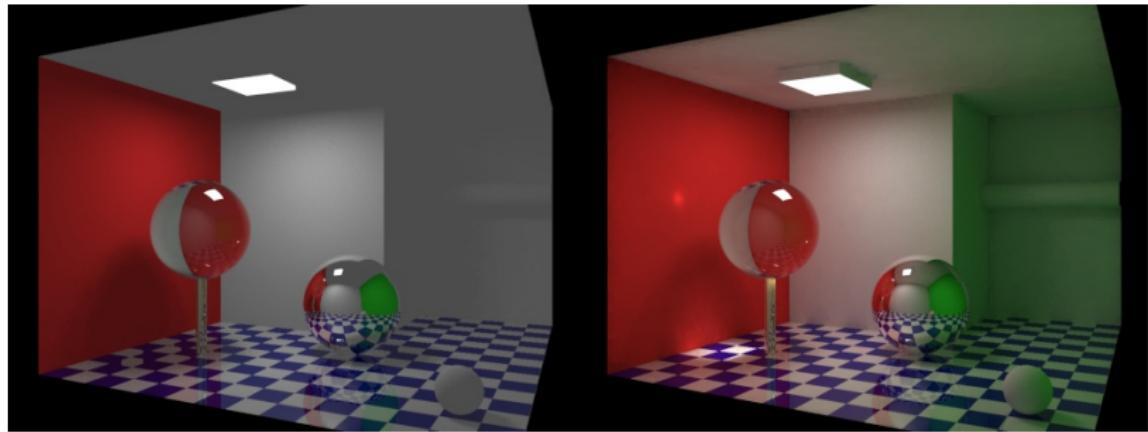
- 빛

- 컴퓨터 그래픽스의 최종 결과는 우리의 눈을 통해 입력되는 시각 정보
- 눈이 인지하는 신호는 전자기파
- 우리가 인지할 수 있는 영역의 전자기파: 가시광선(可視光線)
- 이 가시광선을 일상적으로 ‘빛’이라 부름

- 재질(材質,material)

- 같은 빛이라고 모든 물체가 같은 색으로 보이지는 않음
- 물체는 저마다의 특성에 따라 서로 다른 방식으로 빛을 반사
- 물체가 가진 반사 특성을 재질이라고 함

지역조명과 전역조명



광원 모델

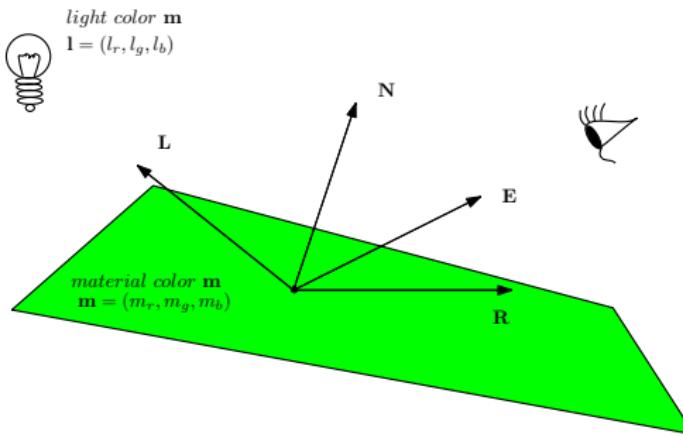
일반적인 광원(光源, light source)은 다루기가 쉽지 않다. 하나의 광원은 일정한 면적이나 체적을 가지기 때문에 이 광원에서 나온 빛들을 적분해야 한다. 실제 실시간 렌더링에서는 광원을 하나의 점이나 방향으로 보는 단순화된 모델을 사용한다. 사용되는 광원은 다음과 같은 것들이 있다.

광원 종류	특징
점광원	광원의 위치와 색으로 결정. 전방향(omnidirection)으로 빛 진행.
집중광원	점광원에서 일부 입체각으로 빛을 제한.
방향광원	특정한 위치가 아니라 방향에 광원 존재.
주변광원	모든 곳에 동일하게 가해지는 빛.

지역 조명 모델 - 풍(Phong) 모델

정반사와 난반사, 주변광 반사 특성을 표현할 수 있는 간단하고 빠른 모델
다음과 같은 정보가 필요

- 광원으로 향하는 벡터 **L**
- 카메라(시점)을 향하는 벡터 **E**
- 법선 벡터 **N**
- 이상적인 반사 방향 **R**



퐁(Phong) 모델: 광원과 재질 - 색상

- 퐁 모델은 난반사, 정반사, 주변광을 각각 독립적으로 계산하여 합성
- 각각의 반사 요소에 따라 결정해야 하는 것은 눈을 향해 오는 빛의 강도(intensity)와 색상
- 색상의 결정
 - 어떤 광원에서 나오는 빛의 색상이 $\mathbf{l} = (l_r, l_g, l_b)$
 - 물체의 재질 색상이 $\mathbf{m} = (m_r, m_g, m_b)$
 - 빛의 색상은 이 빛이 눈에 감지되었을 때 우리가 느끼는 색이
 - 빨간 색을 가진 물체의 재질 색상 $(1.0, 0.0, 0.0)$ 은 도착하는 빛의 RGB 3 개 채널에서 R 채널의 빛을 100% 반사한다는 것을 의미

반사되는 빛의 색상은 다음과 같다.

$$\mathbf{c} = (l_r m_r, l_g m_g, l_b m_b)$$

이를 이렇게 표현한다.

$$\mathbf{c} = \mathbf{l} \otimes \mathbf{m}$$

퐁(Phong) 모델: 광원과 재질 - 광강도

- 이 계산은 눈에 감지되는 빛의 색상을 결정
- 같은 색상이라도 밝고 어두울 수 있음
- 이러한 음영을 고려해야 입체적인 물체로 보임
- 이러한 음영은 광강도(光強度, light intensity)에 의해 결정
- 이 광강도의 값은 스칼라(scalar) 값으로 I 로 표현
- 실제로 눈에 보이는 색은 광원과 재질에 의해 결정되는 색상 \mathbf{c} 와 광강도 I 에 의해 다음과 같이 결정

$$\kappa = I\mathbf{c}$$

퐁(Phong) 모델: 최종 결정 반사색

- 퐁 모델은 각각의 반사 요소를 모두 따로 계산
- 난반사 요소와 관련된 값에는 아래첨자 d
- 정반사에는 s
- 주변광 반사에는 a
- 눈에 관측되는 색상은 다음과 같음

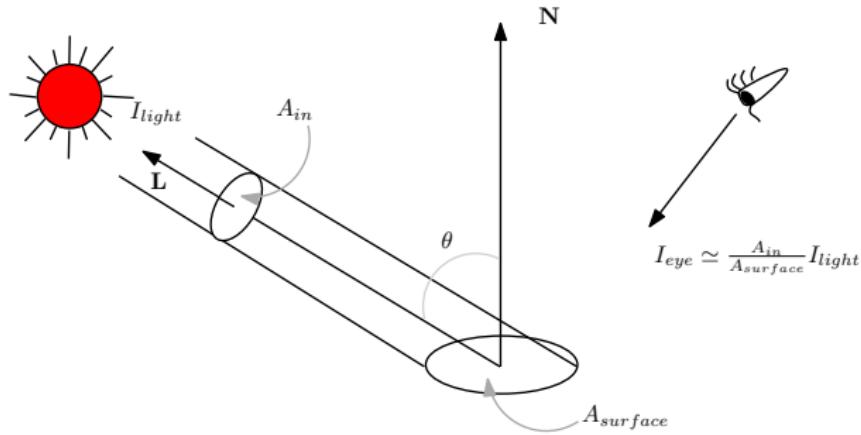
$$\kappa = I_a \mathbf{c}_a + I_d \mathbf{c}_d + I_s \mathbf{c}_s$$

이때, 주변광의 광강도는 상수 값을 가지므로 I_a 는 1로 설정할 수 있다.
따라서 다음 식으로 바꿀 수 있다.

$$\begin{aligned}\kappa &= \mathbf{c}_a + I_d \mathbf{c}_d + I_s \mathbf{c}_s \\ &= \mathbf{l}_a \otimes \mathbf{m}_a + I_d \mathbf{l}_d \otimes \mathbf{m}_d + I_s \mathbf{l}_s \otimes \mathbf{m}_s\end{aligned}$$

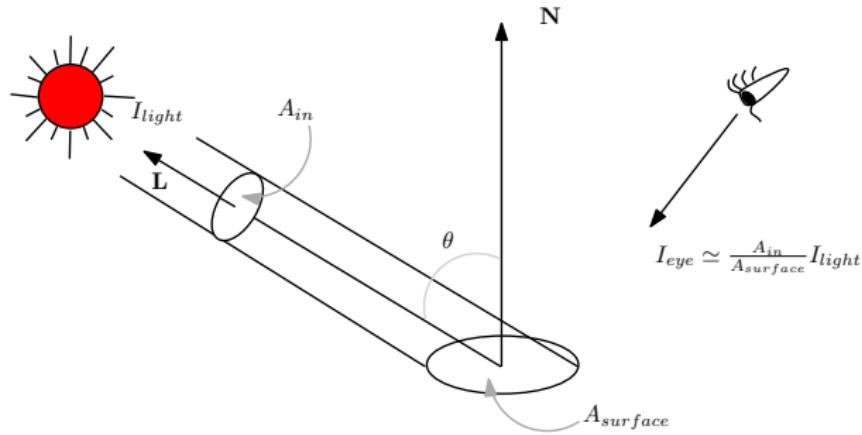
퐁 모델 광강도(intensity)의 계산 - 난반사

- 퐁 모델에서 계산해야 하는 광강도는 난반사 I_d 와 정반사 I_s
- 난반사는 모든 방향에 동등하게 빛이 퍼지는 것으로 가정
- 눈이 어디에 있든지 동일한 색상 관찰
- 눈의 움직임에 따라 변하는 하일라이트(highlight)는 표현하지 못 함
- 색을 칠하려고 하는 한 지점에 대해 어디서 쳐다 보든지 동일한 밝기
- 밝기는 얼마나 많은 에너지가 해당 지점에 떨어지는지에 달려 있음



퐁 모델 광강도(intensity)의 계산 - 난반사 1/2

- 퐁 모델에서 계산해야 하는 광강도는 난반사 I_d 와 정반사 I_s
- 난반사는 모든 방향에 동등하게 빛이 퍼지는 것으로 가정
- 눈이 어디에 있든지 동일한 색상 관찰
- 눈의 움직임에 따라 변하는 하일라이트(highlight)는 표현하지 못 함
- 색을 칠하려고 하는 한 지점에 대해 어디서 쳐다 보든지 동일한 밝기
- 밝기는 얼마나 많은 에너지가 해당 지점에 떨어지는지에 달려 있음

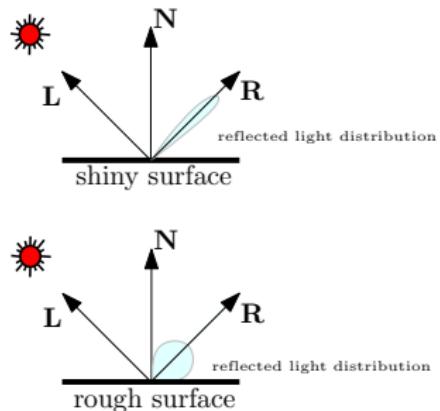
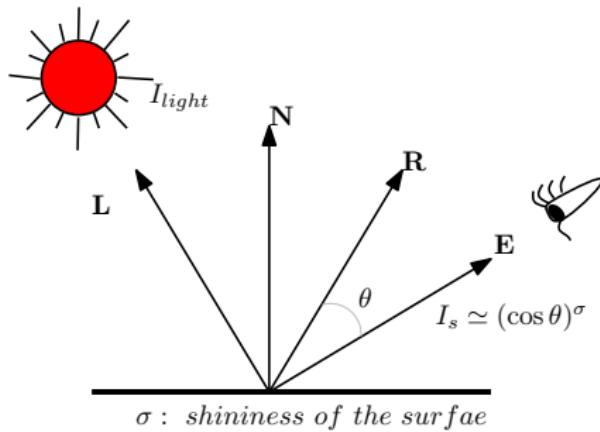


퐁 모델 광강도(intensity)의 계산 - 난반사 2/2

- 이 값은 광원 벡터 \mathbf{L} 과 법선 벡터 \mathbf{N} 이 일치할 때 최대이며, 90도를 이룰 때 0이 됨
- 이 값은 두 벡터의 내적, 즉 두 벡터 사잇각의 코사인(cosine)에 비례한다는 것이 램버트 반사(Lambertian reflectance) 모델
- 따라서 광강도를 계산해야하는 지점에서 빛을 향하는 방향벡터 \mathbf{L} 과 표면 법선벡터 \mathbf{N} 의 내적으로 I_d 를 구할 수 있음

$$I_d = \cos \theta = \mathbf{L} \cdot \mathbf{N} \quad (6)$$

퐁 모델 광강도(intensity)의 계산 - 정반사 1/2



정반사는 거울과 같이 입사각에 대칭되는 방향으로 반사되는 것이다. 그런데, 실제 물체들은 이런 이상적인 정반사가 아니라 반사 방향으로 빛이 강하게 진행하기는 하지만 다른 방향으로 조금씩 빛이 나간다. 퐁 모델에서 사용하는 정반사 모델은 거울과 같은 반사가 아니라 반사 벡터 R 중심으로 퍼지는 반사를 표현한다.

퐁 모델 광강도(intensity)의 계산 - 정반사 2/2

- 정반사는 반사 벡터 \mathbf{R} 근처에서 강하게 관찰되기 때문에 눈을 \mathbf{R} 근처로 가져가야 강한 반사
- 정반사의 광강도 I_s 는 \mathbf{R} 과 \mathbf{E} 의 사잇각의 코사인에 연관
- 물체의 재질에 따라 \mathbf{R} 방향으로 집중되는 정도가 달라짐 - 물체의 반질함(shininess) σ 에 의해 결정

$$I_d = \cos \theta = (\mathbf{R} \cdot \mathbf{E})^\sigma$$

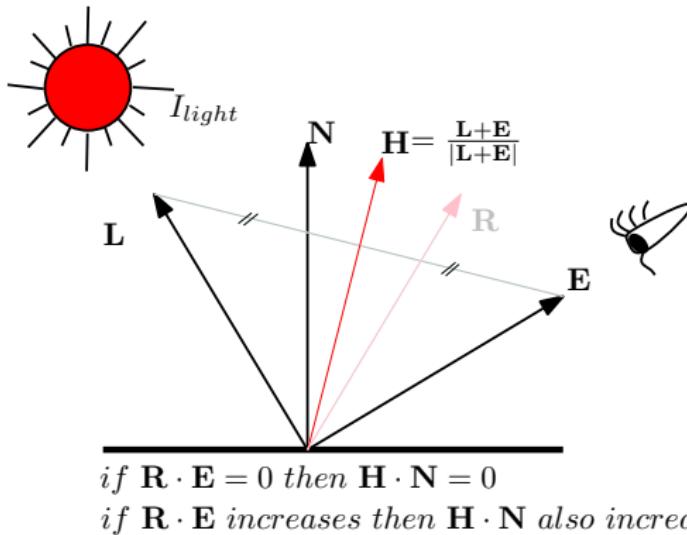
퐁 모델 광강도(intensity)의 계산 - 주변광

주변광의 광강도는 언제나 1이다. 따라서 모든 물체는 주변광과 주변광 반사 재질에 의한 색상 $\mathbf{l}_a \otimes \mathbf{m}_a$ 을 기본적으로 갖게 된다. 이것은 지역조명 기법의 단점인 지나치게 어두운 부분을 없애는 역할을 수행한다.

$$\begin{aligned}\kappa &= \mathbf{c}_a + I_d \mathbf{c}_d + I_s \mathbf{c}_s \\ &= \mathbf{l}_a \otimes \mathbf{m}_a + I_d \mathbf{l}_d \otimes \mathbf{m}_d + I_s \mathbf{l}_s \otimes \mathbf{m}_s\end{aligned}$$

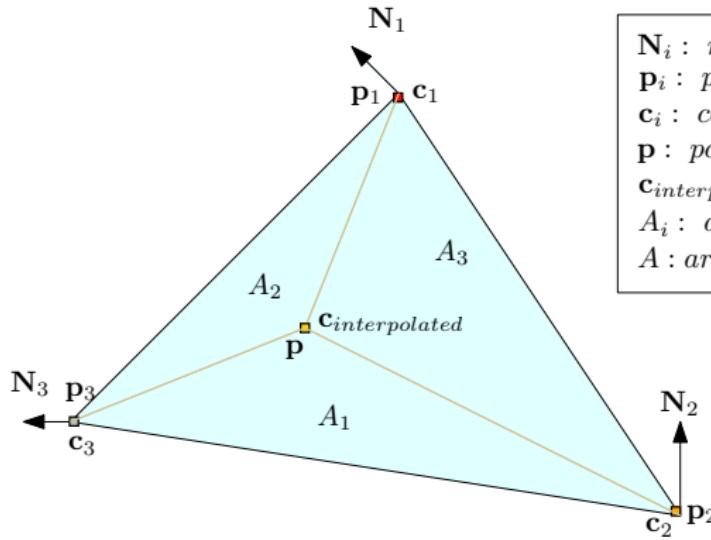
수정된 풍 모델 - 블린(Blinn) 모델

- 풍 모델은 정반사 광강도 계산에서 반사 벡터 \mathbf{R} 을 계산해야 함
- 블린(Blinn)은 이 반사벡터 대신에 반 벡터(halfway vector)라는 것을 도입
- 반 벡터 \mathbf{H} 는 광원 벡터와 시선 벡터 \mathbf{E} 를 더해서 정규화: $\mathbf{H} = \frac{\mathbf{L}+\mathbf{E}}{|\mathbf{L}+\mathbf{E}|}$



구로(Gouraud) 셰이딩

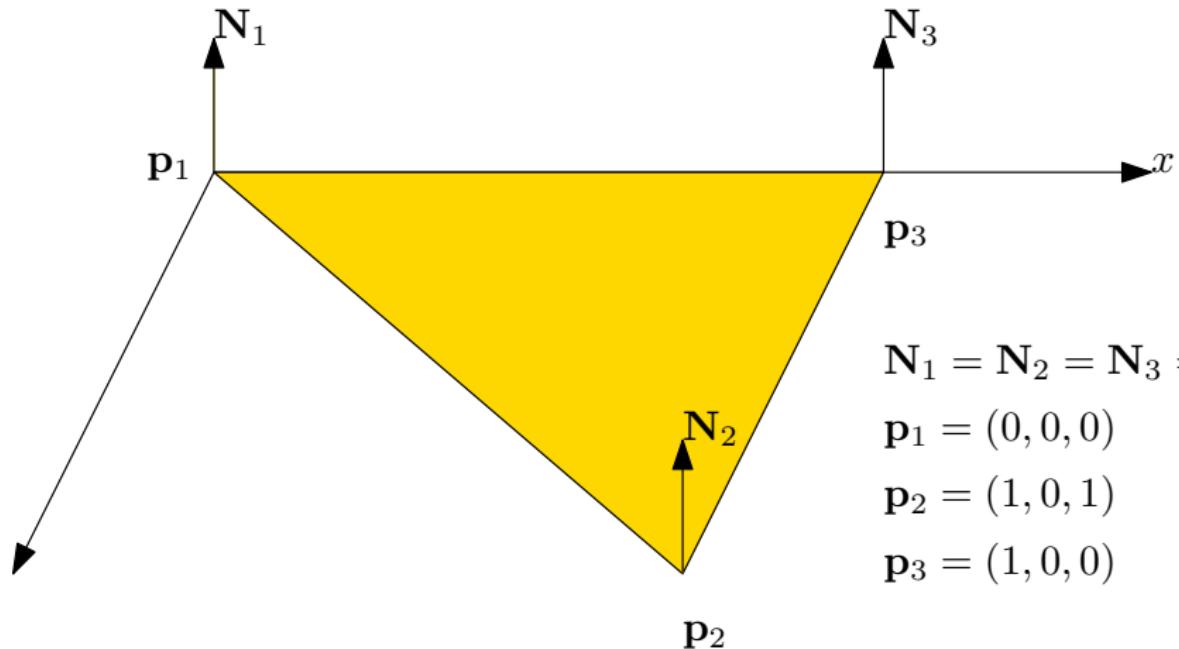
- 각 정점에 법선 벡터 정의
- 법선 벡터와 조명의 관계를 이용하여 정점별 풍 셰이딩
- 정점의 색을 이용하여 내부의 픽셀은 선형보간(linear interpolation)을 통해 얻음



N_i : normal at point i
 p_i : point i
 c_i : color at point i (Phong)
 p : point to be rendered
 $c_{interpolated}$: interpolated color
 A_i : area of $\triangle(p, \forall p_j \neq i)$
 A : area of $\triangle(p_1, p_2, p_3)$

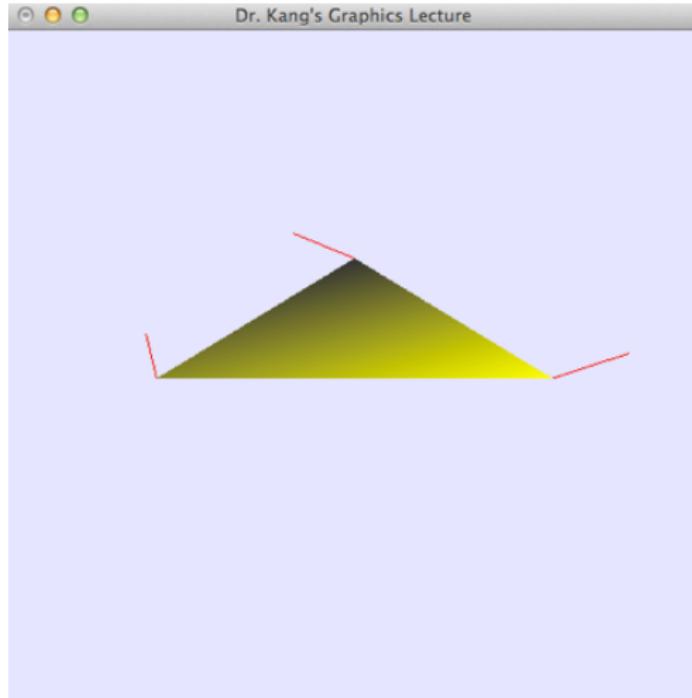
Gouraud Shading
 $c_{interpolated} = \frac{\sum_{i=1}^3 A_i c_i}{A}$

구로(Gouraud) 세이딩 예제



z

구로 세이딩 결과



충돌감지

충돌 처리는 컴퓨터 그래픽스에서 물체를 효율적으로 가시화하거나 자연스러운 애니메이션을 생성할 때 매우 빈번히 요구되는 기술이다. 따라서 효율적인 충돌처리를 위한 다양한 기법이 개발되어 있지만, 여전히 충돌 검출은 많은 경우 성능저하의 주요 원인으로 작용하는 계산상의 병목(bottleneck)이 된다. 이번 장에서는 충돌을 감지하기 위한 다양한 기하적 계산 방법을 소개한다.

충돌 검사의 개념

충돌 검사를 위해서는 충돌 검사 기법이 적용되는 애플리케이션의 환경에 적합한 기법을 선택해야 한다. 이때 기법 선택의 기준이 되는 요소들로는 다음과 같은 것들을 들 수 있다.

- 충돌 객체의 수와 기하적 특성
- 충돌 객체의 물리적 특성
- 충돌 검사의 정확도 요구 정도
- 충돌 검사 결과에서 요구되는 기하적, 물리적 특성
- 계층구조 적용여부

충돌 처리의 두 단계

충돌의 처리는 검사와 반응이라는 두 가지 단계를 거쳐서 이루어진다. 검사 단계는 충돌 처리의 대상이 되는 물체들 가운데 충돌이 일어날 가능성이 있는 물체를 찾아 실제로 서로 부딪히는지를 알아보는 단계이며, 반응 단계는 충돌하는 것으로 판명된 객체들의 충돌을 해소하고 적절한 동작을 취하도록 하는 단계이다. 매우 사실적인 물리 시뮬레이션 환경에서는 충돌된 물체가 어느 시점에 충돌하였는지를 계산하여 그 이후 운동량 보존을 이용하여 물리적으로 적합한 행동을 하도록 하는 작업이다.

거리 기반 검사

충돌 검사의 가장 기초적인 요소는 충돌 검사의 대상이 되는 두 객체가 얼마나 가까이 근접했나를 파악하는 것이다. 어떤 두 물체 사이의 거리는 객체의 기하적 특성에 따라 쉽게 계산할 수 있다. 가장 간단한 점과 점 사이의 충돌부터 구와 평면의 충돌과 같은 다양한 거리 기반 충돌 검사 기법을 살펴 본다.

점(points)들 사이의 충돌

점과 점의 충돌은 두 점이 허용되는 거리 ϵ 이내로 근접했는지를 검사하는 것이다. 충돌 검사의 대상이 되는 두 점을 각각 \mathbf{P} 과 \mathbf{Q} 라고 하면, 검사는 다음과 같이 두 점 사이의 거리 $D(\mathbf{P}, \mathbf{Q})$ 가 ϵ 보다 작은지를 검사하는 것이다.

$$D(\mathbf{P}, \mathbf{Q})^2 = (\mathbf{P}.x - \mathbf{Q}.x)^2 + (\mathbf{P}.y - \mathbf{Q}.y)^2 + (\mathbf{P}.z - \mathbf{Q}.z)^2 < \epsilon^2 \quad (7)$$

이때, \mathbf{P}_x , \mathbf{P}_y , \mathbf{P}_z 는 각각 점 P 위치 벡터의 x, y, z 성분이다.

점과 직선의 거리 1/2

점과 직선의 충돌의 점과 직선의 거리를 구하여 이 값이 허용된 접근 거리 ϵ 이하인지를 검사하는 것이다. 즉, 점과 직선의 거리를 계산하는 방법만 있으면 충돌을 검사할 수 있다. 점 \mathbf{P} 와 직선 \mathbf{L} 의 거리를 계산하는 방법은 점에서 직선으로 수선(垂線)을 내려 그 수선의 발 \mathbf{P}' 과 점 \mathbf{P} 의 거리를 구하는 것이다.

수선의 \mathbf{P}' 를 구하기 위해서는 직선 위의 두 점을 잡아 하나를 시작점 \mathbf{L}_s 라고 하고 다른 하나를 끝점 \mathbf{L}_e 라고 하자. 이 직선은 $\mathbf{L}_s + t(\mathbf{L}_e - \mathbf{L}_s)$ 로 표현되는 점들의 집합이고, 수선의 발을 나타내는 t 의 값 k 는 다음과 같이 구할 수 있다.

$$k = \frac{(\mathbf{P} - \mathbf{L}_s) \cdot (\mathbf{L}_e - \mathbf{L}_s)}{|\mathbf{L}_e - \mathbf{L}_s|^2} \quad (8)$$

점과 직선의 거리 2/2

이렇게 얻은 값을 이용하여 점 \mathbf{P} 에서 직선 \mathbf{L} 로 내린 수선의 발 \mathbf{P}' 를 다음과 같이 구할 수 있다.

$$\mathbf{P}' = \mathbf{L}_s + k(\mathbf{L}_e - \mathbf{L}_s) \quad (9)$$

따라서 점 \mathbf{P} 와 직선 \mathbf{L} 사이의 거리 $\mathcal{D}(\mathbf{P}, \mathbf{L})$ 은 다음과 같이 구할 수 있다.

$$\mathcal{D}(\mathbf{P}, \mathbf{L}) = |\mathbf{P}' - \mathbf{P}| < \epsilon \quad (10)$$

선분 사이의 충돌

선분과 선분의 충돌 역시 두 선분의 거리를 이용하여 검사할 수 있다. 두 선분 \mathbf{L} 과 \mathbf{M} 사이의 거리를 계산한다고 할 때, 가장 먼저 계산해야 하는 것은 두 선분이 가장 가까운 거리로 근접하는 점 \mathbf{L}_c 와 \mathbf{M}_c 를 계산하는 것이다. 이때 \mathbf{L}_c 와 \mathbf{M}_c 는 각각 선분 \mathbf{L} 과 \mathbf{M} 위에 있는 점으로 각각의 선분 위에 있는 점들 하나씩의 선택했을 때, 어느 경우보다 가까운 거리를 갖는 두 점을 의미한다.

이때, 이 두 점 \mathbf{L}_c 와 \mathbf{M}_c 을 연결하는 선분 \mathbf{N} 은 선분 \mathbf{L} 과 \mathbf{M} 에 동시에 수직일 수밖에 없다.

따라서 그 방향은 $\mathbf{L} \times \mathbf{M}$ 과 같은 선분이다. 이 선분의 길이는 \mathbf{L} 과 \mathbf{M} 각각에 임의의 점 하나씩을 선택하여 연결한 벡터를 법선 벡터와 내적한 값이다.

$$D(\mathbf{L}, \mathbf{M}) = \left| (\mathbf{L}_s - \mathbf{M}_s) \cdot \frac{\mathbf{L} \times \mathbf{M}}{|\mathbf{L} \times \mathbf{M}|} \right| \quad (11)$$

평면(平面)과 점

평면 \mathbf{S} 과 점 \mathbf{P} 의 충돌 역시 평면과 점의 거리를 계산하는 것이 핵심이다. 이 거리는 점 \mathbf{P} 에서 평면 \mathbf{S} 에 내린 수선의 발 \mathbf{P}' 과 점 \mathbf{P} 사이의 거리이다. 그런데 이 방법은 수선의 발 \mathbf{P}' 를 구해야 한다는 단점이 있다. 이를 피하고 좀 더 효율적인 방법으로 거리를 계산하는 방법이 있다. 모든 평면은 평면의 법선(法線, normal) 벡터를 가진다. 평면 \mathbf{S} 의 법선 벡터를 \mathbf{N}_S 라고 하자. 이때 법선 벡터 \mathbf{N}_S 는 단위 벡터이다. 또한 평면 위의 한 점을 선택해 이 점을 \mathbf{P}_S 라고 하자. 그러면 이 점 \mathbf{P}_S 에서 \mathbf{P} 로 가는 방향 벡터를 구할 수 있고 이를 \mathbf{D} 라고 하자. 그러면 평면 \mathbf{S} 와 점 \mathbf{P} 사이의 거리 $\mathcal{D}(\mathbf{S}, \mathbf{P})$ 는 다음과 같이 구할 수 있다.

$$\mathcal{D}(\mathbf{S}, \mathbf{P}) = |\mathbf{N}_S \cdot \mathbf{D}| \quad (12)$$

구(球)와 점

구 \mathbf{R} 과 점 \mathbf{P} 의 충돌은 매우 간단하다. 구는 중심 \mathbf{C} 과 반지름 r 을 가진다. 점이 이 구와 충돌한다는 것은 잠과 구의 중심 사이 거리가 반지름 r 보다 작은 값을 가진다는 것이다. 따라서 구와 점의 거리 $D(\mathbf{R}, \mathbf{P})$ 를 이용하여 다음과 같이 검사한다.

$$D(\mathbf{R}, \mathbf{P}) = |\mathbf{P} - \mathbf{C}| < r \quad (13)$$

구(球)들 사이의 충돌

구 \mathbf{R}_1 과 구 \mathbf{R}_2 사이의 충돌은 두 구의 중심 사이의 거리가 두 구의 반지를 r_1 과 r_2 를 합한 값보다 작은지를 검사하면 된다.

$$\mathcal{D}(\mathbf{R}_1, \mathbf{R}_2) = |\mathbf{C}_1 - \mathbf{C}_2| < r_1 + r_2 \quad (14)$$

구(球)와 선분의 충돌 1/2

구 \mathbf{R} 과 선분 \mathbf{L} 의 충돌을 검사하기 위해서는 구와 선분의 거리를 구해야한다. 이때 선분 \mathbf{L} 은 시작점 \mathbf{L}_s 와 끝점 \mathbf{L}_e 로 표현된다. 구와 선분의 거리는 구의 중심과 선분 위의 한 점을 기준으로 계산하는데, 선분위의 한 점은 구의 중심에서 선분으로 수선을 내렸을 때 선분에 닿는 수선의 발이다. 이 수선의 발을 계산하는 방법은 내적(内積)을 이용하여 구할 수 있다. 구의 중심을 \mathbf{C} 라 하고, 수선의 발을 \mathbf{C}' 라고 하면 수선의 발은 선분 위에 있으므로 $\mathbf{L}_s + t(\mathbf{L}_e - \mathbf{L}_s)$ 로 표현되고 여러 점들 가운데 하나이다. 수선의 발을 의미하는 t 파라미터 값 k 는 다음과 같이 쉽게 계산할 수 있다.

$$k = \frac{(\mathbf{C} - \mathbf{L}_s) \cdot (\mathbf{L}_e - \mathbf{L}_s)}{|\mathbf{L}_e - \mathbf{L}_s|^2} \quad (15)$$

구(球)와 선분의 충돌 2/2

구의 중심과 수선의 발 사이의 거리를 이용하여 구 \mathbf{R} 와 선분 \mathbf{L} 의 거리 $\mathcal{D}(\mathbf{R}, \mathbf{L})$ 을 계산하고, 이 값이 구의 반지름 r 보다 작은지를 검사한다.

$$\mathcal{D}(\mathbf{R}, \mathbf{L}) = |\mathbf{L}_s + k(\mathbf{L}_e) - \mathbf{L}_s - C| < r \quad (16)$$

그런데 얻어낸 k 값이 0에서 1 사이의 값이면 수선의 발이 선분 위에 있지만, 0보다 작거나 1보다 크면 선분 밖에 있다. 이때는 선분의 시작점 혹은 끝점과 구의 중심 사이의 거리로 계산해야 한다.

$$0 \leq k \leq 1 \quad (17)$$

$$\mathcal{D}(\mathbf{R}, \mathbf{L}) = |\mathbf{L}_s + k(\mathbf{L}_e) - \mathbf{L}_s - C| < r$$

$$k < 0$$

$$\mathcal{D}(\mathbf{R}, \mathbf{L}) = |\mathbf{L}_s - C| < r$$

$$k > 1$$

$$\mathcal{D}(\mathbf{R}, \mathbf{L}) = |\mathbf{L}_e - C| < r$$

구(球)와 평면의 충돌

구 \mathbf{R} 와 평면 \mathbf{S} 의 충돌을 검사하기 위해 계산해야 하는 구와 평면의 거리는 기본적으로 ??에서 살펴본 점과 평면의 거리와 동일한 계산이 요구된다. 즉, 평면과 구의 중심 \mathbf{C} 의 거리를 계산하여 이 거리가 구의 반지를 r 보다 작은지를 검사하면 된다.

평면 \mathbf{S} 의 단위길이 법선 벡터를 \mathbf{N}_S 라고 하자. 평면 위에 있는 임의의 한 점을 선택해 이 점을 \mathbf{P}_S 라고 하고, 이 점에서 구의 중심으로 가는 벡터 $\mathbf{C} - \mathbf{P}_S$ 를 \mathbf{D} 라고 하자. 그러면 평면 \mathbf{S} 와 구 \mathbf{R} 사이의 거리 $\mathcal{D}(\mathbf{S}, \mathbf{R})$ 는 다음과 같이 구하고 이 값이 구의 반지름 r 보다 작은지를 검사한다.

$$\mathcal{D}(\mathbf{S}, \mathbf{R}) = |\mathbf{N}_S \cdot \mathbf{D}| < r \quad (18)$$

교차점(交叉點)과 교차선(交叉線)

평면과 직선이 평행하지 않다면 반드시 한 점에서 만나게 된다. 이렇게 평면과 직선이 만나는 지점은 다양한 경우에 중요하게 사용될 수 있다. 비슷한 경우로 두 평면이 평행하지 않는 경우 반드시 하나의 직선에서 교차하게 된다. 이러한 교차되는 점과, 직선을 각각 교차점(交叉點, intersection point)과 교차선(交叉線, intersection line)이라고 한다.

선분과 평면의 교차점

평행하지 않는 평면 S 와 선분 L 이 교차하는 점 Q 를 구해 보자. 이 교차점은 당연히 선분 L 위에 존재하므로, 선분의 시작점 L_s 와 끝점 L_e 를 이용하여 다음과 같이 표현되는 점이다. $L_s + t(L_e - L_s)$. 교차점을 찾는 것은 이 파라미터 t 의 적당한 값 k 를 구하는 것이다. 주어진 평면 S 의 법선 벡터를 \mathbf{N}_S 라고 하고, 이 평면 위에 있는 임의의 점을 P 라고 하자. $L_e - L_s$ 벡터와 법선 벡터의 내적(内積)은 이 벡터를 법선 벡터에 사상(寫像, projection)했을 때 얻어지는 선분의 길이이다. 또한 교차점과 시작점 사이의 부분을 법선 벡터에 사상했을 때 얻어지는 선분의 길이는 임의의 점 P 와 시작점 L_s 사이를 잇는 선분을 사상한 것과 같은 결과를 얻는다.

$$k = \frac{\mathbf{N}_S \cdot (\mathbf{P} - \mathbf{L}_s)}{\mathbf{N}_S \cdot (\mathbf{L}_e - \mathbf{L}_s)} \quad (19)$$

따라서 교차점은 다음과 같다.

$$\mathbf{Q} = \mathbf{L}_s + \frac{\mathbf{N}_S \cdot (\mathbf{P} - \mathbf{L}_s)}{\mathbf{N}_S \cdot (\mathbf{L}_e - \mathbf{L}_s)} (\mathbf{L}_e - \mathbf{L}_s) \quad (20)$$

두 평면의 교차선 1/2

두 평면 \mathbf{S} 와 \mathbf{T} 가 교차하면 하나의 직선 \mathbf{L} 을 생성하는데, 이 직선을 교차선이라고 한다. 교차선의 방향은 두 평면이 가진 단위 법선 벡터 \mathbf{N}_S 와 \mathbf{N}_T 의 외적(外積)으로 구할 수 있다. 즉, 교차선의 방향 \mathbf{D}_L 은 다음과 같다.

$$\mathbf{D}_L = \mathbf{N}_S \times \mathbf{N}_T \quad (21)$$

직선은 그 방향만으로는 결정되지 않는다. 따라서 직선이 지나는 점을 구해야 하는데 이를 구하는 방법은 선형 방정식을 푸는 것이다.

평면 \mathbf{S} 와 \mathbf{T} 는 각각 법선 벡터 \mathbf{N}_S 와 \mathbf{N}_T , 그리고 원점에서 법선벡터 방향으로의 이동 d_S, d_T 로 다음과 같이 표현된다.

$$\begin{aligned} \mathbf{S} : \mathbf{N}_S^x x + \mathbf{N}_S^y y + \mathbf{N}_S^z z &= d_S \\ \mathbf{T} : \mathbf{N}_T^x x + \mathbf{N}_T^y y + \mathbf{N}_T^z z &= d_T \end{aligned} \quad (22)$$

두 평면의 교차선 2/2

우리가 구하는 점은 이 두 식을 모두 만족하는 점 \mathbf{P} 이다. 이 직선이 z 축과 평행하지 않다면 반드시 $x = 0$ 이 되는 점이 존재한다. 따라서 점 \mathbf{P} 는 $x = 0$ 인 한 점으로 하여 위 두 식을 다음과 같이 바꾸어 푼다.

$$\begin{aligned}\mathbf{N}_S^y y + \mathbf{N}_S^z z &= d_S \\ \mathbf{N}_T^y y + \mathbf{N}_T^z z &= d_T\end{aligned}\tag{23}$$

위의 두 식을 풀면 \mathbf{P}_y 와 \mathbf{P}_z 의 값은 다음과 같이 구할 수 있다.

$$\begin{aligned}\mathbf{P}_y &= \frac{d_S \mathbf{N}_T^z - d_T \mathbf{N}_S^z}{\mathbf{N}_S^y \mathbf{N}_T^z - \mathbf{N}_T^y \mathbf{N}_S^z} \\ \mathbf{P}_z &= \frac{d_S \mathbf{N}_T^y - d_T \mathbf{N}_S^y}{\mathbf{N}_S^z \mathbf{N}_T^y - \mathbf{N}_T^z \mathbf{N}_S^y}\end{aligned}\tag{24}$$

따라서 교차선 \mathbf{L} 은 다음과 같다.

$$\mathbf{L} = \mathbf{P} + t \mathbf{D}_L\tag{25}$$

직육면체 객체의 충돌

직육면체는 6 개의 면(面), 8 개의 정점(頂點, vertex), 12 개의 선분으로 이루어진 기하객체이다. 12 개의 선분은 4 개가 한 묶음으로 동일한 방향을 취한다. 따라서 직육면체 선분들의 방향은 모두 3 가지 방향을 향하게 된다.

직육면체는 공간상에 임의의 방향으로 변환될 수 있지만, 직육면체를 구성하는 선분들의 방향이 좌표축과 일치하는 경우 매우 간단히 충돌 검사를 수행할 수 있다. 이렇게 직육면체의 모든 선분 각각이 좌표축 3 개 가운데 어느 하나와 일치하는 것을 축정렬(軸整列, axis aligned) 경계 상자라고 하며, 간단히 AABB(axis aligned bounding box)라고 한다. 이와 달리 축에 정렬되지 않고 임의의 방향으로 회전되어 있는 경계 상자를 방향이 있는 경계 상자라고 하고 OBB(oriented bounding box)라고 한다

축정렬 경계 상자(axis aligned bounding box) AABB와 점의 충돌

AABB와 점의 충돌은 점이 AABB의 내부에 들어 왔는지 그렇지 않은지를 검사함으로써 알 수 있다. 어떤 점 \mathbf{P} 가 AABB의 내부에 있을 조건은 다음과 같다.

$$\begin{aligned}m_x &\leq \mathbf{P}_x \leq M_x \\m_y &\leq \mathbf{P}_y \leq M_y \\m_z &\leq \mathbf{P}_z \leq M_z\end{aligned}\tag{26}$$

AABB와 선분의 충돌 1/7

AABB와 선분 L 의 충돌은 AABB의 모든 면으로 주어진 선분을 잘라서 상자의 내부에 있는 것만 남기는 것이다. 이때 선분의 시작점은 L_s 라 하고 끝점은 L_e 라 할 때, 시작점 L_s 는 항상 상자의 외부에 있다고 가정하자. 이제 AABB의 각 축별 최소값과 최대값을 이용하여 선분을 분할하는 방법을 살펴 보자. 우선 최소값을 표현하는 평면을 이용하여 선분을 분할하는 경우를 살펴 보자. 최소값은 각 축마다 있으므로 3 가지가 있다. 이 가운데 x 축 최소값을 표현하는 평면 $x = m_x$ 에 의해 선분을 분할하다고 가정하다. L_s^x 와 L_e^x 모두 m_x 값보다 작은 값을 가진다면 이 선분은 주어진 AABB와 충돌할 수가 없다. 이런 경우는 충돌하지 않는 것으로 판명되는 것이다.

AABB와 선분의 충돌 2/7

충돌이 발생한 경우 즉, \mathbf{L}_s^x 와 \mathbf{L}_e^x 가 이 면의 서로 다른 쪽에 놓여 있는 경우에는 두 값 가운데 m_x 보다 작은 값을 m_x 로 고친다. 이때, x 축 좌표를 변경했으므로 적절히 y , z 축 좌표도 고쳐야 한다. 예를 들어 \mathbf{L}_s^x 가 m_x 보다 작고 \mathbf{L}_e^x 는 m_x 보다 큰 값을 가진다고 하면, $\mathbf{L}_s^x = m_x$ 로 새로 선분을 변경해야 하며, \mathbf{L}_s^y 와 \mathbf{L}_s^z 역시 선분을 구성하는 두 좌표 사이의 x 축 성분의 구간에서 잘려나간 비율만큼 조정이 되어야 한다. 이렇게 x 축 최소값 성분으로 선분을 재설정하는 작업을 $CUT_{min}^x(\mathbf{L})$ 이라고 하면 이 작업은 다음과 같다. $CUT_{min}^y(\mathbf{L})$ 와 $CUT_{min}^z(\mathbf{L})$ 도 비슷한 방법으로 구현할 수 있다.

AABB와 선분의 충돌 3/7

CUT_{min}^x(\mathbf{L}):

if($\mathbf{L}_s^x < m_x \wedge \mathbf{L}_e^x < m_x$) Collision Impossible (return FAI($\mathcal{L}\mathcal{T}$))

if($\mathbf{L}_s^x < m_x \wedge \mathbf{L}_e^x > m_x$)

$$t = \frac{m_x - \mathbf{L}_s^x}{\mathbf{L}_e^x - \mathbf{L}_s^x}$$

$$\mathbf{L}_s^x = m_x$$

$$\mathbf{L}_s^y = \mathbf{L}_s^y + t(\mathbf{L}_e^y - \mathbf{L}_s^y)$$

$$\mathbf{L}_s^z = \mathbf{L}_s^z + t(\mathbf{L}_e^z - \mathbf{L}_s^z)$$

AABB와 선분의 충돌 4/7

CUT_{min}^x(\mathbf{L}):

$$if(\mathbf{L}_s^x > m_x \bigwedge \mathbf{L}_e^x < m_x) \quad (28)$$

$$t = \frac{m_x - \mathbf{L}_e^x}{\mathbf{L}_s^x - \mathbf{L}_e^x}$$

$$\mathbf{L}_e^x = m_x$$

$$\mathbf{L}_e^y = \mathbf{L}_e^y + t(\mathbf{L}_s^y - \mathbf{L}_e^y)$$

$$\mathbf{L}_e^z = \mathbf{L}_e^z + t(\mathbf{L}_s^z - \mathbf{L}_e^z)$$

$$if(\mathbf{L}_s^x > m_x \bigwedge \mathbf{L}_e^x > m_x) \quad (29)$$

Do Nothing

AABB와 선분의 충돌 5/7

비슷한 방법으로 최대값에 의한 선분 분할도 진행해야 한다.

CUT^x_{MAX}(\mathbf{L}):

$$if(\mathbf{L}_s^x > M_x \bigwedge \mathbf{L}_e^x > M_x) \quad (30)$$

Collision Impossible (return FAIL)

$$if(\mathbf{L}_s^x > M_x \bigwedge \mathbf{L}_e^x < M_x)$$

$$t = \frac{M_x - \mathbf{L}_s^x}{\mathbf{L}_e^x - \mathbf{L}_s^x}$$

$$\mathbf{L}_s^x = M_x$$

$$\mathbf{L}_s^y = \mathbf{L}_s^y + t(\mathbf{L}_e^y - \mathbf{L}_s^y)$$

$$\mathbf{L}_s^z = \mathbf{L}_s^z + t(\mathbf{L}_e^z - \mathbf{L}_s^z)$$

AABB와 선분의 충돌 6/7

CUT^x_{MAX}(\mathbf{L}):

$$if(\mathbf{L}_s^x < M_x \bigwedge \mathbf{L}_e^x > M_x) \quad (31)$$

$$t = \frac{M_x - \mathbf{L}_e^x}{\mathbf{L}_s^x - \mathbf{L}_e^x}$$

$$\mathbf{L}_e^x = M_x$$

$$\mathbf{L}_e^y = \mathbf{L}_e^y + t(\mathbf{L}_s^y - \mathbf{L}_e^y)$$

$$\mathbf{L}_e^z = \mathbf{L}_e^z + t(\mathbf{L}_s^z - \mathbf{L}_e^z)$$

$$if(\mathbf{L}_s^x < M_x \bigwedge \mathbf{L}_e^x < M_x) \quad (32)$$

Do Nothing

AABB와 선분의 충돌 7/7

AABB와 선분 \mathbf{L} 의 충돌 검사는 위의 CUT 함수를 선분 \mathbf{L} 에 6 번 (면의 수만큼) 적용하여 남은 선분이 있는지를 검사하는 것이다.

DetectCollision(AABB, \mathbf{L})

if(CUT_{min}^x(\mathbf{L}) == FAIL) stop detection

if(CUT_{min}^y(\mathbf{L}) == FAIL) stop detection

if(CUT_{min}^z(\mathbf{L}) == FAIL) stop detection

if(CUT_{MAX}^x(\mathbf{L}) == FAIL) stop detection

if(CUT_{MAX}^y(\mathbf{L}) == FAIL) stop detection

if(CUT_{MAX}^z(\mathbf{L}) == FAIL) stop detection

Collision Point = \mathbf{L}_s

AABB와 평면의 충돌 1/2

AABB와 평면 \mathbf{S} 의 충돌은 평면과 AABB가 얼마나 가까워질 수 있는지를 계산해야 한다. AABB와 평면에 허용되는 접근 거리는 평면의 방향에 따라 달라진다. 3차원 AABB와 평면의 충돌은 2차원의 축정렬(軸整列) 사각형과 직선의 충돌 문제를 확장한 것으로 볼 수 있다. 이때 직선이 x 축과 동일한 방향이라면 사각형의 중심과 직선 사이에 허용되는 접근 거리는 $|M_y - m_y|/2$ 가 되며 이 직선이 y 축과 같은 방향이라면 허용되는 접근 거리가 $|M_x - m_x|/2$ 가 될 것이다. 물론 직선이 좌표축과 다른 방향을 가진다면 이와 다른 어떤 값이 될 것이다. 이 허용되는 접근 거리의 임계치(臨界值, threshold)를 $\theta(AABB, \mathbf{S})$ 라고 하자. 이 임계치 $\theta(AABB, \mathbf{S})$ 를 구할 수 있다면, AABB와 평면 \mathbf{S} 의 충돌은 AABB의 중심점 \mathbf{C} 와 평면 \mathbf{S} 의 거리가 이 임계치 이상인지 이하인지를 검사하는 문제로 바뀐다.

AABB와 평면의 충돌 2/2

이 값을 구하기 위해 우선 경계 상자의 중점에서 꼭지점으로 향하는 벡터 8 개 가운데 모두 양(positive)의 성분을 갖는 벡터를 \mathbf{D}_r 벡터라 하자. 이 벡터는 다음과 같이 구할 수 있다.

$$\mathbf{D}_r^x = (M_x - m_x)/2 \quad (33)$$

$$\mathbf{D}_r^y = (M_y - m_y)/2$$

$$\mathbf{D}_r^z = (M_z - m_z)/2$$

접근 허용치는 이 벡터와 평면의 단위 법선 벡터 \mathbf{N} 을 이용하여 구할 수 있는데, \mathbf{N} 의 각의 성분에 대해 절대값을 취해 얻는 벡터 \mathbf{N}' 와 \mathbf{D}_r 의 내적이 바로 접근 허용치이다. 따라서 다음과 같이 계산한다.

$$\theta(AABB, \mathbf{S}) = \mathbf{D}_r \cdot \mathbf{N}' \quad (34)$$

AABB와 구의 충돌

정확한 충돌 검사는 계산상의 부담에 비해 특별히 나은 결과를 보이지 않기 때문에, 간단히 AABB를 구의 반지름 r 을 고려하여 확장하고, 구의 중심이 AABB의 내부에 있는지를 검사하는 방법을 사용할 수 있다.

$$\begin{aligned}AABB' &= (m'_x, m'_y, m'_z, M'_x, M'_y, M'_z) \\&= (m_x - r, m_y - r, m_z - r, M_x + r, M_y + r, M_z + r)\end{aligned}\tag{35}$$

이 새로운 AABB'와 구의 중심 \mathbf{C} 와의 충돌을 검사하여 원래의 AABB와 구의 충돌을 근사할 수 있다. 다음은 충돌이 일어나는 조건이다.

$$\begin{aligned}m'_x \leq \mathbf{C}_x \leq M'_x \\m'_y \leq \mathbf{C}_y \leq M'_y \\m'_z \leq \mathbf{C}_z \leq M'_z\end{aligned}\tag{36}$$

AABB와 AABB의 충돌

두 경계 상자 $AABB_a$ 와 $AABB_b$ 가 충돌하기 위해서는 세 축에 따라 설정된 구간에 대해 각 축별로 중첩 검사를 실행하여 모두 중첩되어 있어야 한다. 어떤 구간 $[k, l]$ 과 $[m, n]$ 이 중첩하는지를 검사하여 중첩할 경우 참, 그렇지 않을 경우 거짓을 돌려주는 함수를 $\text{Overlap}([k, l], [m, n])$ 이라고 하면 $AABB_a$ 와 $AABB_b$ 가 충돌할 경우 참, 그렇지 않을 경우 거짓을 돌려주는 함수는 다음과 같다.

$$\begin{aligned} \text{Collide}(AABB_a, AABB_b) = & \\ \text{Overlap}([m_x^a, M_x^a], [m_x^b, M_x^b]) \wedge & \\ \text{Overlap}([m_y^a, M_y^a], [m_y^b, M_y^b]) \wedge & \\ \text{Overlap}([m_z^a, M_z^a], [m_z^b, M_z^b]) & \end{aligned} \tag{37}$$