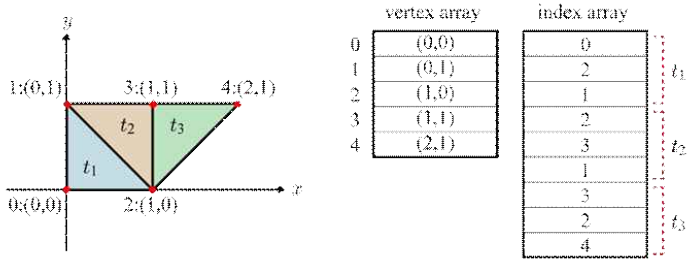
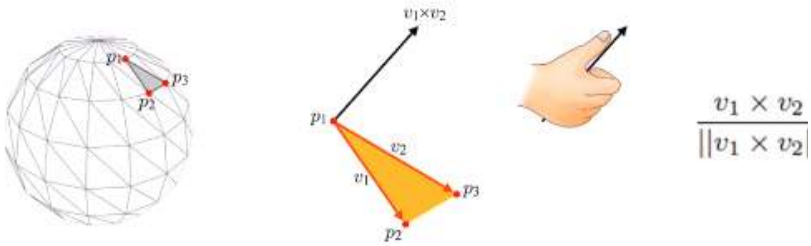
필기노트 - 컴퓨터 그래픽스 3장 모델링

	내 용
Polygon Mesh 폴리곤	● 구를 나타내는 수학적인 공식 -> $(x-C_x)^2+(y-C_y)^2+(z-C_z)^2=r^2$ ㄴ 이러한 것을 음함수라고 함. ● GPU가 음함수 처리에 적합하지 않음. 그래서 부드러운 표면을 Sample함 - 일정한 개수의 Vertices(vertex의 복수형. 꼭짓점. 정점)들을 다각형으로 이어붙인 폴리곤 메쉬를 사용
triangle mesh quad mesh	● triangle(삼각형)으로 만 이루어진 폴리곤 메쉬를 삼각형 메쉬(triangle mesh) ● 꼭짓점의 개수의 두배정도 개수만큼의 삼각형으로 이루어짐 ● 사각형으로 이루어진 것은 quad mesh ● OpenGl ES에서는 삼각형 메쉬 사용하고 모델링에서 사각형 메쉬를 많이 사용함
LOD (Level of Detail)	● 폴리곤 메쉬에서 정점의 개수 ● 해상도라고 표현함 ● LOD컨트롤은 그래필스에서 중요한 요소 - 멀리서 볼때는 simplification하여 적은 정점을 사용하고 - 가까이서 볼때는 refinement한 폴리곤을 사용해야 함
Vertex array Index array	 ● 위 그림처럼 삼각형 세 개가 있을 때 정점들을 배열하는 방식 ● t1은 0.2.1 정점을 이은 삼각형. t2는 2.3.1 정점을 이은 삼각형... ● 가장 일반적인 삼각형 메쉬를 표현하는 형태
Normal 법선 벡터 - triangle normal	● 법선벡터 : 한 평면이나 직선에 대하여 수직인 벡터 (단위벡터로 표현)  ● p1 p2 p3가 이루는 삼격형의 법선벡터를 찾으려고 할 때 - p1에서 각각 p2와 p3로 가는 벡터 v1. v2를 만든다 - 그 두 벡터에서 오른손 법칙을 통해 v1. v2의 외적(Cross product)를 구할수 있다 - v1 X v2 즉 외적은 크기를 가지고 있으므로 외적의 크기로 나누어 방향만 남긴다 ● 그런데 여기서 정점의 순서를 p1. p2. p3가 아닌 p1. p3. p2로 한다면 오른손의 법칙이 반대로 적용돼서 폴리곤 안쪽으로 향한다. 하지만 컴퓨터 그래픽스에서 모든 노말이 물체 바깥을 향하도록 관례를 두고 있음

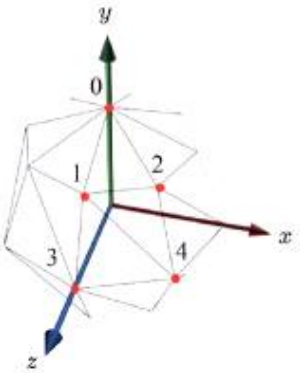
Normal
법선 벡터
- vertex normal

● 삼각형의 normal이 아닌 정점의 normal이 더 중요

● vertex normal은 vertex를 공유하는 triangle normal들의 평균

$$\frac{n_1 + n_2 + n_3 + n_4 + n_5 + n_6}{\|n_1 + n_2 + n_3 + n_4 + n_5 + n_6\|}$$

n1 n2 n3... 들이 각각 triangle normal
다 더해서 단위벡터로 만들어 줌



vertex array

	position	normal
0	(0.000, 1.000, 0.000)	(0.000, 1.000, 0.000)
1	(0.000, 0.707, 0.707)	(0.000, 0.663, 0.748)
2	(0.500, 0.707, 0.500)	(0.529, 0.663, 0.529)
3	(0.000, 0.000, 1.000)	(0.000, 0.000, 1.000)
4	(0.707, 0.000, 0.707)	(0.707, 0.000, 0.707)
⋮	⋮	⋮
25	(0.000, -1.000, 0.000)	(0.000, -1.000, 0.000)

index array

0	0
1	1
2	2
3	1
4	3
5	4
⋮	⋮
143	16

vertex array에 위와같은 방식으로 위치와 normal까지 저장할 수 있음