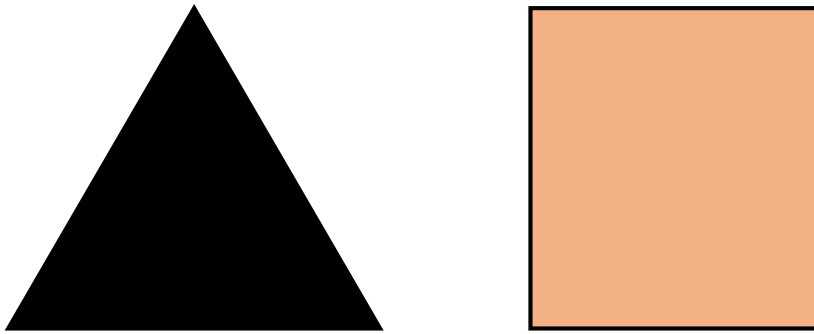
1. Modeling

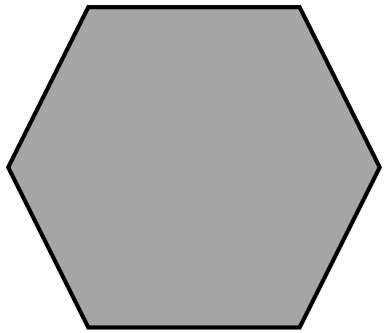
Prof. HyeongYeop Kang
siamiz@khu.ac.kr
YouTube: HKang IIIXR LAB
IIIXR LAB

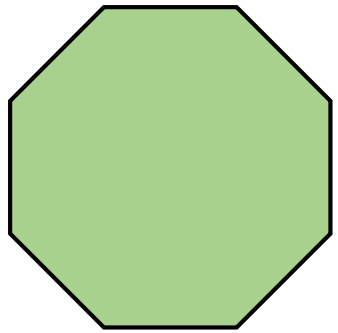
Polygon

A polygon is a plane shape that is described by a finite number of straight line segments.

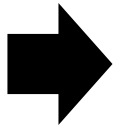
- Among the many types of polygons, the representation with triangle polygon is most popularly used in game graphics.
- In contrast, quad polygon is widely used in the field of filmmaking.

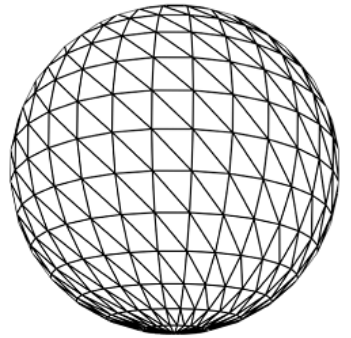






polygon examples



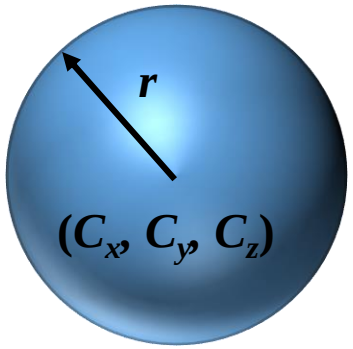


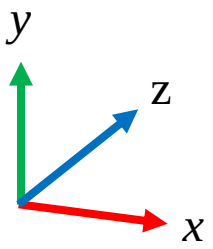
triangle example

Polygon Mesh

Sphere can be represented by either implicit way or explicit way.

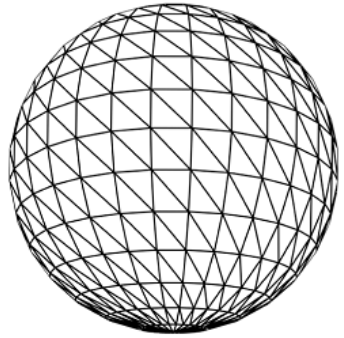
- The polygon mesh representation (explicit representation) is preferred in game graphics because the GPU is optimized for processing polygons.
- The mesh's vertices are the points that *sample* the smooth surface and therefore the polygon mesh is not an accurate representation but an approximate one.





implicit representation

$(x - C_x)^2 + (y - C_y)^2 + (z - C_z)^2 = r^2$

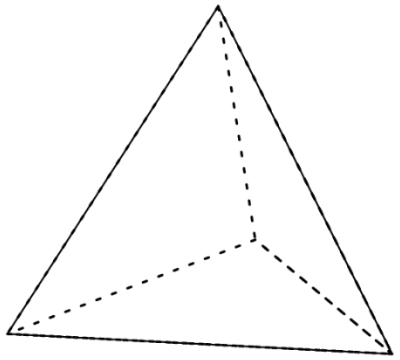


explicit representation
(polygon mesh)

Polygon Mesh

In a typical closed mesh (with no hole), the number of triangles is approximately twice the number of vertices, e.g., given 100 vertices, we have about 200 triangles.

- $v - e + f = 2$ (where v, e, and f are respectively the number of vertices, edges, and faces of the mesh) - the Euler's polyhedron formula
- In a closed triangle mesh, every edge is shared by two faces and every face has three edges. Therefore, we can say $2e = 3f$.
- When we replace e by $\frac{3}{2} f$, we obtain $v - \frac{3}{2} f - f = 2$. Therefore, $f = 2v - 4$.
- As the mesh's size increases, the number of faces converges to twice the number of vertices.

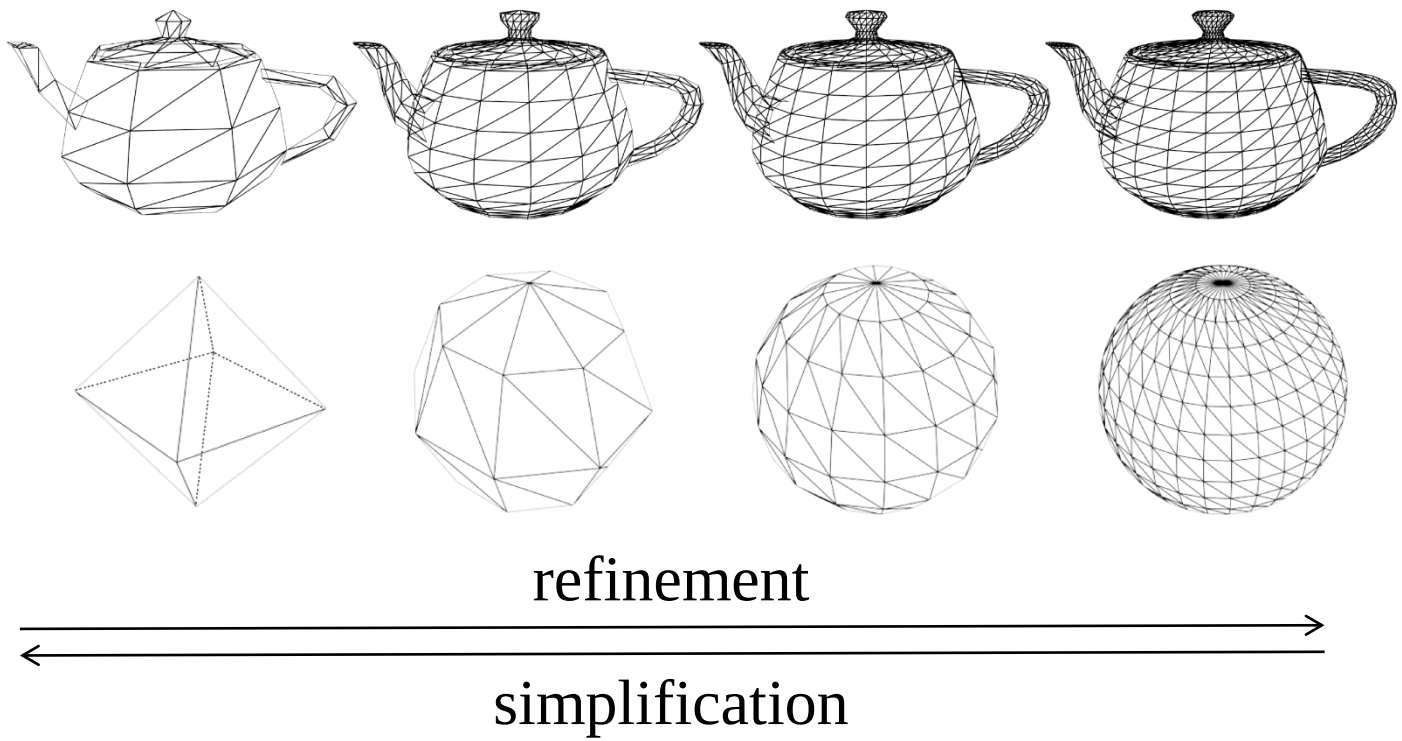


$2e = 3f = 12$

Polygon Mesh

Tradeoff between accuracy and efficiency

- The vertex count of a polygon mesh is described as a *resolution* or *level of detail* (LOD)
- As the resolution increases, the shape of the mesh becomes closer to the original smooth surface, but the time needed for processing the mesh also increases.

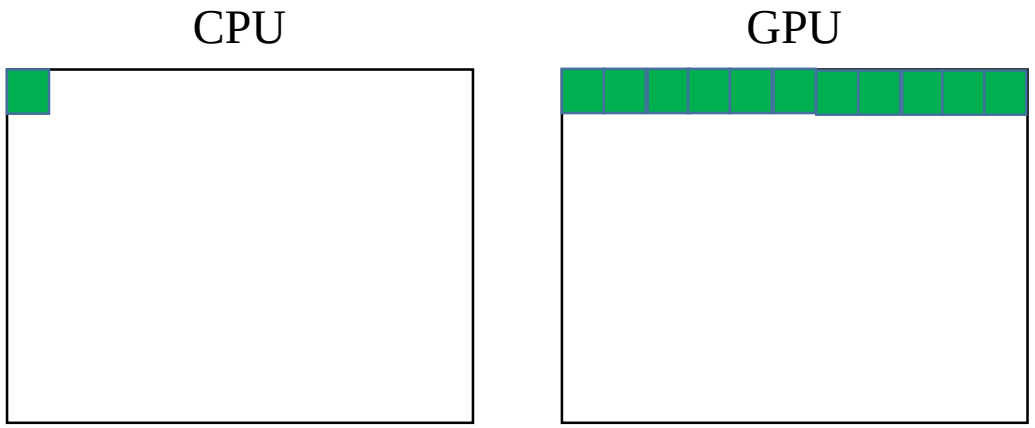


Graphics processing unit (GPU)

A specialized electronic circuit for rendering.

- GPUs are designed to have parallel structure which makes GPUs more efficient than general-purpose central processing units (CPUs) for rendering.
- Thanks to their parallel structure, GPUs are also widely used for algorithms that process large blocks of data in parallel such as neural networks, image processing, etc.

Nvidia’s GPU
(https://www.nvidia.com/)

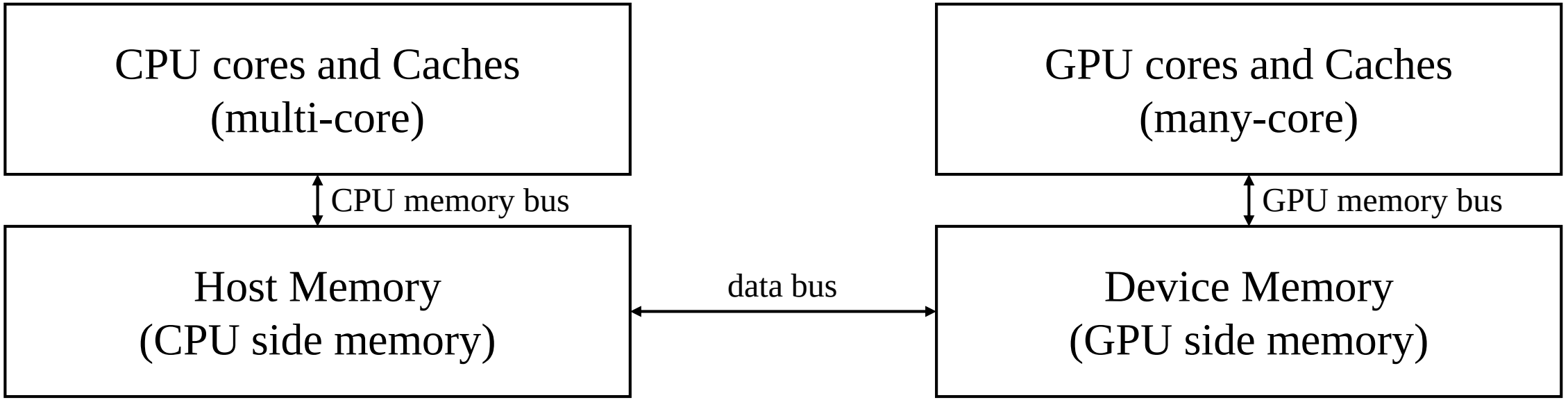


CPU vs GPU

Graphics processing unit (GPU)

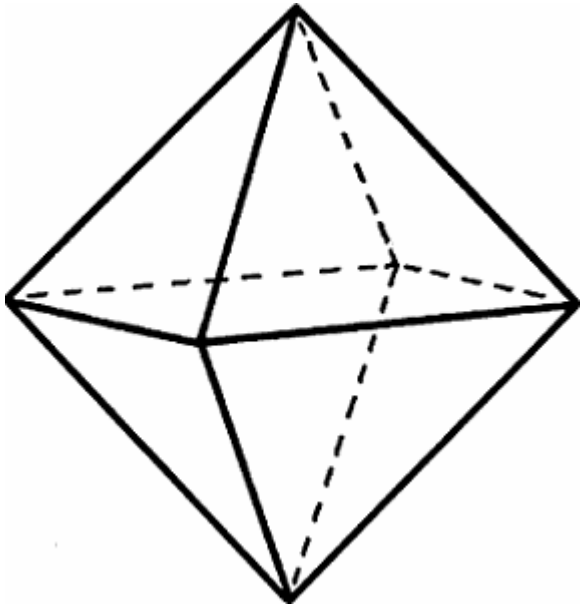
A specialized electronic circuit for rendering.

- However, GPUs cannot run the O/S such as Windows or Linux by themselves.
- For this reason, the required data for rendering should be transmitted from the CPU side to the GPU side through a data bus (such as PCI-e bus).



Practice

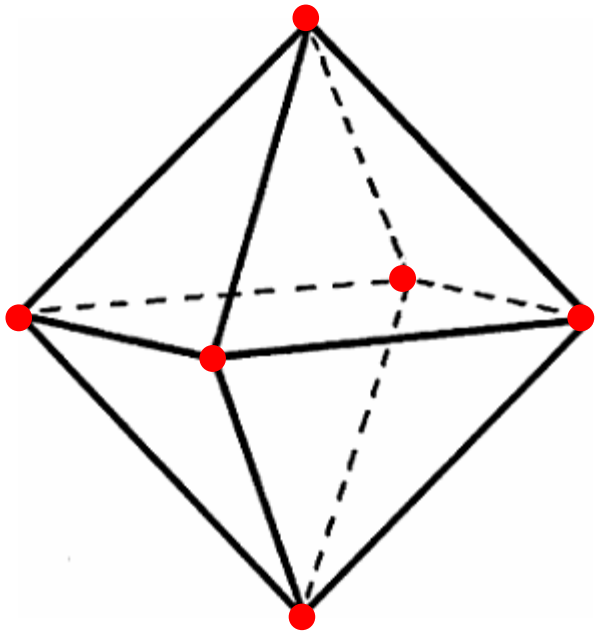
Let v, e, and f denote the numbers of vertices, edges, and faces of closed triangle mesh, respectively. Consider a octahedron, find its v, e, and f.



octahedron

Practice - solution

Let v, e, and f denote the numbers of vertices, edges, and faces of closed triangle mesh, respectively. Consider a octahedron, find its v, e, and f.



$v = 6$

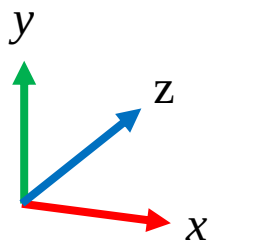
$e = 12$

$f = 8$

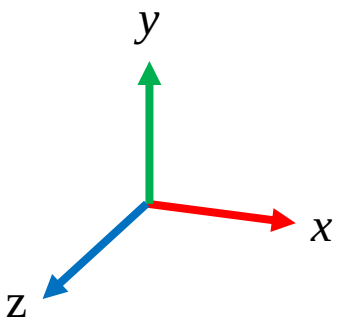
Polygon Mesh Representation

Non-indexed representation

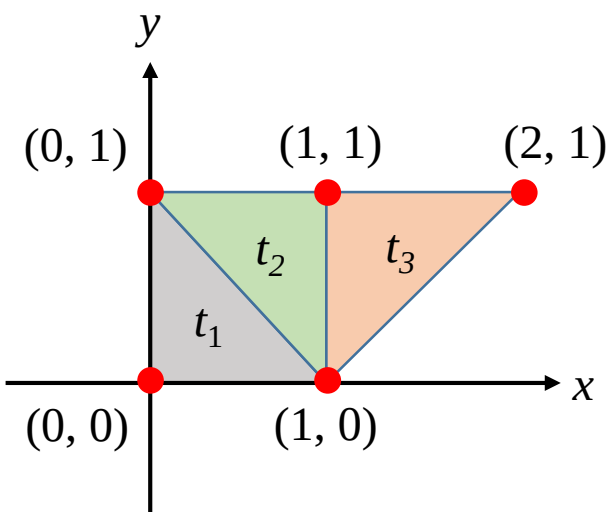
- The vertices are enumerated in a memory space, named vertex array.
- Three vertices are read in linear order to make up a triangle.
- The order of vertices depends on the coordinate system used: Left-handed Cartesian coordinate system (LHS) and Right-handed Cartesian coordinate system (RHS).



LHS (Direct X)



RHS (OpenGL)



vertex array (LHS)
(0, 0)
(0, 1)
(1, 0)
(1, 0)
(0, 1)
(1, 1)
(1, 1)
(2, 1)
(1, 0)

vertex array (RHS)
(0, 0)
(1, 0)
(0, 1)
(0, 1)
(1, 0)
(1, 1)
(1, 1)
(1, 0)
(2, 1)

Polygon Mesh Representation

Non-indexed representation

- Non-indexed representation is inefficient because the vertex array contains redundant data.
- To make matters worse, the data stored in the vertex array are not restricted to vertex positions but include additional information (color, normal, etc.).

vertex array (LHS)
(0, 0)
(0, 1)
(1, 0)
(1, 0)
(0, 1)
(1, 1)
(1, 1)
(2, 1)
(1, 0)

there are actually five distinct vertices

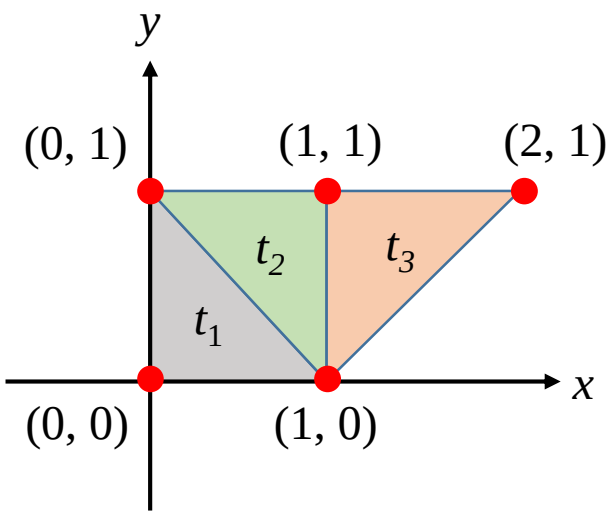
real vertex array
(0, 0), (1, 0), (0.3, 0.2), (0.21, 0.33, 0.76), ...
...
...

a real vertex array contains a lot of different data

Polygon Mesh Representation

Indexed representation

- A better method is using a separate index array.
- A vertex appears “only once” in the vertex array.
- Three indices per triangle are stored in the index array.
- The vertex array storage saved by removing the duplicate data outweighs the additional storage needed for the index array.

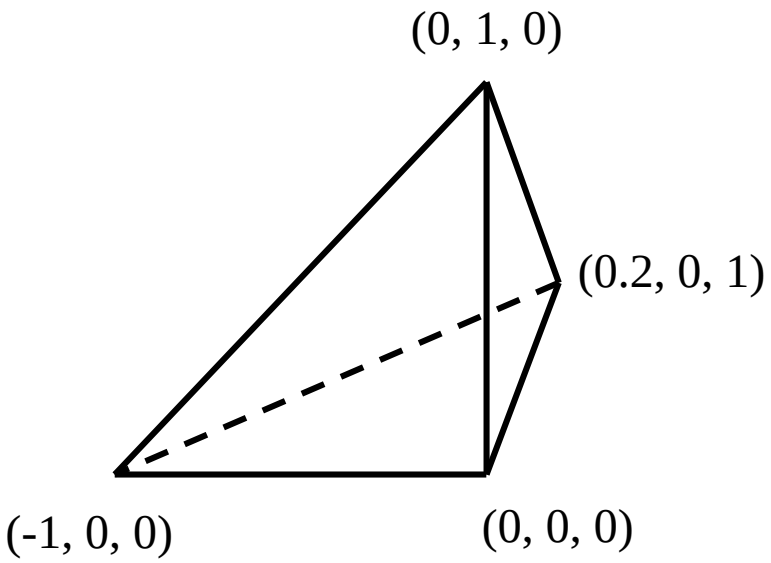


vertex array
(0, 0)
(0, 1)
(1, 0)
(1, 1)
(2, 1)

index array (RHS)
0
2
1
1
2
3
3
2
4

Practice

Given the triangle mesh shown below, fill in the vertex and index arrays for its indexed mesh representation (RHS and LHS).



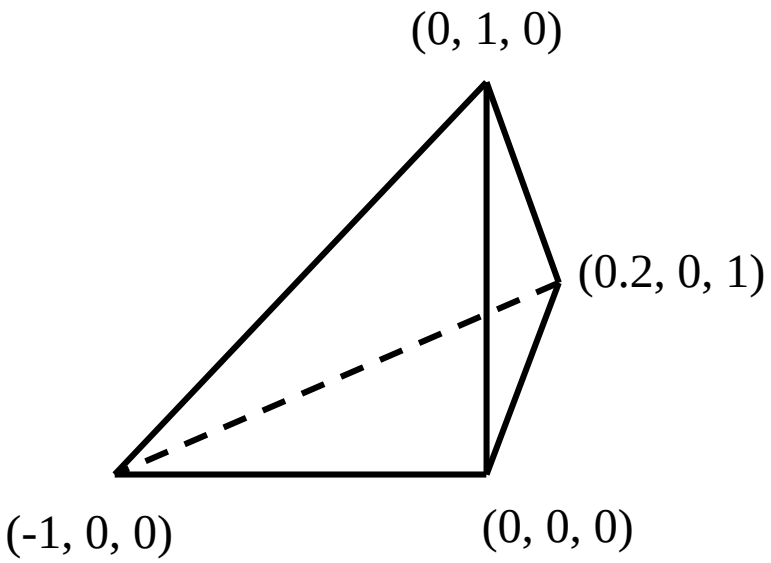
vertex array
(0, 0, 0)

index array (RHS)

index array (LHS)

Practice - solution

Given the triangle mesh shown below, fill in the vertex and index arrays for its indexed mesh representation (RHS and LHS).



vertex array
(0, 0, 0)
(0, 1, 0)
(0.2, 0, 1)
(-1, 0, 0)

index array (RHS)
0
1
3
0
2
1
1
2
3
0
3
2

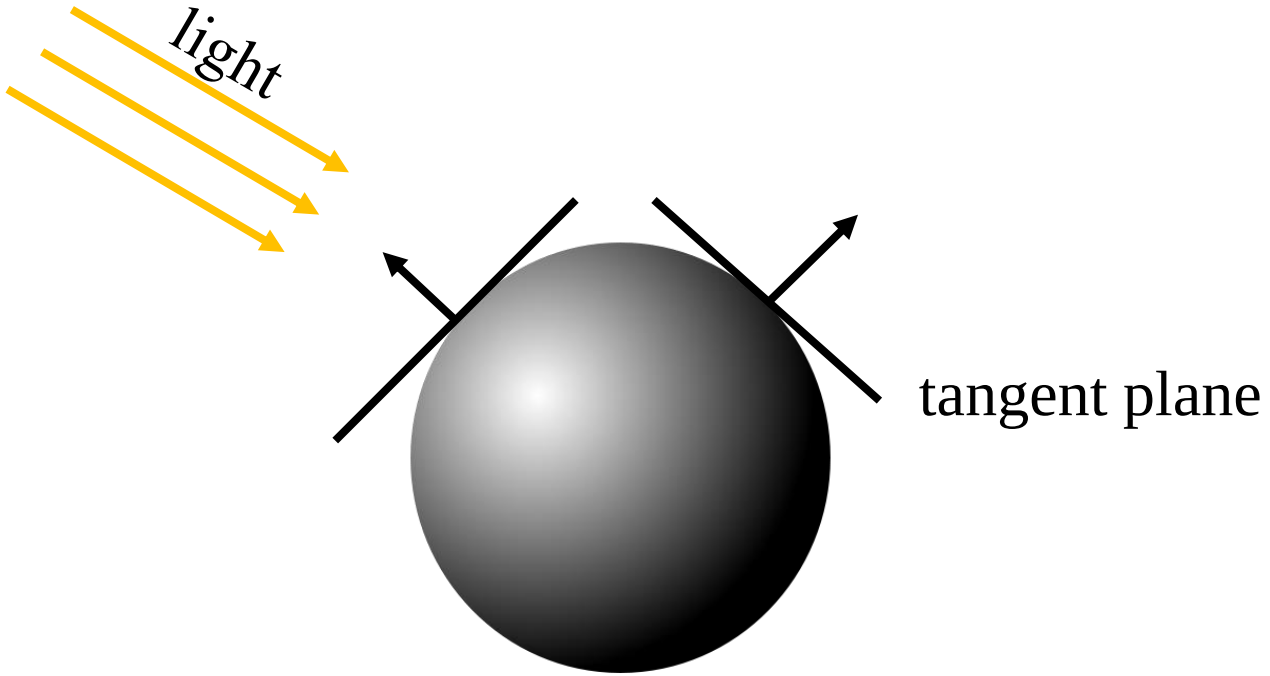
index array (LHS)
0
3
1
0
1
2
1
3
2
0
2
3

Normals

Surface normals in graphics

- Normal is the vector perpendicular to the surface tangent at a given point.
- The brightness of the object surface at certain point depends on the angle between normal at the point and the light direction (this will be covered in a later chapter).

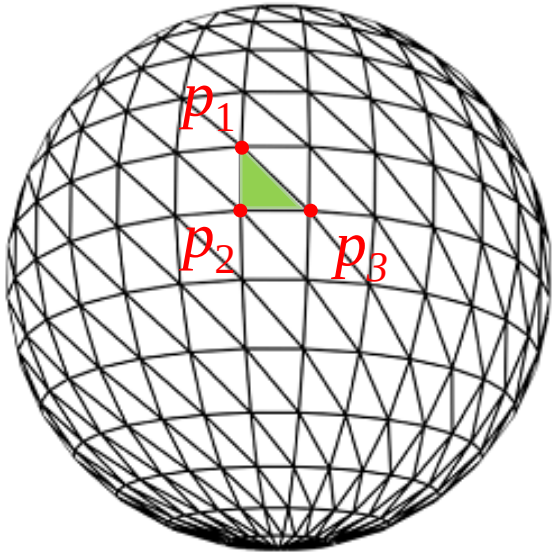
Surface normals are classified into *triangle* normals and *vertex* normals.

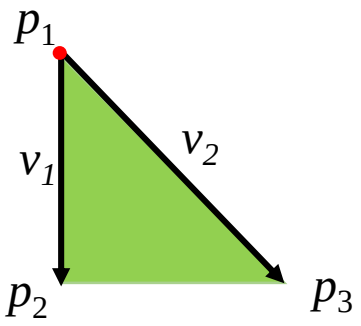


Triangle Normals

Triangle normal

- Given triangle (p_1, p_2, p_3), let v_1 denote the vector connecting the first vertex (p_1) and the second (p_2).
- Similarly, the vector connecting the first vertex (p_1) and the third (p_3) is denoted by v_2.
- Note that the order of p_1, p_2, and p_3 are counter-clockwise (CCW).

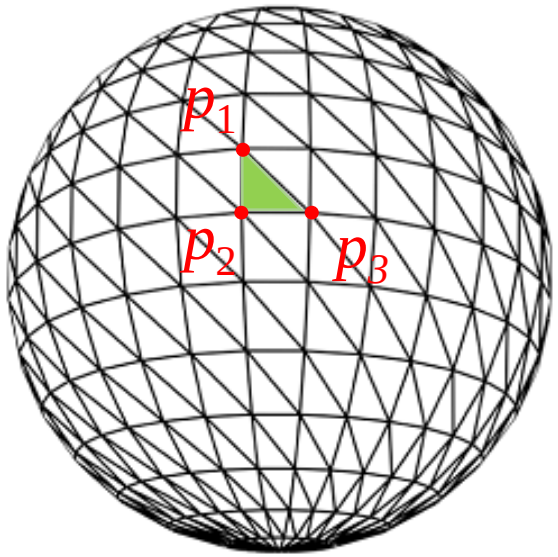




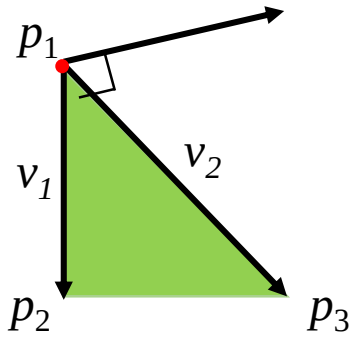
Triangle Normals

Triangle normal

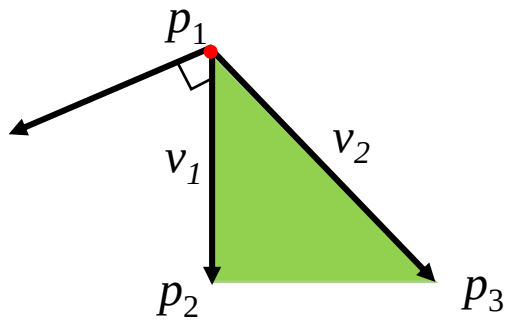
- Then, the triangle normal can be computed using the cross product based on either the left-hand or right-hand rule.
- Every normal vector is made to be a unit vector by default.



$$\frac{v_1 \times v_2}{||v_1 \times v_2||}$$



RHS

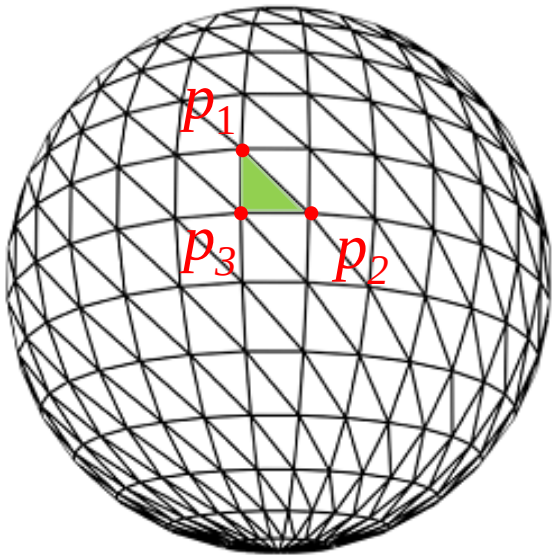


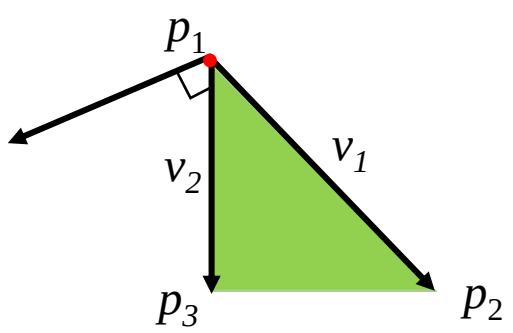
LHS

Triangle Normals

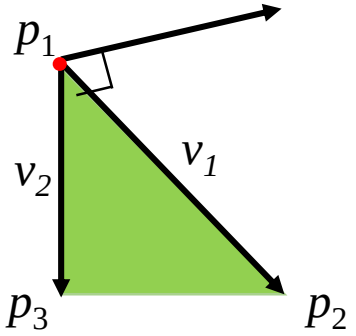
Counterclockwise vs. clockwise

- What if the vertices are ordered clockwise (CW)?





RHS



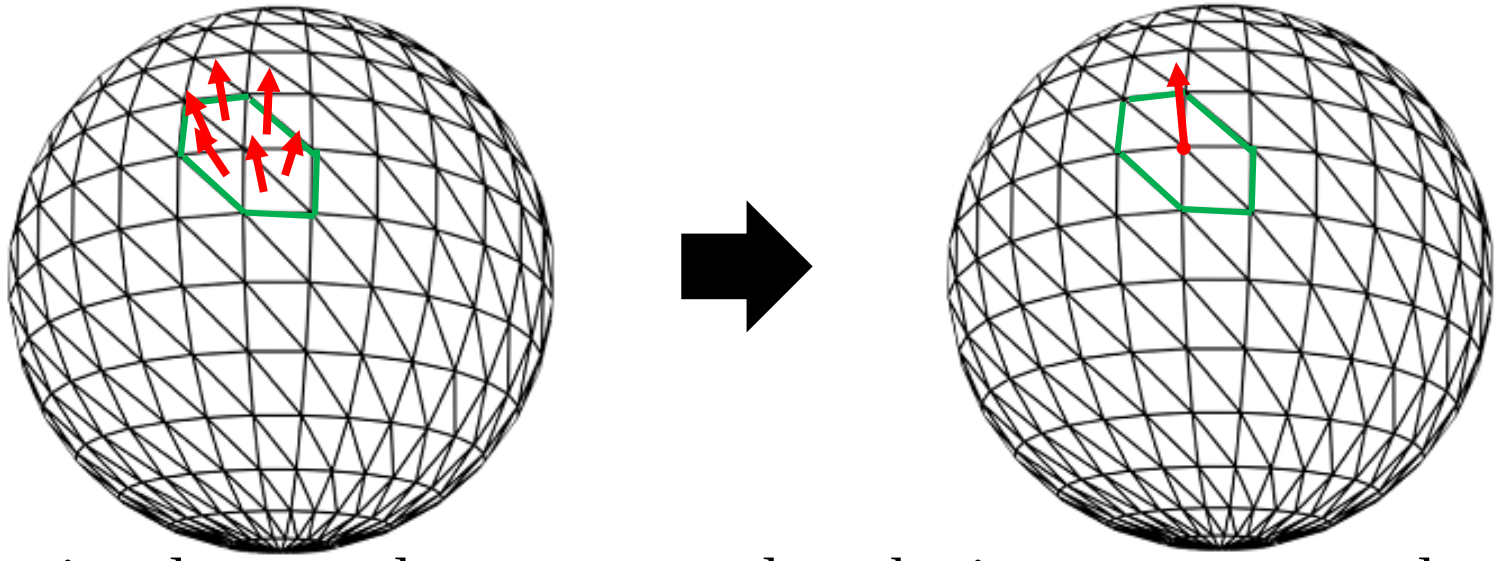
LHS

- The normal is in the opposite direction.
- The normal direction depends on the vertex order.
- In computer graphics, surface normals are supposed to point out of the polyhedron.

Vertex Normals

Vertex normal

- We have discussed the *triangle normals*, but more important are the *vertex normals*. A normal can be assigned to a vertex such that the vertex normal approximates the normal of the smooth surface's point that the vertex samples.
- A vertex normal can be defined by averaging the normals of all the triangles sharing the vertex.
- Vertex normals are data that is almost always transmitted to GPU through a vertex array.

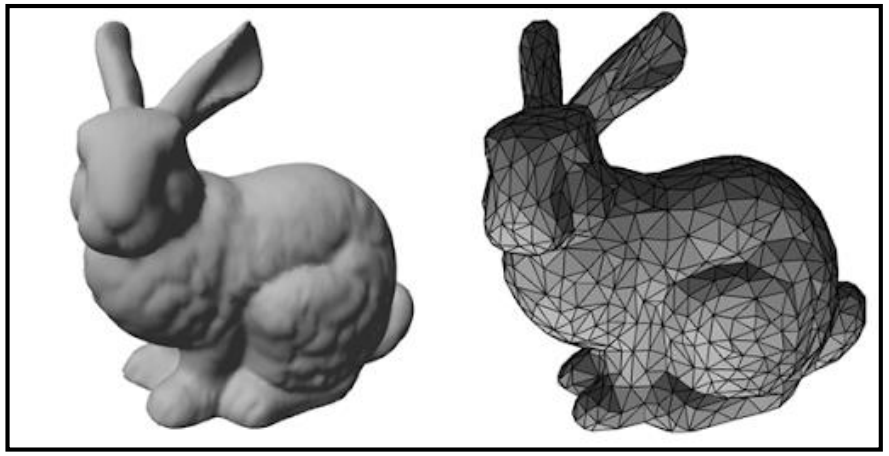


triangle normals are averaged to obtain a vertex normal

Export and Import

Polygon mesh to run-time 3D application

- Polygon meshes and related data created using off-line graphics packages are stored in files and passed to the run-time 3D application program.
- The process of outputting the data in a format suitable for other applications is called *export*.
- On the other hand, taking such exported data is called *import*.
- There are a lot of file formats. Among the popular is .obj file.



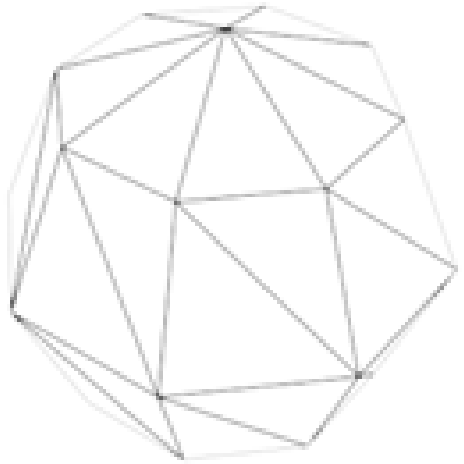
```
# OBJ file format with ext .obj
# vertex count = 2503
# face count = 4968
v -3.4101800e-003 1.3031957e-001 2.1754370e-002
v -8.1719160e-002 1.5250145e-001 2.9656090e-002
v -3.0543480e-002 1.2477885e-001 1.0983400e-003
v -2.4901590e-002 1.1211138e-001 3.7560240e-002
v -1.8405680e-002 1.7843055e-001 -2.4219580e-002
v 1.9067940e-002 1.2144925e-001 3.1968440e-002
v 6.0412000e-003 1.2494359e-001 3.2652890e-002
v -1.3469030e-002 1.6299355e-001 -1.2000020e-002
v -3.4393240e-002 1.7236688e-001 -9.8213000e-004
```

Stanford bunny and its .obj file

Export and Import

Consider a low-resolution mesh of a unit sphere.

- The sphere surface is uniformly sampled at every 45 degrees such that the mesh is composed of 26 vertices and 48 triangles.
- Shown below are some snippets of the .obj file for the mesh.



```
v  0.000  1.000  0.000
v  0.000  0.707  0.707
v  0.500  0.707  0.500
v  0.000  0.000  1.000
v  0.707  0.000  0.707
          ⋮
v  0.000 -1.000  0.000
# 26 vertices
```

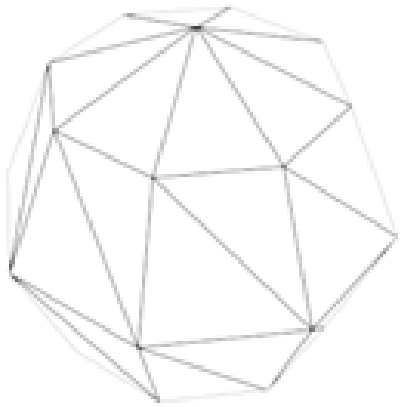
```
vn  0.000  1.000  0.000
vn  0.000  0.663  0.748
vn  0.529  0.663  0.529
vn  0.000  0.000  1.000
vn  0.707  0.000  0.707
           ⋮
vn  0.000 -1.000  0.000
# 26 vertex normals
```

```
g Sphere001
f  1//1  2//2  3//3
f  2//2  4//4  5//5
f  5//5  3//3  2//2
          ⋮
# 48 faces
```

Export and Import

Importing to the 3D application

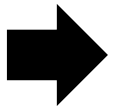
- The data stored in a file is imported into the vertex and index arrays of the 3D application.
- As the mesh is composed of 48 triangles, the index array has 144 (48 × 3) elements.



```
v  0.000  1.000  0.000
v  0.000  0.707  0.707
v  0.500  0.707  0.500
v  0.000  0.000  1.000
v  0.707  0.000  0.707
      ⋮
v  0.000 -1.000  0.000
# 26 vertices
```

```
vn  0.000  1.000  0.000
vn  0.000  0.663  0.748
vn  0.529  0.663  0.529
vn  0.000  0.000  1.000
vn  0.707  0.000  0.707
      ⋮
vn  0.000 -1.000  0.000
# 26 vertex normals
```

```
g Sphere001
f  1//1  2//2  3//3
f  2//2  4//4  5//5
f  5//5  3//3  2//2
      ⋮
# 48 faces
```



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