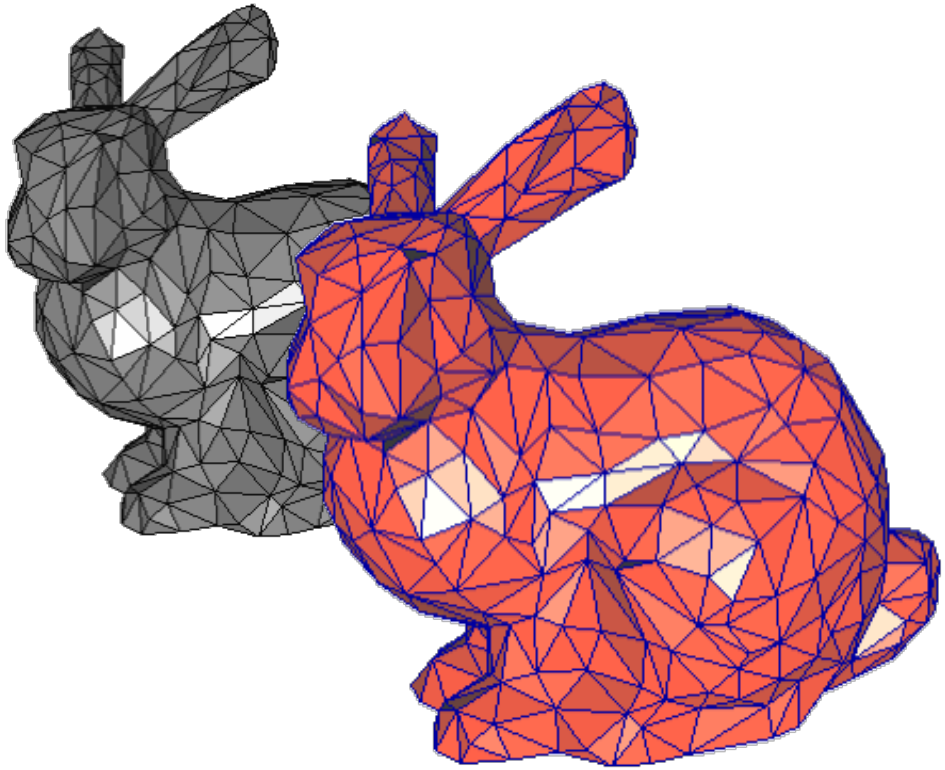
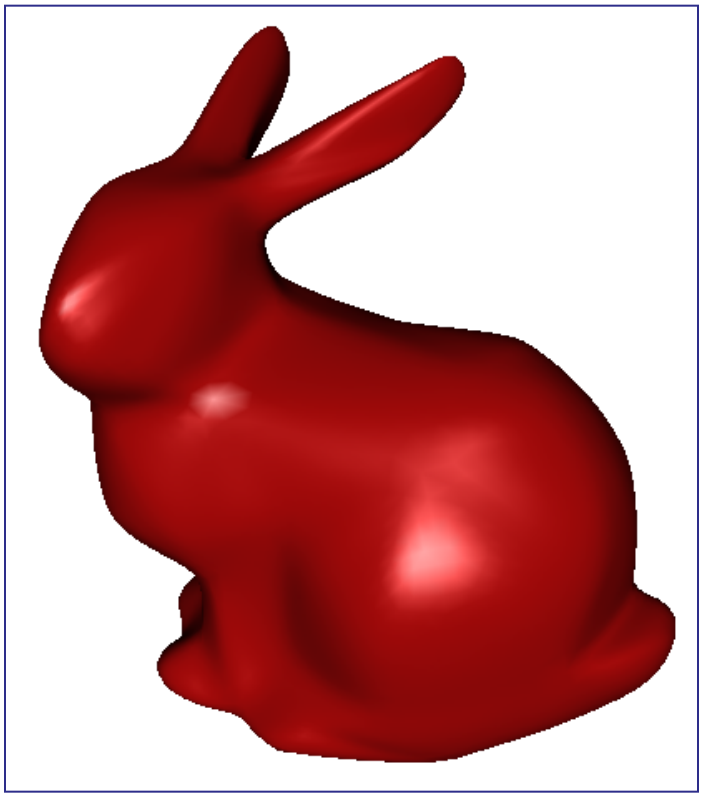
Mesh 3D Poligonali



Introduzione

Il modo più comune di rappresentare un oggetto è quello di rappresentare il suo contorno, anche detto B-Rep (Boundary Representation).

Si assume per ipotesi che il contorno degli oggetti solidi sia una superficie o varietà due-dimensionale (two-manifold*).



(*)l'intorno di ogni punto della superficie è omeomorfo ad un disco piano; se il punto è sul bordo si considera un semidisco.

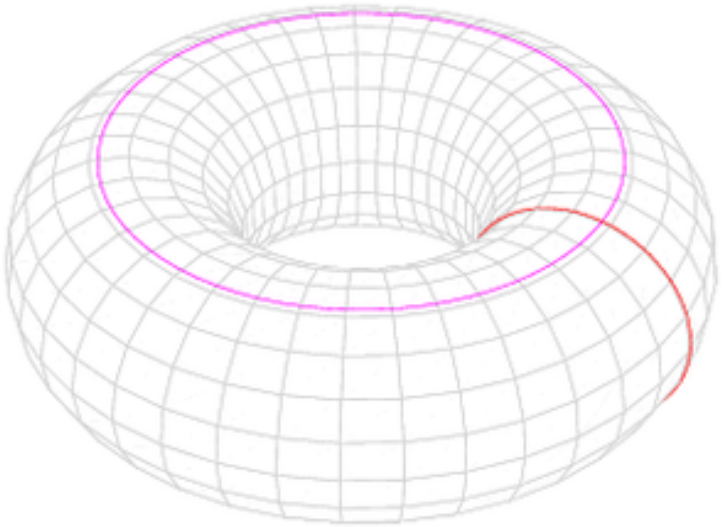
Omeomorfismo

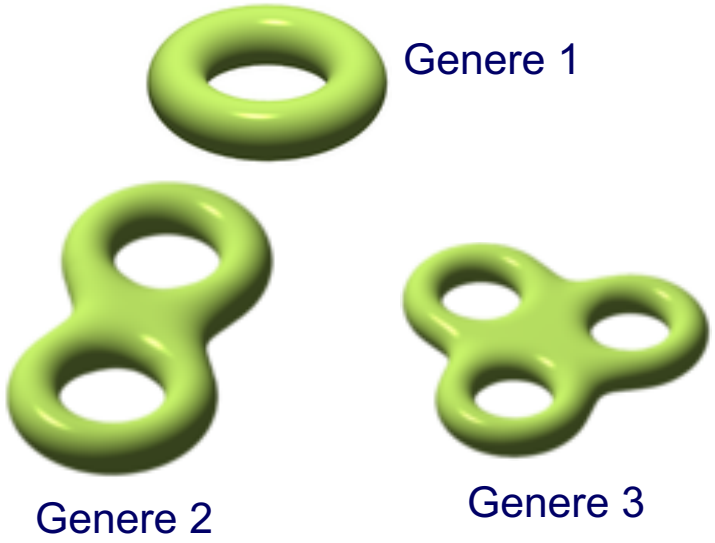
In topologia, un omeomorfismo (dal greco homoios = identica e morphe = forma), è una particolare funzione fra spazi topologici che modellizza l'idea intuitiva di "deformazione senza strappi".

Una tazza ed una ciambella sono omeomorfi. Dalla "deformazione senza strappi" mostrata in figura si può infatti costruire un omeomorfismo fra i due oggetti.

Genere di una superficie

In topologia, il genere di una superficie viene definito come il numero più grande di curve semplici chiuse disgiunte che possono essere disegnate sulla superficie senza separarla in parti non connesse. Una sfera ha genere 0.

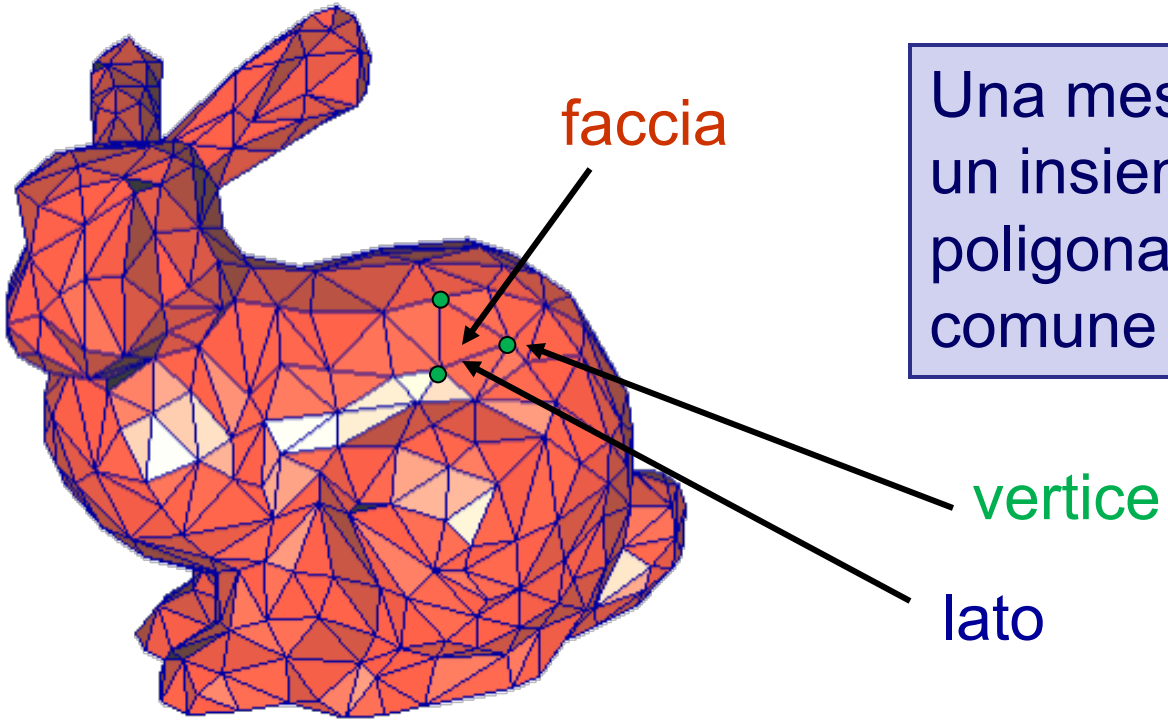




Nel caso in cui la superficie sia orientabile, il genere può essere definito più informalmente come il "numero di buchi":

Definizione di Mesh 3D Poligonale

Per semplificare la descrizione del contorno di un oggetto, ai fini di una grafica 3D real-time si usa una approssimazione poligonale, che chiameremo mesh 3D poligonale;



Una mesh 3D poligonale è un insieme di facce poligonali che hanno in comune vertici e lati

faccia poligonale:= un poligono chiuso

Quali facce/poligoni usare?

In pratica si usano facce poligonali piane e convesse; ma quali poligoni piani? E perché convessi?

I triangoli sono i più utilizzati perché:

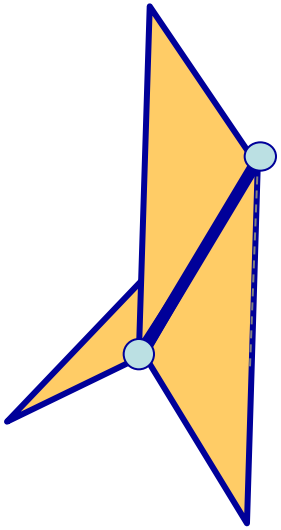
- matematicamente semplici
- sempre piani
- sempre convessi
- facili da rasterizzare (l'hardware grafico è basato su triangoli)
- strutture dati più semplici

In questo caso la mesh 3D poligonale è una superficie piana a tratti e se le facce sono tutte triangoli si dirà mesh triangolare; a volte si usano mesh a facce quadrilatere, a volte miste triangoli e quadrilateri a volte mesh a facce generiche.

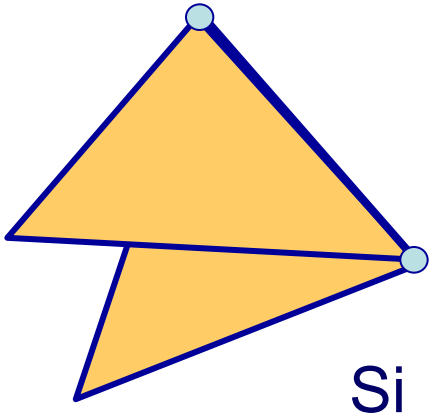
Two-Manifold

Avendo assunto che il contorno di un oggetto sia una varietà due-dimensionale (two-manifold), si richiede che la mesh che lo approssima sia a sua volta two-manifold. Bisogna allora imporre che:

1-un lato deve appartenere al massimo a due facce;



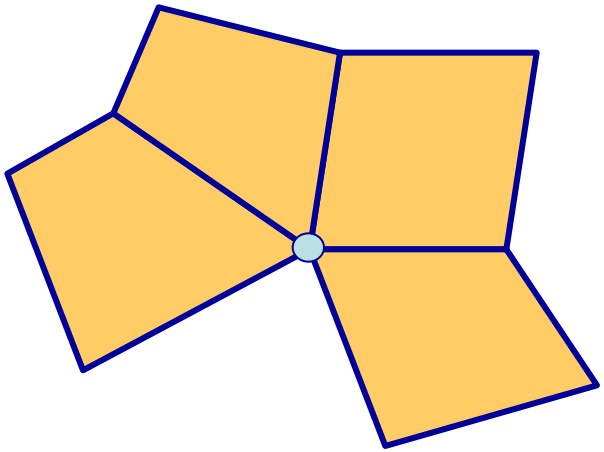
No



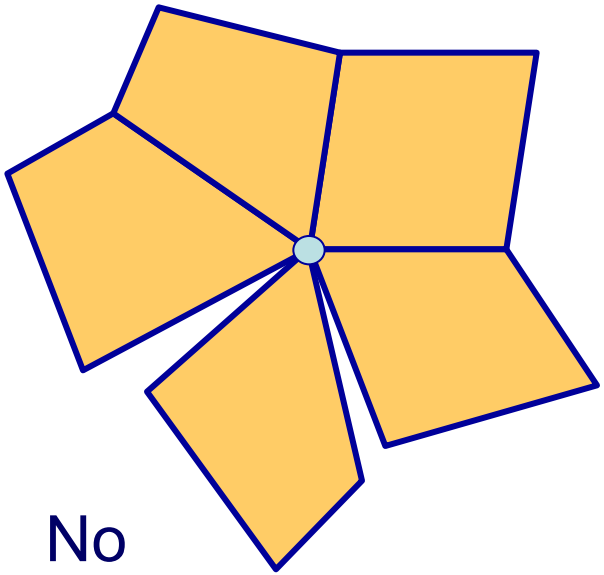
Si

Two-Manifold

2-se due o più facce incidono sullo stesso vertice allora devono formare un ventaglio (fan).



Si

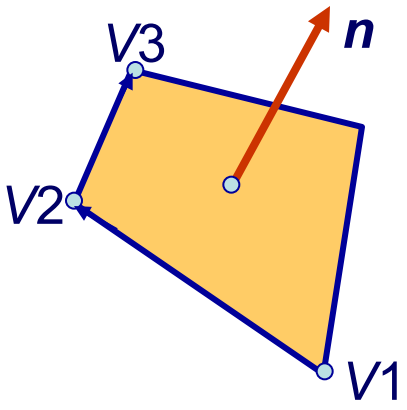


No

Normale ad una faccia

La normale ***n*** ad una faccia è data dal prodotto vettoriale di due suoi lati consecutivi non collineari (bisogna stare attenti al verso: la normale è *uscente* dal *front* della faccia).
Per tre vertici ($V1$, $V2$, $V3$) si ha:

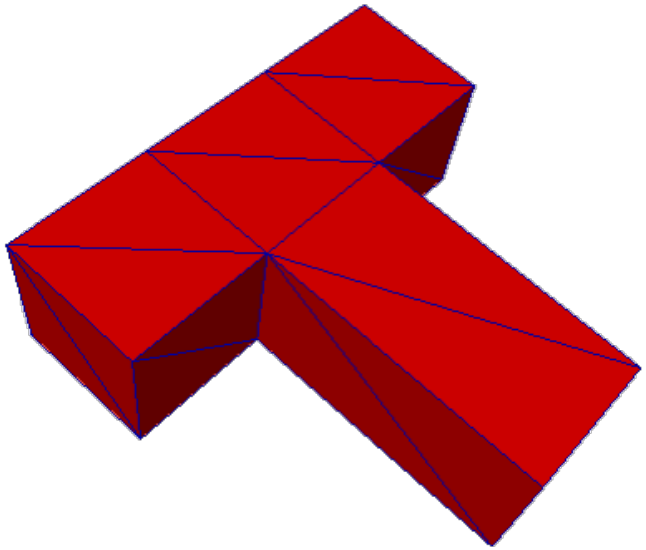
$$\boldsymbol{n} = (V3 - V2) \times (V2 - V1)$$



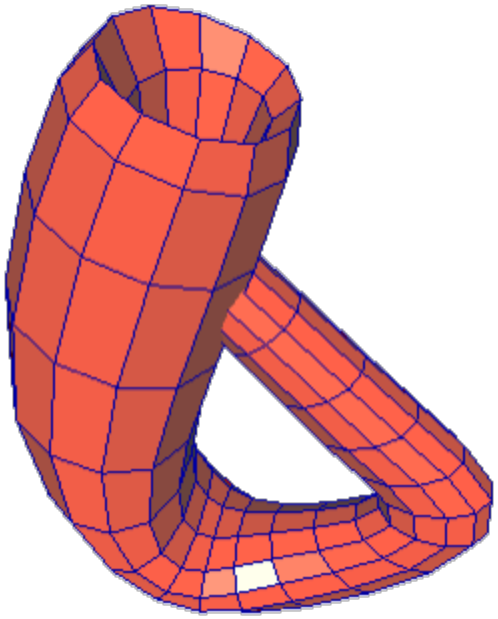
L'orientazione di una faccia è data dall'ordine ciclico (orario o antiorario) dei suoi vertici. L'orientazione determina il fronte ed il retro della faccia. Una convenzione (usata anche da OpenGL e WebGL) è che la faccia mostra il fronte se i vertici sono disposti in senso antiorario.

Orientabile

Una mesh si dice orientabile se è possibile determinare una normale coerente in ogni punto della mesh; con coerente si intende che le normali siano tutte orientate da interno ad esterno o viceversa.



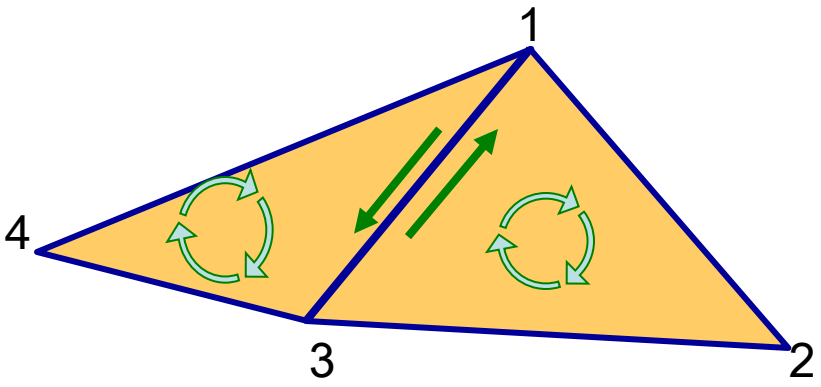
Orientabile

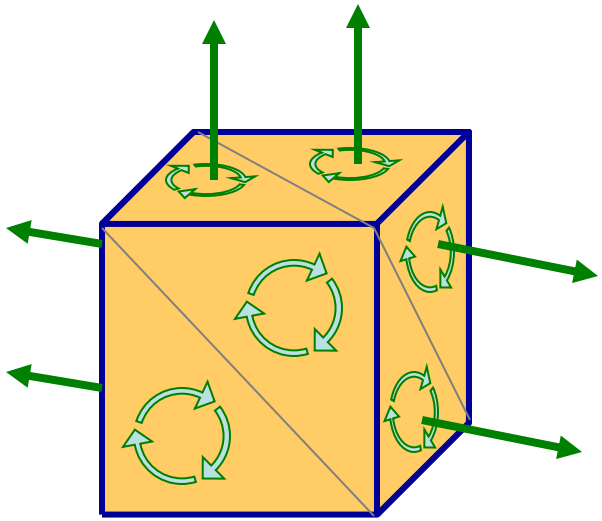


Non orientabile

Orientamento della Mesh

Non basta che una mesh sia orientabile, è necessario che ogni faccia sia descritta in maniera coerente alle altre.





L'orientamento della mesh serve sia per il suo disegno che per avere una definizione corretta dell' oggetto solido che rappresenta

Orientamento della Mesh

L'orientazione di due facce adiacenti è coerente se i due vertici del loro lato in comune sono in ordine inverso. Vuol dire che l'orientazione non cambia attraversando il lato in comune.

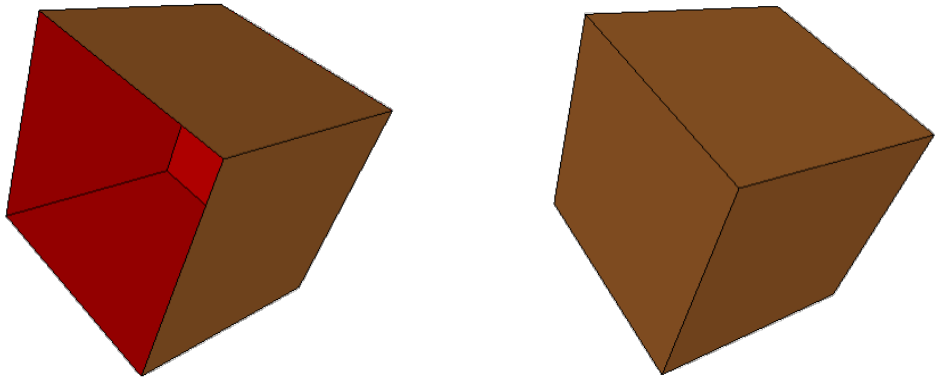
Se una mesh è descritta in maniera coerente, allora si può determinare se è orientabile controllando che le normali siano tutte interne o tutte esterne.

La mesh si dice orientabile se esiste una scelta dell'orientazione delle facce che rende compatibili tutte le coppie di facce adiacenti.

Mesh chiusa o aperta

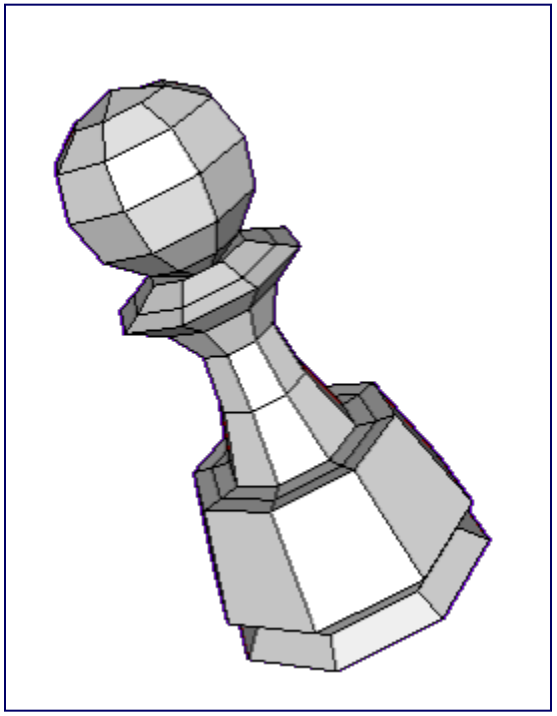
La superficie di un oggetto solido è rappresentata da una mesh chiusa in contrapposizione ad aperta.

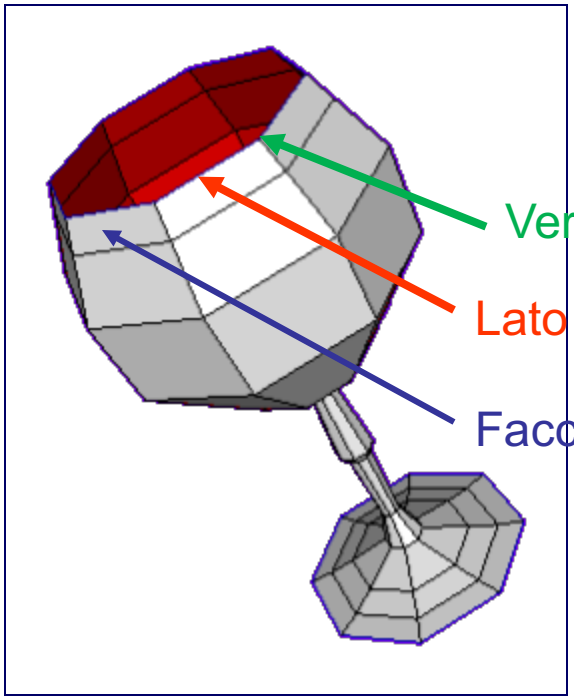
Una mesh aperta si contraddistingue per il fatto di avere un contorno, cioè esistono dei lati, delle facce e dei vertici di bordo.



Mesh chiusa o aperta

Lato di bordo: lato che appartiene ad una sola faccia;
Vertice di bordo: estremo di un lato di bordo;
Faccia di bordo: faccia con un lato di bordo.



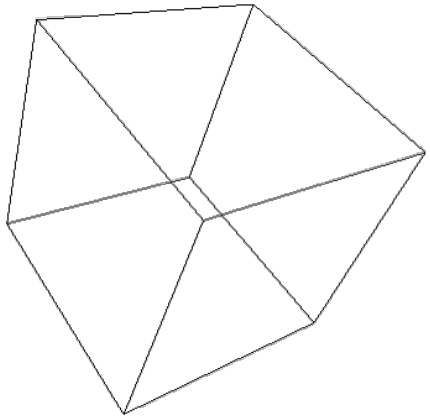


Topologia e Geometria

Di una mesh si distingue:

- La definizione geometrica (dove sono posizionati nella spazio 3D i vertici)
- La definizione topologica (come sono connessi i vertici da lati e facce)

Nota: due mesh possono avere stessa geometria, ma differente topologia rappresentando due oggetti differenti.

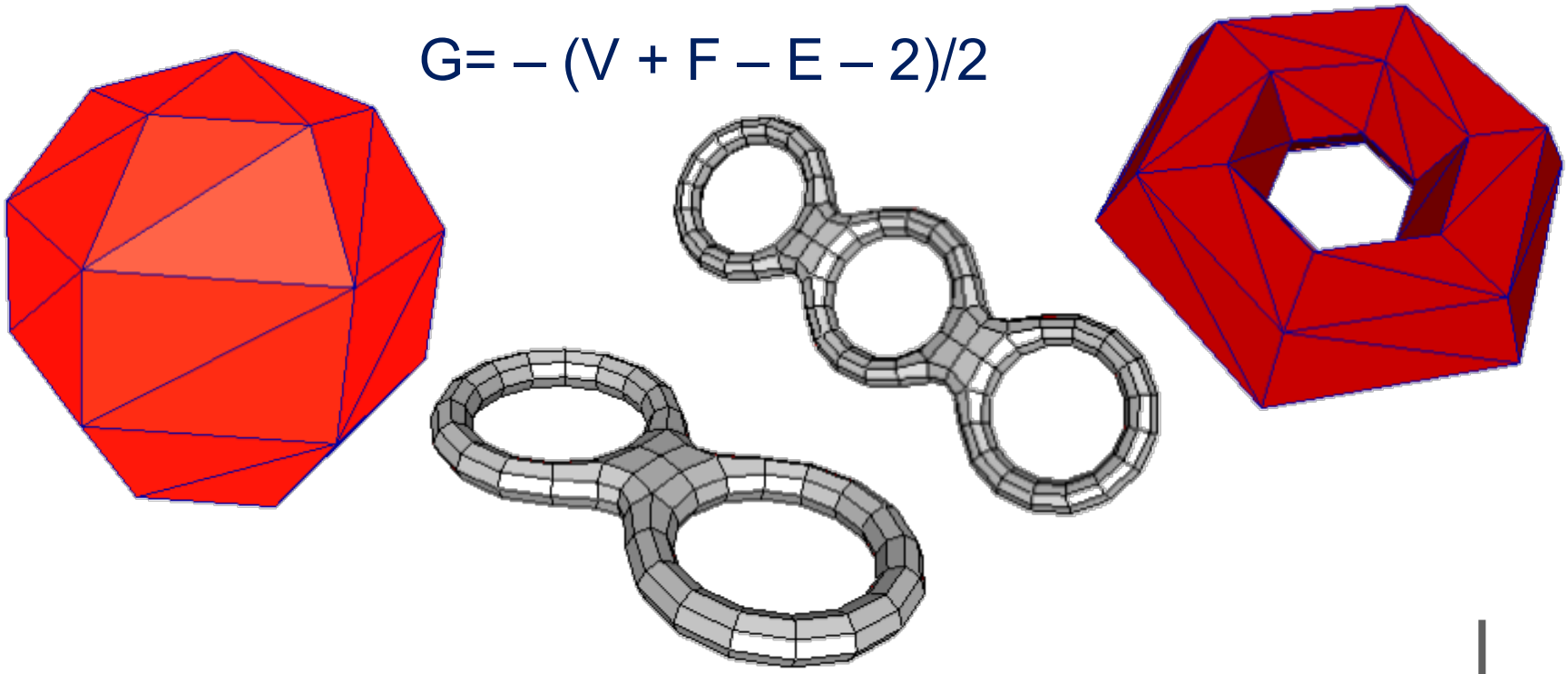


Topologia di una mesh

Genere

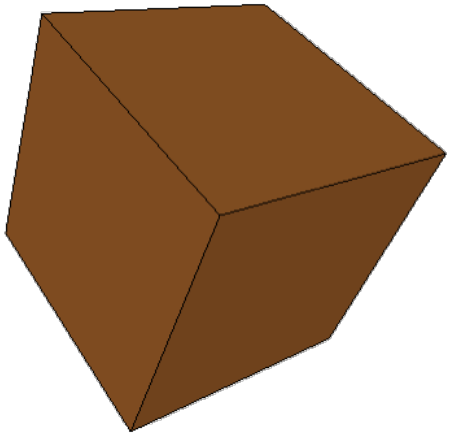
se la mesh è chiusa e orientabile, si possono contare i “buchi” dell’oggetto che definiscono il genere (esempio sfera per genere 0 e toro per genere 1)

$$G = -(V + F - E - 2)/2$$

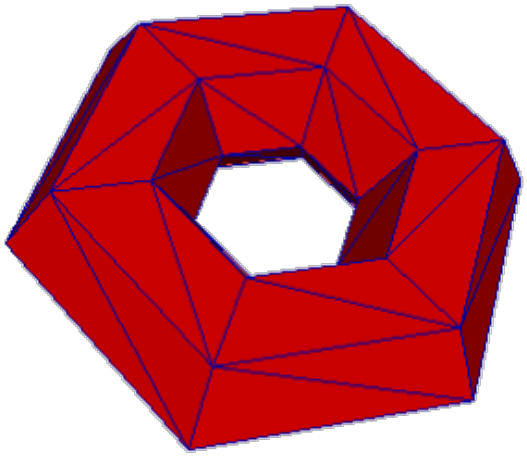


Topologia di una mesh

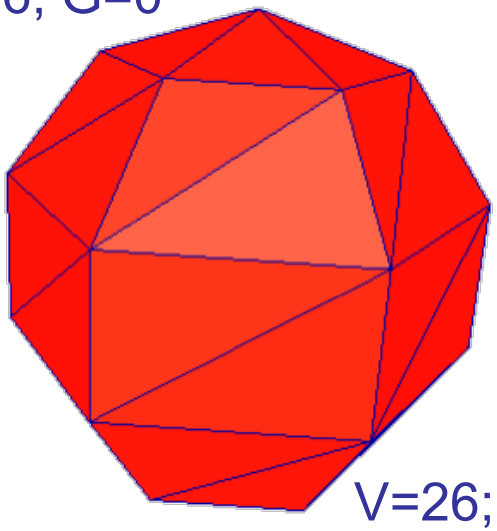
$$G = -(V + F - E - 2)/2$$



V=8; E=12; F=6; G=0



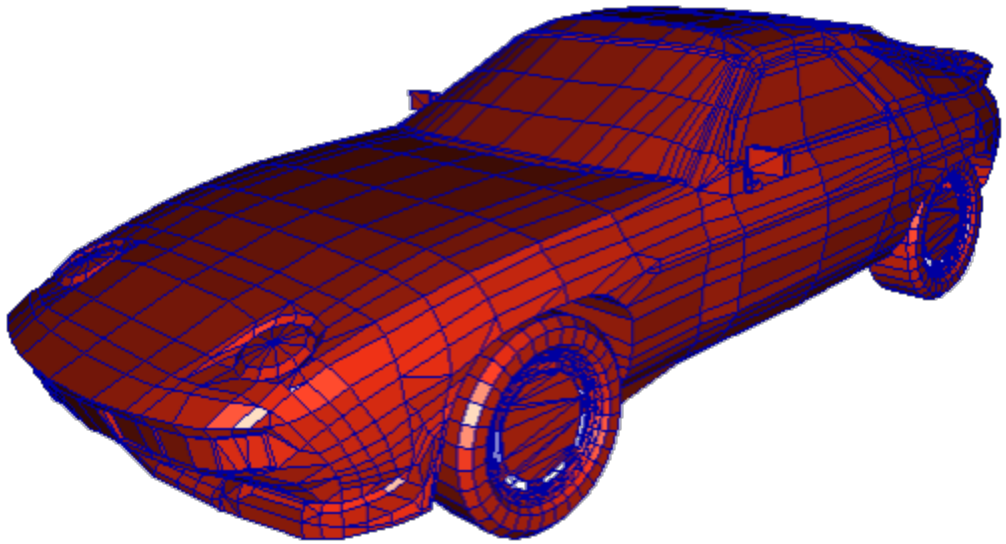
V=36; E=108; F=72; G=1



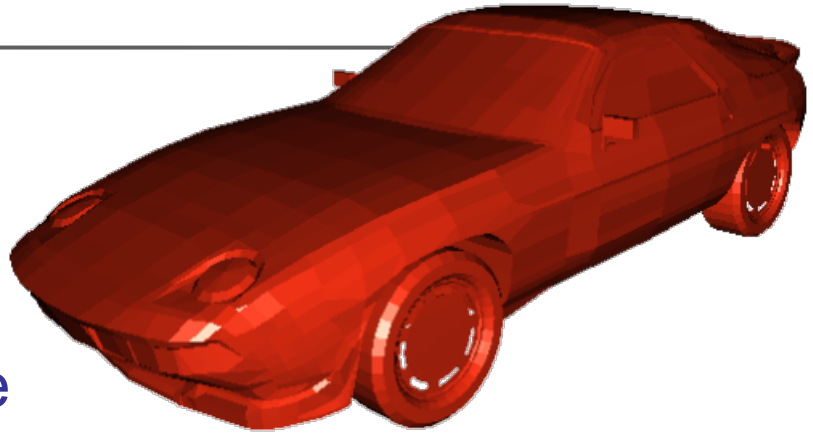
V=26; E=72; F=48; G=0

Memorizzazione di una Mesh

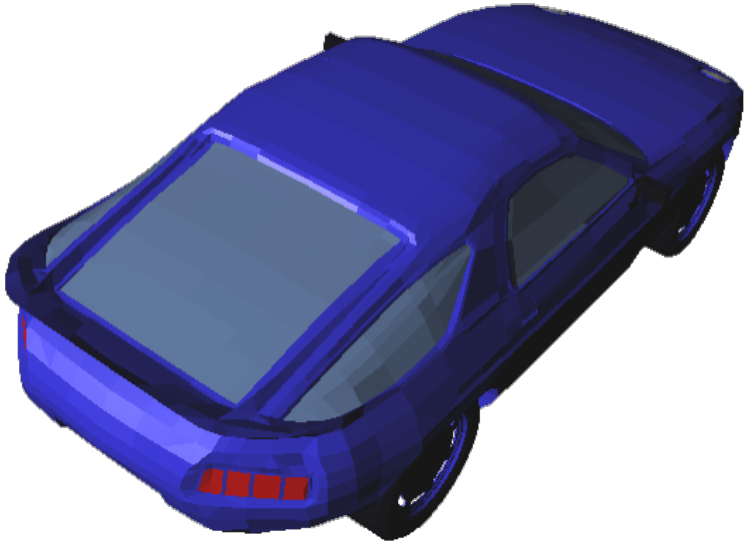
- Modo **indicizzato**
  - Lista di vertici
    - per ogni vertice le sue coord. X, Y, Z
  - Lista di facce orientate
    - per ogni faccia, indici dei vertici



Disegno di Mesh



- Definizione attributi colore
  - per **vertice**
    - un colore per ogni vertice
  - per **faccia**
    - un colore per ogni faccia



- Definizione attributi texture
  - per **vertice**
    - una coppia di coordinate (u,v) texture

Formato OBJ

E' un formato commerciale della Alias-Wavefront molto diffuso. Puo' essere in binario, o in ASCII (testo). Permette la memorizzazione di geometria, topologia, materiali e texture.

```
# Blender v2.80 (sub 75) OBJ File: ''
# www.blender.org
mtllib cube.mtl
o Cube
v  1.000000  1.000000 -1.000000
v  1.000000 -1.000000 -1.000000
v  1.000000  1.000000  1.000000
v  1.000000 -1.000000  1.000000
v -1.000000  1.000000 -1.000000
v -1.000000 -1.000000 -1.000000
v -1.000000  1.000000  1.000000
v -1.000000 -1.000000  1.000000
...
```

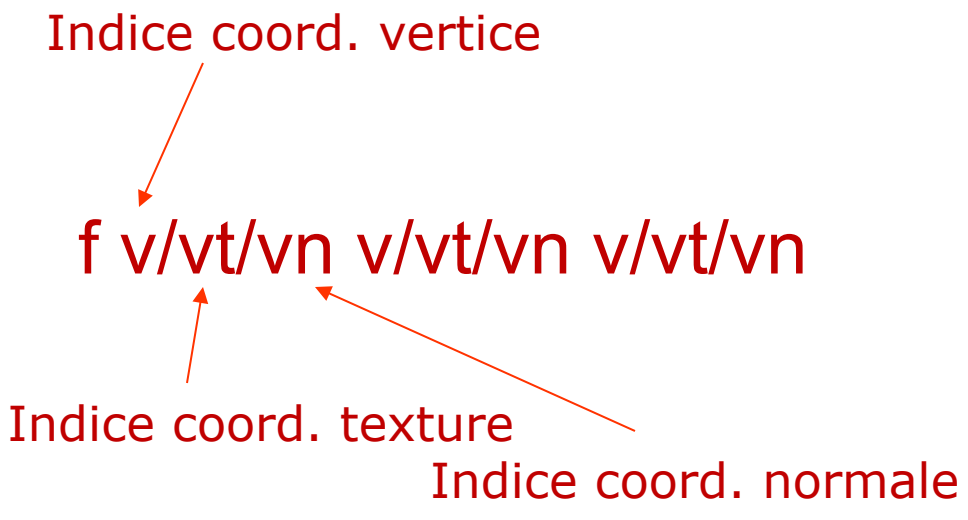
https://en.wikipedia.org/wiki/Wavefront_.obj_file

Formato OBJ continua

```
…
vt 0.000000 0.000000
vt 1.000000 0.000000
vt 1.000000 1.000000
vt 0.000000 1.000000

vn  0.0000  1.0000  0.0000
vn  0.0000  0.0000  1.0000
vn -1.0000  0.0000  0.0000
vn  0.0000 -1.0000  0.0000
vn  1.0000  0.0000  0.0000
vn  0.0000  0.0000 -1.0000

usemtl Material_1
f 1/1/1 5/2/1 7/3/1 3/4/1
f 4/1/2 3/2/2 7/3/2 8/4/2
f 8/1/3 7/2/3 5/3/3 6/4/3
f 6/1/4 2/2/4 4/3/4 8/4/4
f 2/1/5 1/2/5 3/3/5 4/4/5
f 6/1/6 5/2/6 1/3/6 2/4/6
```



Formato OBJ continua

```
# Blender MTL File: 'None'#
Material Count: 2
newmtl Material_1
Ns 400.0
Ka 1.000000 0.000000 1.000000
Kd 1.000000 1.000000 0.000000
Ks 5.000000 5.000000 5.000000
Ke 0.0 0.0 0.0
Ni 1.0
d 1.000000
illum 2
map_Kd webgl-marble.png
newmtl Material_2
Ns 323.999994
Ka 1.000000 1.000000 1.000000
Kd 0.000000 0.000000 0.800000
Ks 0.500000 0.500000 0.500000
Ke 0.0 0.0 0.0
Ni 1.450000
d 1.000000
illum 2
```

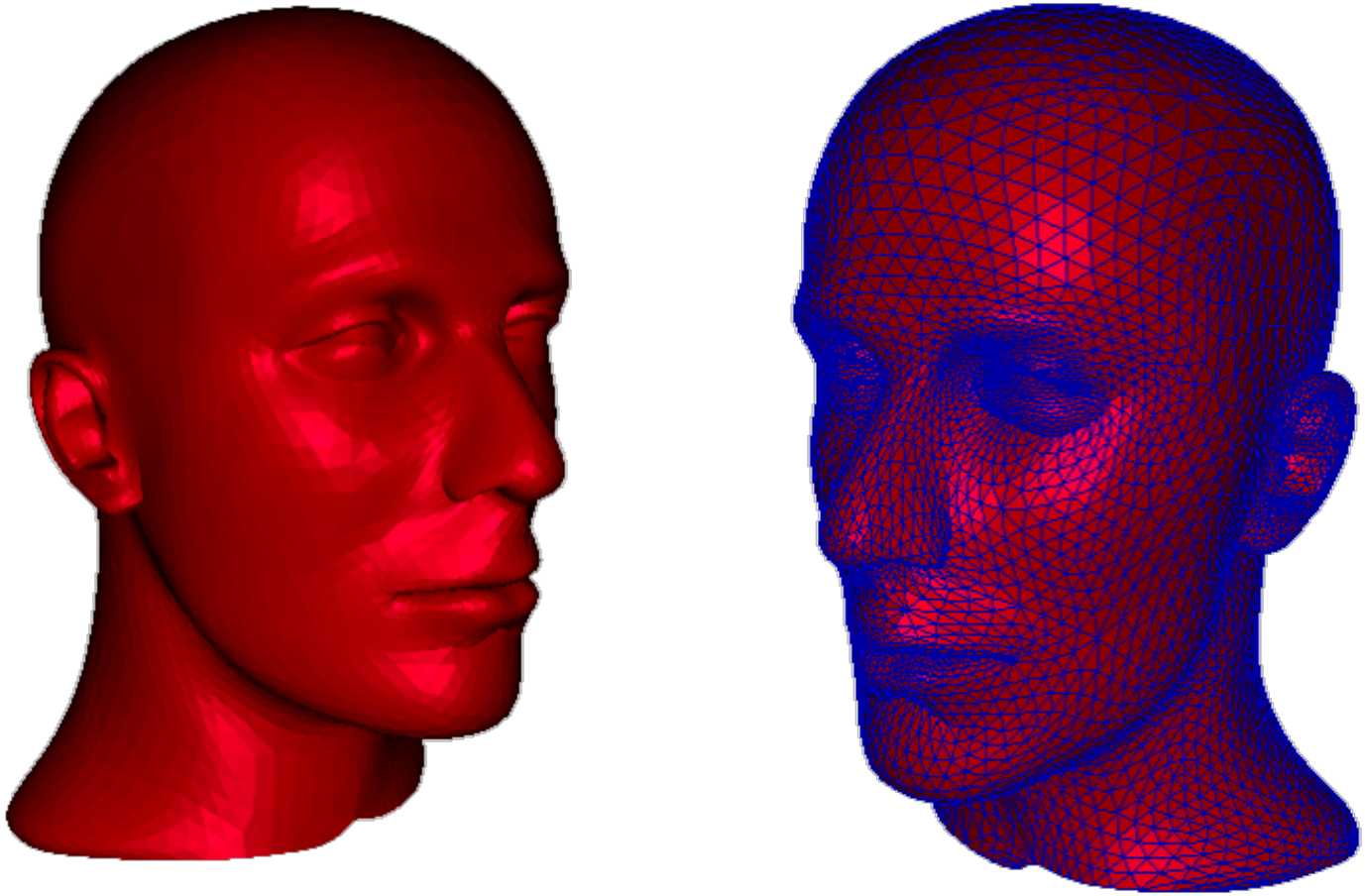
Numero di Materiali

Definizione Material_1

Texture del Material_1

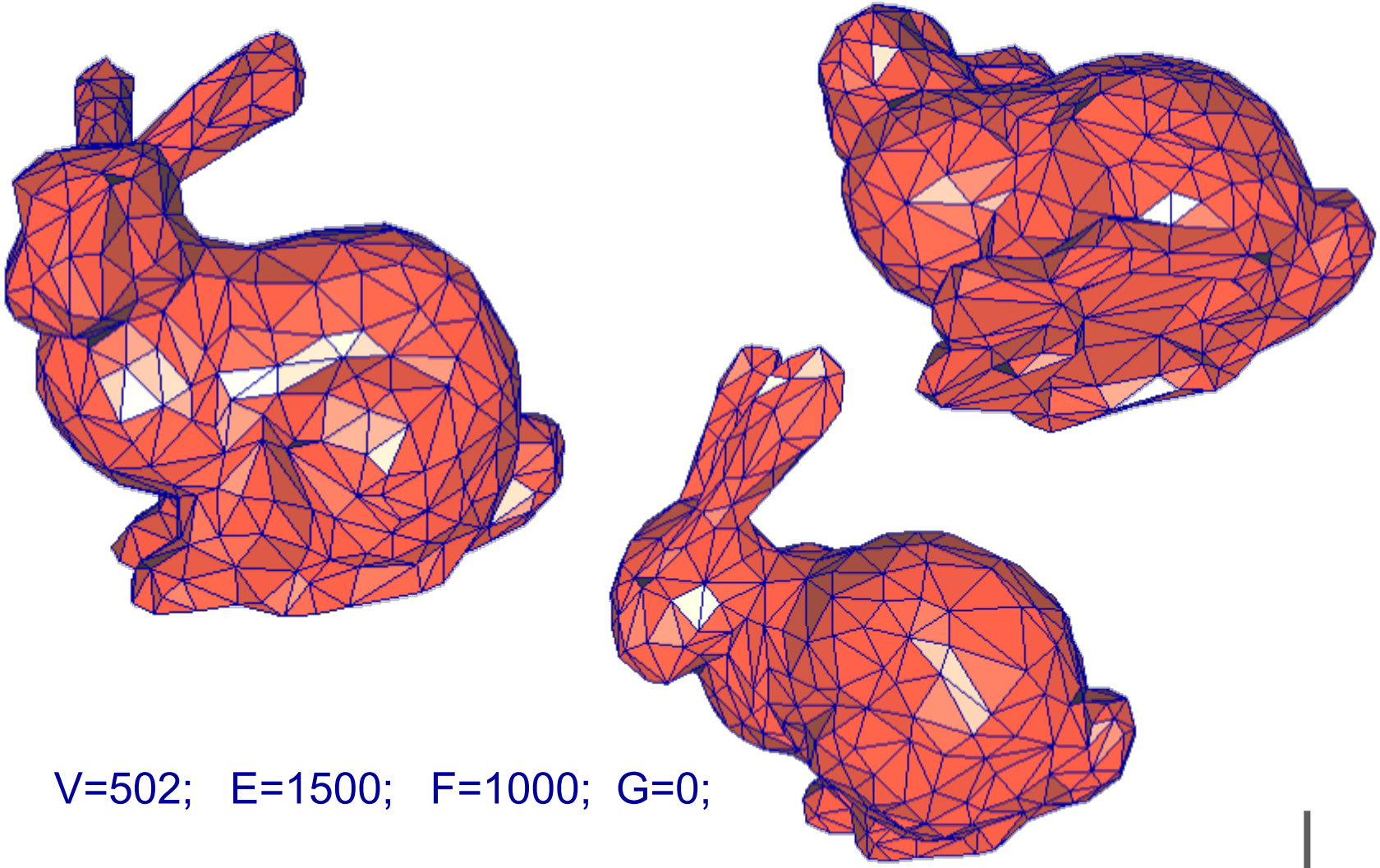
Definizione Material_2

Esempio: mannequin



V=6737; E=20144; F=13408;

Esempio: bunny



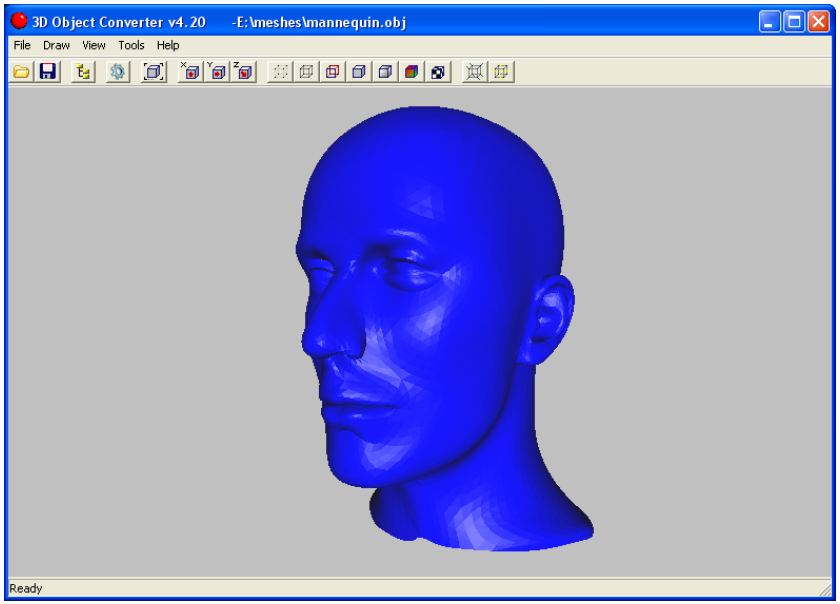
V=502; E=1500; F=1000; G=0;

Convertitori

Per massima compatibilità, ideato un formato, gli stessi creatori forniscono dei convertitori per poter esportare/importare i propri modelli in altri pacchetti.

Un pacchetto utile per Windows, macOS e Linux:

3D Object Converter



Modellatori

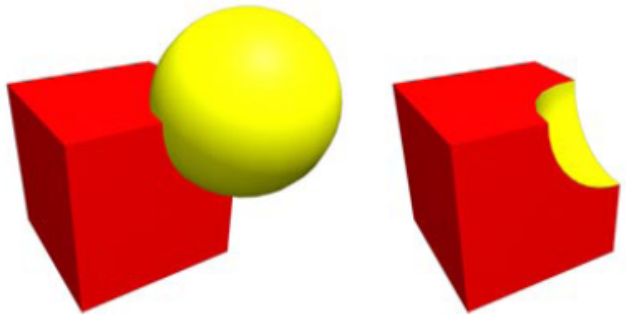
La modellazione 3D è il processo di definizione di una forma tridimensionale in uno spazio virtuale su computer; questi oggetti, chiamati modelli 3D vengono realizzati utilizzando dei software, chiamati modellatori 3D. Questi software permettono di editare la mesh del modello 3D.

I software di modellazione, solitamente permettono di importare ed esportare modelli in differenti formati, oltre ad un formato proprietario.

noi vedremo Blender 3.0

Editing di Mesh 3D

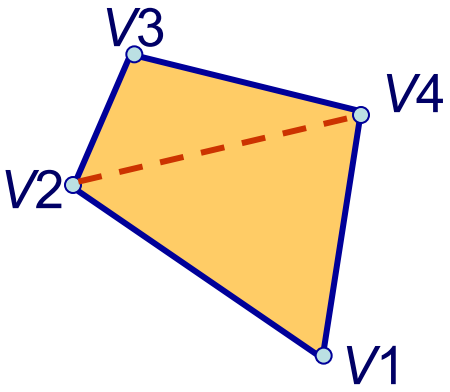
Vertex Editing: delete, move, rotate, scale, …
Edge Editing: delete, move, rotate, scale, collapse, swap, split, remove, …
Face Editing: add, delete, move, rotate, scale, collapse, split, subdivide, breakQuad, flip, …
Mesh Editing: subdivide, breakQuad, joinTriangle, flip, delete invalid face, stitch, fillHole, fillAllHoles, …
Create: triangulates points 3D, triangulate polygon, triangulate polygons
Extract: feature edge, boundary edge, hole, all holes
Modelling: chamfer (VEF), extrude (VEF), offset (VEF), smooth (VEF, mesh), cut, slice, shell, reduce
Boolean Operations: union, intersection, difference



differenza
cubo-sfera

Operazioni invarianti la topologia

Gli operatori di Eulero trasformano una mesh in una differente aggiungendo o rimuovendo vertici, lati e facce nel rispetto della topologia. Per esempio:



1.aggiungere (o cancellare) una faccia inserendo (o eliminando) un lato fra vertici esistenti;

2.aggiungere o cancellare un vertice;

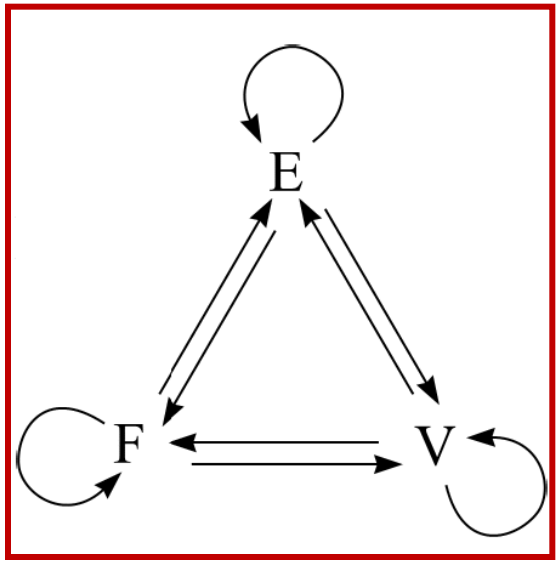
Si osservi come tali operazioni non modifichino il genere della mesh; se per es. si aggiunge un lato, allora E aumenta di 1, ma anche F aumenta di 1 lasciando F-E inalterato.

$$G = -(V + F - E - 2) / 2$$

Adiacenze

Abbiamo visto come sia comodo descrivere una mesh poligonale per memorizzarla in un file, per conservarla o trasportarla, ma all'interno di un programma risulta più efficiente usare una descrizione diversa.

Nelle fasi di elaborazione di una mesh risulta determinante poter accedere ai suoi elementi e ai loro adiacenti nella topologia dell'oggetto.



Per semplicità useremo una coppia ordinata di lettere per indicare le entità coinvolte:

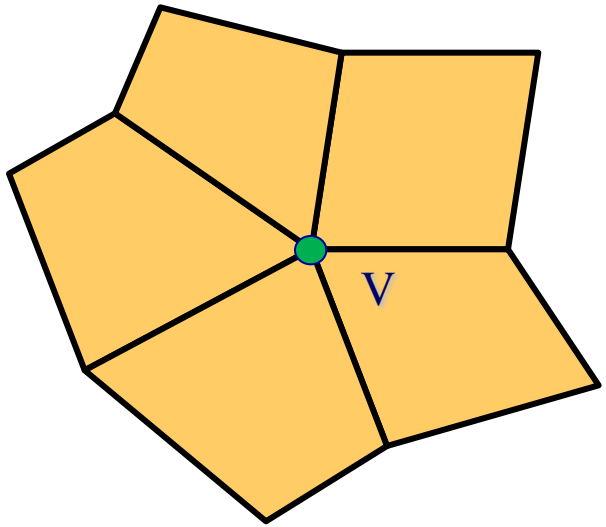
FV: Vertices di una Face

FE: Edges che definiscono la Face

FF: Faces adiacenti a ring di una Face

Adiacenze

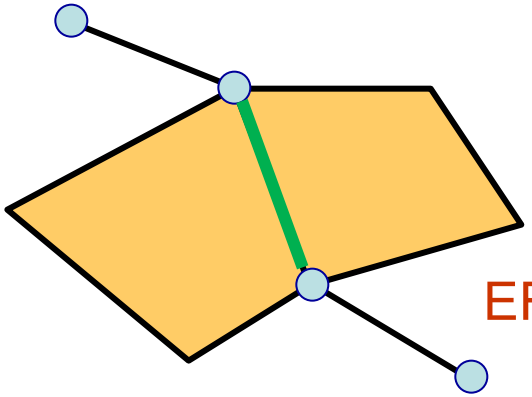
VV: Vertices del primo ring di ogni Vertex
VE: Edges che hanno come estremo un fissato Vertex
VF: Faces a ring incidenti in un Vertex
EF: Faces che incidono su un Edge
EV: Vertices di un Edge
EE: Edge precedenti e consecutivi di un Edge



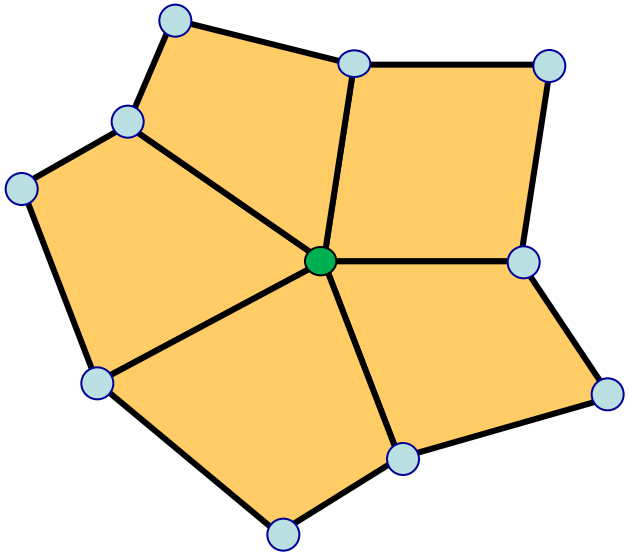
Fra tutte le nove possibile relazioni di adiacenza solitamente se ne considera e mantiene solo un sottoinsieme; le altre vengono ricavate proceduralmente a run-time

VF: primo ring di Faces di un Vertex

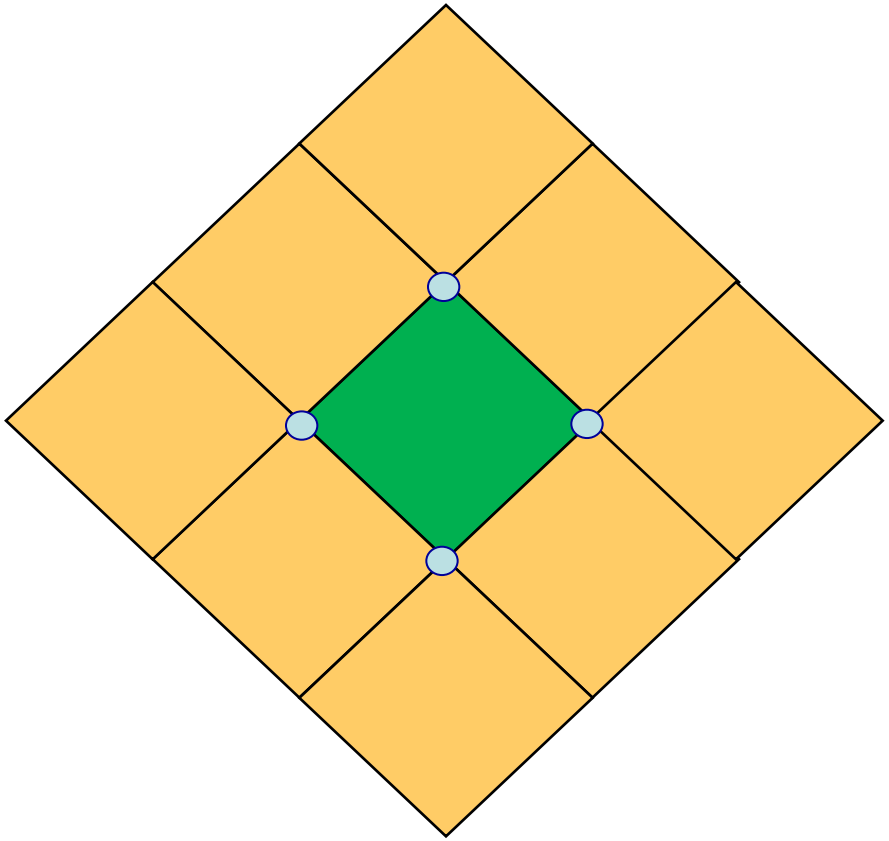
Adiacenze



EF EV EE



VF VV VE



FF FV FE

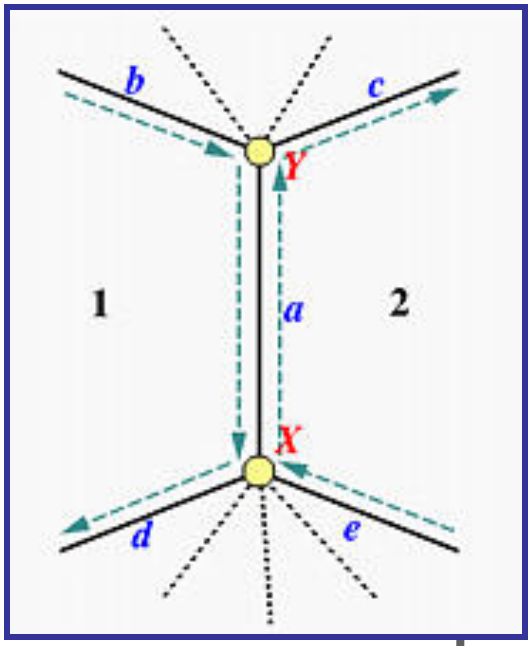
Mesh 3D: strutture dati

- Lista semplice di Vertici (coord. Cartesiane) + lista ordinata degli indici dei vertici che formano le Facce (informazioni aggiuntive possono descrivere una lista di buchi);

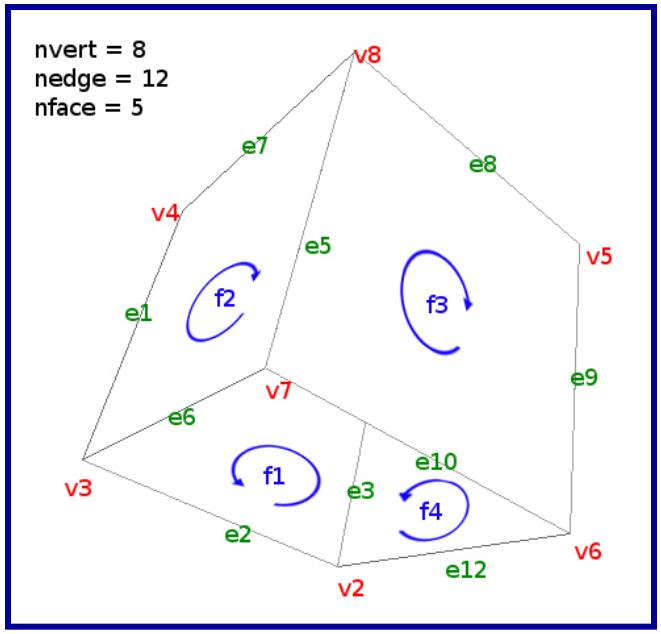
- Lista semplice di Vertici (coord. Cartesiane) + lista di Lati (coppie di indici) + lista di Facce adiacenti per lati;

- Struttura dati Winged Edge

Edge	*Vertices*		*Faces*		*Left Traverse*		*Right Traverse*	
Name	*Start*	*End*	*Left*	*Right*	*Pred*	*Succ*	*Pred*	*Succ*
a	X	Y	1	2	*b*	*d*	*e*	*c*



Esempio Cubo Aperto

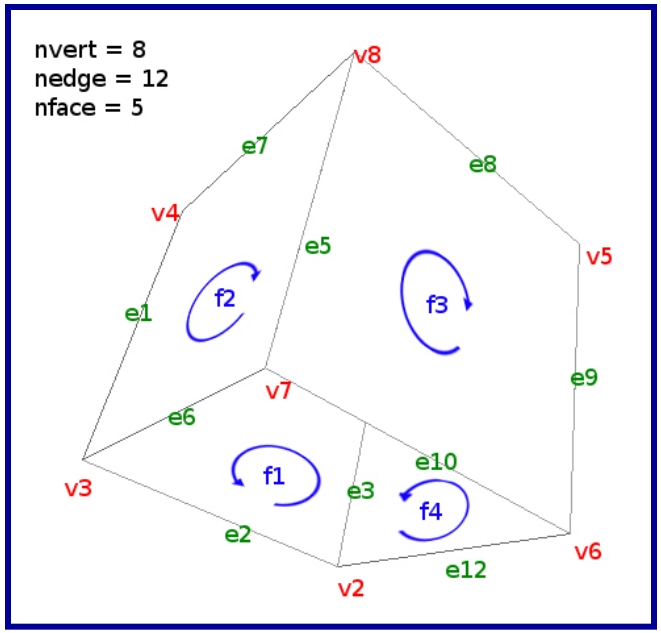



```
Coordinate (X,Y,Z) dei Vertices
1> 1.000000 -1.000000 1.000000
2> -1.000000 -1.000000 1.000000
3> -1.000000 1.000000 1.000000
4> 1.000000 1.000000 1.000000
5> 1.000000 -1.000000 -1.000000
6> -1.000000 -1.000000 -1.000000
7> -1.000000 1.000000 -1.000000
8> 1.000000 1.000000 -1.000000
FV:Indici dei Vertices di ogni Face
1> 4 3 2 1
2> 8 7 3 4
3> 7 8 5 6
4> 5 1 2 6
5> 8 4 1 5
```

Esempio Cubo Aperto

EF:Indici delle Faces di ogni Edge (0 indica Edge di bordo)

```
1:  1  2
2:  1  0
3:  1  4
4:  1  5
5:  2  3
6:  2  0
7:  2  5
8:  3  5
9:  3  4
10:  3  0
11:  4  5
12:  4  0
```



FE:Indici degli Edges di ogni Face (senso antiorario)

```
1:  1  2  3  4
2:  5  6  1  7
3:  5  8  9  10
4:  11  3  12  9
5:  7  4  11  8
```

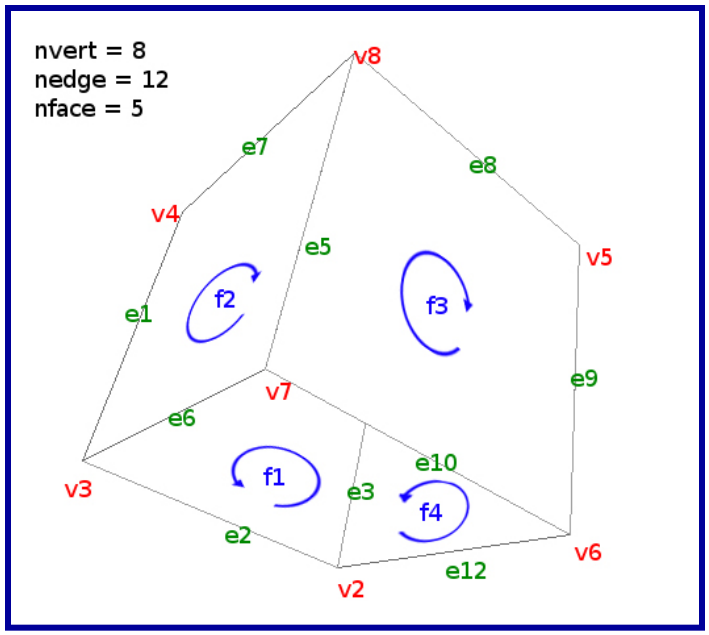
Esempio Cubo Aperto

VF:Indici delle Faces intorno ad ogni Vertex (senso antiorario)

1: 5 1 4
2: 4 1
3: 1 2
4: 1 5 2
5: 4 3 5
6: 3 4
7: 2 3
8: 2 5 3

VV:Indici dei Vertices del primo ring di ogni Vertex

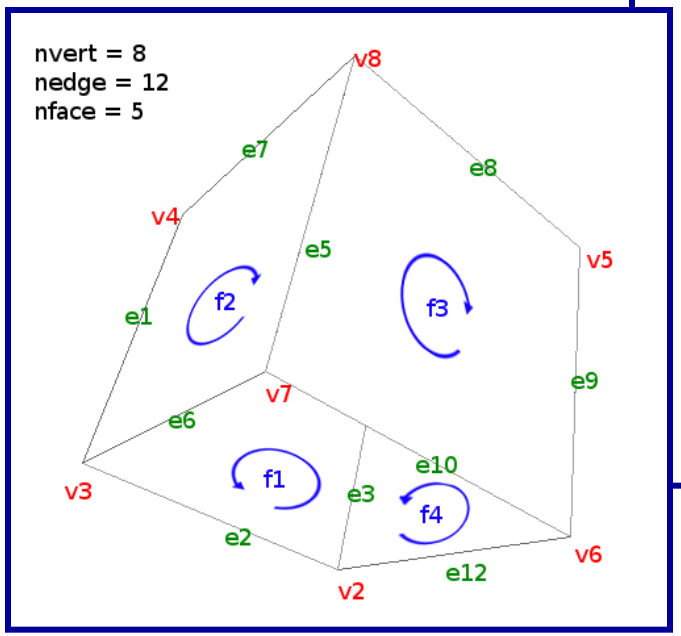
1: 5 8 4 3 2 6
2: 6 5 1 4 3
3: 2 1 4 8 7
4: 3 2 1 5 8 7
5: 1 2 6 7 8 4
6: 7 8 5 1 2
7: 3 4 8 5 6
8: 7 3 4 1 5 6

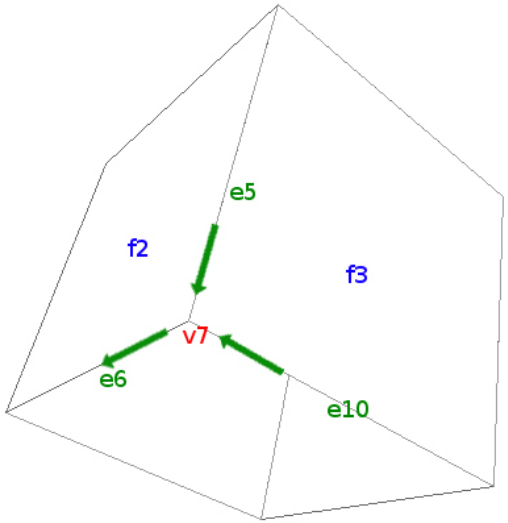


Esempio Cubo Aperto

VE:Indici degli Edges uscenti da ogni Vertex (senso antiorario; un indice negativo indica che l'Edge entra nel Vertex)

1: 4 -3 -11
2: 12 3 -2
3: 2 -1 -6
4: -4 7 1
5: 9 -8 11
6: 10 -9 -12
7: 6 -5 -10
8: -7 8 5





Libreria di funzioni su mesh

Libreria del progetto *XCModel*, per la gestione di unstructured mesh; alcune funzioni utili sono in

HTML5_2d_2/resources/mesh_utils.js

e in particolare segnalo la seguente funzione:

LoadSubdMesh: partendo da una definizione di mesh in termini di vertici V e facce FV (Vertici di ogni Faccia), determina tutte le informazioni FE, FF, VV, VE, VF, EF e EV (non gestisce EE), memorizzandole in opportune strutture dati; altre funzioni presenti ed utili sono:

find_in_vert_face( ) //cerca nella struttura VF un dato indice di faccia

find_in_face( ) //cerca nella struttura FF un dato indice di vertice

Esempio

In HTML5_webgl_3 analizzare e sperimentare il codice:

cube_obj_file_webgl.html e .js

load_mesh.js

che carica il file cube.obj con materiali e texture definite in cube.mtl utilizzando la libreria glm_utils.js.

Analizzare con attenzione l'oggetto mesh prodotto:

```
mesh.SourceMesh  //path del file .obj dato in input
mesh.data        //oggetto con tanti campi
mesh.fileMTL     //path del file .mtl
mesh.materials   //array contenente info sui
                 //materiali e texture
```

Esempio

Scommentando nella funzione LoadMeshFromOBJ nel file load_mesh.js la chiamata alla funzione LoadSubdivMesh, viene utilizzata la libreria mesh_utils.js che calcola tutte le adiacenze di vertici, lati e facce della mesh caricata.

Si analizzi in anche in questo caso l'oggetto mesh prodotto ed in particolare il campo mesh.data

mesh.data.vert
mesh.data.face
mesh.data.edge
mesh.data.nvert
mesh.data.nface
mesh.data.nedge
…

…
mesh.data.facetnorms
mesh.data.mormal
mesh.data.textCoords
mesh.data.group

ai fini del rendering analizzare anche mesh.materials

Repository di modelli 3D

https://www.turbosquid.com/Search/3D-Models

https://grabcad.com/

https://www.blender-models.com/

ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Giulio Casciola

Dip. di Matematica

giulio.casciola at unibo.it