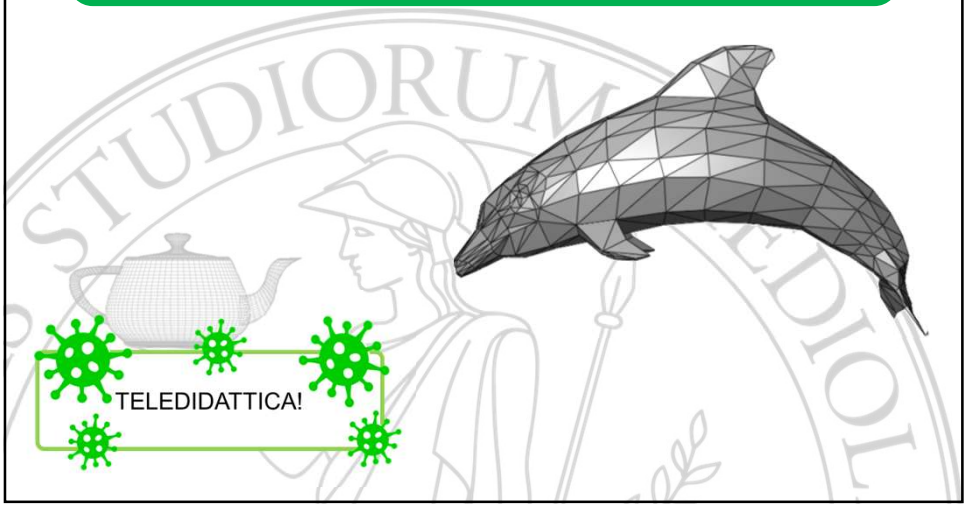
Marco Tarini - Computer Graphics 2019/2020
Università degli Studi di Milano

3D Models: Polygonal Representations



1

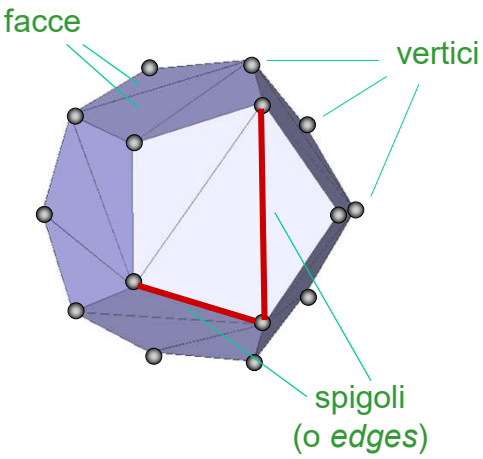
Polygonal Mesh (mesh poligonale)

- ✓ Superficie approssimata da “facce” poligonali
  - ⇒ adiacienti lato a lato (di solito)
- ✓ Il modello digitale più diffuso!
  - ⇒ spesso (e.g.: nei games) sinonimo di modello 3D
  - ⇒ GPU friendly: le schede video sono fatte per renderizzare le mesh (attraverso uno specifico algoritmo, che vedremo: raterization based)

2

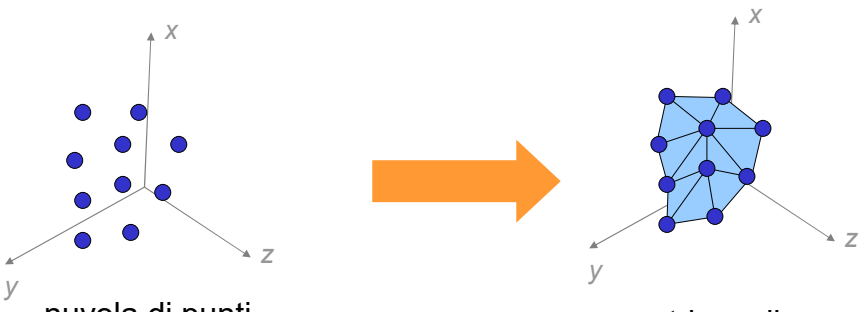
Mesh triangolare (o mesh simpliciale)

✓Un insieme di poligoni adiacenti



3

Da Nuvola di punti a Triangle Mesh



problema abbastanza studiato,

Classi di algoritmi: “Front Advancing” (come “Ball Pivoting”), o volumetric reconstructions

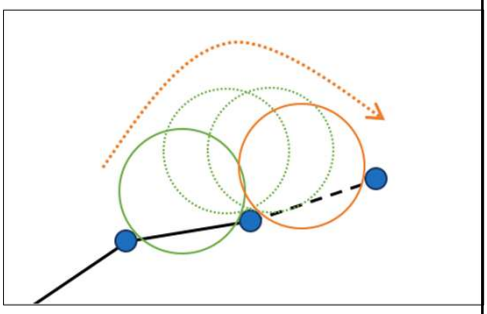
4

Da Nuvola di punti a Triangle Mesh

✓Algoritmi per farlo

⇒Front Advancing:

- es:
  «ball pivoting»

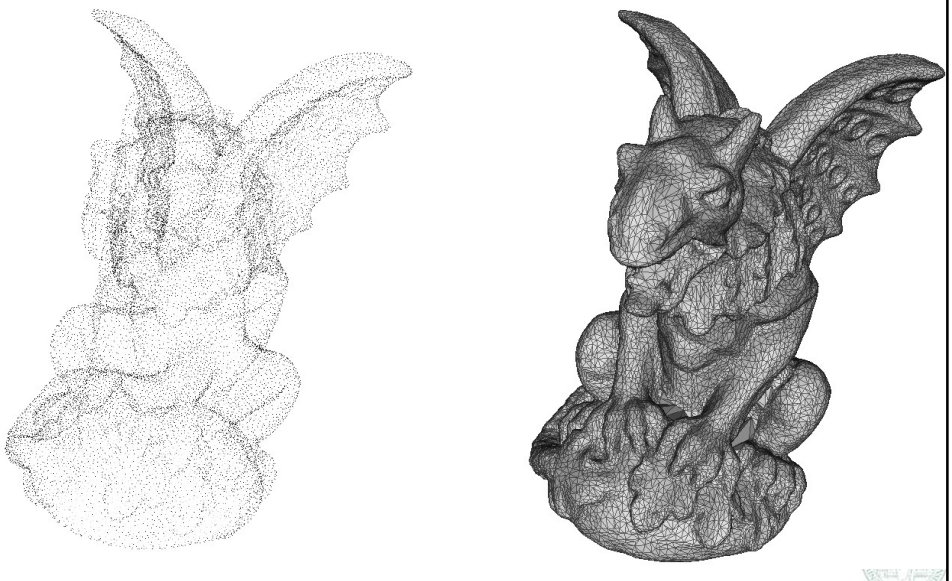


⇒Oppure: Volumetric reconstruction

- Idea:
  1) da nuvola di punti a modello volumetrico,
     (es: attraverso «Poisson reconstruction»),
  2) da modello volumetrico a mesh poligonale
     (es: attraverso «marching cubes»)
- Vedere lezione su modelli volumetrici

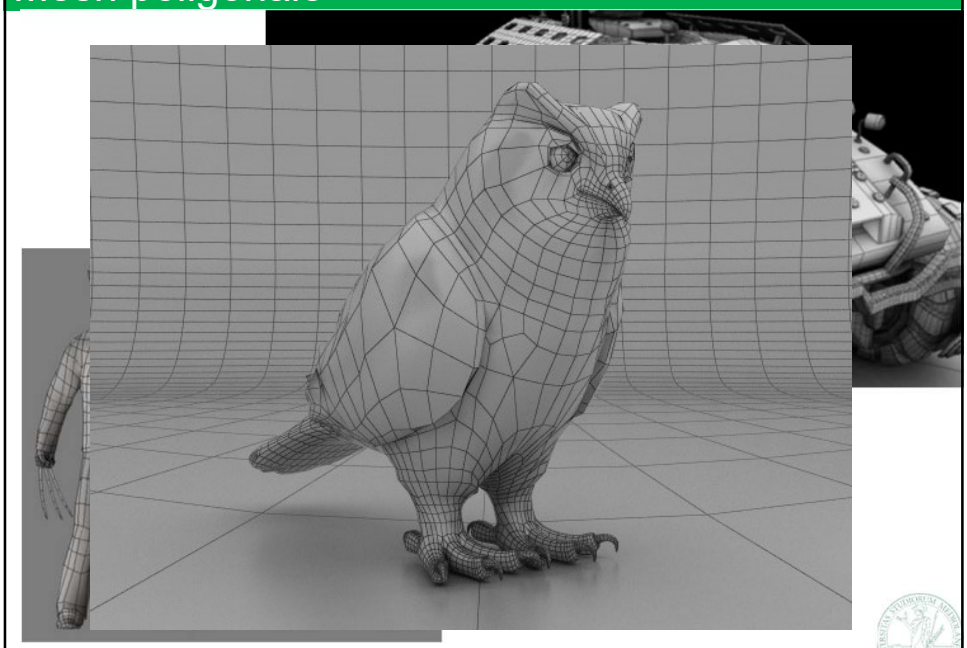
5

Esempio



6

Mesh poligonale



7

Mesh poligonali

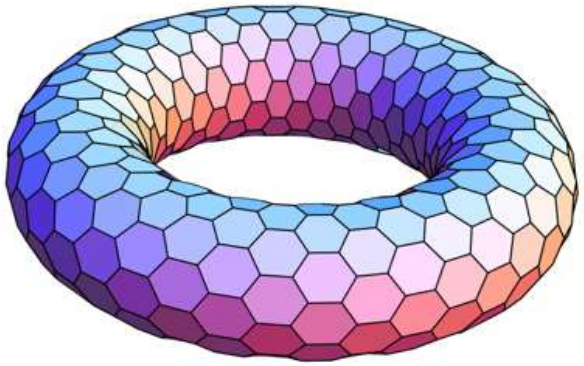
I poligoni possono essere:

- ✓tutti triangoli:
  - ⇒Triangular mesh, o tri-mesh, o “mesh simpliciale”
- ✓tutti quadrilateri (nello slang della CG: “quads”)
  - ⇒Quad-qesh (a volte pure-quad mesh)
- ✓*quasi* tutti quadrilateri
  (ma alcuni triangoli, pentagoni, etc)
  - ⇒ho una “quad-dominant” mesh
- ✓poligoni generici
  (triangoli, quadrilateri, pentagoni, esagoni, etc)
  - ⇒mesh poligonale (generica)

10

Mesh poligonali

✓volendo, esistono anche le hexa-mesh
(e hexa-dominant mesh) (ma assai poco usate)



11

Mesh Polionali: risoluzione

✓ Risoluzione:
il numero di facce (o di vertici) che compongono la mesh

⇒Hi-res: più accuratezza

⇒Low-res: più efficienza

⇒Mesh low res:
detta anche low-poly mesh

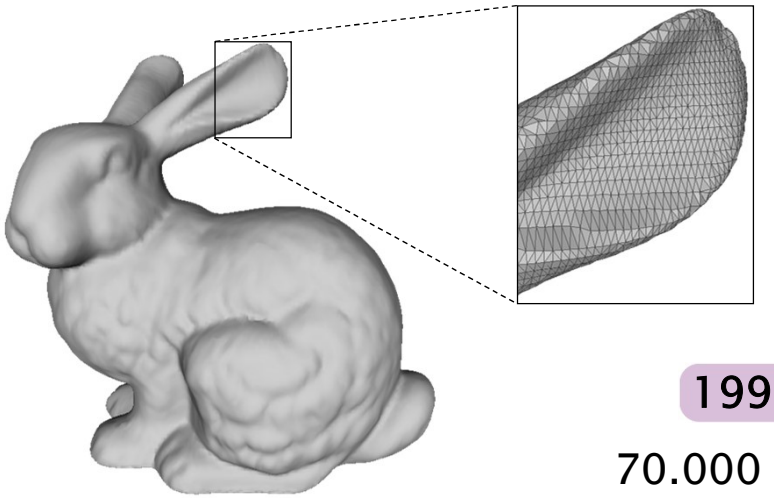
num facce lineare con num vertici.
Statisticamente:
per tri-mesh:
num facce ≅ 2 × num vertici
per quad-mesh:
num facce ≅ num vertici

⇒La risoluzione di una mesh può essere **adattiva**:
tassellamento più fine (campionamento più fitto)
dove necessario

- Per es, dove la curvatura della mesh è alta
  Dove la mesh è piatta, bastano meno triangoli
- Per paragone: la risoluzione di una immagine rasterizzata non è adattiva (rate costante di num pixel per unità di superficie)

12

Risoluzione tipica: crescente negli anni

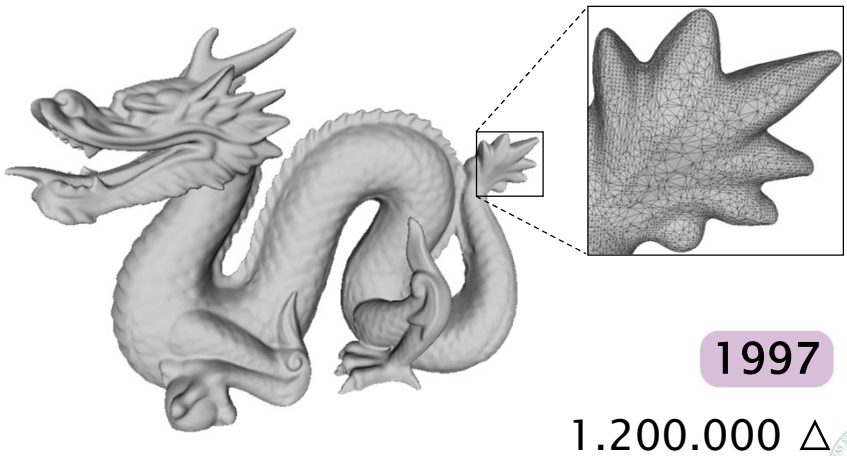


1994

70.000 △

13

Risoluzione tipica: crescente negli anni

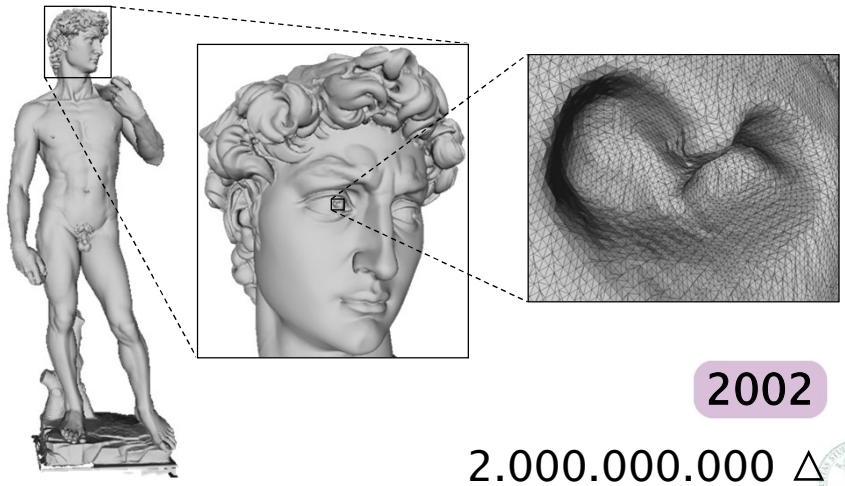


1997

1.200.000 △

14

Risoluzione tipica: crescente negli anni

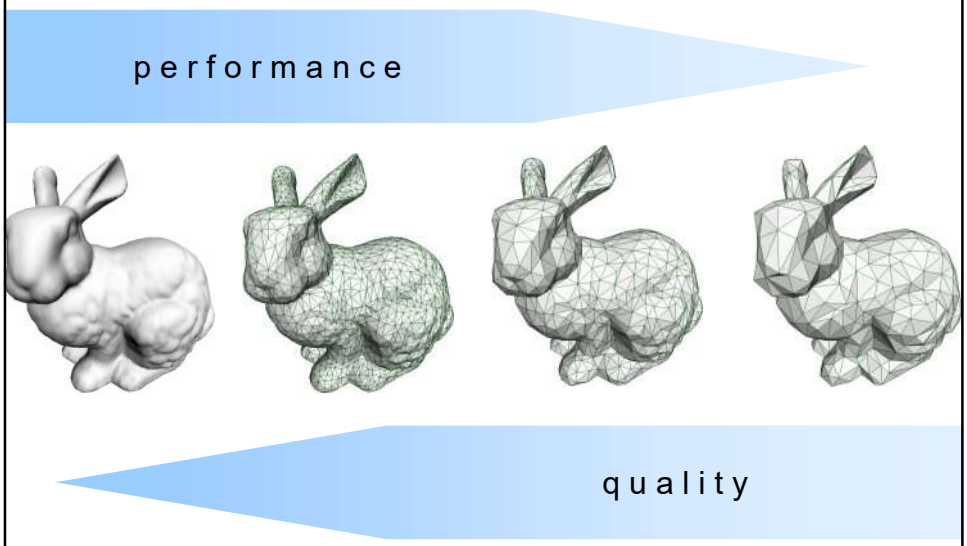


2002

2.000.000.000 △

15

3D Meshes: resolution



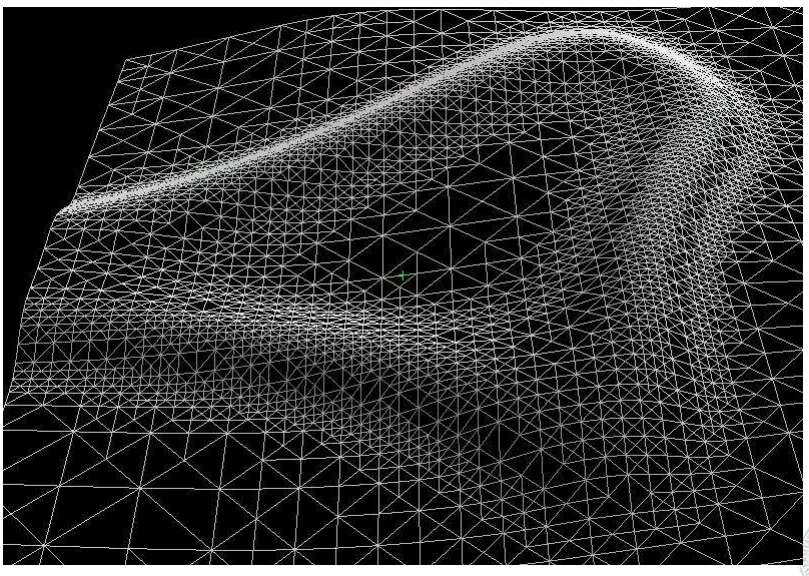
16

Mesh Polionali: risoluzione

- ✓La risoluzione di una mesh può essere adattiva: tassellamento più fine (campionamento più fitto) dove necessario
  - ⇒Per es:
    dove la curvatura della mesh è alta: più triangoli;
    dove invece la mesh è piatta, meno triangoli
  - ⇒Per es:
    dove la mesh è semanticamente importante
    (es sul volto di un personaggio): più triangoli

17

3D Meshes: *adaptive* resolution



18

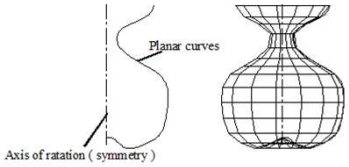
Mesh Triangolare o Tri-meshes

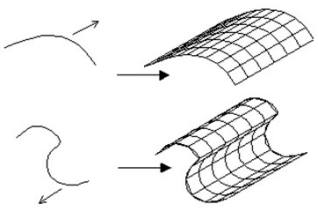
- ✓ Vantaggio: facce sempre planari
  - ⇒ tre punti nello spazio sono sempre co-planari
- ✓ Vantaggio: semplice modo di interpolare attributi (vedi poi)
- ✓ Modo preferito dai matematici
  - ⇒ la chiamano “mesh simpliciale”
  - ⇒ matematicamente: è una una approssimazione “lineare a tratti” della superficie che stiamo rappresentando
- ✓ Modo preferito dalle GPU
  - ⇒ vedremo, l’unico tipo di mesh che possono essere renderizzate direttamente dall’Hardware specializzato per la CG (schede video)
  - ⇒ altri poligoni vengono scomposti in triangoli!
- ✓ Di gran lunga il tipo di mesh più usato in praticamente tutte le applicazioni di CG

19

Pure quad-mesh

- ✓ Generalmente, più difficili da costruire
  - ⇒Esistono un numero minore di tecniche per produrle
  - ⇒Alcune eccezioni:
  - ⇒superfici «di rivoluzione»
  - ⇒per es, superfici «di spazzata»

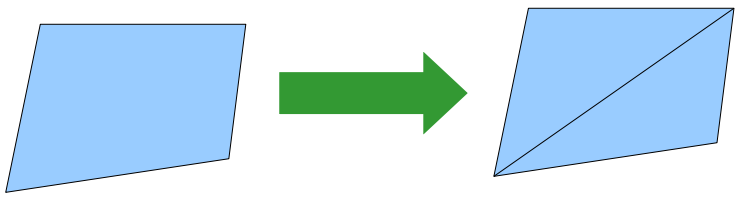




- ✓ Richieste da alcuni algoritmi
  - ⇒es, di suddivisione (vedi poi)

20

Da *Quad meshes* a *Triangle meshes*



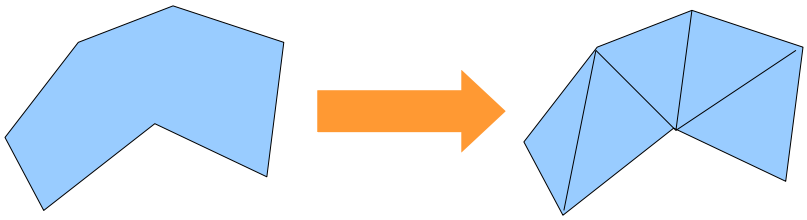
"un quad"

due triangoli!

"diagonal split"

21

Da *Polygonal meshes* a triangle meshes



un poligono a *n* lati?

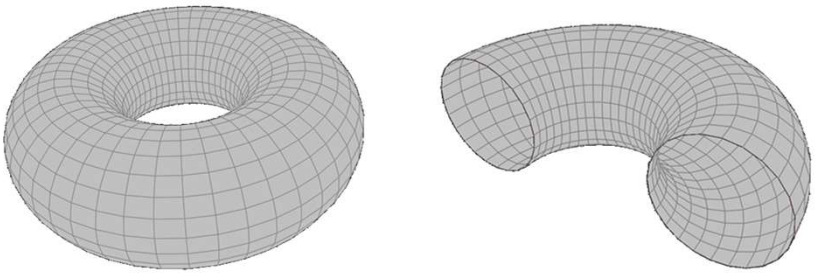
(n-2) triangoli!

triangolarizzazione di poligono:
(in 3D, un problema non del tutto banale...)

22

Mesh chiuse e aperte

- ✓ un edge condiviso da solo 1 faccia è di bordo;
- ✓ un edge condiviso da 2 faccie è interno;
- ✓ se una mesh non ha edge di bordo, è chiusa (altrimenti, è aperta)



23

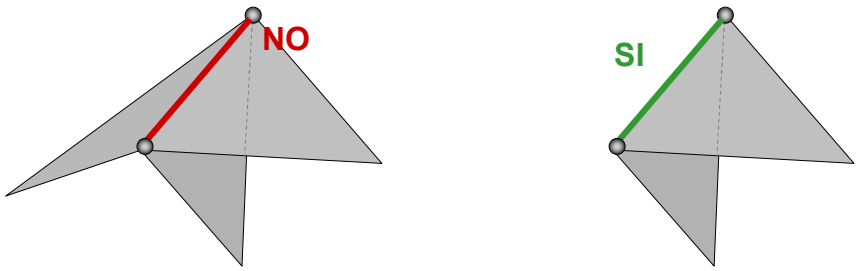
Mesh two-manifold e non

- ✓ una mesh è detta two-manifold (una "varietà due") se *rappresenta in effetti una superficie*
  - ⇒molti algoritmi di geometry processing necessitano che questo sia il caso!
- ✓ Non tutte le mesh (= insiemi di poligoni che condividono dei vertici e degli edge) lo sono!
  - ⇒le facce di una mesh rappresentano sempre (pezzi di) superficie – tutto ok
  - ⇒su edge e vertici le cose possono andare storte
  - ⇒quali condizioni devono verificare?

25

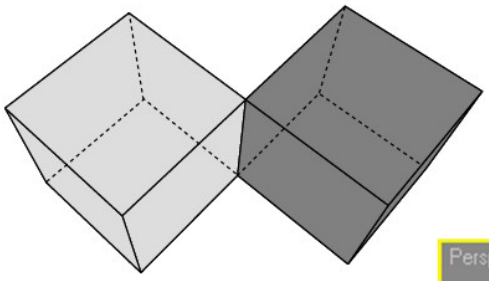
Edge two manifold

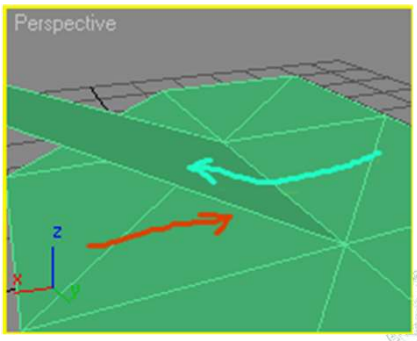
✓ un edge interno è two-manifold se è condiviso da al massimo due facce



26

Esempi di edge non 2-manifolds



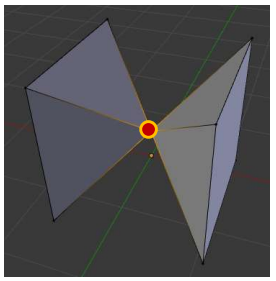


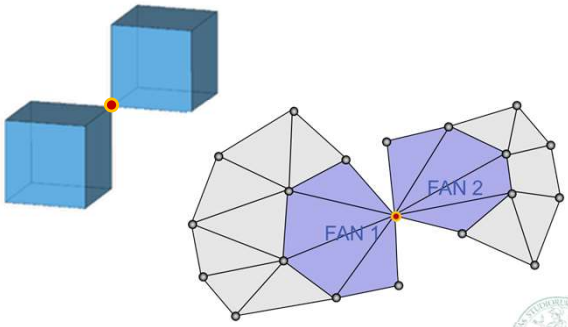
27

Vertice two-manifold

- ✓ due facce sono adiacenti se condividono un edge
- ✓ L'insieme di facce che condividono un vertice tutte adiacenti a coppie = un fan fan (lett.: un ventaglio)
- ✓ un vertice è two manifold se ha un solo fan

Esempi di vertici non 2-Manifold:





30

Mesh di poligonale: struttura dati

✓ Componenti:

⇒**geometria**
- i vertici, ciascuno con pos (x,y,z)
- un campionamento della superficie!

⇒**connettività** (detta anche: "topologia")
- come sono connessi i vertici
- ogni poligono connette alcuni vertici

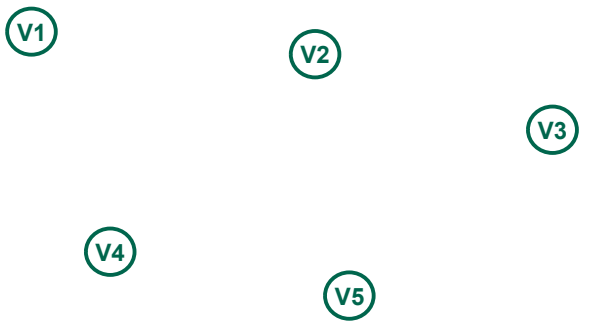
⇒**attributi**
- Definiti sulla superficie
- es: colore, normali, UV, (indice di) materiali, …

32

Mesh: geometria

✓Insieme di posizioni dei vertici

⇒Un vettore posizione (x,y,z) per ogni vertice



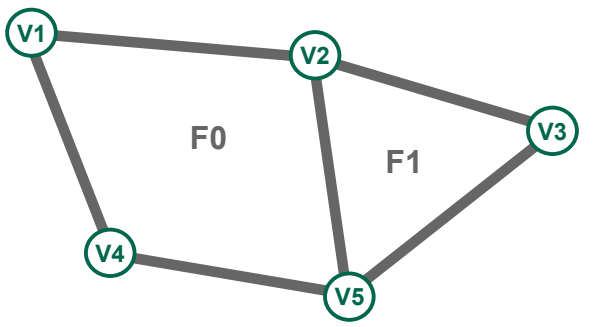
33

Mesh: connettività (o topologia)

✓Facce

⇒poligoni che connettono fra loro i vertici

⇒simile a: nodi connessi da archi, in un grafo



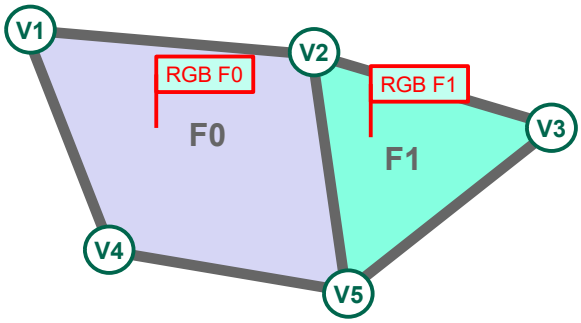
34

Mesh: attributi

✓Quantità che variano sulla superficie

⇒es: colore RGB

⇒definiti per faccia, costanti su ogni faccia?



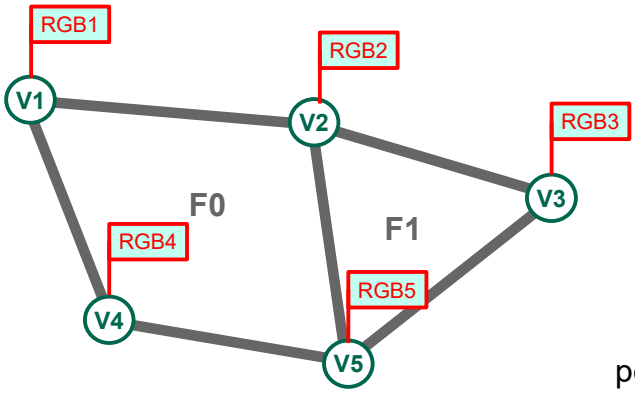
per faccia

35

Mesh: attributi

✓Quantità che variano sulla superficie

⇒Campionati per vertice, interpolati nelle facce

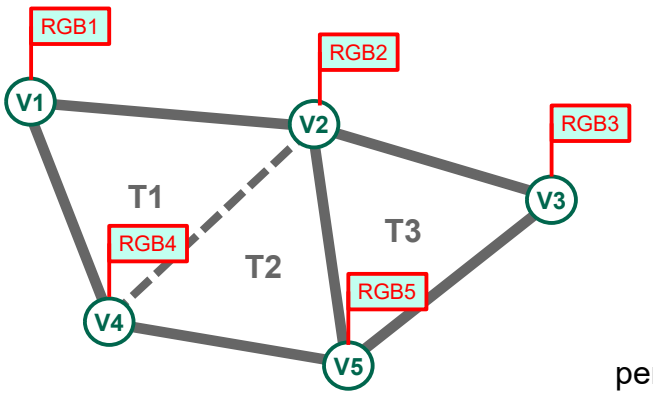


per vertice

36

Mesh: attributi

- ✓ L'interpolazione degli attributi è definita in modo facile dentro a facce triangolari
  - ⇒ Come stiamo per vedere
- ✓ Quindi, per prima cosa, gli altri poligoni devono essere suddivisi in triangoli



per vertice

38

Mesh: attributi

- ✓ Modellano quantità che variano sulla superficie
- ✓ Esempi:
  - ⇒ Colore
    - tipicamente espresso come RGB
  - ⇒ Vettore «Normale»
    - quale orientamento ha la superficie in quel punto?
  - ⇒ Le cose più varie, dipendenti dall'applicazione:
    - per es: temperatura e pressione alla superficie (in una simulazione fisica)
    - per es: «coefficiente di vulnerabilità» (in un gioco)
    - per es: qualità della ricostruzione, in un modello di un oggetto reale (quanto è accurata la superficie qui?)

39

Mesh: attributi

- ✓ Possono essere:
  - ⇒ scalari (es: temperatura, qualità…),
  - ⇒ vettori (es: colore, normale…)
- ✓ memorizzati per vertice
  - ⇒ è caso più comune
  - ⇒ come definire il valore in tutti i gli altri punti della sup? INTERPOLAZIONE LINEARE (see next)
- ✓ oppure memorizzati per faccia
  - ⇒ vengono considerati constanti su quella faccia
  - ⇒ quindi discontinuità C0 fra le facce
    - in corrispondenza cioè degli edge
  - ⇒ caso più raro

40

Interpolazione attributi per vertice nella mesh

- ✓ All'interno del poligono, gli attributi definiti sui vertici (di qualsiasi natura) vengono *interpolati*
  - ⇒ cioè: ogni punto sulla mesh ha, implicitamente, un attributo chè una *interpolazione* degli attributi assegnati ai vertici del poligono a cui appartiene
    - interpolazione = combinazione lineare (convessa) = comb. lineare con pesi non neg a somma 1
- ✓ come definire i pesi dell'interpolazione?
  - ⇒ passo 0: dividere ogni poligono in triangoli inserendo edges
    - procedimento automatico, fatto dietro le quinte
    - es: 1 quad diventa 2 tri, con diagonal split
    - nota: la scelta degli edge inserti influenza il risultato finale
    - in MeshLab, i nuovi edge vengono "faux edges", it: edge "finti", per distinguerli dagli edge originalmente presenti nell mesh

41

Interpolazione attributi dentro un triangolo

Caso analogo (ma più sempilce): interpolazione attributi in **un segmento**

- ✓ osservazione: un segmento di estremi V_0 e V_1 è composto da tutte e sole le interpolazioni lineari di V_0 e V_1
- ✓ cioè, per ogni P nel segmento (estremi inclusi), ho

  $P = a_0 \cdot V_0 + a_1 \cdot V_1$

  per due "pesi" scalari, opportunamente scelti, a_0 e a_1

  non negativi: $a_0 , a_1 \geq 0$

  a somma 1: $a_0 + a_1 = 1$

  e quindi anche $a_0 , a_1 \leq 1$
- ✓ dato P, esistono unici (a_0 , a_1), e viceversa
- ✓ (a_0 , a_1) sono dette le coordinate baricentriche di P nel segmento V_0 , V_1
- ✓ se definiamo un attributo colore $(r,g,b)_0$, in V_0 e un attributo colore $(r,g,b)_1$ su V_1, allora a P assegniamo il colore:

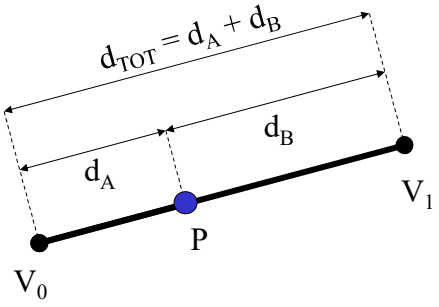
  $a_0 \cdot (r,g,b)_0 + a_1 \cdot (r,g,b)_1$
- ✓ nota: sfruttiamo l'abilità di interpolare qualsiasi attributo (colore, normale, temperatura, etc)

42

Interpolazione attributi dentro un triangolo

Caso analogo (ma più sempilce): interpolazione attributi su di **un segmento**

- ✓ come troviamo le coordinate baricentrice di P nel segmento V_0 , V_1 ?



$$a_0 = d_B / d_{TOT}$$
$$a_1 = d_A / d_{TOT}$$

43

Interpolazione attributi dentro un triangolo

- ✓ osservazione: un triangolo di vertici V_0 V_1 e V_2 è composto da tutte e sole le interpolazioni lineari di V_0 V_1 e V_2
- ✓ cioè, per ogni P nel triangolo (bordi inclusi), ho

  $P = a_0 \cdot V_0 + a_1 \cdot V_1 + a_2 \cdot V_2$

  per tre "pesi" scalari, opportunamente scelti, a_0 a_1 e a_2

  non negativi: $a_0, a_1, a_2 \geq 0$

  a somma 1: $a_0 + a_1 + a_2 = 1$ e quindi $a_0, a_1, a_2 \leq 1$
- ✓ dato P, esistono unici (a_0, a_1, a_2), e viceversa
- ✓ (a_0, a_1, a_2) sono dette le coordinate baricentriche di P nel tri V_0, V_1, V_2
- ✓ se definiamo gli attributi colore

  $(r,g,b)_0$ in V_0 $(r,g,b)_1$ in V_1 $(r,g,b)_2$ in V_2

  allora a P assegniamo il colore:

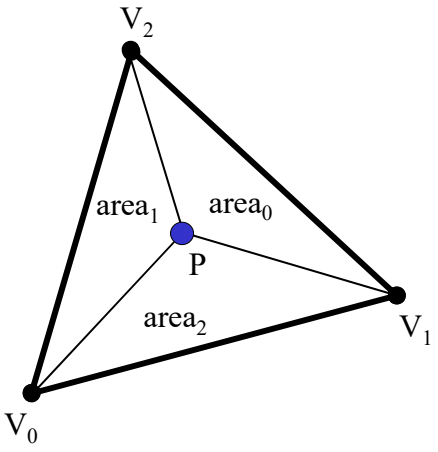
  $a_0 \cdot (r,g,b)_0 + a_1 \cdot (r,g,b)_1 + a_2 \cdot (r,g,b)_2$

44

Interpolazione attributi dentro un triangolo

- ✓ come troviamo le coordinate baricentrice di P nel triangolo V0, V1, V2 ?

$\text{area}_{TOT} = \text{area}_0 + \text{area}_1 + \text{area}_2$



$a_0 = \text{area}_0 / \text{area}_{TOT}$
$a_1 = \text{area}_1 / \text{area}_{TOT}$
$a_2 = \text{area}_2 / \text{area}_{TOT}$

45

Attributi per faccia: normali

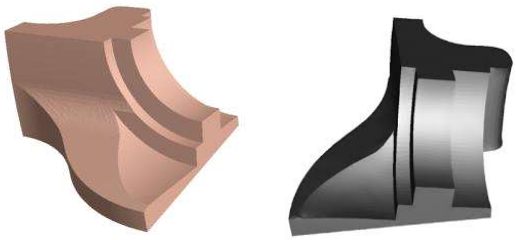
Nel caso in cui l'attributo è la normale

- ✓ per faccia:
  - ⇒normale costante per faccia
  - ⇒facce (dall'aspetto) piatto
  - ⇒discontinuità C0 sugli edge:
    edge sono «spigoli taglienti»,
    detti edge di crease, o «hard» edges
- ✓ per vertice
  - ⇒normale che varia sulla faccia
  - ⇒facce (dall'aspetto) curvo
  - ⇒continuità C0: → aspetto «smooth» (liscio),
  - ⇒(ma non continuità C1)

46

Normali di una superficie: osservazioni

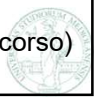
- ✓ Normali costanti = superficie piatta
- ✓ Normali che variano con continuità = superficie curva
- ✓ Normali che variano con discontinuità = superficie con un *crease*
  - ⇒Il crease è costituito dai punti dove la normale è discontinua



47

Normali come attributi per vertice di una mesh

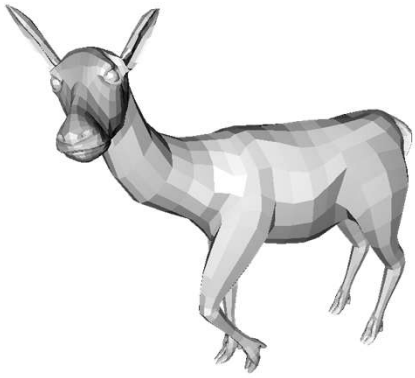
- ✓ Normali definite per faccia:
  - ⇒ Le normali sono costanti sulle facce:
  - ⇒ Le facce appaiono piatte
  - ⇒ coerentemente col modello digitale, ma a differenza (spesso) dell'oggetto 3D che si intendeva rappresentare!
  - ⇒ Appaiono crease su ogni edge della mesh
- ✓ Normali definite per vertice:
  - ⇒ Le normali variano sulle facce (vengono interpolate dentro le facce, come qualsiasi altro attributo)
  - ⇒ Le facce appaiono in generale curve
  - ⇒ Come ottengo una faccia che appaia piatta? Uso una stessa normale come attributo dei tre vertici di un triangolo
  - ⇒ Come ottengo un crease (una discontinuità di normale)? (vedi next)
- ✓ Nota: le normali risultano visibili all'osservatore attraverso l'illuminazione del modello digitale
  - ⇒ Il calcolo dell'illuninazione è un task del rendering (2nda metà del corso)

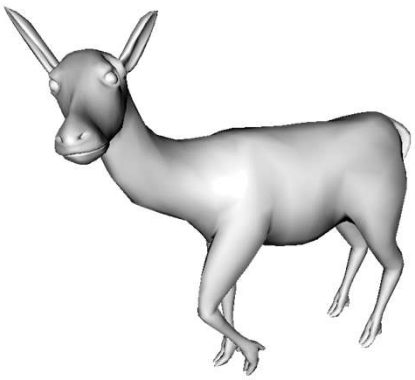


48

Attributi per faccia: normali

Normali per faccia | Normali per vertice





Nota: le normali sono rivelate dall'illuminazione del rendering

49