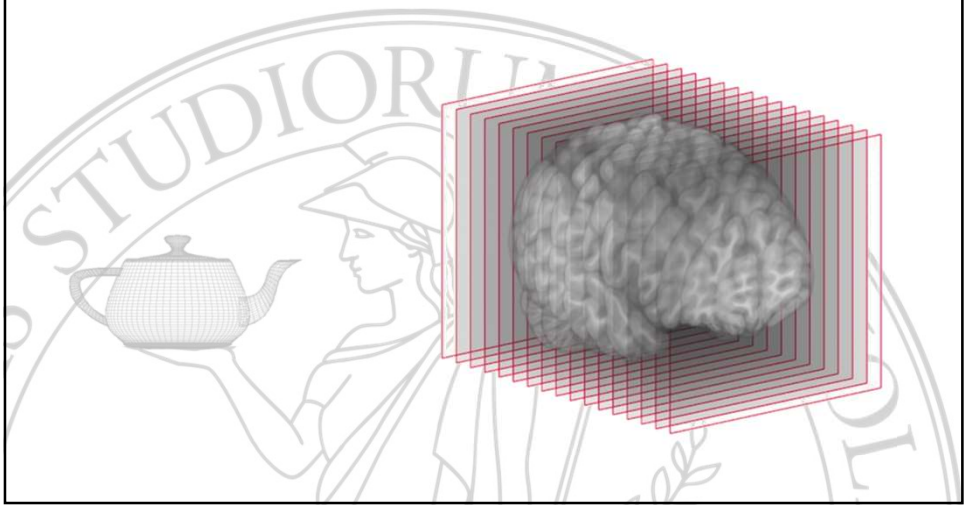
Marco Tarini - Computer Graphics 2020/2021
Università degli Studi di Milano

Rappresentazioni Volumetriche



1

Una (imperfetta) categorizzazione dei tipi di modelli digitali 3D

		ELEMENTI DISCRETI			CONTINUI
		regolari *«a griglia»*	semi-regolari o irregolari: elementi simpliciali	semi-regolari o irregolari: elementi non simpliciali	
SUPERFICIALI	2-manifold *«rappresenta una vera superficie»*	**Height Field** **Range Scan** **Geometry Images**	**Triangle Mesh**	**Polygonal Mesh** **Quad Mesh** **Quad dominant Mesh**	**Subdivision surfaces** **Parametric Surfaces (es. B-splines)**
	non-manifold *«non rappresenta una sup»*	**Set di Range Scan**	**Point Cloud**		
VOLUMETRICI	(3-manifold)	**Voxelized Volume** **Volumetric Textures**	**Tetra Mesh**	**Hexa Mesh**	**Implicit models (es. CSG)**

3

Modelli 3D Volumetrici

1. Discreti & regolari: dataset voxellizati
   ⇒analogo di un immagine rasterizzata, ma in 3D
   ⇒una griglia di voxel

2. Discreti & irregolari: mesh poliedrali
   ⇒analogo di una mesh poligonale (ma nel volume)
   ⇒insieme di poliedri adiacenti faccia a faccia

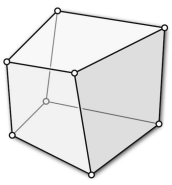
3. Continui: modelli impliciti
   ⇒rappresentazione basata su funzioni volumetriche
   ⇒superficie come luogo di zeri di una funzione

4

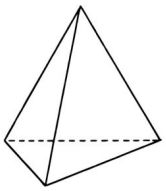
Mesh poliedrale

✓Corrispondente volumetrico delle mesh poligonali

✓Composta da poliedri quali…



esaedro
(o “hexa” per brevità)

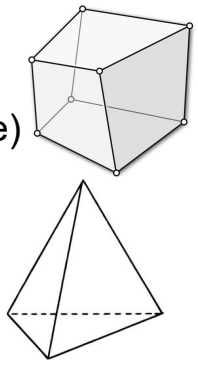


tetraedro
(o “tetra” per brevità)

5

Modelli 3D volumetrici ad **elementi finiti**

- ✓Tipo degli elementi:
  - ⇒hexahedra (anche detti “cuboidi”)
  - ⇒tetrahedra (piramidi a base triangolare)
  - ⇒poliedri generici (raro)



- ✓mesh poliedrale = mesh composta da elementi poliedrici adiacenti faccia a faccia
  - ⇒hexahedron mesh, o hexa-mesh
  - ⇒tetrahedron mesh tetra-mesh
- ✓esempio di visualizzatore: www.hexalab.net

6

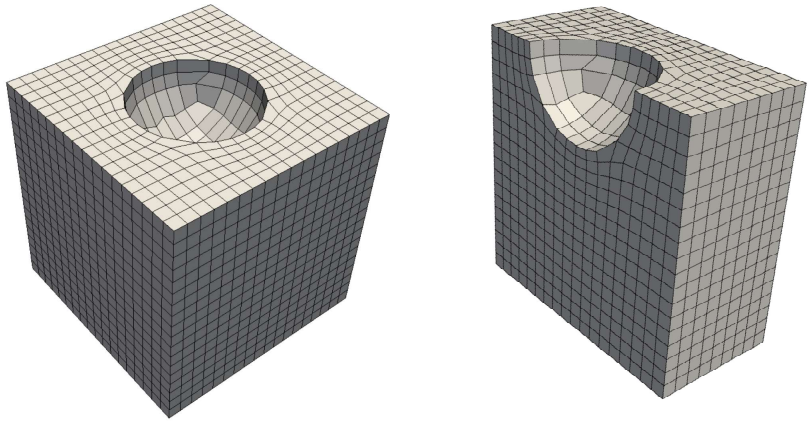
Mesh poliedrale

- ✓Composta da
  - ⇒geometria:
    - vertici (0D), con pos (x,y,z)
  - ⇒connettività:
    - poliedri (3D)
    - facce (2D)
    - edge (1D)

    che connettono i vertici
  - ⇒attributi
    - sui vertici,
    - Implicitamente inteprolati dentro gli elementi
- ✓Struttura dati: simile alla mesh poligonale
  - ⇒indicizzata:
    - lista vertici,
    - lista poliedri
  - ⇒Esistono anche varianti per mesh poliedrali di struttura basata su half-edge

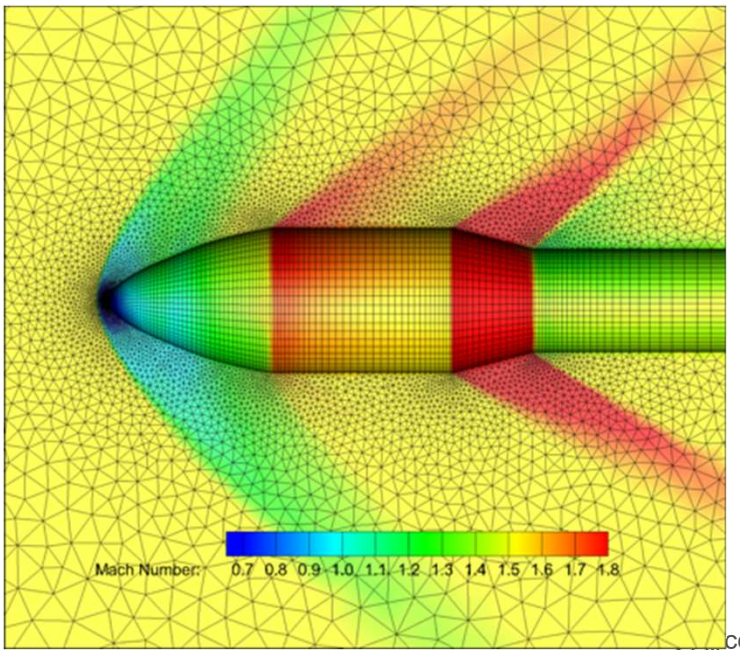
7

Hexa-mesh



8

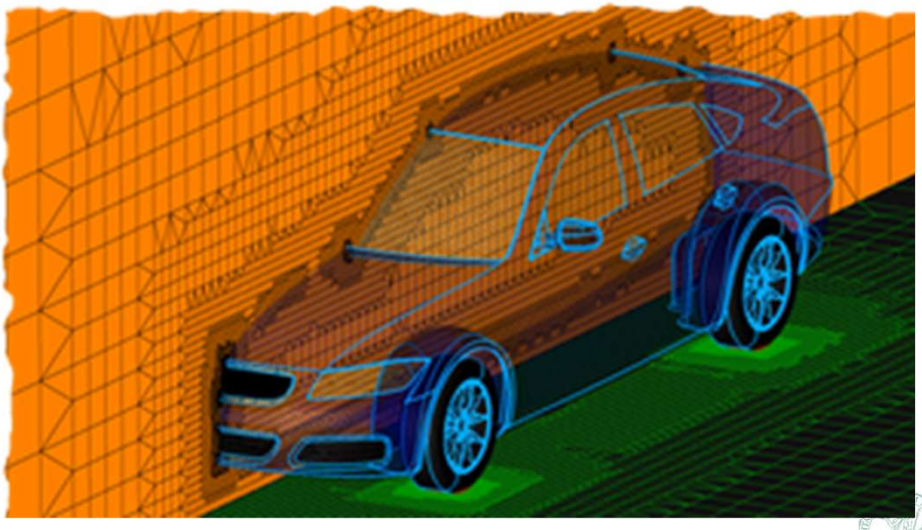
context: FEA (Finite Element Analysis)



ce: pointwise.com

10

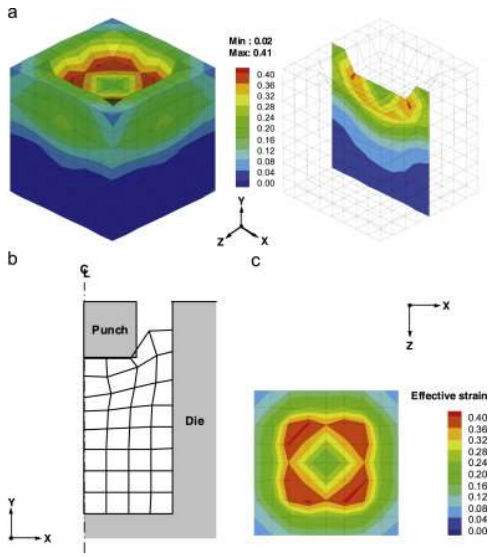
context: FEA (Finite Element Analysis)



source: pointwise.com

11

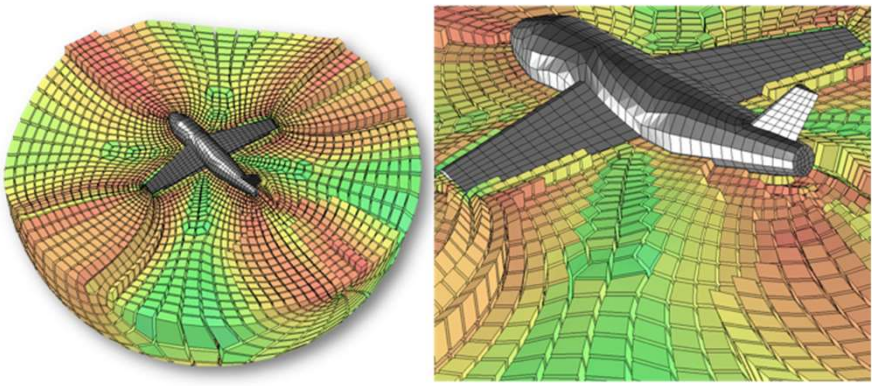
context: FEA (Finite Element Analysis) simulation



Soure: M.C.Lee et al: "3D simulation of forging"
in: Finite Elements in Analysis and Design journal

12

FEA simulation



Source: [Ruiz-Girones Roca Sarrate 2012]

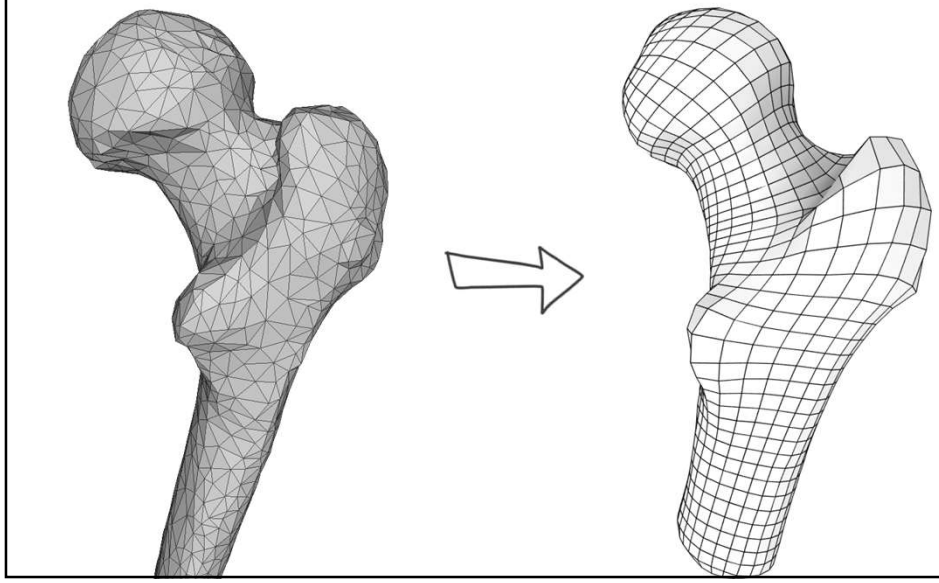
13

Uso tipico: simulazioni fisiche

- ✓ FEM / FEA
  (Finite Element Method /
  Finite Element Analysis)
  - ⇒ Usata in ingegneria per verificare virtualmente le proprietà strutturali degli oggetti rappresentati
  - ⇒ Esempio: simulazione di carico:
    questo palazzo sostiene il suo peso?
    questo ponte sostiene il suo carico?
  - ⇒ Esempio: simulazione di termodinamica:
    come si diffonde il calore all'interno di questo oggetto?
  - ⇒ Simulazione dinamica, o statica
- ✓ Le simulazioni sono il principale uso delle mesh poliedrali
- ✓ Problema (difficile): hexa-meshing o tetra-meshing:
  costuire una hexa-mesh o tetra-mesh
  a partire da una rappresentazione superficiale
  - ⇒ tipicamente, da una mesh poligonale (es: triangolare)

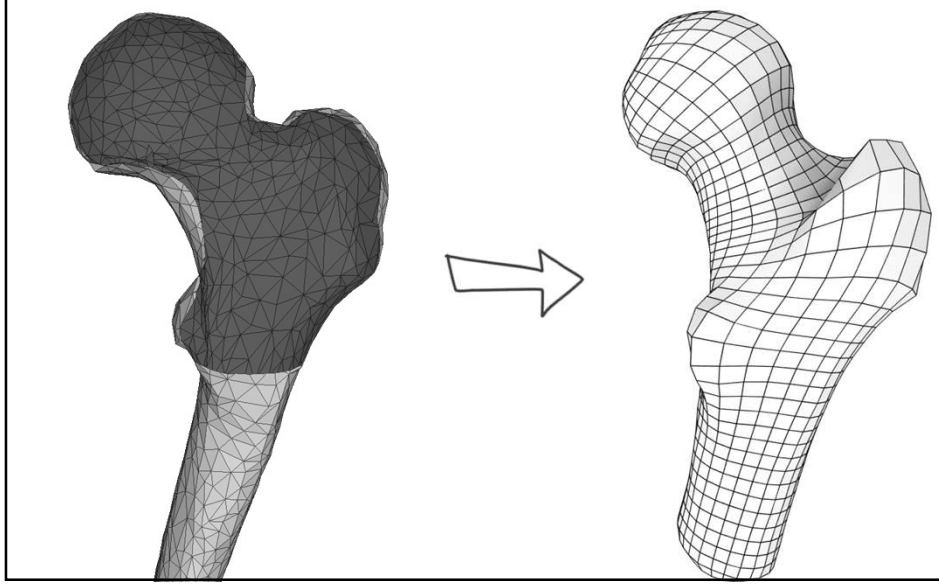
14

Hexa-Mesh construction



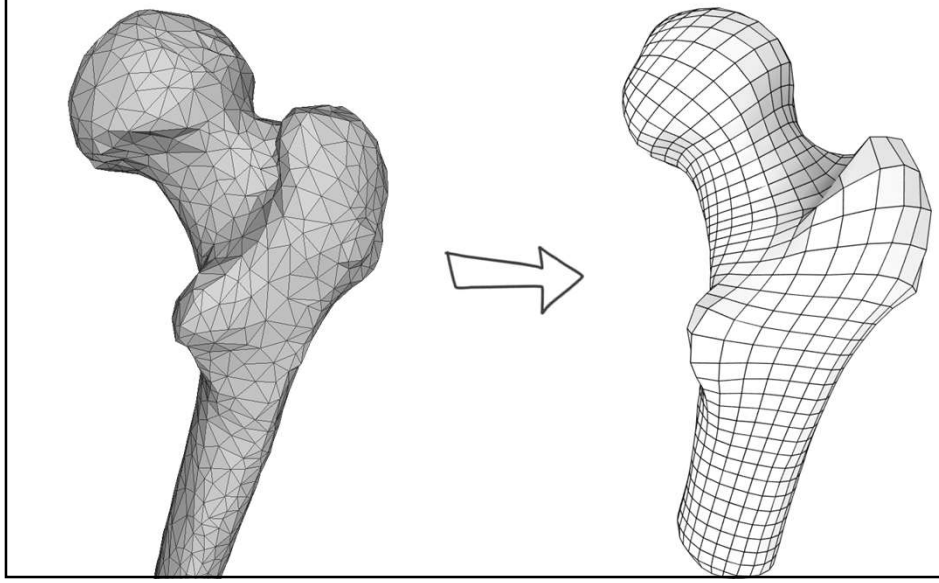
15

Hexa-Mesh construction



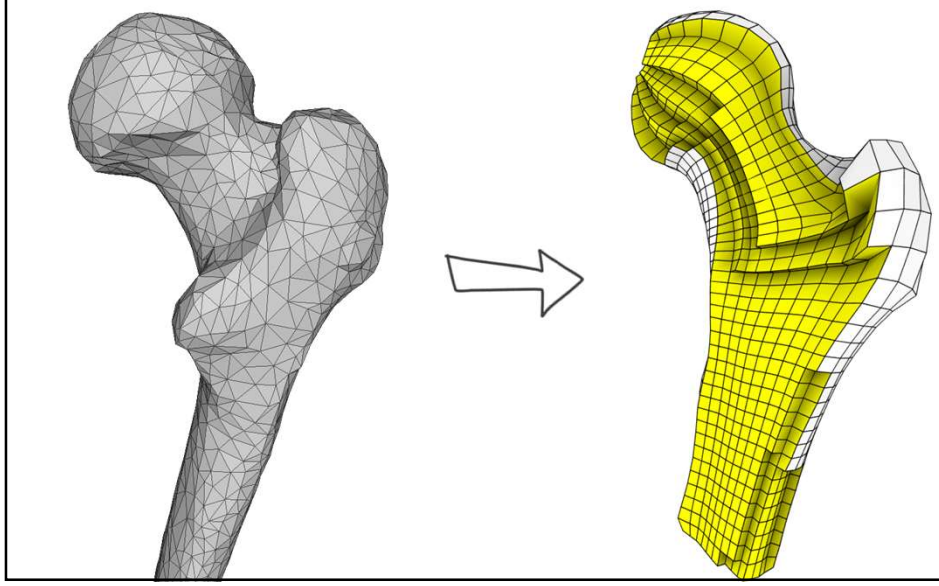
16

Hexa-Mesh construction



17

Hexa-Mesh construction

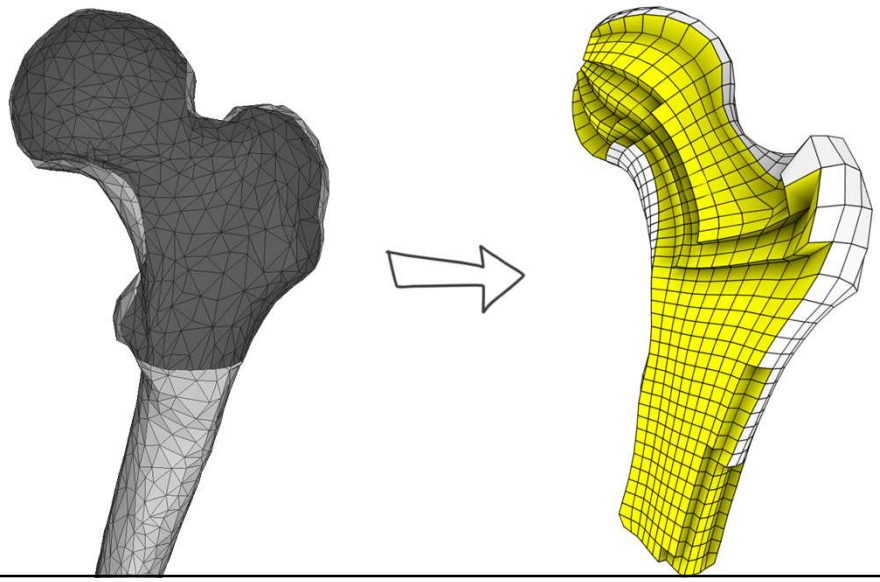


18

Polyhedral-Mesh construction

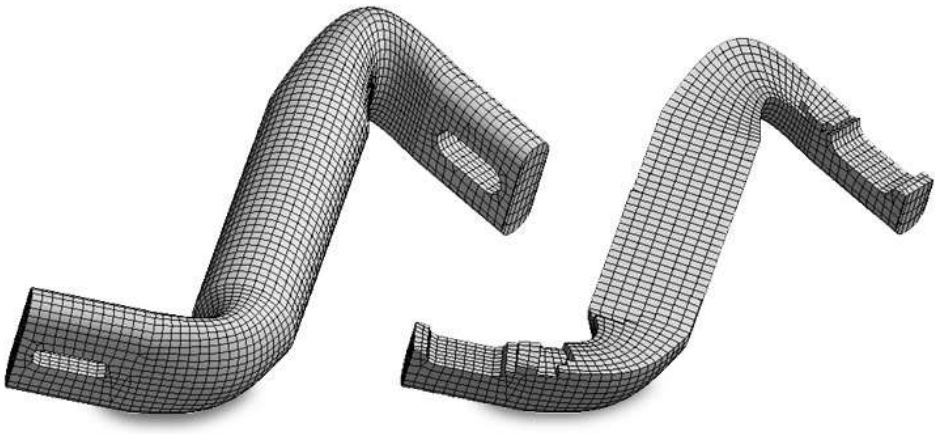
INPUT:
una mesh superficiale
(es: una tri mesh) M

OUTPUT:
una mesh poliedrale (es: una hexa-mesh)
il cui bordo approssima (o è) M



20

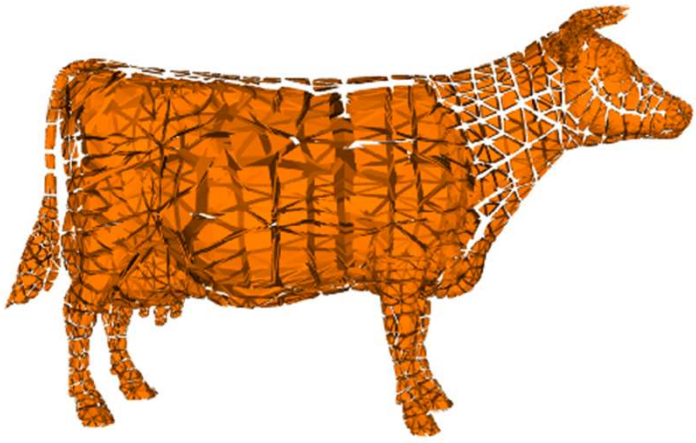
Hexahedron Mesh (Hex-mesh, hexa-mesh)



vedere esempi attraverso il visualizzatore
di hexa mesh online: www.hexalab.net

21

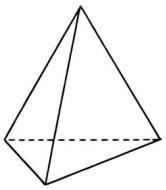
Tetra Mesh



22

Tetraedri

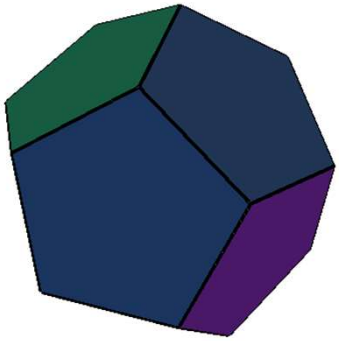
- ✓ Tetraedro: struttura *simpliciale* del volume
  - ⇒ come il triangolo è lo è della superficie.
  - ⇒ cioè: un tetraedro è il luogo di punti che sono l'interpolazione lineare fra i suoi quattro vertici
  - ⇒ cioè: ogni punto P dentro un tetraedro T (superficie compresa) è esprimibile come una (e una sola) combinazione lineare dei quattro vertici di T
  - ⇒ i 4 pesi di questa combinazione (quattro scalari) sono detti le coordinate baricentriche di P dentro T
  - ⇒ posso usare le coordinate baricentriche per interpolare fra gli attributi definiti sui vertici di T
- ✓ In tutto questo: tetra-mesh (volume) del tutto analoga alla tri-mesh (superficie)

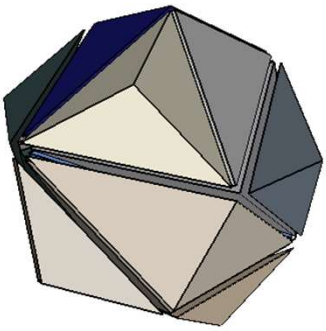


23

Tetrahedralization

- ✓ Ogni poliedro può essere scomposto in tetraedri
  - ⇒Così come ogni poligono in triangoli





24

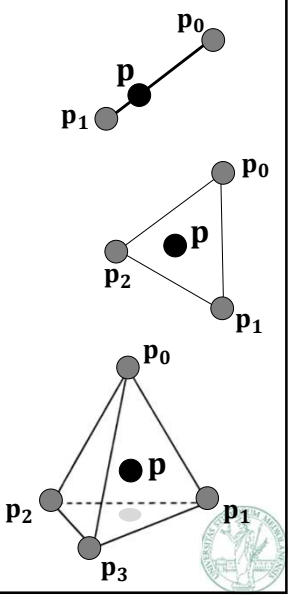
Computo delle «coordinate baricentriche» dentro un qualsiasi elemento *simpliciale*

- ✓ E' lo stesso problema in tutte le dimensioni con la stessa soluzione!
- ✓ Dato un tetraedro (o un triangolo, o un segmento) costituito dai suoi 4 (o 3, o 2) vertici $\mathbf{p}_0 \dots \mathbf{p}_n$ e un punto $\mathbf{p}$ al suo interno, trovare i 4 (3, 2) le «coordinate baricentriche» di p dentro a quel tetraedro (o tri, o sec))
- ✓ cioè i 4 (o 3, o 2) valori scalari $t_0 \dots t_n$ tali che

$$\mathbf{p} = \sum_i t_i \ \mathbf{p}_i \qquad \sum_i t_i = 1 \qquad \forall i : \ 0 \leq t_i \leq 1$$

- ✓ Fatto questo, il valore dell'attributo a in $\mathbf{p}$ sarà così dato dalla stessa combinazione lineare degli attributi $a_0 \dots a_n$ definiti sui vertici :

$$a = \sum_i t_i \ a_i$$

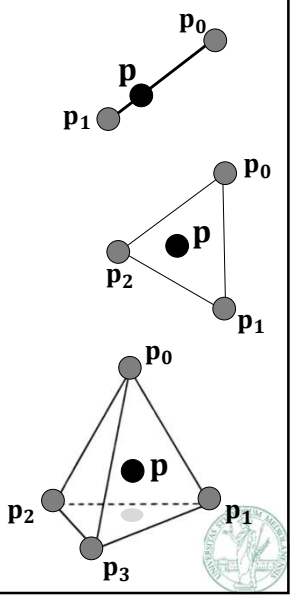


26

Computo delle «coordinate baricentriche» dentro un qualsiasi elemento *simpliciale*

- ✓ Schema generale della soluzione
1. Unire i 4 (3, 2) vertici $\mathbf{p}_0 \ldots \mathbf{p}_n$ al punto $\mathbf{p}$
2. Hai ottenuto 4 (3,2) nuovi sotto-tetraedri (-triangoli, -segmenti) che scompongono il tetraedro (triangolo, segmento) originale
3. Calcola l'estensione (cioè il volume, l'area, la lunghezza) di questi nuovi sotto-elementi
4. La coordinata baricentrica t_i è data dall' estensione dell'elemento opposto al vertice $\mathbf{p}_i$ diviso la somma delle estensioni (cioè diviso l'estensione dell'elemento originale)

- ✓ Esercizio: scrivere la formula nei tre casi, ipotizzando di avere una funzione volume(p0,p1,p2,p3) che restituisce il volume di un tetraedro di 4 vertici dati



27

Tetra meshes o Hexa Meshes?

- ✓ Risoluzione di una mesh poliedrale: n. di poliedri (o di vertici)
  - ⇒ maggiore risoluzione: simulazioni più accurate ma più lente
  - ⇒ nota: numero di elementi è CUBICO con 1/dimensione lineare
- ✓ Può essere adattiva (e spesso lo è)
- ✓ Multi-risoluzione: piramidi di livello di dettaglio sono possibili
  - ⇒ similmente alle mesh poligonali
- ✓ Categorie, simili a mesh poligonali:
  - ⇒ «Pure» hexa-mesh: solo elementi hexa
  - ⇒ «Hexa-dominant» mesh: grande maggioranza di elementi Hexa
- ✓ Esiste un concetto di regolarità locale anche per le hexa meshes / tri meshes
  - ⇒ analogo a quello delle alle mesh, ma definito sugli edge: un edge di una hexa mesh è regolare sse è condiviso da 4 hexa un edge di una tetra mesh è regolare sse è condiviso da 6 tetra
  - ⇒ mesh semiregolare: maggioranza di edge regolari.

29

Tetra meshes o Hexa Meshes

- ✓ Per poter essere utilizzata in una simulazione, una hexa mesh / tetra mesh deve avere elementi di buona «qualità», cioè (semplificando), la loro forma deve essere lontana dall'essere degenere (cioè piatta o, peggio, concava)
  - ⇒ la generazione automatica di mesh poliedrali con questa caratteristica è un problema aperto e difficile
  - ⇒ la generazione è fatta a partire da una struttura superficiale, es una mesh polignonale (deve essere two-manifold, chiusa, ben orientata)
- ✓ Hexa meshes:
  - ⇒ più difficile da costruire
  - ⇒ ma le simulazioni su hexa mesh sono più efficienti (a parità di risoluzione) o più accurate (a parità di tempo di esecuzione)
  - ⇒ recentemente, questo assunto è stato messo in discussione da alcuni risultati teorici di ricerca, che sembrano negare i vantaggi delle hexa-mesh

30

Una (imperfetta) categorizzazione dei tipi di modelli digitali 3D

		ELEMENTI DISCRETI			CONTINUI
		regolari *«a griglia»*	semi-regolari o irregolari: elementi simpliciali	semi-regolari o irregolari: elementi non simpliciali	
SUPERFICIALI	2-manifold *«rappresenta una vera superficie»*	**Height Field**, **Range Scan**, **Geometry Images**	**Triangle Mesh**	**Polygonal Mesh**, **Quad Mesh**, **Quad dominant Mesh**	**Subdivision surfaces**; **Parametric Surfaces** (es. B-splines)
	non-manifold *«non rappresenta una sup»*	**Set di Range Scan**	**Point Cloud**		
VOLUMETRICI	(3-manifold)	**Voxelized Volume**, **Volumetric Textures**	**Tetra Mesh**	**Hexa Mesh**	**Implicit models** (es. CSG)

31

Modelli 3D Volumetrici

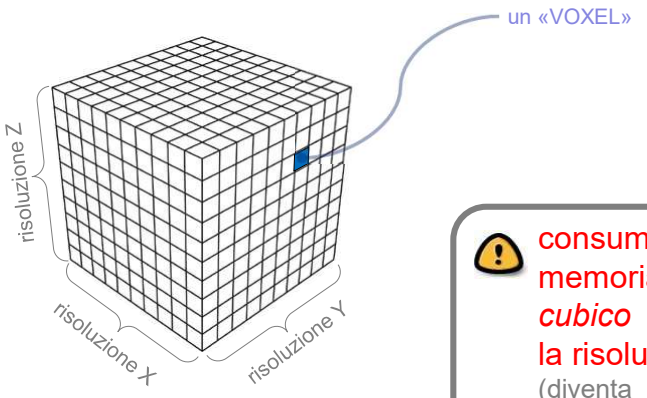
1. Discreti & regolari: dataset voxellizati
   ⇒analogo di un immagine rasterizzata, ma in 3D
   ⇒una griglia 3D regolare di voxel

2. Discreti & irregolari: mesh poliedrali
   ⇒Tetra-mesh, hexa-mesh
   ⇒insieme di poliedri adiacenti faccia a faccia

3. Continui: modelli impliciti
   ⇒rappresentazione basata su funzioni volumetriche
   ⇒superficie: luogo di zeri di questa funzione

32

Modello 3D a voxel (o voxellizato)



Griglia regolare 3D
(o lattice)

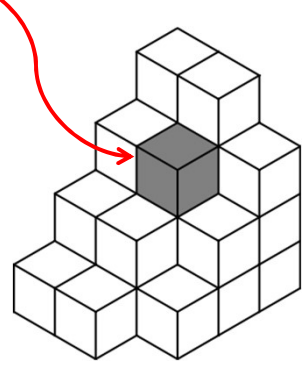
consumo memoria *cubico* con la risoluzione (diventa facilmente ingestibile)

```
array [RES_X] [RES_Y] [RES_Z] of Voxels
```

33

Modelli Voxellizati

- ✓"Voxel" = Volume element
  - ⇒Così come…
    - "Pixel" = Picture Element
    - "Texel" = Texture Element
- ✓Elemento di una griglia regolare 3D
  - ⇒che è anche detta un lattice



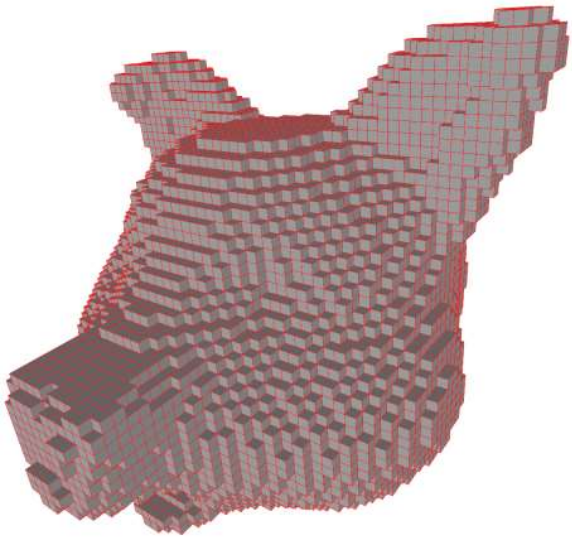
- ✓In codice:

```
Voxel[][][] data = new Voxel[resX][resY][resZ];
```

Esempio in Java

34

In questo caso, 1 Voxel = 1 Boolean (1 bit)

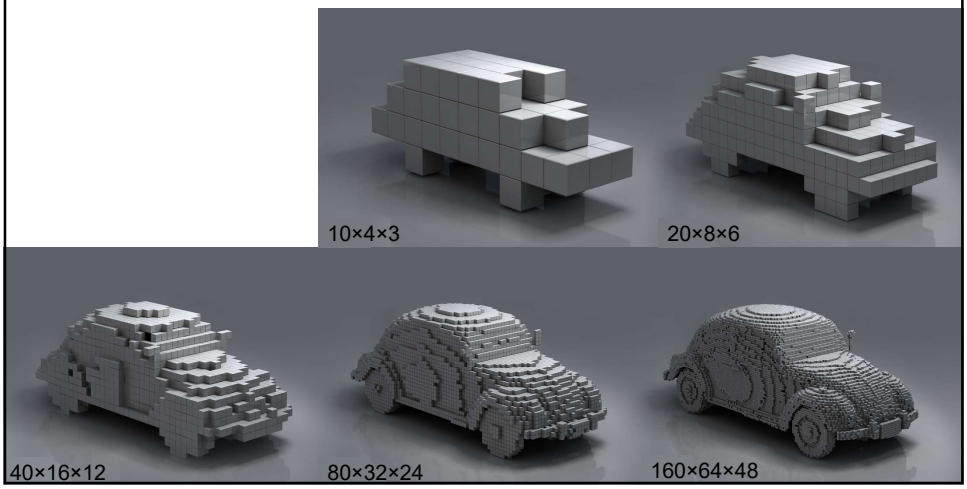


ogni *voxel* è pieno (1) o vuoto (0)

35

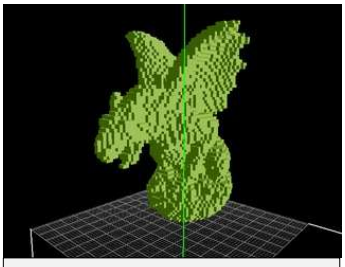
Risoluzione e costo in memoria

✓risoluzione: un intero per lato res = (X,Y,Z)



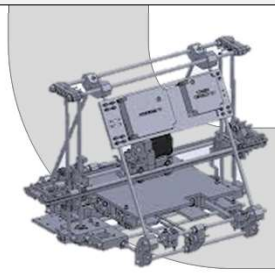
36

Voxel = 1 bit (pieno / vuoto)



Voxelized dataset
Voxel: pieno/vuoto

✓Input naturale di (alcuni)
3D printing devices
⇒Rapid prototyping devices per stampa “additiva”



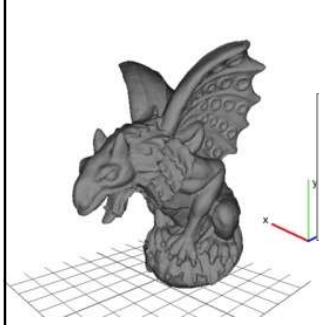
3D printer

oggetti stampati

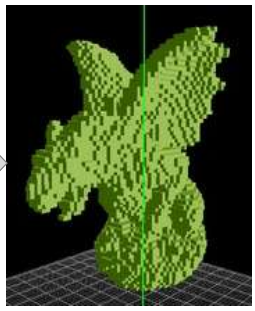
37

Voxelized Volumes:

- ✓Input naturale dei **3D printing devices**



Voxelization
scelta:
resx , resy , resz



Tri Mesh
ben orientata
e chiusa!

Voxelized volume
– 1bit × voxel

to printer

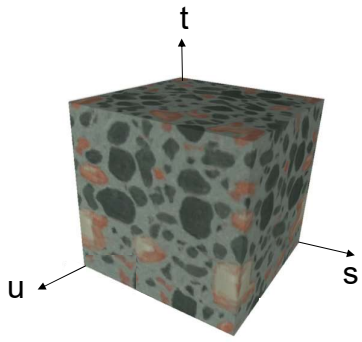
38

Occupazione spaziale dei dataset voxelizzati

- ✓Lo spazio è cubico con la risoluzione (linare)
- ✓E' di solito un prezzo troppo alto
  - ⇒Es: 1024^3 voxel = 1 gigavoxel
  - ⇒Molto oneroso, persino nel caso, come abbiamo visto fin'ora, di 1 solo bit per voxel (1 = pieno / 0 = vuoto)
  - ⇒Quando si memorizza 1 byte, 1 float, 1 double, 1 colore… etc, la situazione peggiora
- ✓Detta la «curse of dimensionality»

40

Se Voxel = 1 colore RGB: “texture volumetriche”



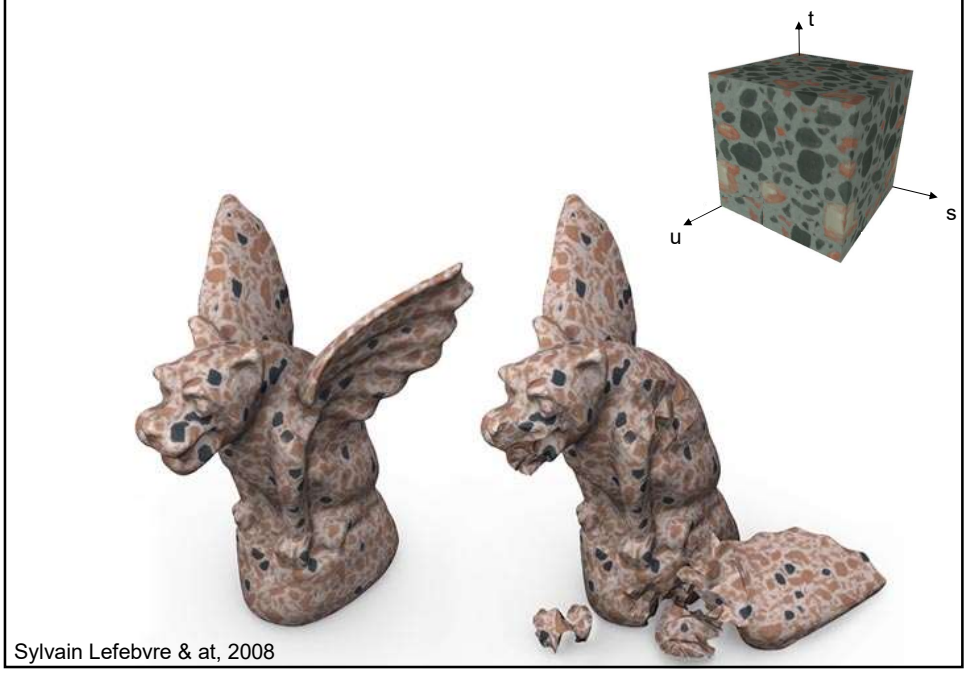
43

Volumetric Textures (o “solid Textures”)

- ✓ 1 texel = 1 voxel
  - ⇒esempio, solid RGB textures: 1 texel = 1 RGB color
- ✓ E’ supportata dall’Hardware, come ogni altra tessitura:
  - ⇒occupa la RAM della scheda video
  - ⇒accesso HW accelerato durante il rendering
  - ⇒interpolazione **tri**-lineare durante l’accesso …
- ✓ Modella il segnale (es. il colore) *dentro* al volume
  - ⇒per es: come gli oggetti sono colorati all’interno
  - ⇒utile per modelli che si possono rompere
  - ⇒utile per pattern come legno, marmo…
- ✓ Non richiede alcuna parametrizzazione della superficie!
  - ⇒La tessitura viene indicizzata dalle posizioni dei vertici
- ⚠ Solito problema, occupazione di memoria
  - ⇒es: quanto per 1 tessitura 1024^3 8-bits-per-channel RGBA?
  - ⇒es: quanto per 1 tessitura 265^3 8-bits-per-channel RGBA?

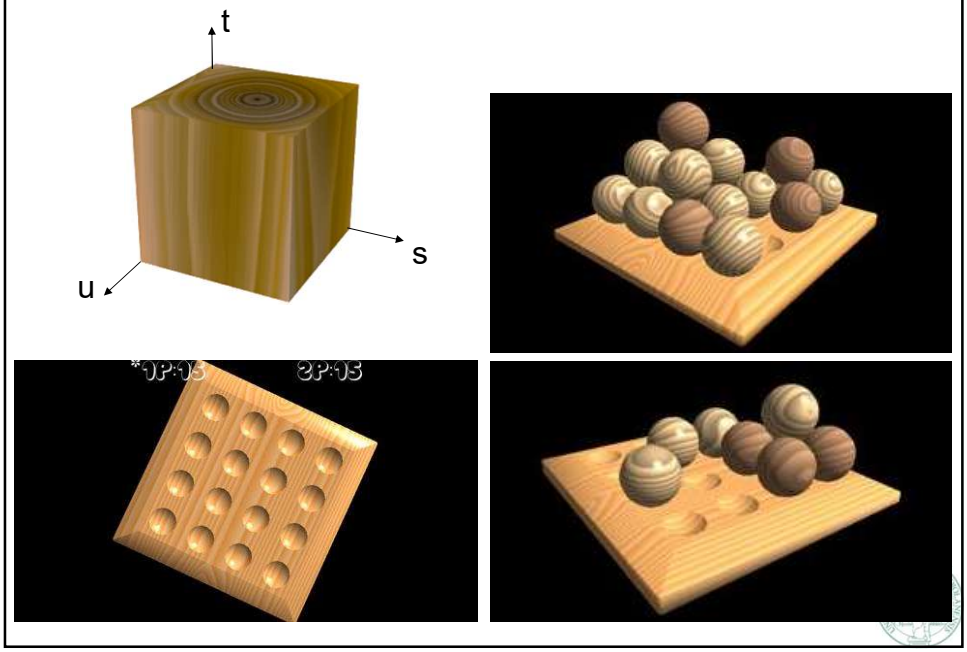
44

Un *colore* per voxel: “texture volumetriche”



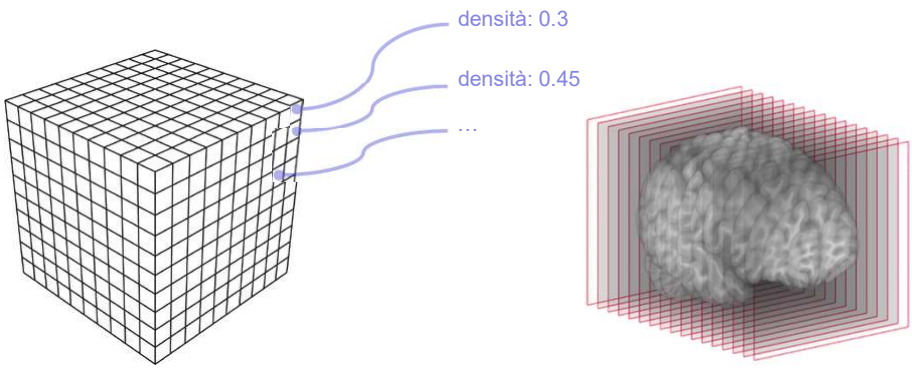
45

Volumetric Textures (o “solid Textures”)



46

Voxel = 1 scalare (es fra 0.0 e 1.0)

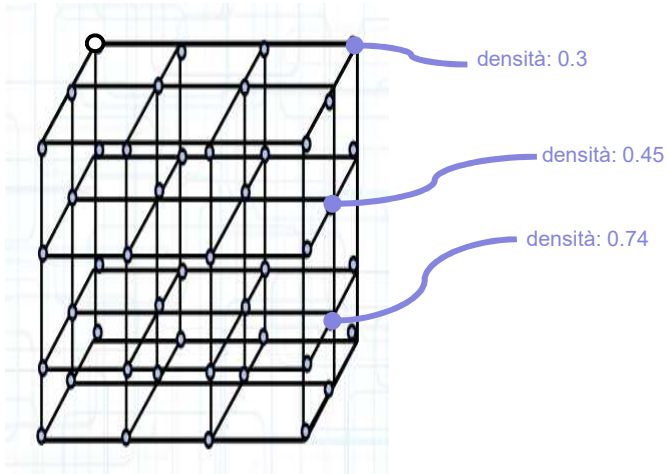


Esempio di File format: **DICOM**
(medicina)

```
Volume float [RES_X] [RES_Y] [RES_Z]
```

47

Voxelized models: uno scalare per voxel

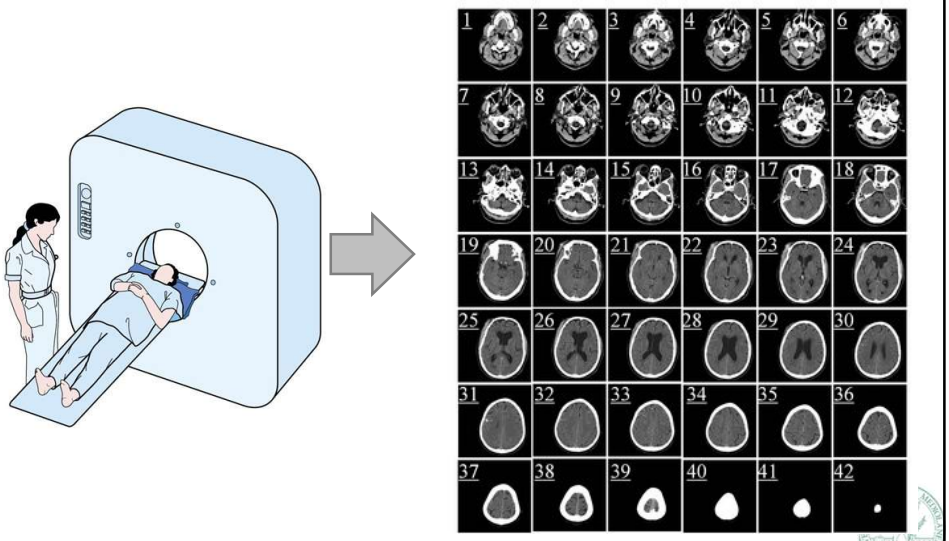



```
float volume[RES_X] [RES_Y] [RES_Z]
```

48

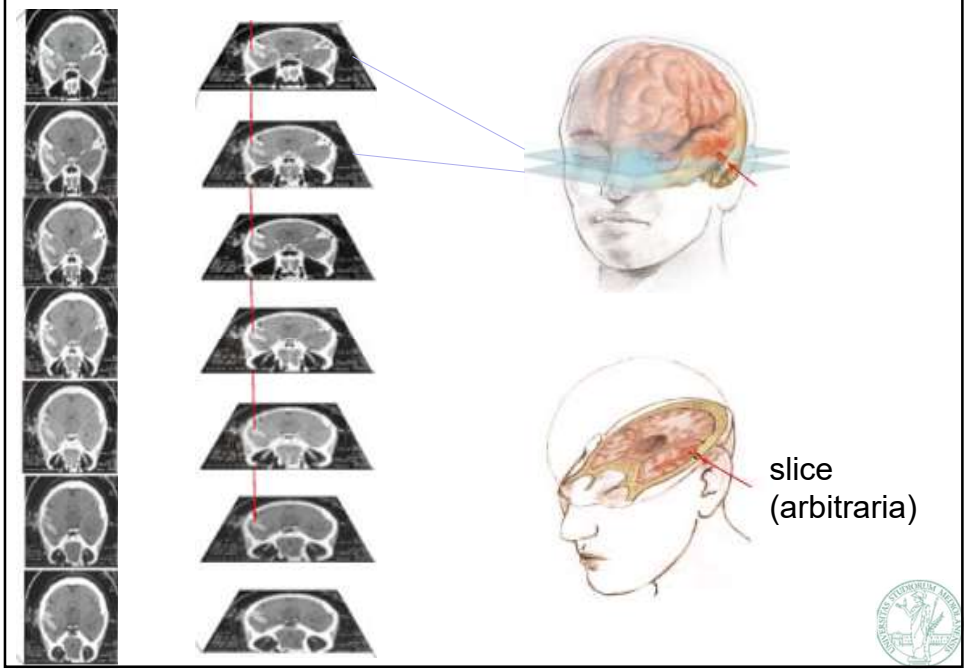
Se 1 voxel = 1 float (valori di densità)

✓Output naturale di **CT scans**



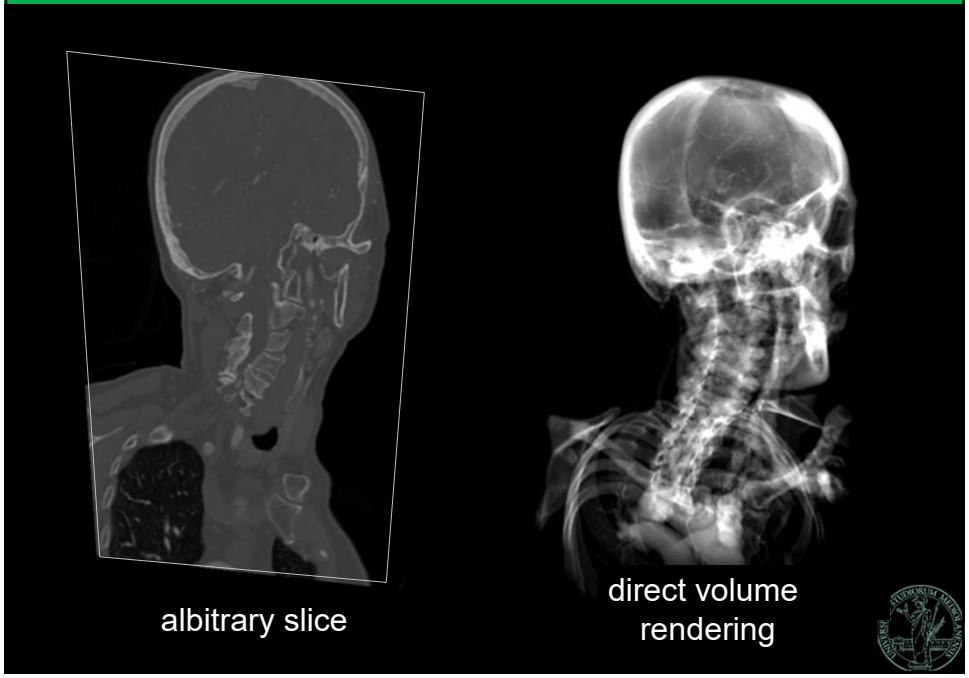
49

Se 1 voxel = 1 float (valori di densità)



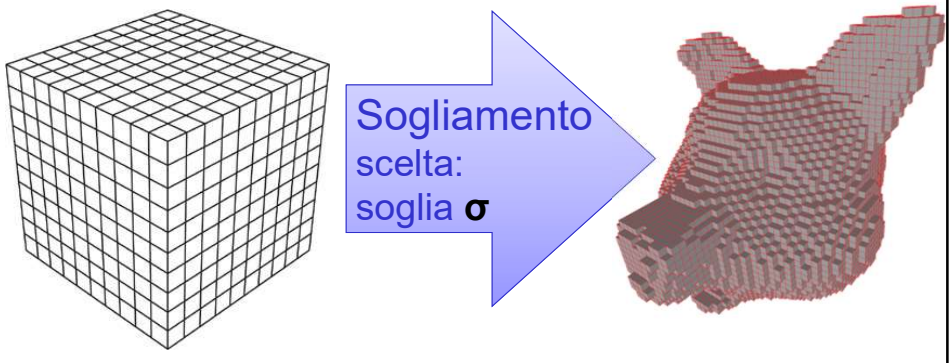
52

Voxelized volumes con voxel = 1 float fra 0 e 1



53

Soglia sulla densità



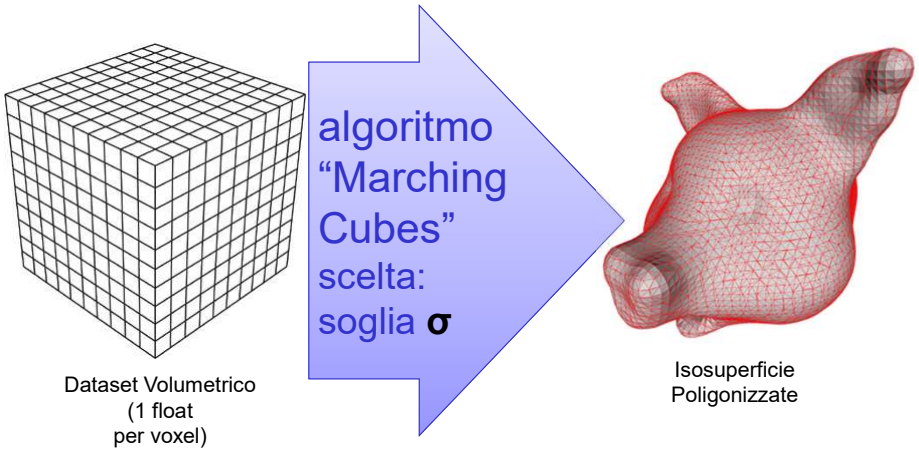
dataset voxelizzato
1 float per voxel
(tipcamente: densità)

dataset voxelizzato
dentro/fuori
per voxel

54

Poligonizzazione di un modello volumetrico

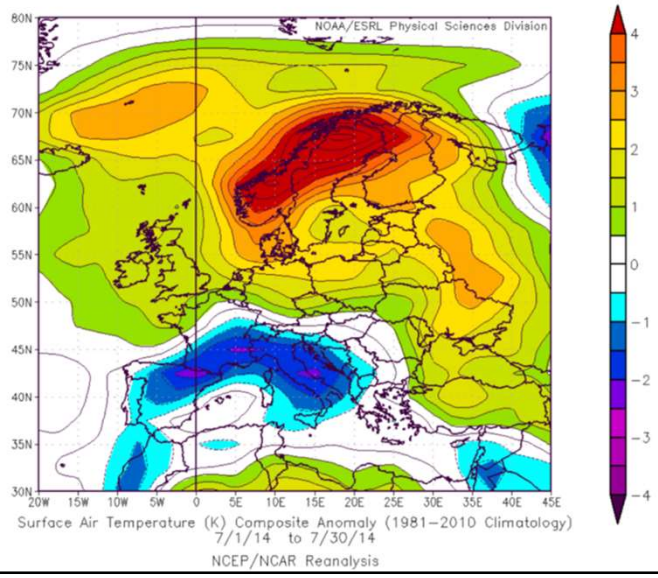
o segmentazione



55

Esempio di isolinea

✓Isolinee di pressione



56

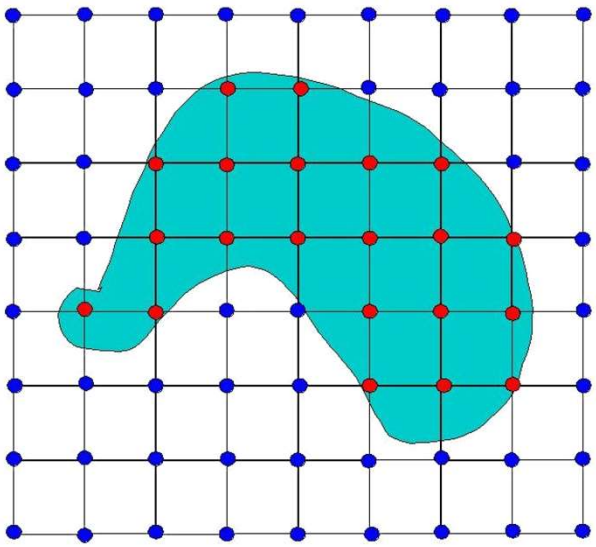
Isolinee e isosuperfici

- ✓ Su un piano (2D):
  - ⇒ ogni punto del piano ha un valore scalare (esempio: pressione, o altezza – height field)
  - ⇒ prendo tutta la regione 2D con valore > **σ**
  - ⇒ il bordo di questa regione è una linea … i cui punti hanno valore tutti **σ** e che racchiude tutti i valori di valore > **σ**
  - ⇒ è la « isolinea di valore **σ** » (linea di isovalori)
- ✓ Su un volume (3D):
  - ⇒ ogni punto dello spazio ha un valore scalare (esempio: densità, pressione, temperatura…)
  - ⇒ prendo la regione 3D con valore > **σ**
  - ⇒ il bordo di questa regione è una superficie… i cui punti hanno tutti valore è **σ** che racchiude tutti i valori di valore > **σ**
  - ⇒ è la « iso-superficie di valore **σ** »

57

Algoritmo marching squares (per 2D)

- ✓ Sogliare voxels



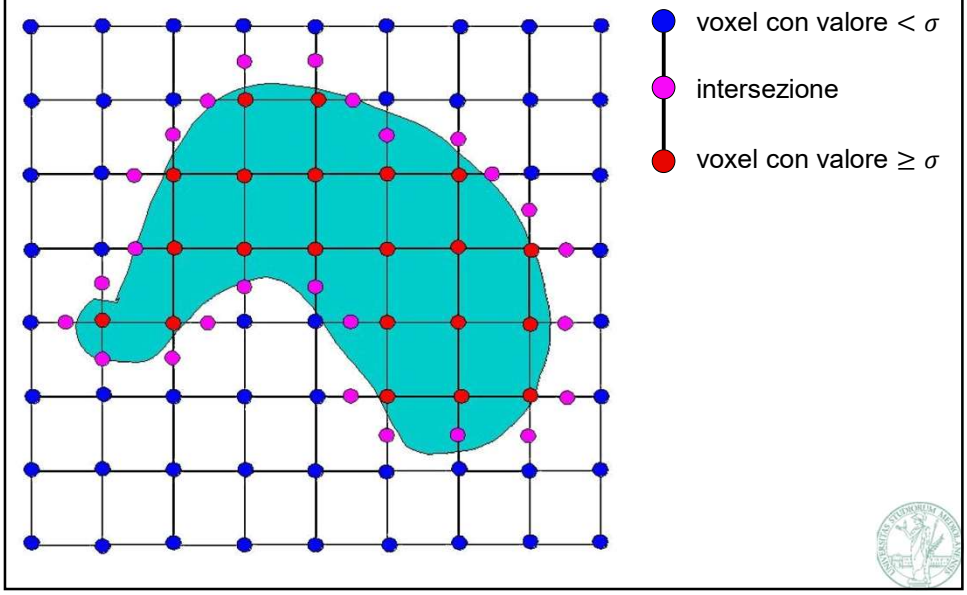
voxel con valore $< \sigma$

voxel con valore $\geq \sigma$

59

Algoritmo marching squares (per 2D)

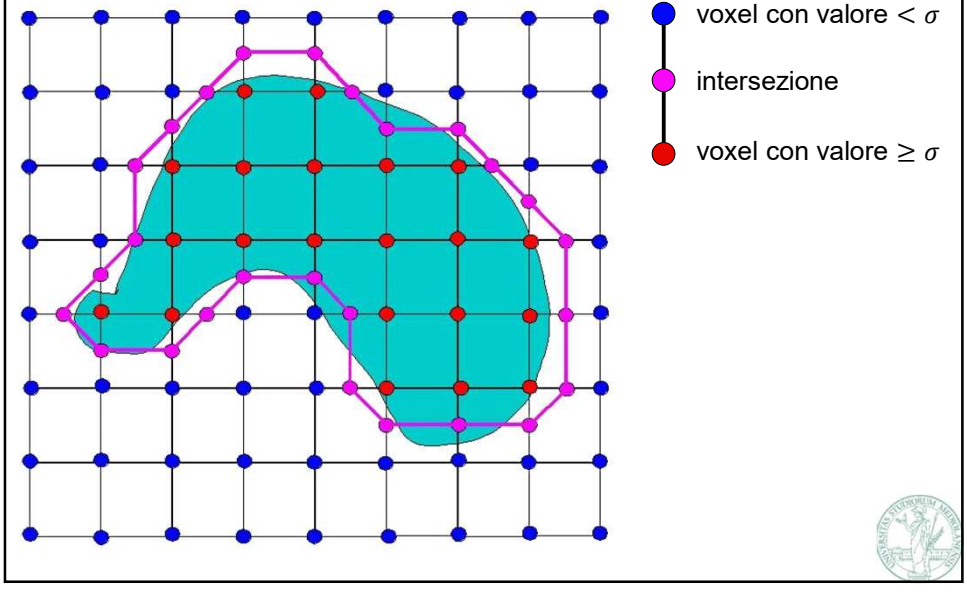
✓Trovare intersezioni



60

Algoritmo marching squares (per 2D)

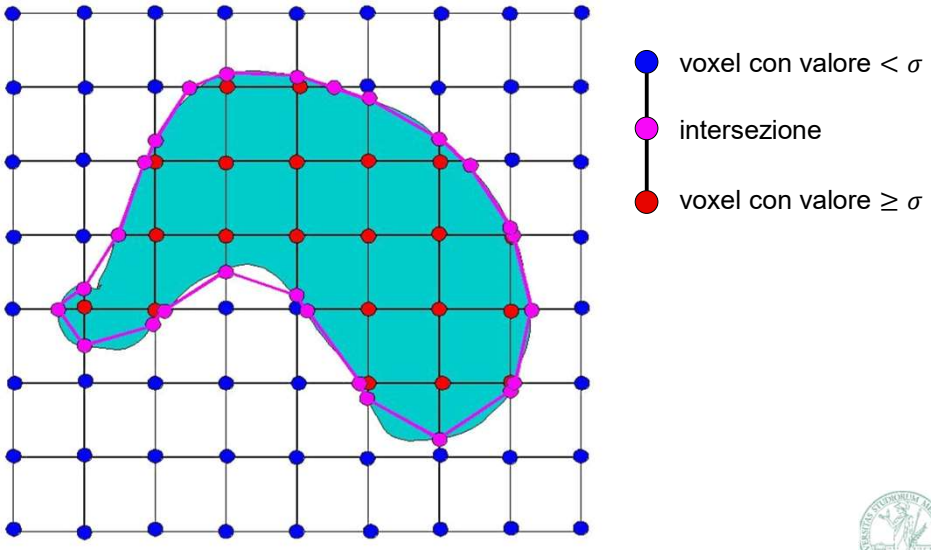
✓Unire intersezionicreando segmenti



61

Algoritmo marching squares (per 2D)

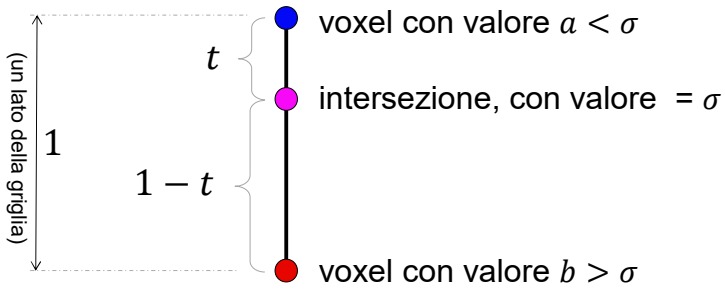
✓Variante: trovare intersezioni e unirle



62

Algoritmo marching squares (per 2D)

✓Come trovare le intersezioni



✓Ipotesi: segnale interpolato linearmente:

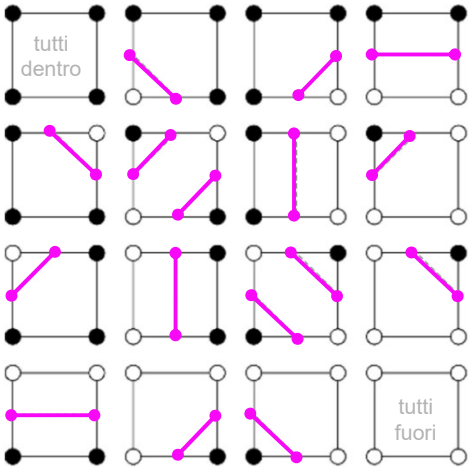
$$a(1-t)+b\,t = \sigma \quad \Leftrightarrow \quad t = \frac{\sigma - a}{b - a}$$

63

Algoritmo marching squares (per 2D)

Come trovare
i segmenti che connettono
le intersezioni?

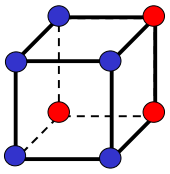
- ✓ Consideriamo un quadrato fra 4 voxel
- ✓ Ogni suo vertice è «dentro» $< \sigma$ o «fuori» $\geq \sigma$
- ✓ Per ogni combinazione, (e sono solo $2^4 = 16$) decido, una volta per tutte, i segmenti da costruire per unire le intersezioni
- ✓ ottengo questa tabella:



64

Generalizzando a 3D: algoritmo marching cubes

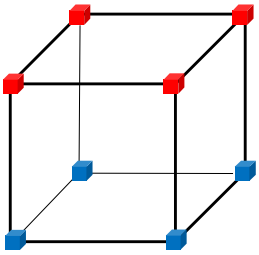
- ✓ Ogni voxel della griglia è dentro o fuori
  - ⇒valore > o < della soglia prescelta
- ✓ Ogni edge della griglia (orientato lungo la X, Y o Z), che connetta un vertice dentro ad uno fuori, ha una intersezione con l'isosuperficie cercata
  - ⇒la si trova trovata come nel caso 2D
  - ⇒per ciascuna intersezione, creo un vertice della mesh
  - ⇒ho ottenuto la **geometria** della mesh!
- ✓ Come ottengo la **connettività**?
  - ⇒Scompongo la griglia in cubetti 1x1x1
  - ⇒Ogni cubo ha 8 vertici, ciascuno dei quali è dentro o fuori ➔ $2^8 = 256$ casi
  - ⇒Uso una tabella per analizza, in ciascun caso, quali poligoni creare per connettere i vertici sui lati della griglia



65

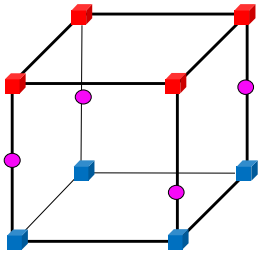
Generalizzando a 3D: algoritmo marching cubes

✓Esempio di uno dei 256 casi

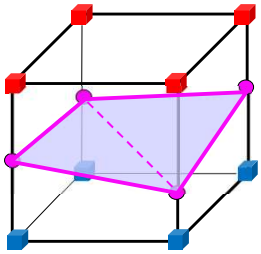


4 Voxel sotto soglia

4 Voxel sopra soglia



4 Intersezioni (calcolate sugli edge)

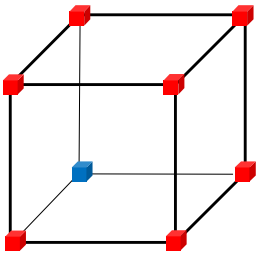


Connesse da due triangoli

66

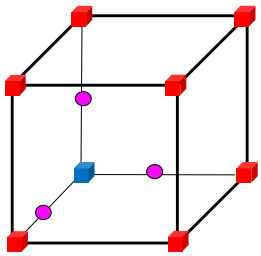
Generalizzando a 3D: algoritmo marching cubes

✓Esempio di uno dei 256 casi

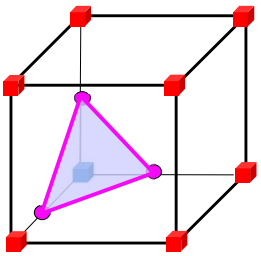


7 Voxel sotto soglia

1 Voxel sopra soglia



3 Intersezioni (calcolate sugli edge)

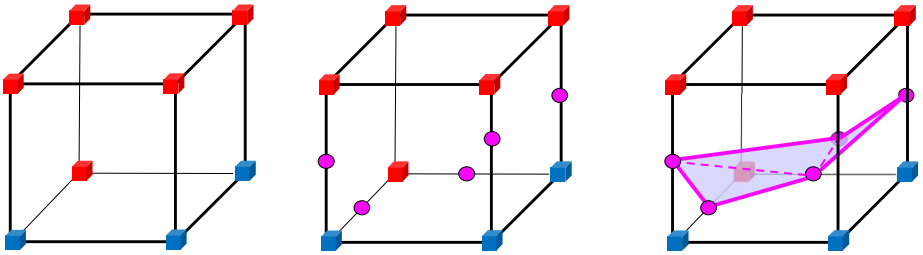


Connesse da un triangolo

67

Generalizzando a 3D: algoritmo marching cubes

✓Esempio di uno dei 256 casi



5 Voxel sotto soglia

3 Voxel sopra soglia

5 Intersezioni (calcolate sugli edge)

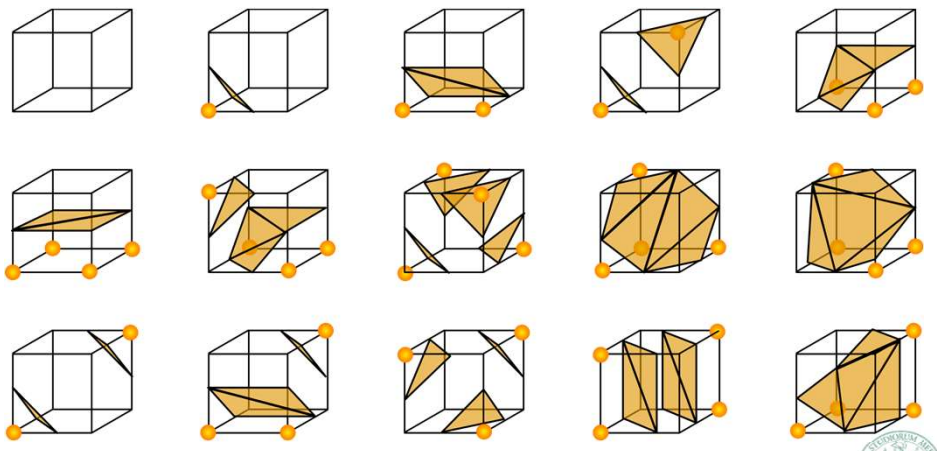
Connesse da tre triangoli

68

Marching cube table

✓Altri esempi

⇒Tutti gli altri sono analoghi a uno di questi 15, per simmetria



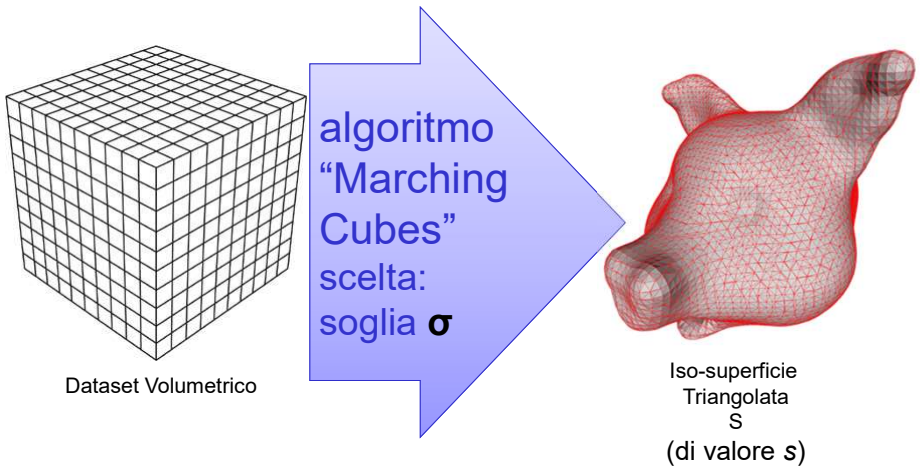
69

Marching cubes

- ✓ Problema: efficienza.
  - ⇒ Curse of dimensionality: numero cubico di cubi da analizzare
- ✓ Soluzione possibile: evitare di processare il gran numero di cubi vuoti
  - ⇒ Molti dei cubi sono «tutti dentro» o «tutti fuori»
  - ⇒ Cioè non contengono né intersezioni sugli spigoli, né facce all'interno
- ✓ Idea: far "marciare" i cubi:
  - ⇒ Trovo un cubo non vuoto → lo processo
  - ⇒ Passo a processare i cubi non vuoti vicini
  - ⇒ Continuo fino ad aver completato la mesh connessa e chiusa

70

Da modello voxelizzato a Mesh poligonale

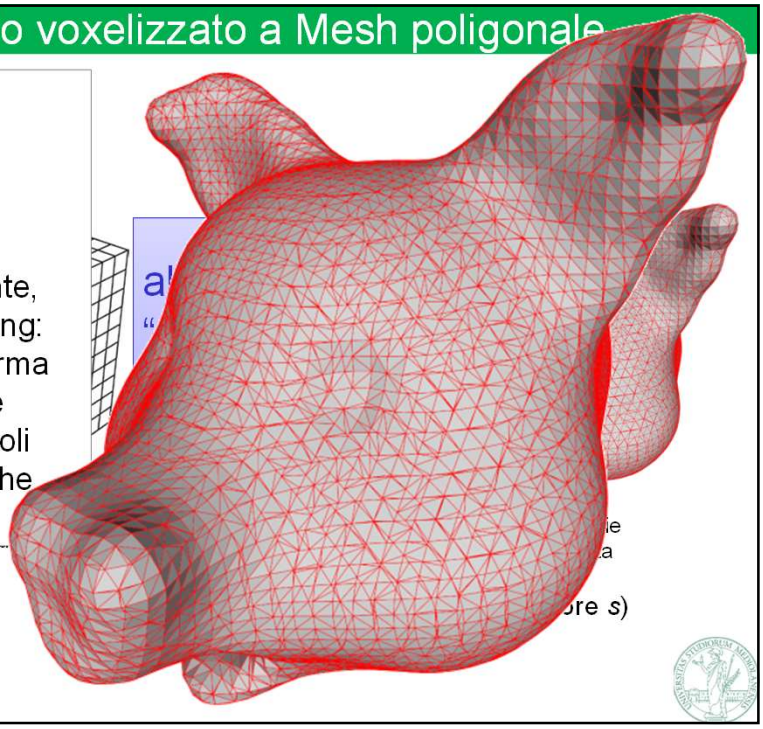


71

Da modello voxelizzato a Mesh poligonale

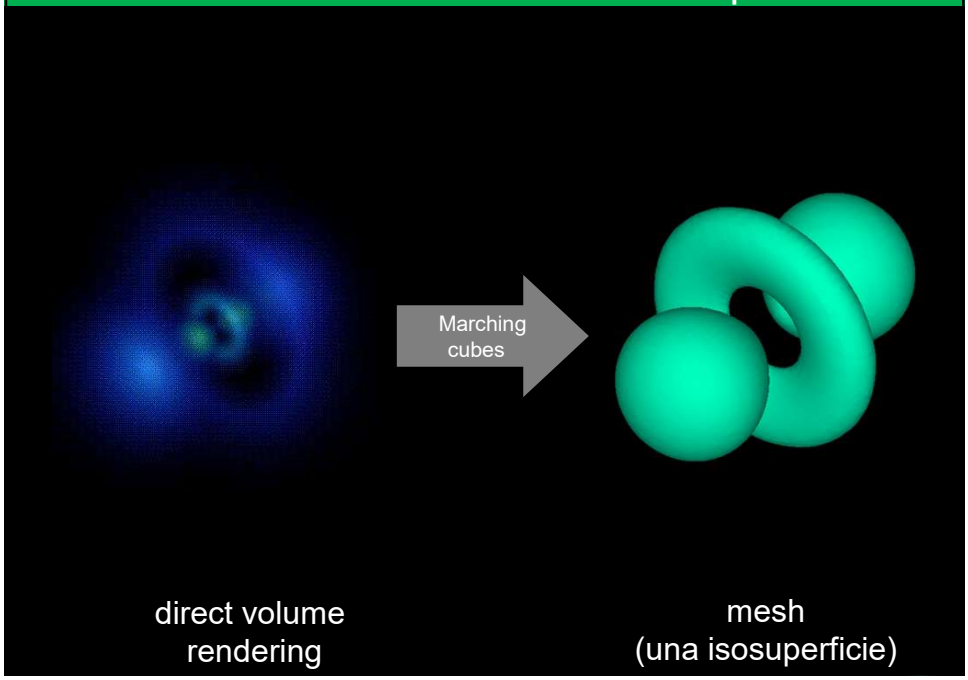
Output:
mesh chiusa,
two-manifold,
ben orientata

Ma tipicamente,
cattivo meshing:
triangoli di forma
molto lunga e
stretta, triangoli
piccoli, o anche
degeneri



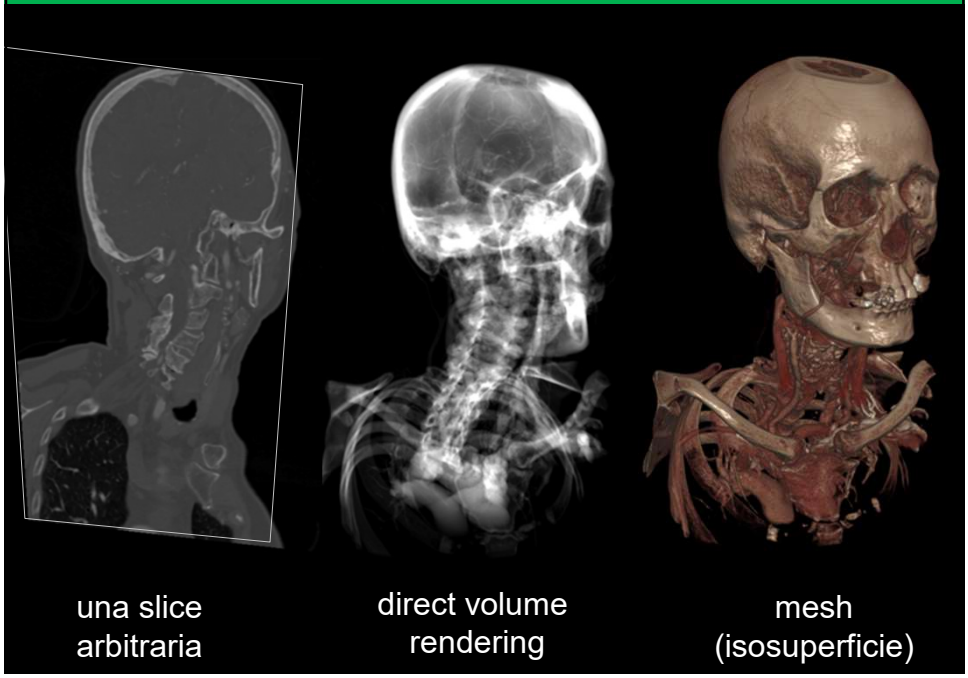
72

Da dataset volumetrico a mesh: esempio



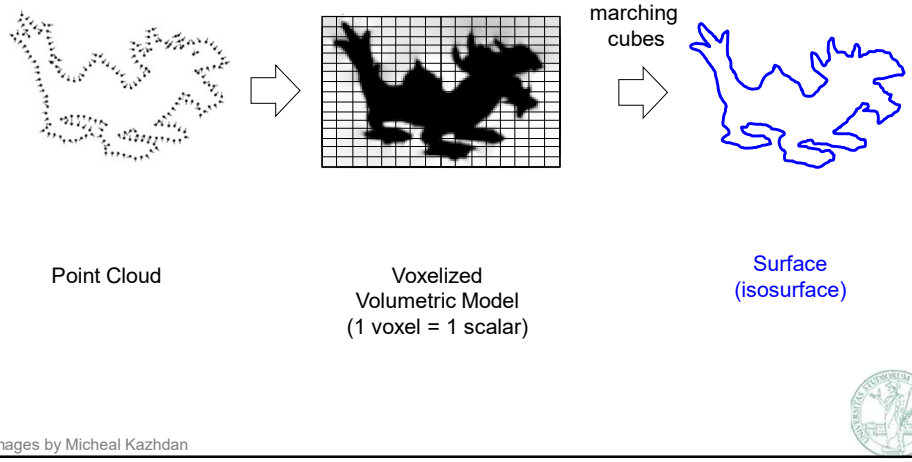
73

Voxelized volumes con voxel = 1 float fra 0 e 1



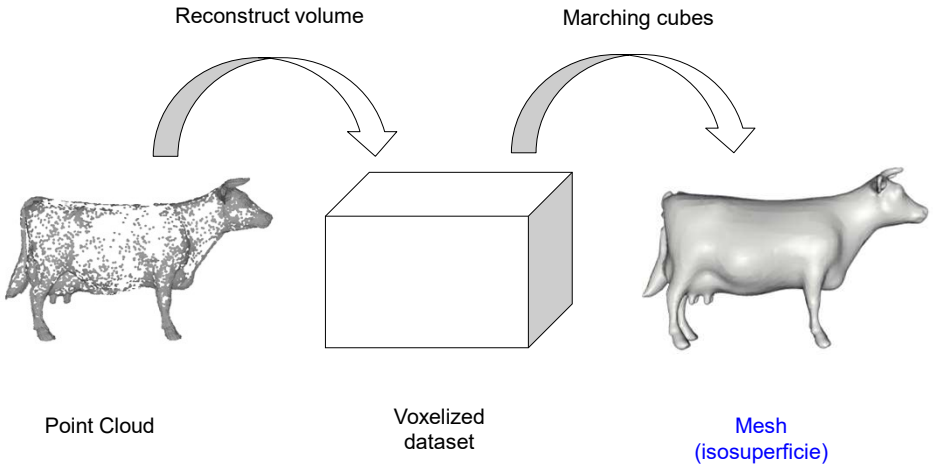
75

Strutture volumetriche come dataset intermedio



76

Strutture volumetriche come dataset intermedio



Images by H. Hoppe

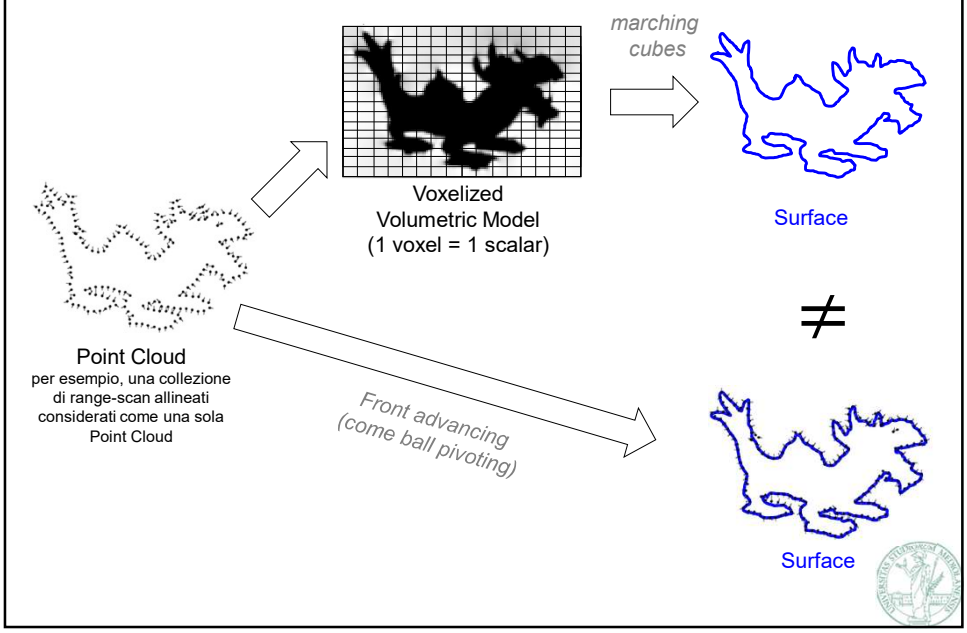
77

Strutture volumetriche come dataset intermedio

- ✓ Un modo comune di convertire una point cloud (e altro) in una mesh è passare attraverso a un struttura volumetrica a voxel
- ✓ Idea:
  - ⇒ 1. Stendere una griglia volumetrica (o lattice) nel volume coinvolto
  - ⇒ 2. Memorizzare una «signed distance function» nel volume (1 valore scalare per voxel, che memorizza una stima della distanza dalla superficie, negativo se il punto è dentro la superficie)
  - ⇒ 3. estrarre isosuperficie (marching cubes)
- ✓ Nel passo 2: ogni punto della nuvola di posizione *p* e normale *n* «vuole» che…
  - ⇒ che il valore della funzione in *p* sia 0
  - ⇒ che il gradiente della funzione in quella posizione sia allineato a *n*

78

Quale metodo è migliore?



79

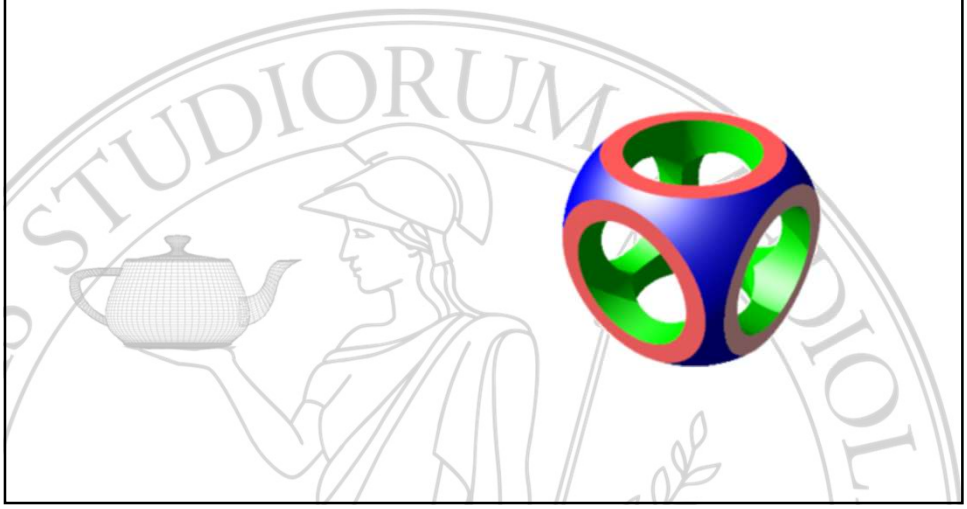
Una (imperfetta) categorizzazione dei tipi di modelli digitali 3D

		ELEMENTI DISCRETI			CONTINUI
		regolari «a griglia»	semi-regolari o irregolari: elementi simpliciali	semi-regolari o irregolari: elementi non simpliciali	
SUPERFICIALI	2-manifold «rappresenta una vera superficie»	**Height Field** **Range Scan** **Geometry Images**	**Triangle Mesh**	**Polygonal Mesh** **Quad Mesh** **Quad dominant Mesh**	**Subdivision surfaces** **Parametric Surfaces (es. B-splines)**
	non-manifold «non rappresenta una sup»	**Set di Range Scan**	**Point Cloud**		
VOLUMETRICI	(3-manifold)	**Voxelized Volume** **Volumetric Textures**	**Tetra Mesh**	**Hexa Mesh**	**Implicit models (es. CSG)**

97

Marco Tarini - Computer Graphics 2021/2022
Università degli Studi di Milano

Modelli impliciti



98

Modelli 3D Volumetrici

1. Discreti & regolari: dataset voxellizati
   ⇒analogo di un immagine rasterizzata, ma in 3D
   ⇒una griglia di voxel

2. Discreti & irregolari: mesh poliedrali
   ⇒analogo di una mesh poligonale (ma nel volume)
   ⇒insieme di poliedri adiacenti faccia a faccia

3. Continui: modelli impliciti
   ⇒rappresentazione basata su funzioni volumetriche
   ⇒superficie come luogo di zeri di una funzione

99

Modello implicito

- ✓ Un oggetto definito (implicitamente) da una funzione contina f
  che va da punti dello punti 3D a valori scalari
  (detta funz. generatrice o funz. primitiva)

$$f: \mathbb{R}^3 \rightarrow \mathbb{R}$$

- ✓ Il valore di f definisce, per un punto dato $\mathbf{p}$, se è dentro oppure fuori dall'oggetto:
  - ⇒ $f(\mathbf{p}) < 0$ <==> $\mathbf{p}$ dentro
  - ⇒ $f(\mathbf{p}) > 0$ <==> $\mathbf{p}$ fuori
  - ⇒ $f(\mathbf{p}) = 0$ <==> $\mathbf{p}$ sulla superficie

100

Modelli impliciti: semantica del valore scalare

- ✓ Un'interpretazione comune:
  «*valore "fuzzy" di appartenenza all'oggetto*»
  - ⇒ 1 : dentro (appartiene all'oggetto)
  - ⇒ 0 : fuori (non appartiene all'oggetto)
  - ⇒ ½ : posizione della superficie
    (là dove si passa da «più fuori che dentro» a «più dentro che fuori)
- ✓ Un'altra interpretazione comune:
  «*distanza con segno (approssimata) dalla superficie*»
  - ⇒ Valori negativi : dentro (distanza negativa dalla superficie)
  - ⇒ Valori positivi : fuori (distanza positiva dalla superficie)
  - ⇒ Zero : posizione della superficie (per definizione)
  - ⇒ Nota: eccetto dove 0, non deve essere esattamente la distanza
- ✓ Note:
  - ⇒ È facile convertire questi valori uno nell'altro (come?)
  - ⇒ Il gradiente della funzione è invertito nei due casi
    (primo caso: gradiente verso il dentro. Secondo caso: verso il fuori)
  - ⇒ In questi lucidi, assumeremo il secondo caso

101

Superficie implicita

- ✓ E' la **superficie** che delimita il modello implicito (volumetrico)
- ✓ E' dato il **luogo degli zeri** di una funzione :

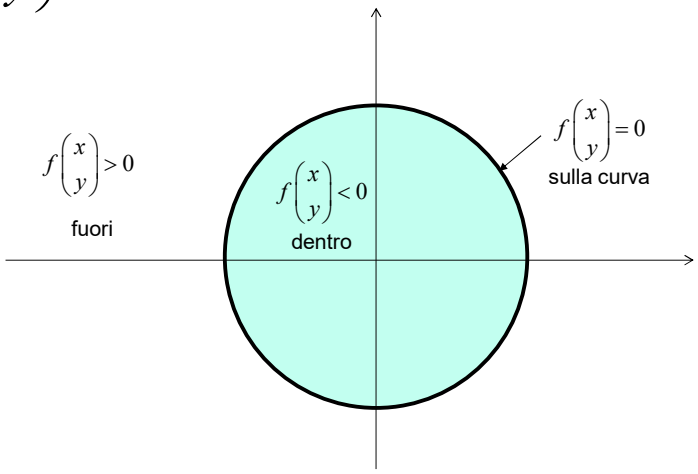
$$f: \mathbb{R}^3 \rightarrow \mathbb{R}$$

(cioè l'insieme di tutti i punti 3D $\mathbf{p}$ t.c. $f(\mathbf{p}) = 0$)

102

Esempio ridotto in 2D: un cerchio

$$f\begin{pmatrix} x \\ y \end{pmatrix} = x^2 + y^2 - r^2$$



103

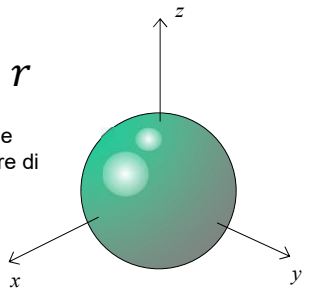
Esempio di modello implicito: una sfera

$$f\begin{pmatrix}x\\y\\z\end{pmatrix} = \sqrt{x^2+y^2+z^2} - r$$

cioè

$$f(\mathbf{p}) = \|\mathbf{p}-\mathbf{0}\| - r$$

Origine (vettore di 0,0,0)



Sfera centrata nell'origine

104

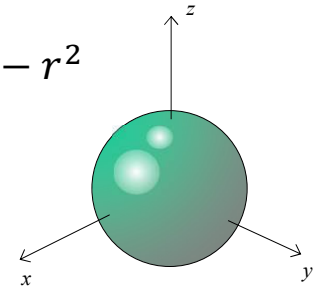
Esempio di modello implicito: una sfera (variante)

$$f\begin{pmatrix}x\\y\\z\end{pmatrix} = x^2+y^2+z^2-r^2$$

Funzione diversa, stessa superficie

cioè

$$f(\mathbf{p}) = \|\mathbf{p}-\mathbf{0}\|^2 - r^2$$



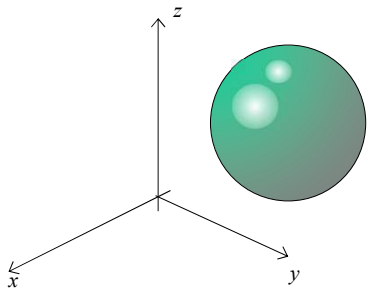
Sfera centrata nell'origine

105

Esempio di modello implicito: una sfera

$$f(\mathbf{p}) = \|\mathbf{p} - \mathbf{c}\|^2 - r^2$$

«La superfice della sfera è il luogo dei punti $\mathbf{p}$ che distano r dal suo centro $\mathbf{c}$»



Sfera centrata nel punto $\mathbf{c}$

106

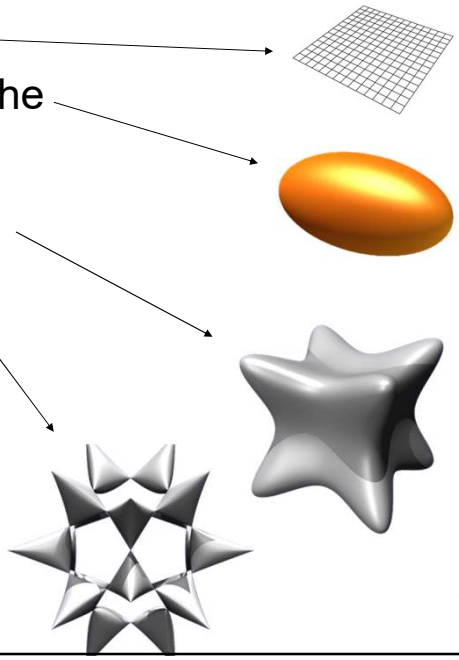
Categorie di modelli impliciti

- ✓ Superfici algebriche: $f()$ è un polinomio
- ✓ Superfici quadriche : $f()$ è di grado 2
  - ⇒classe importante!
    - equazioni semplici, ma buon potere esressivo
    - per esempio, sfere perfette (l'esempio sopra)
- ✓ Superfici cubiche : $f()$ è di grado 3
- ✓ Superfici quartiche : $f()$ è di grado 4
- ✓ etc

107

Superfici **algebriche**

- ✓ Grado 1: Lineari
- ✓ Grado 2: Quadratiche
- ✓ Grado 3: Cubiche
- ✓ Grado 4: Quartiche
- ✓ Grado 5: Quintiche
- ✓ Grado 6: Sestiche
- ✓ ...



108

Normali di una superficie implicita

- ✓ Sia $\mathbf{p}$ un punto sulla superficie, cioè $f(\mathbf{p}) = 0$
- ✓ Quanto vale la normale alla superficie in $\mathbf{p}$?
  - ⇒ cioè: quale direzione $\hat{\boldsymbol{n}}$ è ortogonale alla superficie nel punto $\mathbf{p}$
  - ⇒ cioè: in che direzione $\hat{\boldsymbol{n}}$ devo spostarmi da $\mathbf{p}$ per allontanarmi il più possibile dalla superficie
  - ⇒ cioè: in che direzione $\hat{\boldsymbol{n}}$ devo spostarmi da $\mathbf{p}$ per aumentare il più possibile il valore di $f\ (\mathbf{p} + k\,\hat{\boldsymbol{n}})$
- ✓ Risposta: basta (per def) prendere il gradiente di $f(\mathbf{p})$
  - ⇒ gradiente f vettore delle derivate parziali in calcolato in $\mathbf{p}$ (rinormalizzato)

$$\hat{\boldsymbol{n}} = \frac{\nabla f(\mathbf{p})}{\|\nabla f(\mathbf{p})\|}$$

109

Gradiente di una funzione (ripasso)

- ✓ Sia $f(\mathbf{p})$ è una funzione da punti a scalari
$$f: \mathbb{R}^3 \rightarrow \mathbb{R}$$
- ✓ Quindi da x, y, z ad un valore scalare
- ✓ Il suo gradiente $\nabla f(\mathbf{p})$ nel punto $\mathbf{p}$ è definito dal vettore delle sue derivate parziali in x, y, z (calcoalte in $\mathbf{p}$)
$$\nabla f(\mathbf{p}) = \begin{pmatrix} \partial f/\partial x \\ \partial f/\partial y \\ \partial f/\partial z \end{pmatrix}$$
- ✓ Ad esempio…

$$f\begin{pmatrix} x \\ y \\ z \end{pmatrix} = x^2 + y^2 + z^2 - r^2 \qquad \nabla f\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} 2x \\ 2y \\ 2z \end{pmatrix}$$

←derivata di f in x
←derivata di f in y
←derivata di f in z

funzione primitiva della sfera

gradiente di f, cioè normale della sfera nel punto x, y, z (a meno di normalizzazione)

110

Vantaggi delle superfici implicite

- ✓ E' una rappresentazione volumetrica
  - ⇒ per es, facile determinare se un punto $\mathbf{p}$ sia all'interno o all'esterno di un oggetto – basta valutare f su $\mathbf{p}$!
- ✓ E' molto compatta
  - ⇒ a differenza di quelle discrete!
- ✓ Rappresenta superfici curve
  - ⇒ Paragona con tri-mesh, che rappresenta superfici lineari a tratti (cioè localmente piatti)
- ✓ Buon modello per superfici che variano nel tempo
  - ⇒ es quelle dei fluidi
- ✓ Consentono potenti operazioni di editing
  - ⇒ operazioni volumetriche booleane: vedi sotto

111

Da modello implicito a mesh

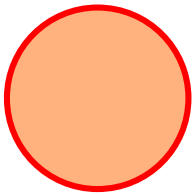
- ✓ Modo comune (ma non l'unico): passare da voxel
- ✓ Passi:
  - ⇒ **campionare** f ad (una data risoluzione resx, resy, resz) ottenendo un **campo scalare voxelizzato**
    (Parametri dell'algoritmo)
  - ⇒ estrarre una mesh poligonale M attraverso **Marching Cubes** (con soglia 0)
  - ⇒ calcolare le **normali** nei vertici di M, con attraverso i **gradienti** di f calcolati nelle posizioni dei vertici

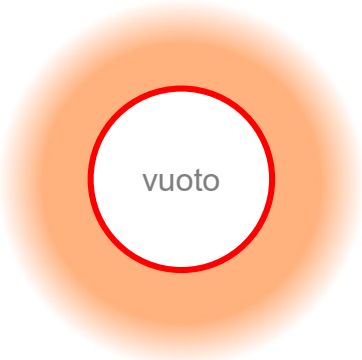
    (preferibile al calcolo delle normali dei vertici attraverso le normali dei triangoli, come abbiamo visto per una mesh generiche)

112

Operazioni booleane volumetriche

- ✓ Opposto: dentro diventa fuori e viceversa





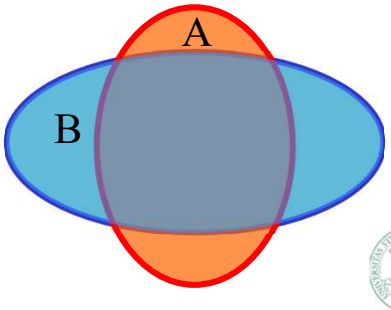
f_A $\quad -f_A$

- ⇒ Posso vedere gli scavi come intersezioni con gli opposti

113

Operazioni booleane volumetriche

- ✓ Siano A e B modelli impliciti con funzioni primitive f_A e f_B
- ✓ Posso definire (come modelli impliciti) la funzione primitive della loro…
  - ⇒inversione: $-f_A$
  - ⇒intersezione: $\max(f_A, f_B)$
  - ⇒unione: $\min(f_A, f_B)$
  - ⇒scavo di B da A: $\min(f_A, -f_B)$
  - ⇒Scavo di A da B: $\min(-f_A, f_B)$



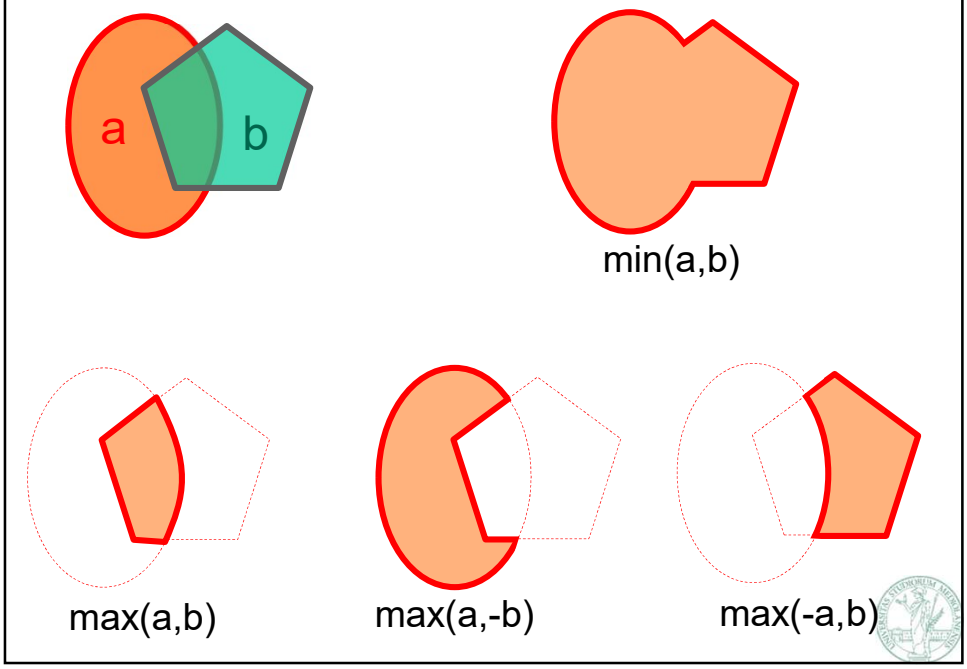
114

Operazioni booleane volumetriche

Operazione volumetrica booleana	Vista come operazione fra insiemi di punti	Vista come operazione booleana	Operaz delle due funzioni primitive
Opposto	$\overline{A}$	$\sim A$	$-f_A$
Intersezione	$A \cap B$	$A \wedge B$	$\max(f_A, f_B)$
Unione	$A \cup B$	$A \vee B$	$\min(f_A, f_B)$
Scavo di B da A	$A \cap \overline{B}$	$A \wedge \sim B$	$\max(f_A, -f_B)$
Scavo di A da B	$\overline{A} \cap B$	$(\sim A) \wedge B$	$\max(-f_A, f_B)$

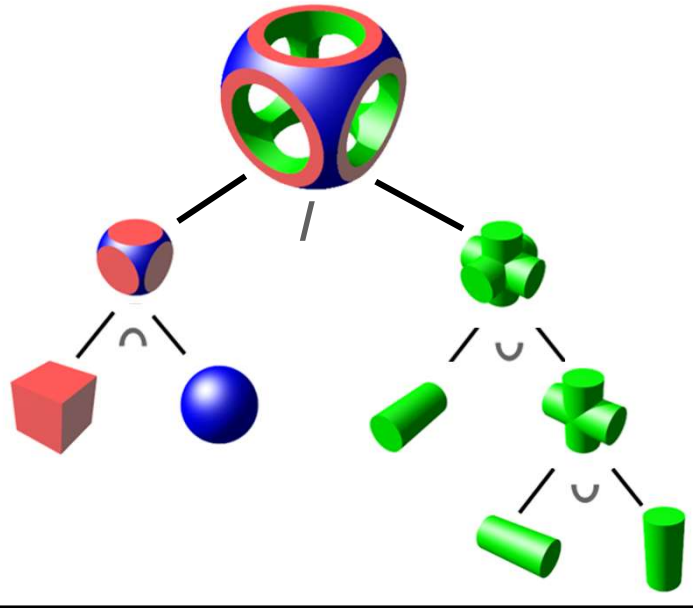
115

Operazioni booleane volumetriche di modelli impliciti



116

Constructive Solid Geometry (CSG)

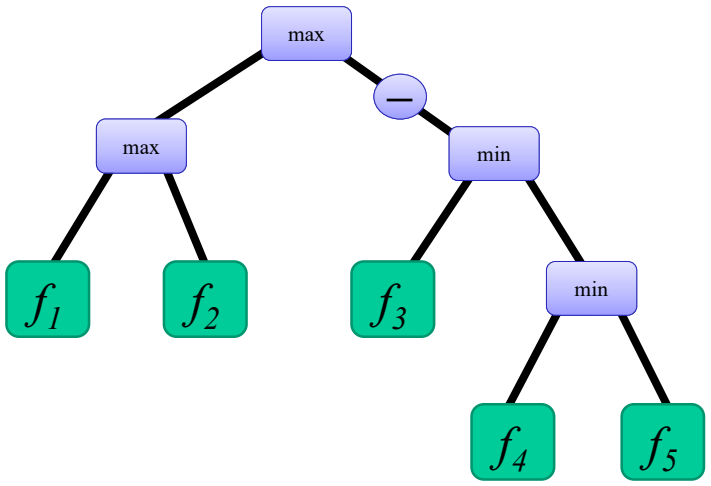


118

Geometric Solid Modelling

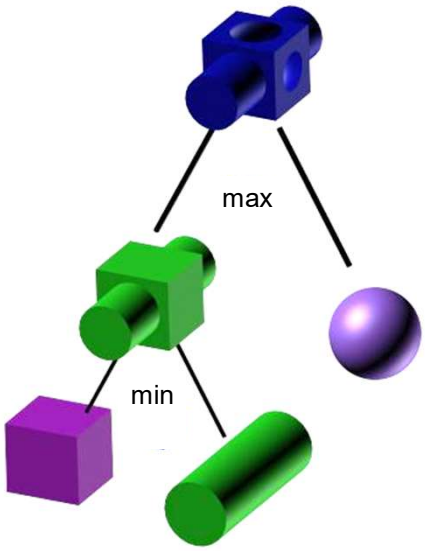
Albero di parsing delle operazioni

$$f(\mathbf{p}) = \max\left(\max(f_1(\mathbf{p}), f_2(\mathbf{p})), -\min\left(f_3(\mathbf{p}), \min(f_4(\mathbf{p}), f_5(\mathbf{p}))\right)\right)$$



119

Constructive Solid Geometry CSG: altro esempio



120

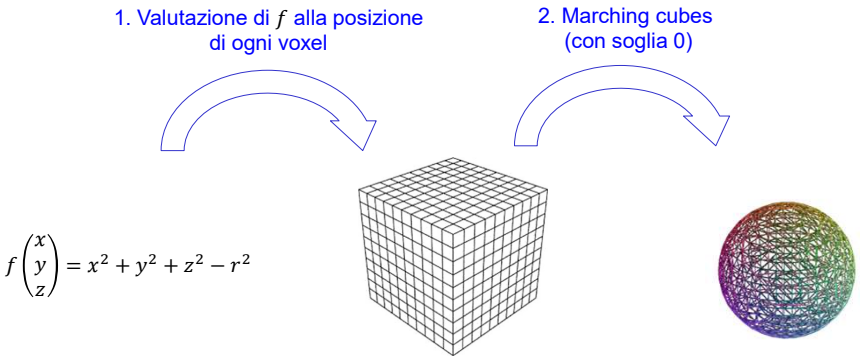
Constructive Solid Geometry

- ✓ Idea: modellare oggetti solidi 3D a partire dalle forme più semplici («primitive») che li compongono
- ✓ Utile per modellare oggetti **meccanici / artificiali**
  - ⇒Contesto CAD
- ✓ Il modello è rappresentato da una struttura ad albero
  - ⇒radice = l'intero oggetto
  - ⇒nodi interni = operazioni booleane dei nodi figli
  - ⇒foglie = forme basilari – associate a funzioni primitive (esempio: sfere, cilindri, semispazi, cubi…)
- ✓ E' l'albero di parsing di un'espressione di operazioni booleane

121

Da modello implicito a mesh («esplicitare» un modello implicito»)

- ✓ Un modo semplice (non necessariamente l'unico) di farlo è…



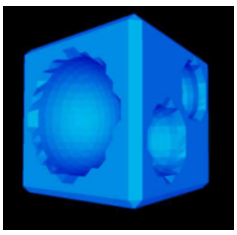
Scelta: a che risoluzione?

122

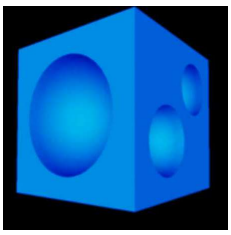
Rendering di un modello implicito

- ✓ un modello implicito può essere renderizzato direttamente!
  - ⇒ attraverso algoritmi di ray-casting (o ray-tracing)
  - ⇒ (vedi seconda parte del corso!)

senza conversione in una mesh



Rendering di un mesh ottenuta (passando da un volume a bassa risoluzone)



Rendering diretto con un algoritmo di ray-tracing

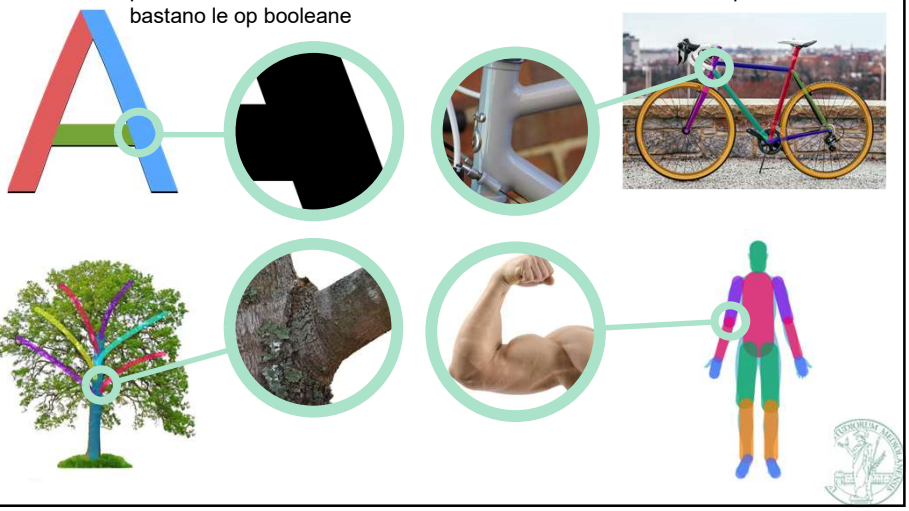
123

Oltre alle sempici operazioni booleane…

- ✓ Per molti oggetti, è utile unire le sottoparti con operazioni diverse da quelle booleane

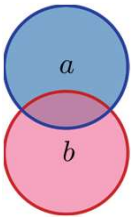
Esempio:
per una lettera di un font, bastano le op booleane

Ma la fusione fra due tubi in una biciletta (welding) non è ben approssimata da una op booleana

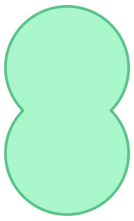


125

Funzioni per combinare modelli impliciti

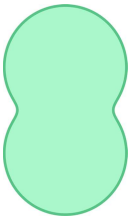


union
[Sabin 1968]



$\min(a, b)$

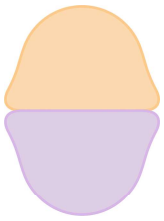
blend
[Blinn 1982]
[Ricci 1973]



$a + b$

$\sqrt{a^n + b^n}^{\frac{1}{n}}$

contact & bulge
[Cani 1993]



$$\begin{cases} a - b + 1 & \text{if } b > 1 \\ a + h(a, b) & \text{else} \end{cases}$$

see «meta-balls»

126