

1

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

		ELEMENTI DISCRETI			CONTINUI
		regolari «a griglia»	semi-regolari o irregolari		
			elementi simpliciali	elementi non simpliciali	
SUPERFICIALI	2-manifold «rappresenta una vera superficie»	Height Field Range Scan	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

## Height fields & Range scans

- ✓ Sono tre strutture dati **superficiali** simili che si basano su un concetto simile:
  - ⇒ Campionamento della superficie
  - ⇒ I campioni sono memorizzati su una **griglia regolare 2D**
  - ⇒ **Connettività implicita**: ogni campione è *implicitamente* connesso ai propri vicini sulla griglia

(Un altro caso di questo tipo, che non vedremo, è costituito dalle Geometry Images)



5

## Height fields & Range scans

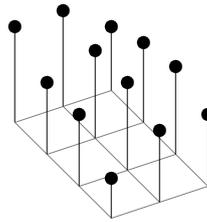
- ✓ Vantaggi comuni di queste strutture:
  - ⇒ La connettività è **implicita** (non deve essere memorizzata)
  - ⇒ Anche le relazioni di adiacenza fra elementi sono **implicite**
  - ⇒ **Multi-risoluzione** semplice da ottenere (come per le immagini 2D rasterizzate – cioè di pixel)
  - ⇒ Geometry Processing può essere **parallelizzato**: molto facile da ottenere (usando la GPU)
  - ⇒ **Machine learning** su forme 3D: reso semplice dalla regolarità (è come Machine Learning su immagini)
  - ⇒ Analogia con **immagini 2D** (molti vantaggi – es compressione, editing, filtering)
- ✓ Alcuni svantaggi:
  - ⇒ La risoluzione **non è adattiva**
  - ⇒ L'**espressività** è limitata (solo alcune forme possono essere rappresentate)



6

## Campo di altezze

- ✓ Campo di altezze (height-field)
  - ⇒ o mappa di altezze (height map)
  - ⇒ o immagine di profondità (depth image)
  - ⇒ o mappa di profondità (depth-map)
  - ⇒ o modello «2.5D» (2.5D model)



```
float[][] height = new float[resX][resY];
```

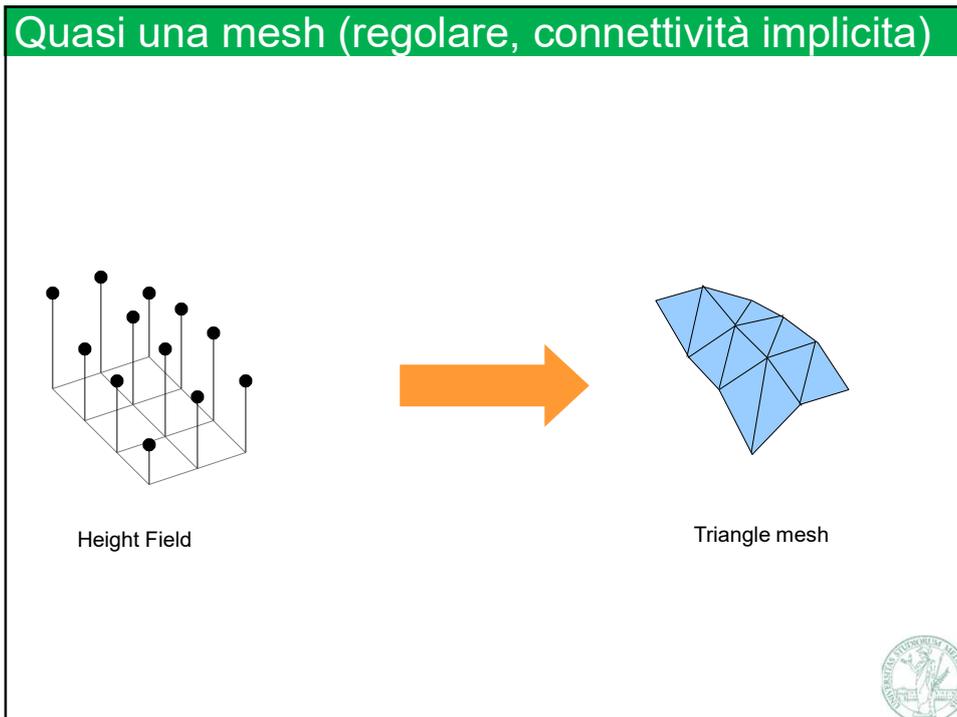
7

## Campo di altezza

```
var height[resX][resY] float;
```

- ✓ Array 2D di valori *scalari*
  - ⇒ Il valore ad indice  $x, y$  rappresenta il punto 3D di coordinate  $(x, y, \text{height}[x][y])$
  - ⇒ Vantaggio: 2 su 3 delle coordinate non devono essere memorizzate! Solo la  $z$
  - ⇒ Molto utilizzato per terreni (per es, in videogames)
  - ⇒ E' una funzione (tabellata) da  $x, y$  a  $z$ 
    - Quindi non sono rappresentabili: sottopassaggi, tunnel, ponti, caverne, burroni spioventi...

8

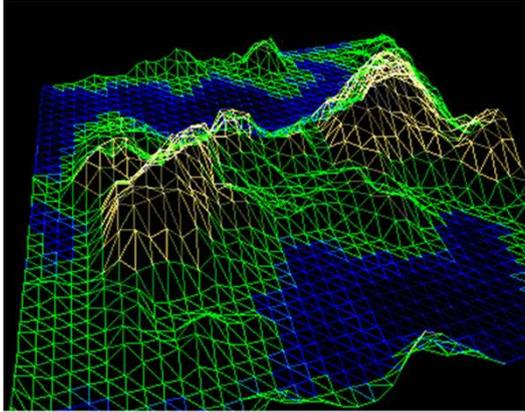


9



10

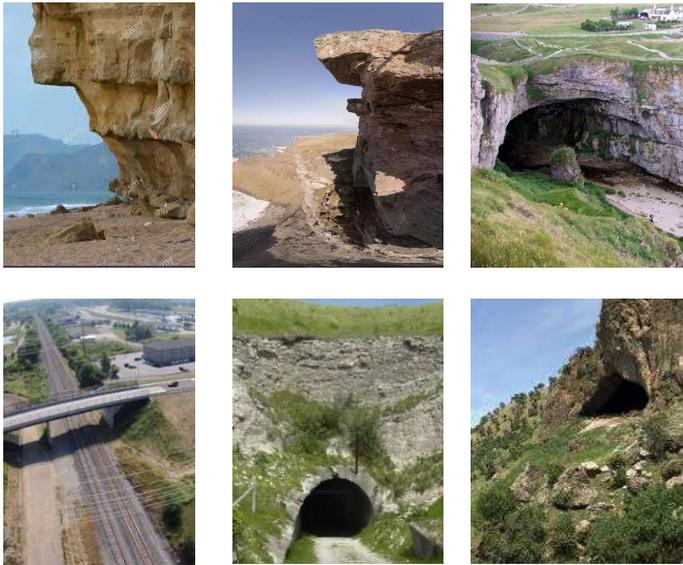
### Tipico uso: modellare un terreno



Vedere il primo lucido per un altro esempio, più high res

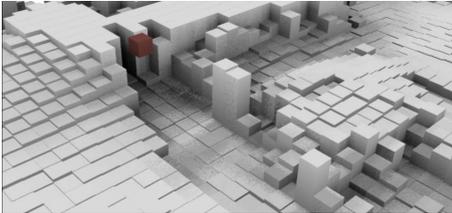
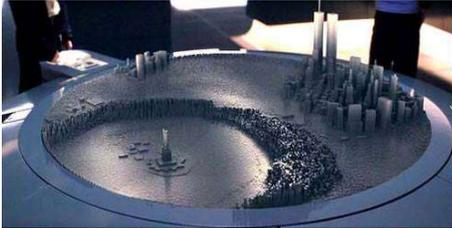
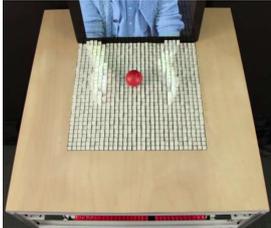
12

### Non tutti i terreni possono essere modellati da campi di altezza...



13

### Curiosità: height maps come dispositivi

Immaginati dalla fantascienza	o reali
 <p>(CGI)</p>	 <p>(real)</p>
 <p>(CGI)</p>	 <p>(real)</p>

18

### Range Scan (or Depth Image)

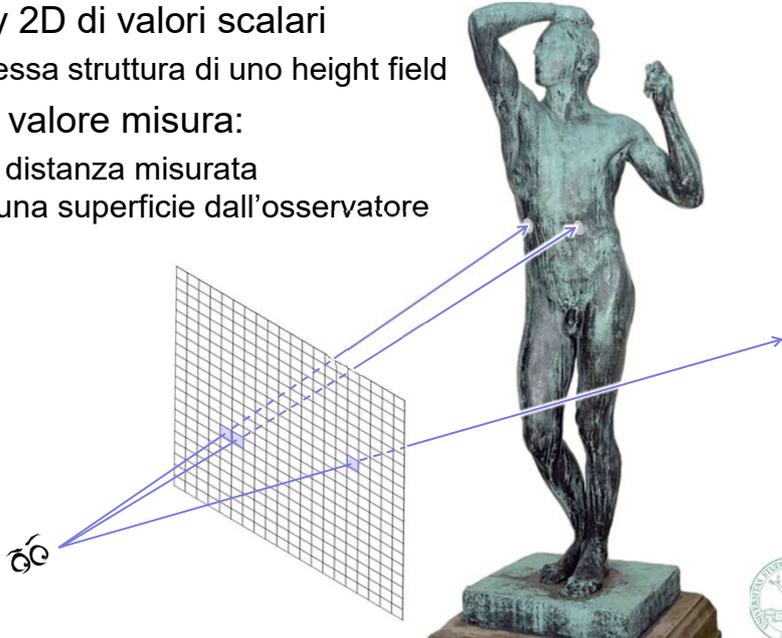
- ✓ Un array 2D di misurazioni di profondità
  - ⇒ Effettuate da un determinato dispositivo
  - ⇒ Sono l'output di molte tecniche di acquisizione automatica
    - sia attive – basate su laser, o luce strutturata
    - che passive – basate su foto
  - ⇒ «Range scanning»: una tecnica di acquisizione 3D che produce range-scans.
  - ⇒ «Depth camera»: macchina fotografica / videocamera che cattura una range-scan (per fotogramma, o per scatto)
    - cioè una profondità per pixel piuttosto che solo un RGB
  - ⇒ «RGBD camera»: una depth camera che produce *anche* una normale immagine a color (canali R, G e B per pixel)

```
var rangeScan [resX] [resY] float;
```

19

### Range Scan (or Depth Image)

- ✓ Array 2D di valori scalari
  - ⇒ Stessa struttura di uno height field
- ✓ Ogni valore misura:
  - ⇒ La distanza misurata di una superficie dall'osservatore



20

### Range Scan (or Depth Image)

- ✓ Array 2D di valori scalari
  - ⇒ Stessa struttura di uno height field
- ✓ Ogni valore misura:
  - ⇒ La distanza misurata di una superficie dall'osservatore

$d_0$	$d_1$	$d_2$	$d_3$	$d_4$
$d_5$	$d_6$	$d_7$	$d_8$	$d_9$
$d_{10}$	$d_{11}$	$d_{12}$	$d_{13}$	$d_{14}$
$d_{15}$	$d_{16}$	$d_{17}$	$d_{18}$	$d_{19}$
$d_{20}$	$d_{21}$	$d_{22}$	$d_{23}$	$d_{24}$
$d_{25}$	$d_{26}$	$d_{27}$	$d_{28}$	$d_{29}$

Una distanza (1 scalare)  
oppure il valore "invalido"

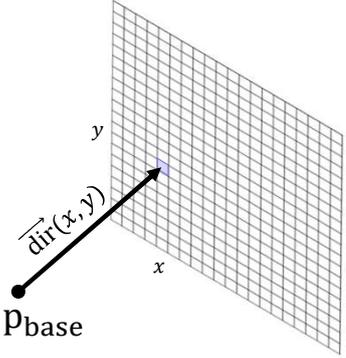


21

### Range scan

```
var rangeScan [resX][resY]float;
```

✓ Il campione a coordinate  $x, y$  rappresenta il punto  
 $p_{base} + rangeScan[x][y] \cdot \vec{dir}(x, y)$



Il punto  $p_{base}$   
 e la  
 funzione  $\vec{dir}(x, y)$   
 devono  
 essere noti, per poter  
 interpretare la  
 rangeScan!

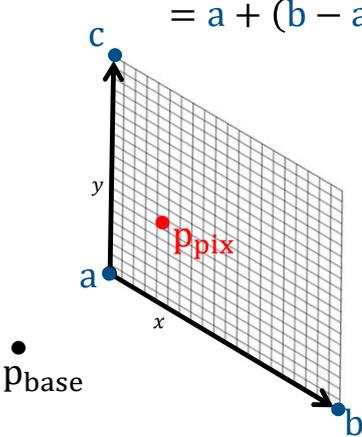


22

### Range scan

```
var rangeScan [resX][resY]float;
```

$$\vec{dir}(x, y) = p_{pix} - p_{base}$$

$$= a + (b - a) \frac{x}{resX} + (c - a) \frac{y}{resY} - p_{base}$$



23

## Range Scan

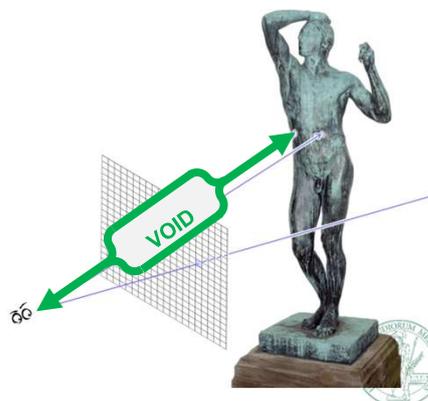
- ✓ Campioni possono essere marcati come vuoti
  - ⇒ Usando un valore scalare arbitrario (per es, 0, o 0xBAD)
- ✓ Come tutti i dataset catturati, le Range Scan tipicamente presentano molti difetti di misurazione
  - ⇒ Rumore, missing data, outliers, etc (come le point cloud)
  - ⇒ La rimozione di questi difetti può avvalersi della regolarità della griglia (definizione veloce e robusta dei vicini)
  - ⇒ Es 1: per stimare una normale per campione
  - ⇒ Es 2: rimozione dei campioni prossimi ad «depth jump», salti sulla Z (sono inaffidabili e rumorosi)
- ✓ Una range scan cattura un «solo lato» di un modello
  - ⇒ E' «2.5D»: molte forme non possono essere modellate
  - ⇒ Ne sono necessarie molte (un «**range scan set**») per modellare un oggetto a tutto tondo.



24

## Range Scan, o Depth Image

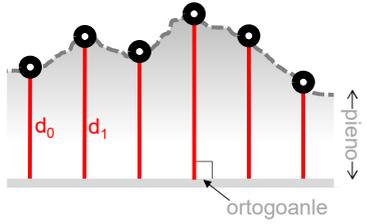
- ✓ In una **range-scan**, ogni «pixel» rappresenta un punto xyz
- ✓ Quindi la si può considerare una **point-cloud**
  - ⇒ A volte, i due termini sono usati intercambiabilmente
- ✓ Alcune differenze:
  - La range scan ha una struttura a griglia (che consente di definire un relazione di adiacenza fra campioni)
  - La range-scan cattura un'informazione sull'occlusione:  
1 campione = «so che lo spazio fra l'osservatore e questo punto è vuoto»



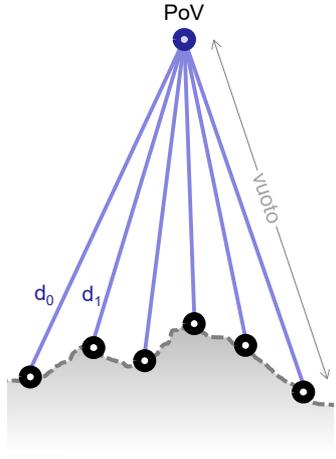
26

### Range Scan VS Height field

Stessa struttura,  
 ma la semantica  
 è (leggermente)  
 differente



**height field**



**range scan**

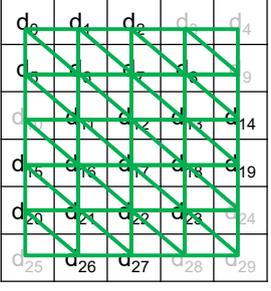


27

### Range Scan, or Depth Image

La connettività implicita di una range scan  
 ne fa una **Triangular Mesh**  
 (competamente regolare, 2 manifold, ben orientata)

$d_0$	$d_1$	$d_2$	$d_3$	$d_4$
$d_5$	$d_6$	$d_7$	$d_8$	$d_9$
$d_{10}$	$d_{11}$	$d_{12}$	$d_{13}$	$d_{14}$
$d_{15}$	$d_{16}$	$d_{17}$	$d_{18}$	$d_{19}$
$d_{20}$	$d_{21}$	$d_{22}$	$d_{23}$	$d_{24}$
$d_{25}$	$d_{26}$	$d_{27}$	$d_{28}$	$d_{29}$



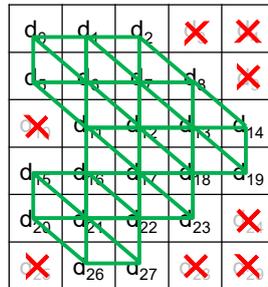
Eccetto per  
 eventuali vertici non  
 2 manifold...



28

## Range Scan, or Depth Image

La connettività implicita di una range scan ne fa una **Triangular Mesh**  
 (competentemente regolare, 2 manifold, ben orientata)



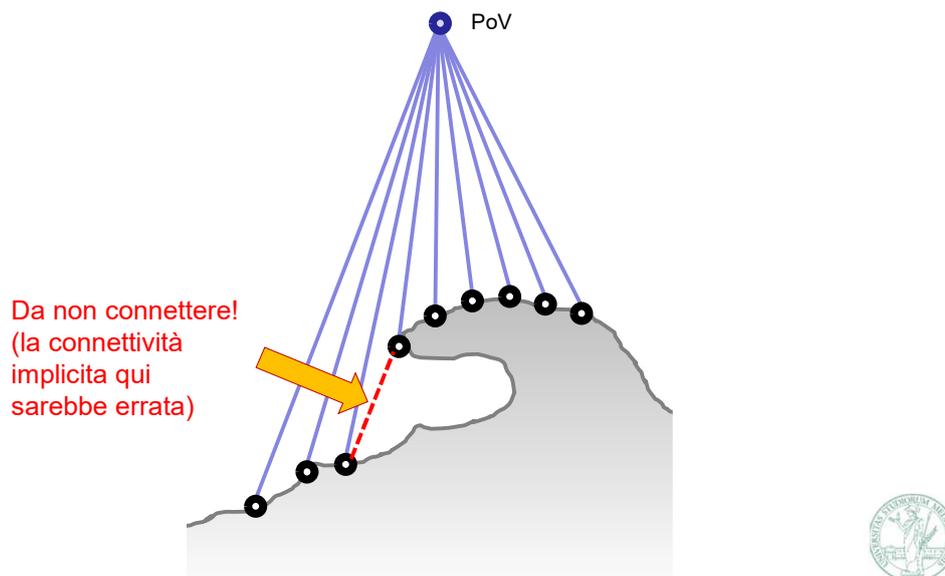
Vanno però evitati i triangoli che hanno vertici definiti come nulli o che sono attraversano salti sulla zeta  
 (possibile criterio: scartare i triangoli con area eccessiva)



29

## Depth jumps in range scans: superfici mancanti

Un «depth jumps» può segnalare una superficie ignota



30

## Range scan e depth map

- ✓ Possono essere considerate alla stregua di...
  - ⇒ ...immagini rasterizzate monocanale (quindi, in scala di grigi)
  - ⇒ ...nuvole di punti
  - ⇒ ...mesh triangolari o quadrangolari regolari
- ✓ ed essere quindi memorizzate / renderizzate / processate come tali
- ✓ Con alcune differenze!



31

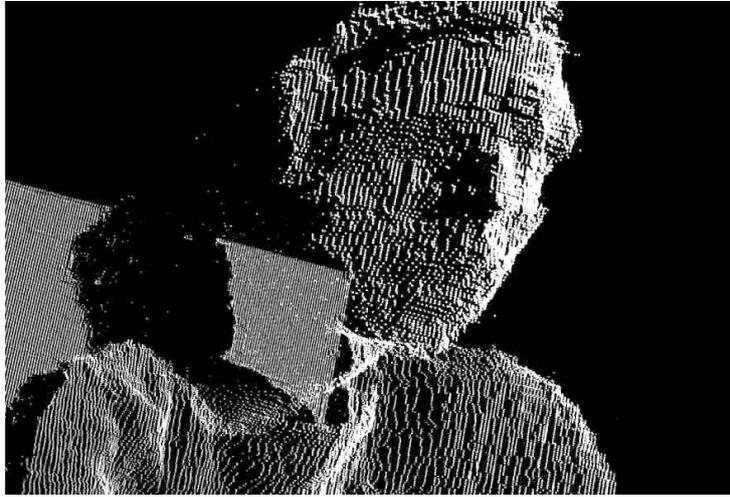
## Range scan: come immagine a scala di grigi

Posso considerarle (visualizzarle, salvarle, etc) come immagini raster ad un canale (gray-scale)



32

### Depth (motion) cameras! (e.g. Microsoft Kinect)



Output del kinect (una Depth camera)  
visualizzato come una nuvola di punti (attraverso point splatting)

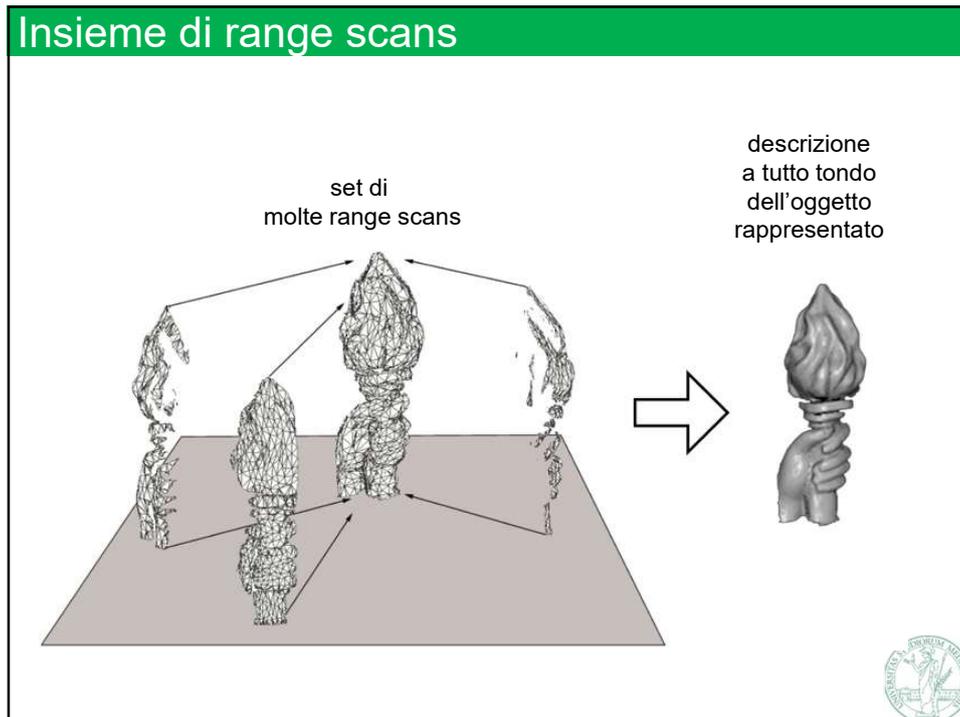


33

### Range Scan renderizzata come mesh triangolare



34



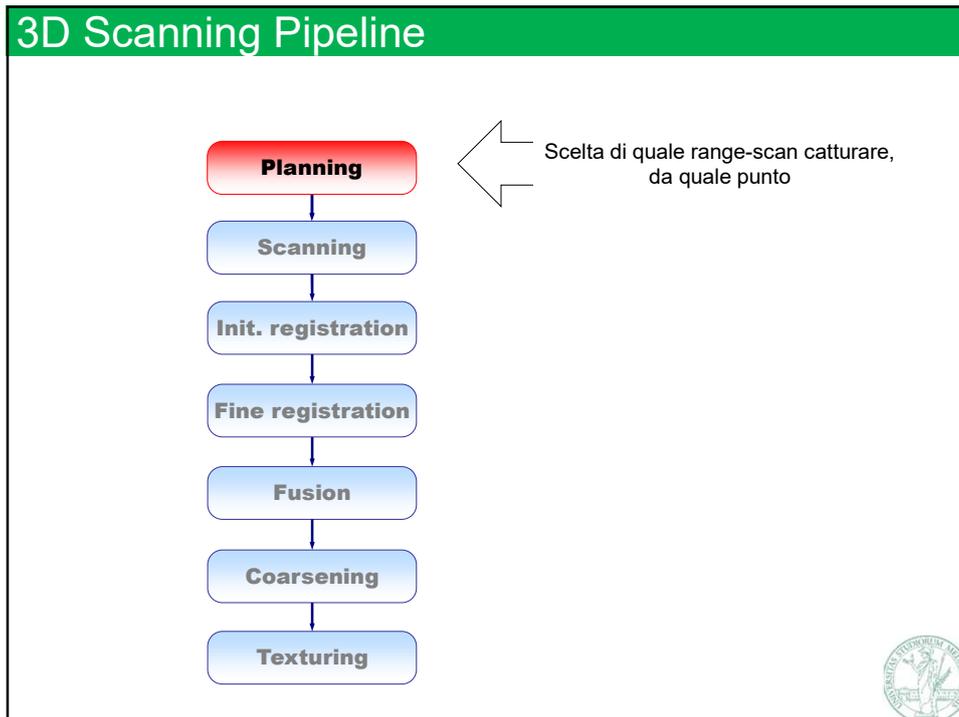
35

### Insieme di range scans

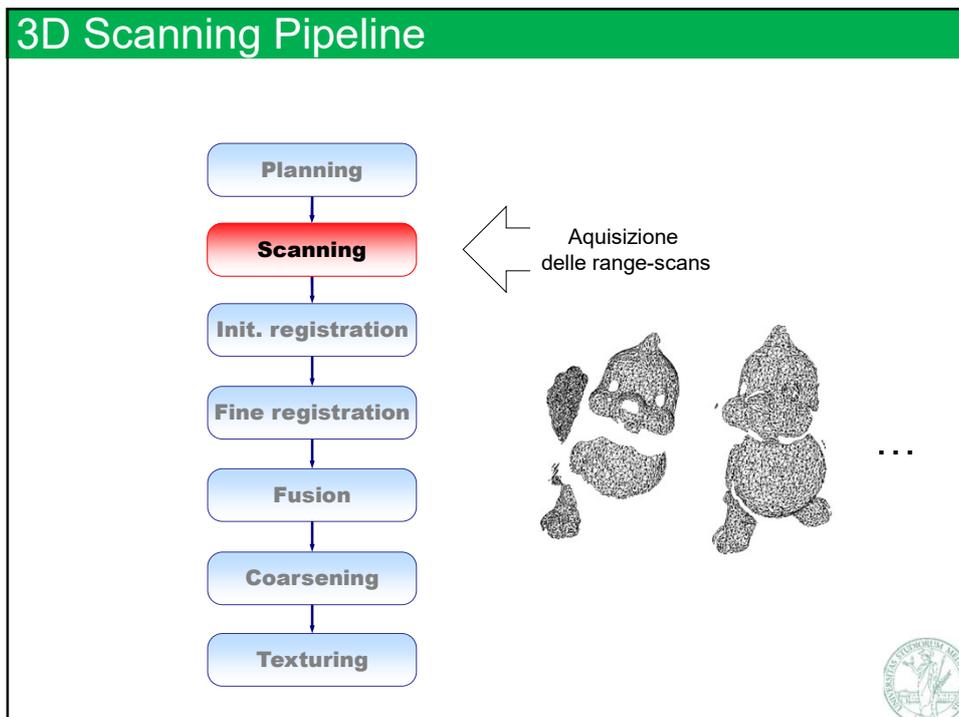
- ✓ Molte tecniche di acquisizione (dette collettivamente di *range scanning*) producono molte **range scans** sovrapposte che rappresentano descrizioni parziali di uno stesso oggetto
  - ⇒ Laser scanning, time of flight :
- ✓ Processing necessario (una pipeline di processi):
  - ⇒ Allineamento reciproco delle range scan (attraverso ICP)
  - ⇒ Fusione in un'unica mesh dell'unione di tutte le nuvole di punti in una sola (attraverso Poisson Reconstruction, o Front advancing)
- ✓ Spesso seguito da:
  - ⇒ Mesh simplification
  - ⇒ Mesh parametrization
  - ⇒ Texture baking
- ✓ Risultato:
  - ⇒ Una mesh adattiva, a media risoluzione (adattiva), UV mapped, e textured (pronta per l'uso in, mettiamo, un videogame)



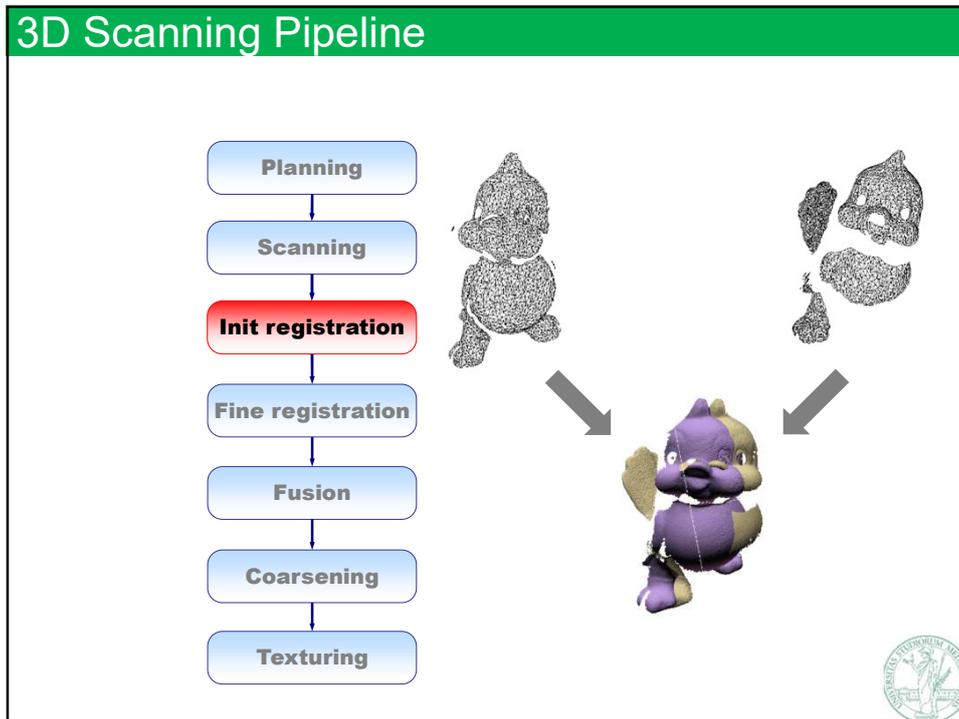
36



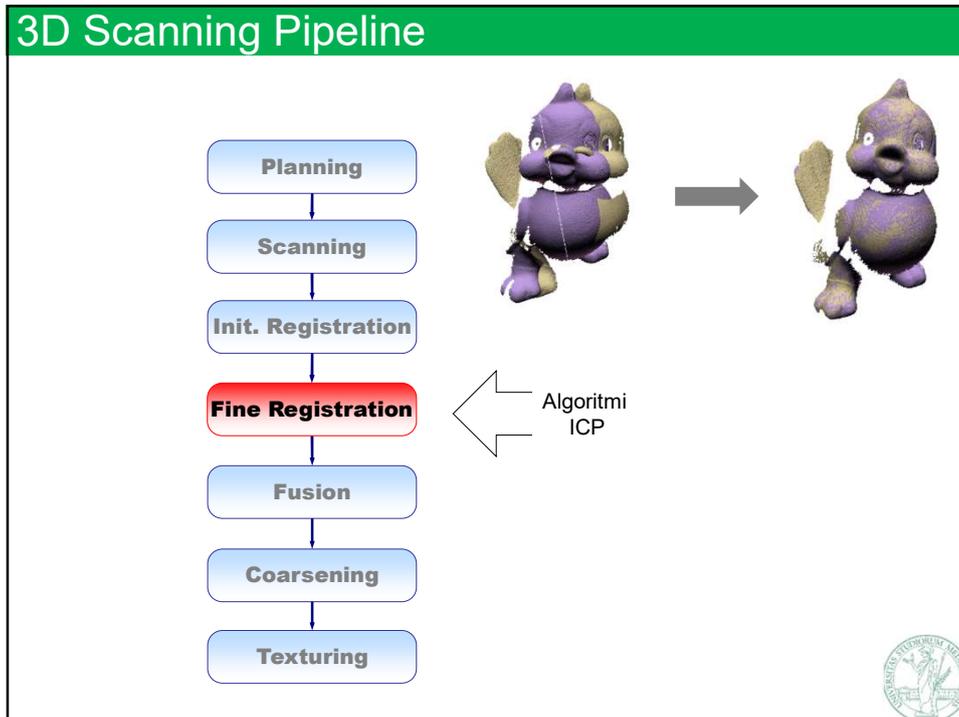
37



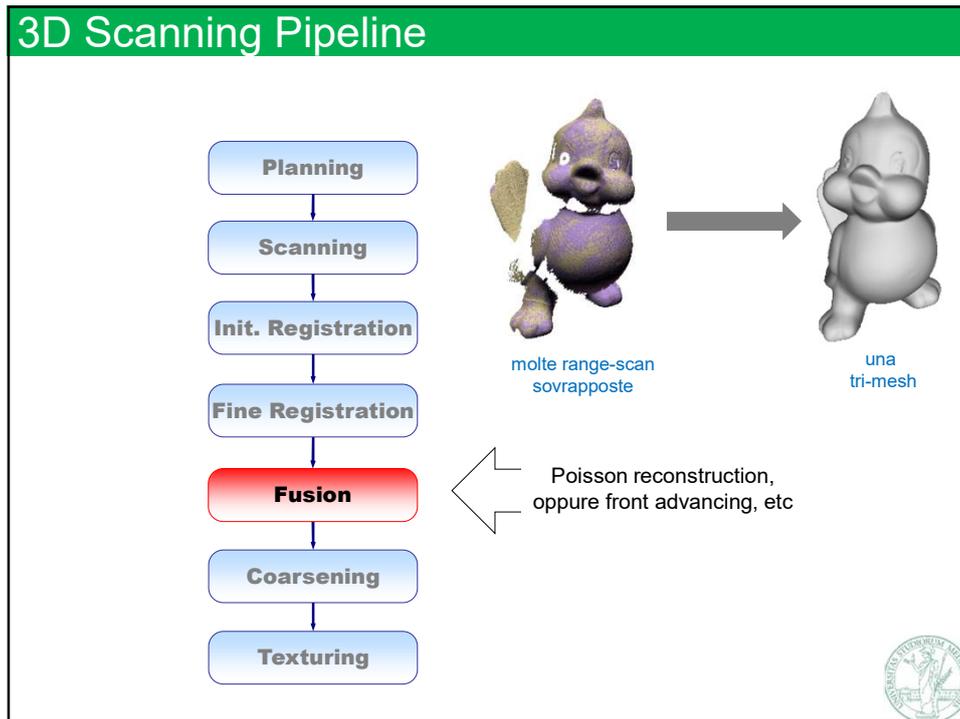
38



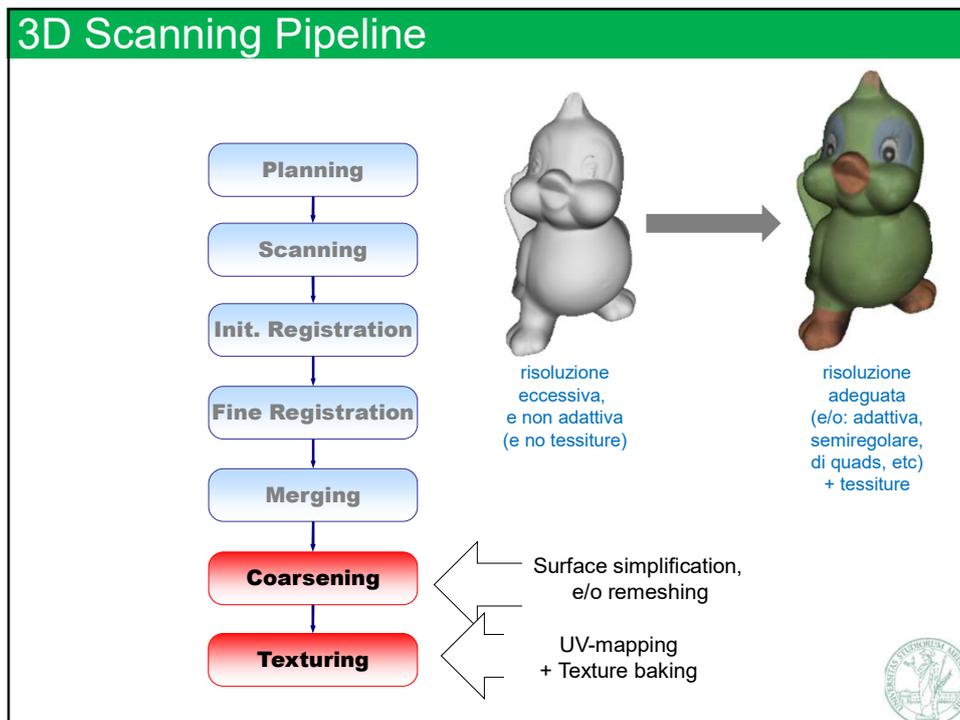
39



40



41



42