



# **Il colore: acquisizione e visualizzazione**

Lezione 15: 10 Maggio 2013

# The importance of color information

---

## Precision vs. Perception



**3D scanned geometry**

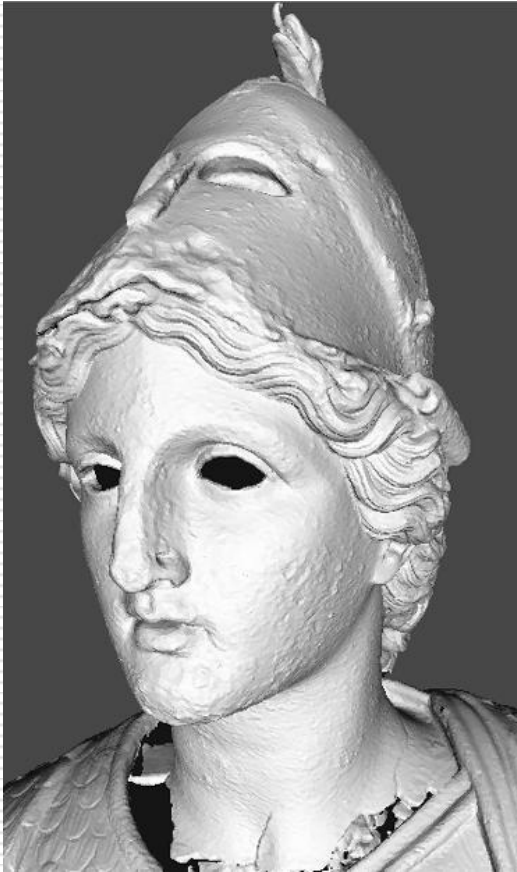


**Photo**

---

# Color and appearance

---



Pure geometry



“Pure” color



Rendering of  
material properties

---

# What is color?

---

**Color is light!** So how do we represent it?

□ **Apparent color**

- No lighting effects, no moving highlights

□ **Unshaded texture**

- Removal of shading & highlights

□ **Spatially varying reflection properties**  
(Bidirectional Reflection Distribution Function, **BRDF**)

- Relightable representation of the real object interaction with light

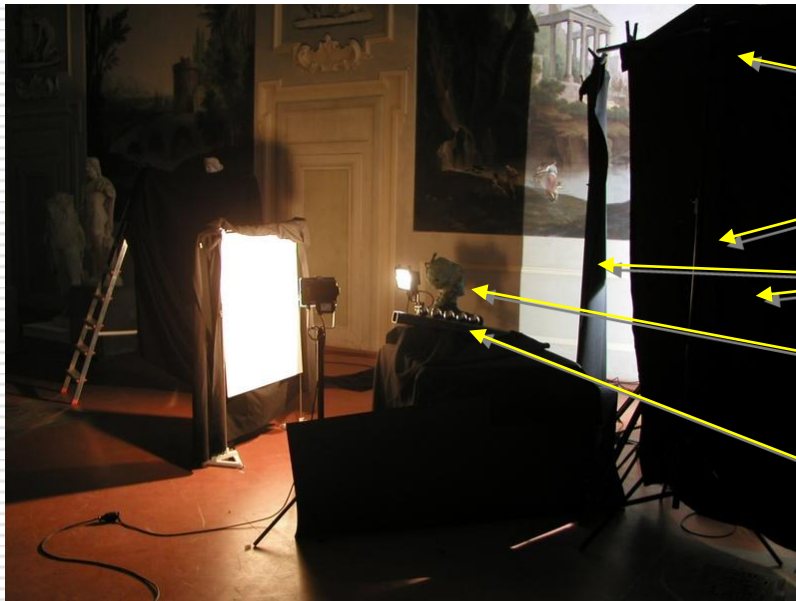
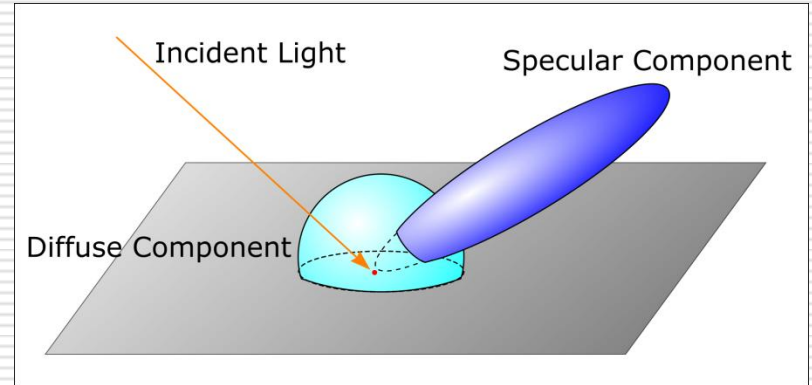


Image by MPI (Lensch, Goesele)

---

# BRDF acquisition

**$BRDF(\theta_i, \Phi_i, \theta_o, \Phi_o, u, v)$**



- light source
- camera
- black felt
- Minerva head
- calibration target

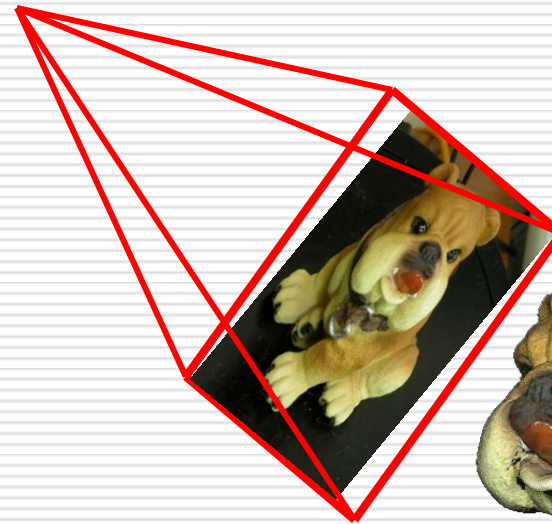
# An alternative solution: color projection

---

Alternatively, we can start from a set of photos covering the surface of the object. In a photo, color information is stored according to optical laws of perspective ...

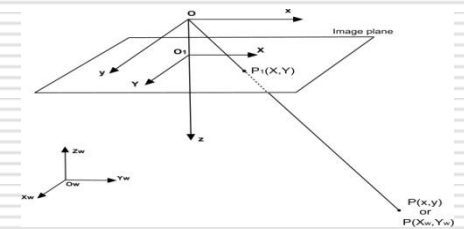
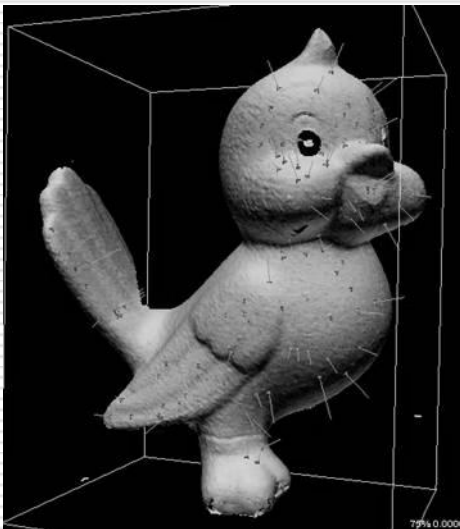
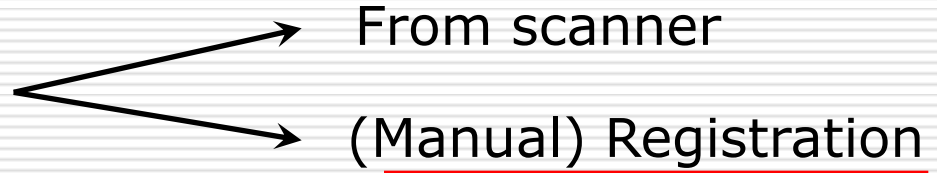
If camera parameters can be recovered, it is possible to project back the information onto the geometry

Simple and effective...



# Texture building from photos: Input data

- A complete 3D model
- A set of photos
- Registration info  
(camera data)



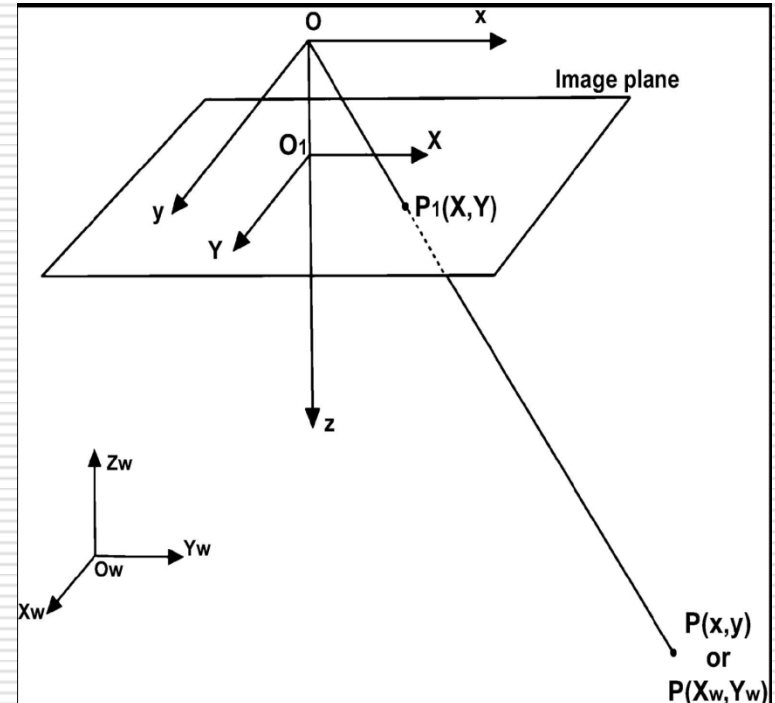
- Position
- Orientation
- Focus distance

.....

# Registration info: parameters estimation

## Intrinsic and extrinsic parameters

- Extrinsic parameters: rotation matrix and translation vector
- Intrinsic parameters: focal length, lens distortion...



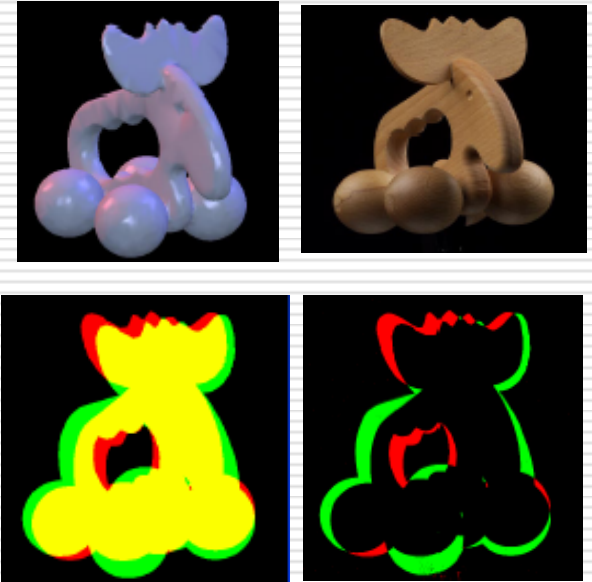


# Automatic parameters estimation

---

Automatic camera computation via silhouette matching

- Compute silhouette on image
- Render the 3D model
- Compute error as render-silhouette image difference
- Greedy iteration, by small rotations, until silhouette matching error is below a threshold



Limitations:

all object visible in each image ==> just small objects!  
hard or impossible to generate silhouette

**Automated Texture Registration and Stitching for Real World Models**

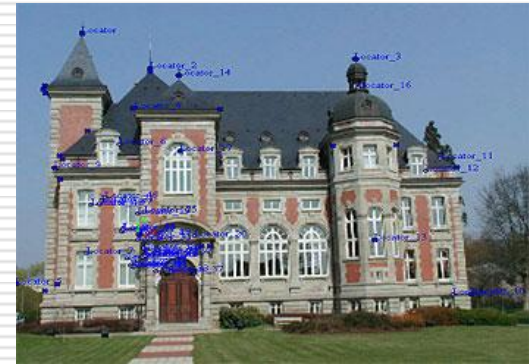
Lensch et al, Pacific Graphics 2000

---

# Estimation using photogrammetry tools

---

Photogrammetry tools do estimate the camera parameters and can be used to recover intrinsics and extrinsics to project color on a 3D model. Many tools can as well do the entire texturing process, for small models



<http://imagemodeler.realviz.com>

---

# Parameters estimation using correspondences

Parameters estimation:

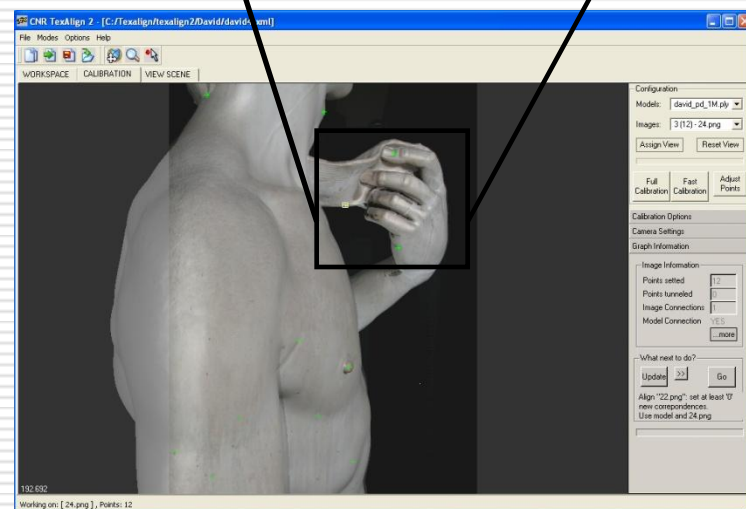
- Setting of some correspondences between image and geometry
- Minimization of error function

Different algorithms and implementations:

- TSAI (old faithful)
- GARCIA (fast but need good start)
- intel OpenCV (hard to integrate)

**Minimizing user intervention in registering 2D images to 3D models**

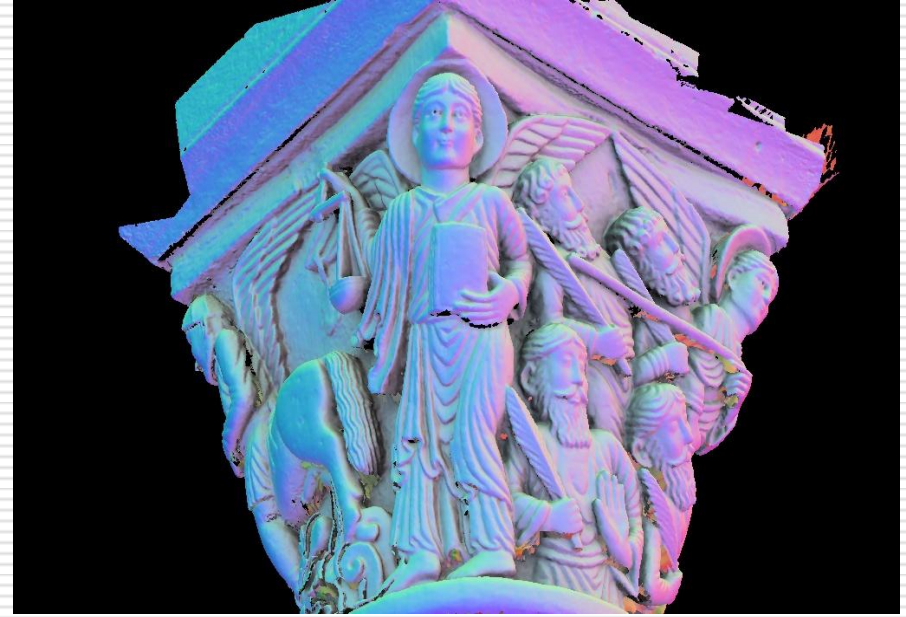
T. Franken, M. Dellepiane, F. Ganovelli, P. Cignoni, C. Montani, R. Scopigno 2005



# Automatic alignment using mutual information

---

- ❑ Mutual information is used with geometric features ***correlated in some way*** to the visual appearance of the objects but *invariant* to the lighting environment.



# Image registration: pros and cons

---



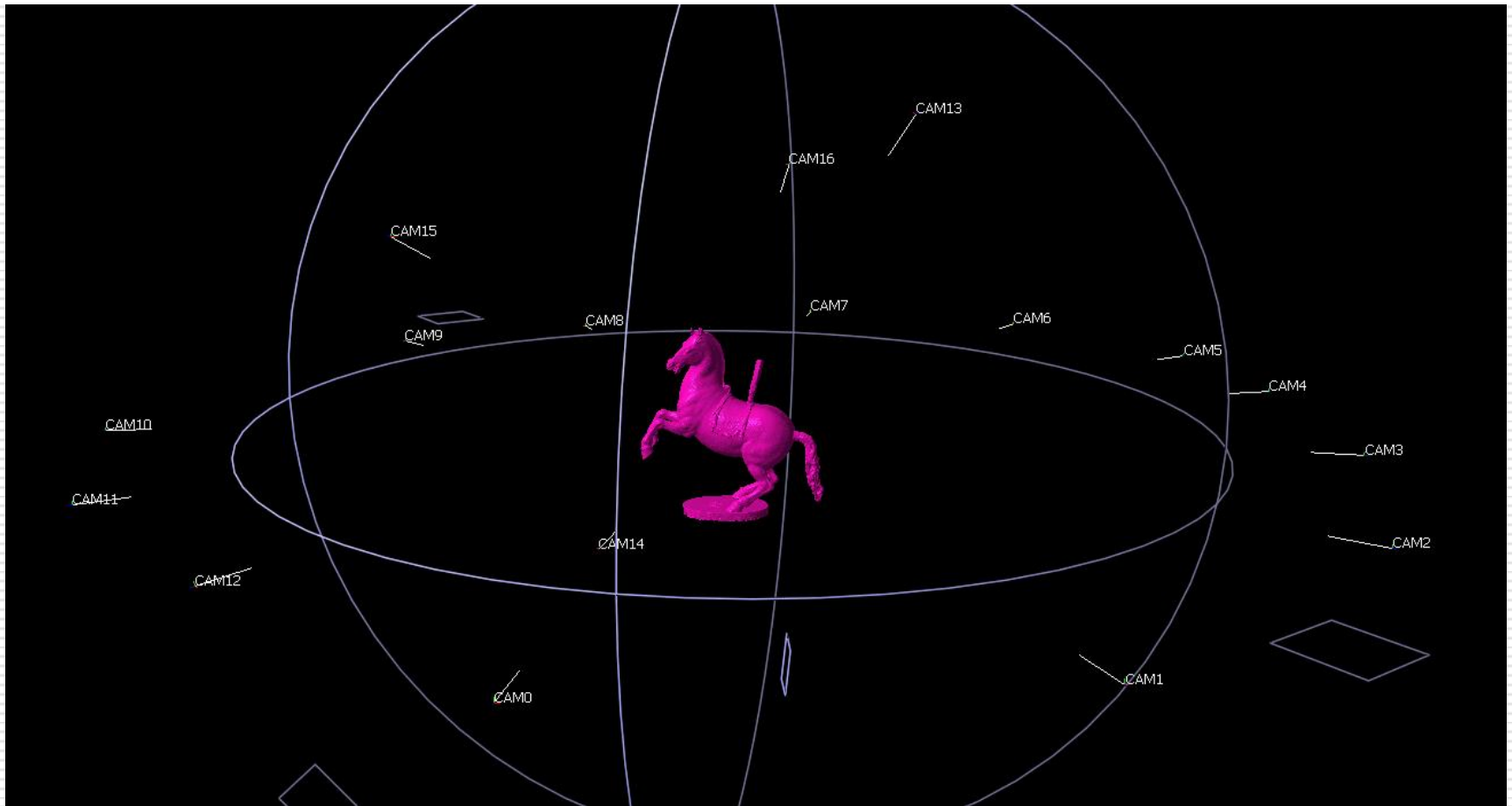
- User friendly
- Tens of images on one model
- Very flexible (from statues to buildings)
- Extensible

- Extrinsic/intrinsic
- Dependent on correspondences / Starting point
- Measure of alignment quality



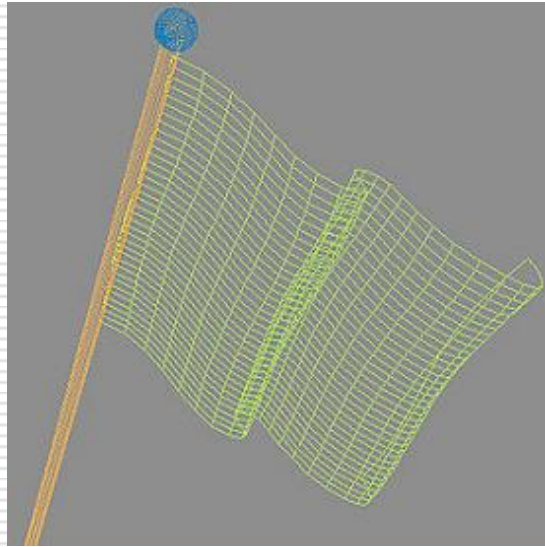
# 3D model, photos, camera parameters... and now?

---

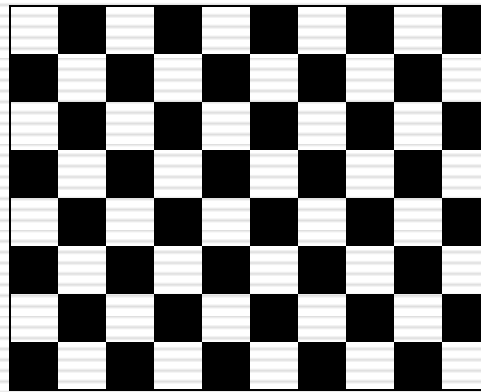


# Encoding the color information: Texture mapping

---



+



=



3D geometry

RGB texture  
2D

(color-map)

---

# Encoding the color information: Texture mapping

---



+



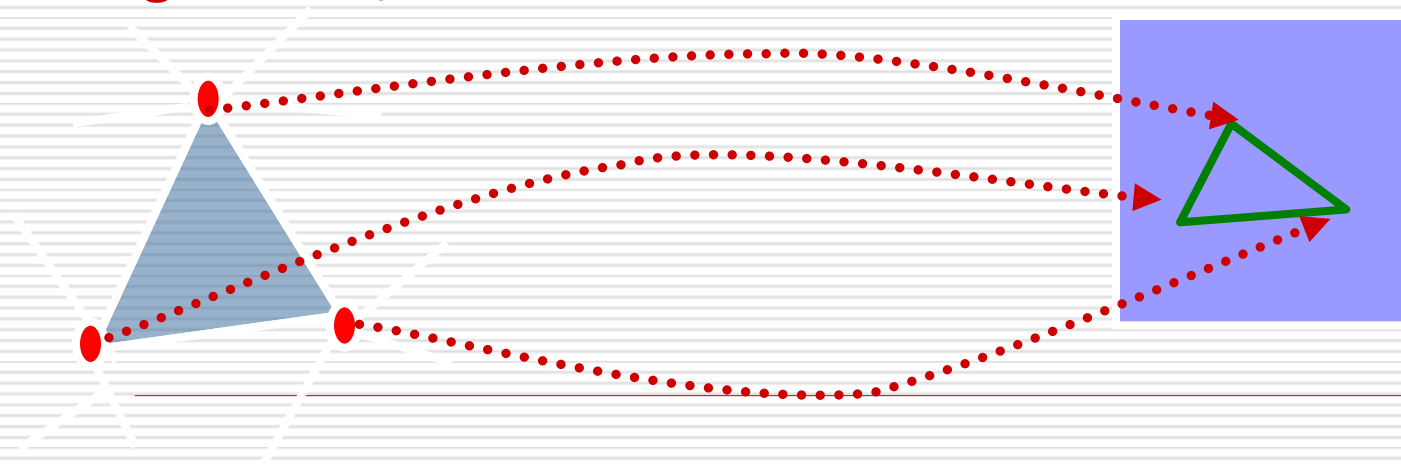
=



Texture: image...

3D geometry

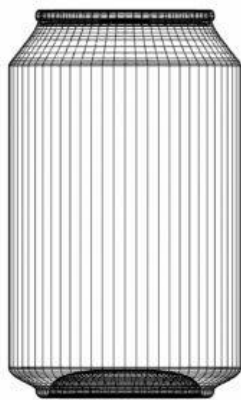
Texture mapped rendering





# Encoding the color information: Texture mapping

---

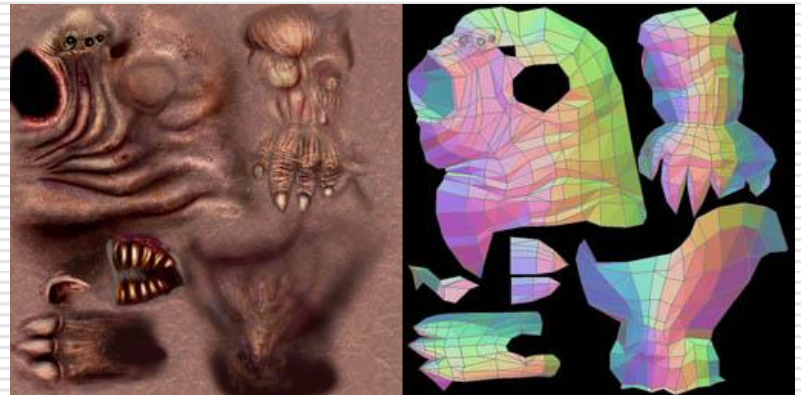


# Encoding the color information: Texture mapping

---



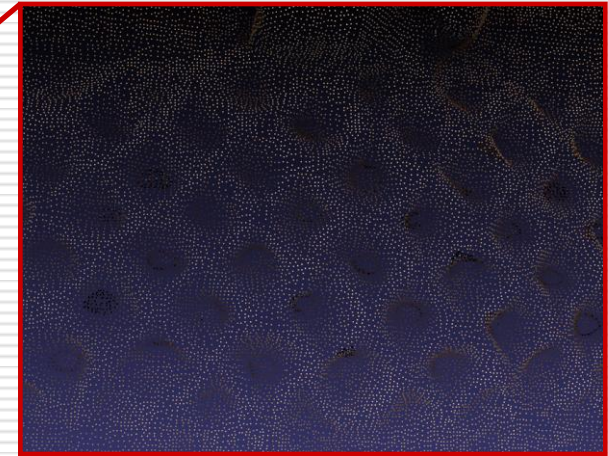
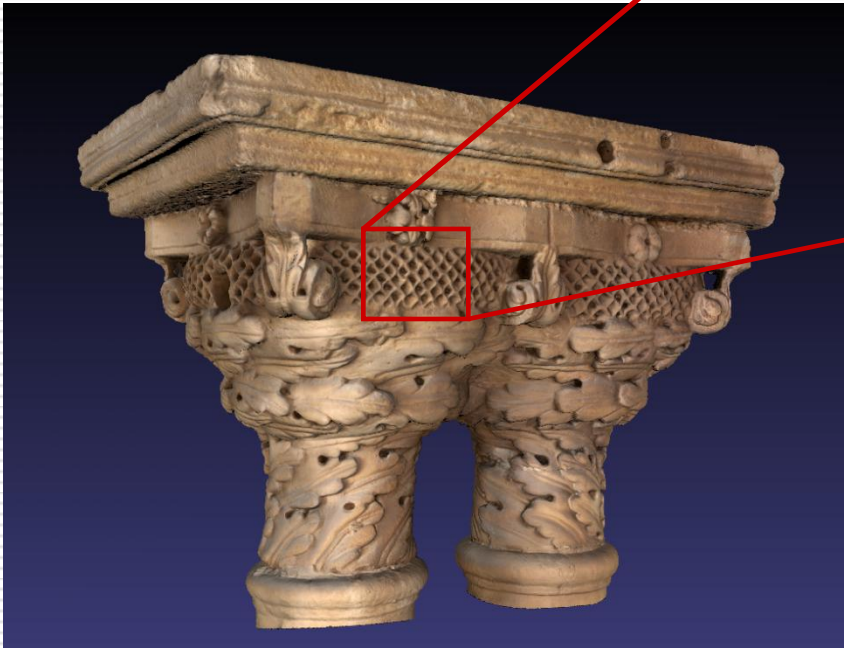
Hand made, or  
automatized



## Encoding the color information: Color per vertex

---

A color value is assigned to each vertex of the model.  
The space between points is filled via interpolation.



# Encoding the color information

---

## Texture Mapping

- Independent of geometric density
- Compact 2D Structure
- Editable, compressible, easily accessible structure
- Parameterization
- Use with "multiresolution" or adaptive structures
- Need to pack data without losing detail
- Blending between photos

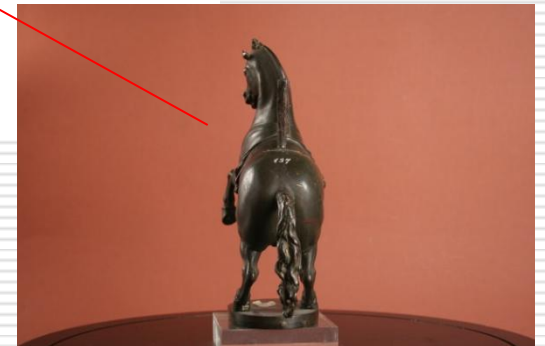
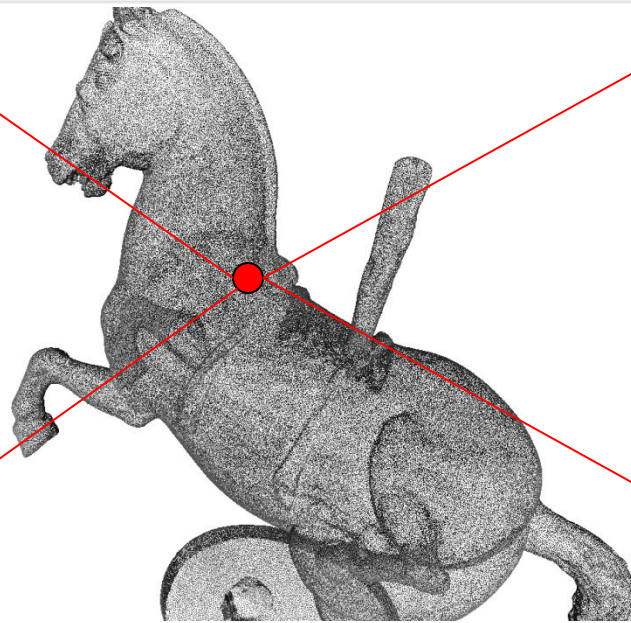
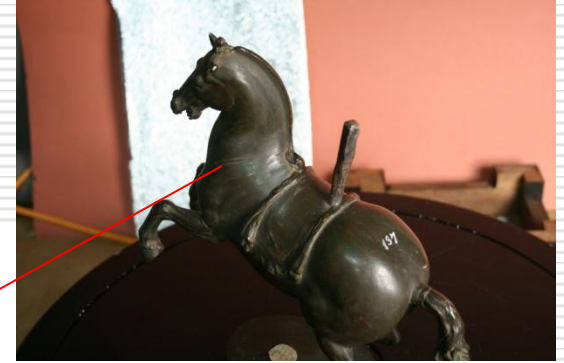
vs



## Color per vertex

- Easy structure
- Compatible with "multiresolution" or adaptive structures
- No need for parametrization
- Very dependent on geometric density
- Harder to access or "boost" (for now)
- Texture mapping is more widely used

# Mapping the color information



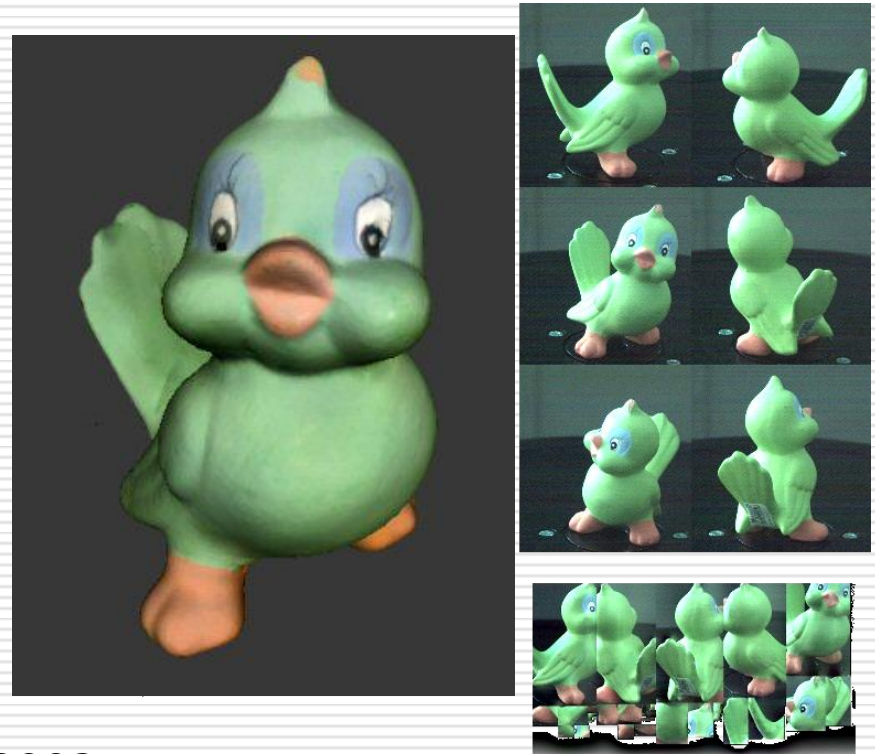
Which color value?

# Mapping the color: automatic texture mapping

---

*Weaver*, a tool for the generation of texture maps

- ❑ For each area, the better (orthogonal) photo is chosen
- ❑ Mesh is splitted according to the photo allocation and parametrized using perspective projection
- ❑ From photos, the used area is cut and packed in the texture
- ❑ Color discordances on borders are corrected



## Reconstructing Textured Meshes From Multiple Range RGB Maps

M. Callieri, P. Cignoni, and R. Scopigno 2002

---

# Mapping the color: masked photo blending

---

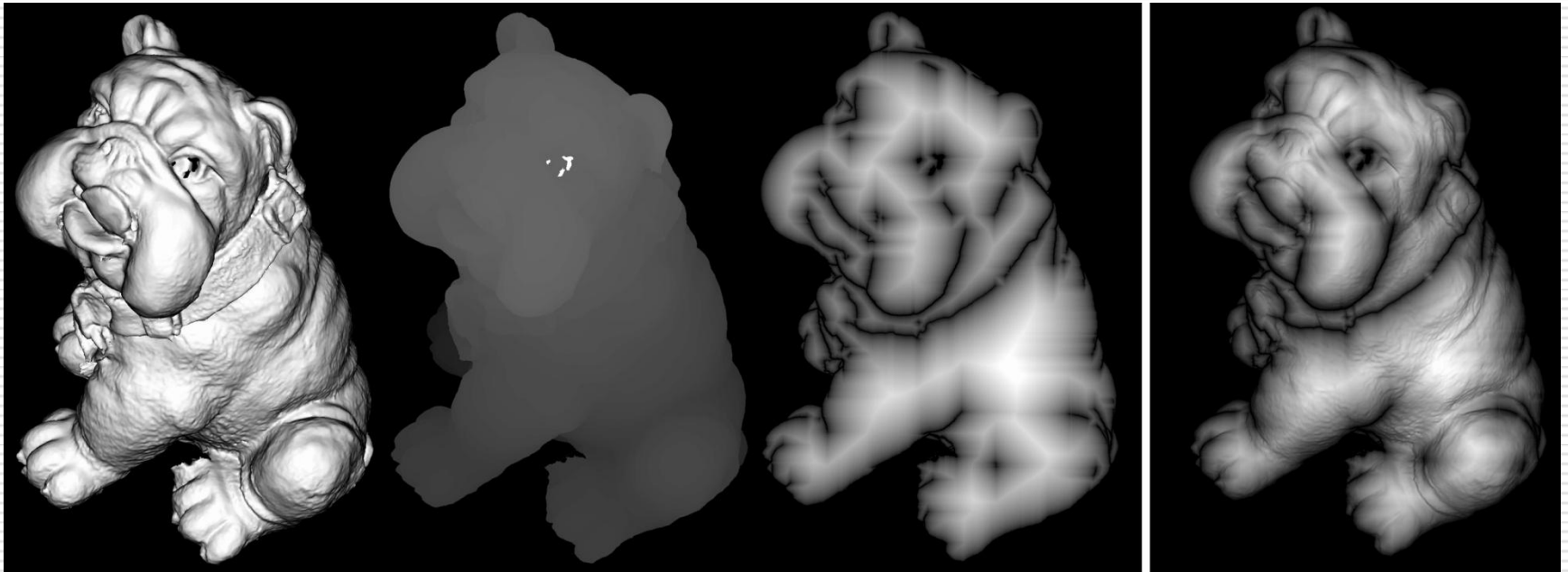
For each image, a set of quality masks is calculated.

Angle  
Mask

Depth  
Mask

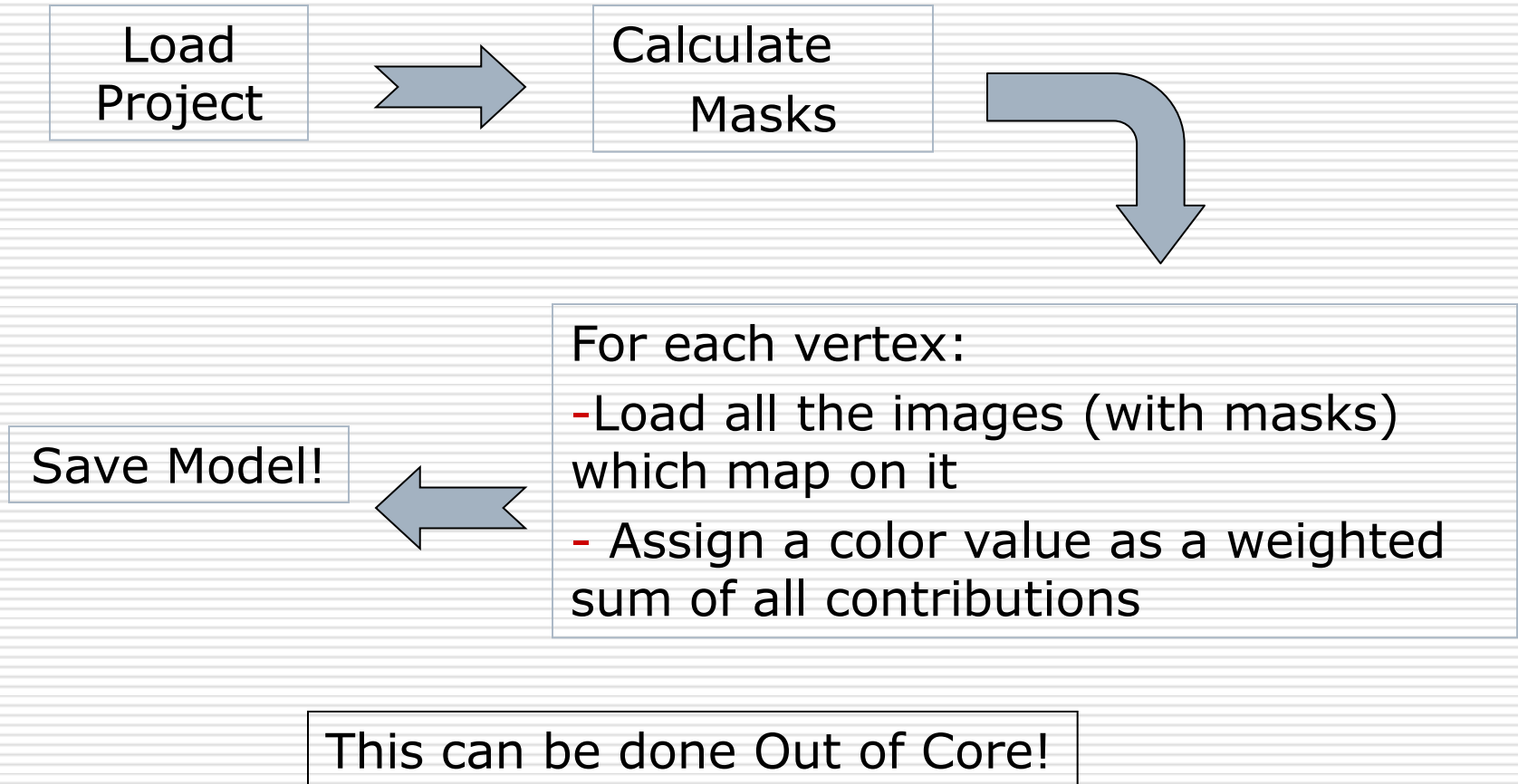
Border  
Mask

Final  
Mask



# Mapping the color: masked photo blending

---



## **Masked photo blending**

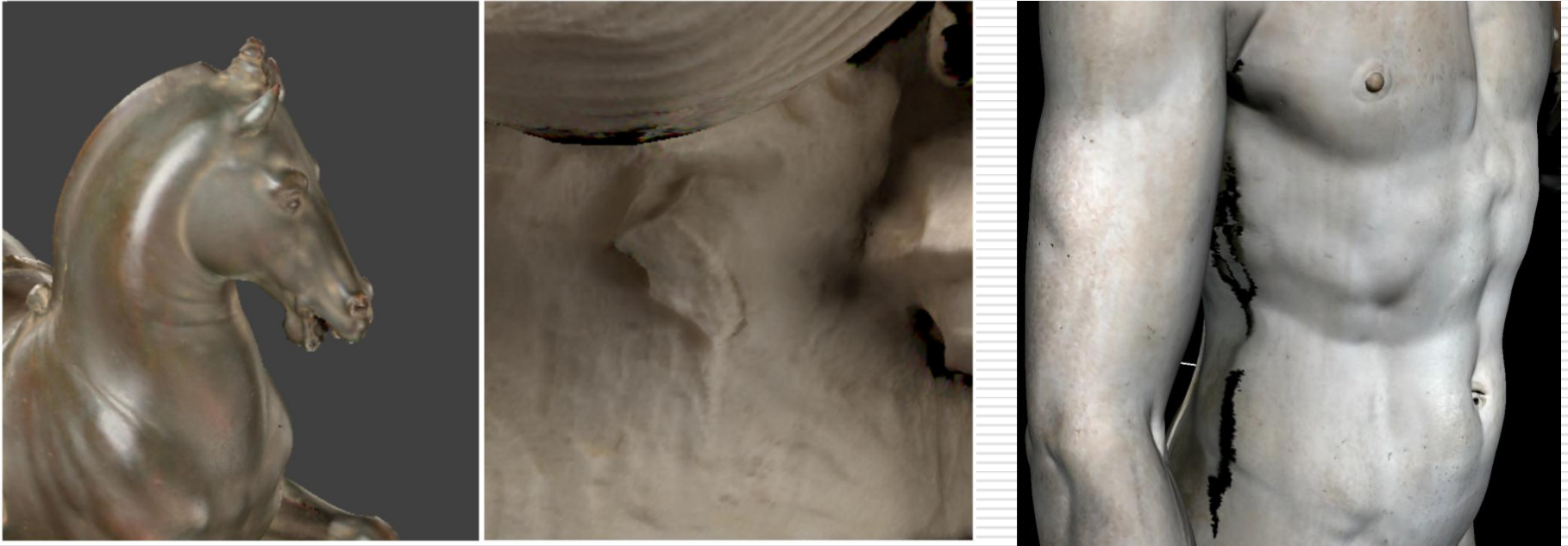
M. Callieri, P. Cignoni, M. Corsini and R. Scopigno

---



# Color projection: open issues

---

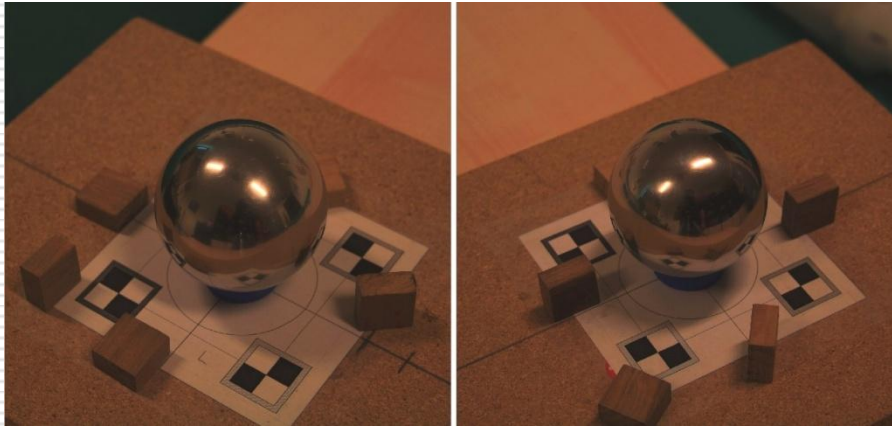


The quality of color depends mainly on:

- the original photo set (shadows, highlights, uneven lighting, bad coverage)
  - the quality of image registration
-

# Color projection: controlling the light environment

---



Use an acquisition device to estimate the lights in the scene.

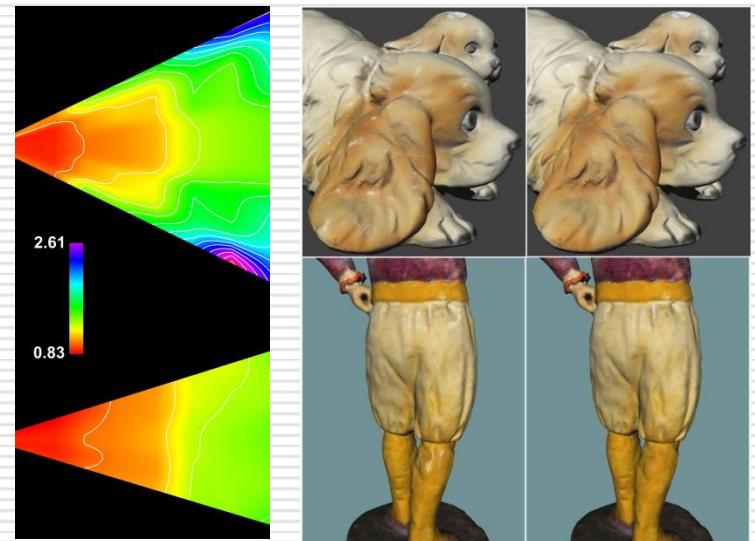
## **Stereo light probe**

M. Corsini et al. 2008

“Calibrate” a light source to correct image artifacts before and during projection

## **Flash lighting space sampling**

M. Dellepiane et al. 2009



# MeshLab in full color

---



## Image Alignment

Filters->Camera->Image registration: Mutual information

Usage:

- 1) Get Shot
- 2) Apply

Note: Focal length issue

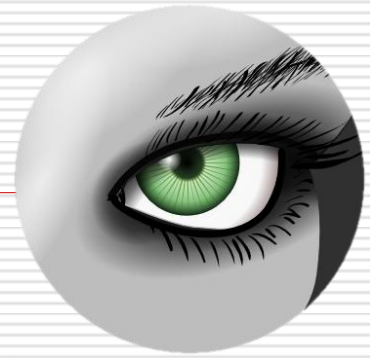
Coming (not) soon:

Use of correspondences/hybrid method (Sottile et al 2010)



# MeshLab in full color

---



Color projection

Filters->Camera->Project active rasters to current mesh

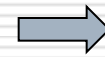
Usage:

1) Apply

Color per vertex

or

Texture if you already have a  
parametrization



# MeshLab in full color

---



Parameterization

Filters->Texture-> Parameterization + Texturing from images

Usage:

- 1) Define texture name and resolution
- 2) Apply
- 3) Save model with texture

Note: will be present in the official release



# Color projection: wrap up

---

## **Big issues in color projection**

- Photo shooting (lights setup, surface coverage)
- Material estimation
- Image registration (semi-automatic)
- Color encoding
- Color projection
- Visualization

**Pseudo-conclusion:** the approach depends mainly on the object and the application

---

# Next in line...

---

Next lesson:

- Complex geometries and remove visualization

Contacts:

Matteo Dellepiane

c/o ISTI-CNR Via G. Moruzzi 1

56124 Pisa (PI)

Tel. 0503152925

E-Mail: [dellepiane@isti.cnr.it](mailto:dellepiane@isti.cnr.it)

Personal website: <http://vcg.isti.cnr.it/~dellepiane/>

VCG website: <http://vcg.isti.cnr.it>

---