Meshlab as a Complete Open Tool for the Integration of Photos and Colour with High-Resolution 3D Geometry Data

Marco Callieri, Guido Ranzuglia, Matteo Dellepiane, Paolo Cignoni and Roberto Scopigno
Visual computing Lab, ISTI - CNR, Italy

Abstract:
Even if a precise, detailed geometry is essential for the study and documentation of a Cultural Heritage (CH) artifact, lots of information also comes from the colour and appearance data. Since a complete reconstruction of the optical properties of an artifact is still hard (or impossible) to achieve, what is normally done is to rely on colour data coming from photographic dataset. In this paper, we will describe the tools that MeshLab offers for the generation and processing of colour information on a high-resolution 3D model. We will show the basic operations available and present actual projects where these tools have been successfully used. The availability of such advanced colour management tools in an Open Source tool is important for the archaeological and CH community. MeshLab is a complete, state-of-the-art open source software for the creation and manipulation of high-resolution 3D models starting from real-world 3D scanned data, at par with the specific commercial tools available on the market.

Keywords:
MeshLab, 3D Scanning, Color Mapping, Photographic Registration, Texture Mapping

1. Introduction

In digital 3D models, the reproduction of apparent colour, obtained by mapping photographic information on a 3D model is still the most common approach to describe the visual appearance of an object, especially in the cultural heritage field.

While a more complex representation of the material of the object, like a Bi-Directional Reflectance Distribution Function (BRDF) would have been much more effective, it is often impractical (due to time and resources constraints) to employ the sampling techniques needed to acquire such complex representations.

The advance of 3D scanning hardware has been, in the last few years, quite significant; on the market are nowadays available many different devices, able to capture objects with a wide range of materials/sizes/complexity[2].

Unfortunately, the same technological advance had not much effect on the ability of devices to capture the colour of scanned objects. Most 3D scanners do not capture colour. In the devices with such sensors, the generated colour is generally poor in terms of resolution (the colour sensor is often an embedded camera derived from phone/mobile photographic devices) and photometric quality (no explicit calibration is possible, shooting parameters are chosen by the device and no manual mode is available, the software does not support saving in uncompressed formats, hidden colour processing is done in the driver software). It may be enough as a preview or for some industrial processes, but not for the needs of Cultural Heritage operators.

For this reason, a 3D Scanning campaign is usually supported by a photographic campaign, since the quality of the colour acquired by scanner is not always sufficient, or the acquisition is performed in light conditions that are bad for photography (but do not influence the scanner).

Corresponding author: callieri@isti.cnr.it

2 Since 3D scanning hardware is no more a research topic, there are no updated technical surveys on the devices available on the market; two partial lists are available at www.simple3d.com and www.sculptor.org/3D/Scanning/imaging.htm.

The task of effectively using photographs to generate documentation for 3D models, and to generate a colour mapping of the 3D surface is still one of the most problematic steps in the processing pipeline of 3D scanning data.

2. Previous Work

While most of the intermediate steps in the colour mapping pipeline have been active topics of research, it is difficult to find examples of complete software tools able to manage in a complete way the process of applying colour to 3D models. Each research group active in this field has produced software prototypes for their methods; these tools, however, are generally available by request, and it may prove to be difficult to use them to build a complete software pipeline.

Some commercial 3D tools, in particular those oriented to the management of data produced by 3D scanners, like RapidForm or Geomagic, do implement (in part or completely) the colour mapping pipeline. MeshLab, on the other hand, does implement the entire colour management pipeline.

The first step in the colour projection pipeline is image registration, since in most cases, the camera parameters associated with each image are not known in advance. Several automatic (Ikeuchi et al. 2007; Brunie et al. 1992; Lensch et al. 2000; Wolberg and Zokai 2006) and semi-automatic (Franken et al. 2005) methods for image- to-geometry registration have been proposed. They are mainly based on an analysis of the geometric features of the model (e.g., silhouette and orthogonality), or on some input given by the user (2D-to-3D correspondences).

The most correct way to represent the material properties of an object is to describe them through a reflection function (i.e. BRDF), which attempts to model the observed scattering behavior of a class of real surfaces. A detailed presentation of its theory and applications can be found in (Dorsey et al. 2007). Unfortunately, state-of-the-art BRDF calculation approaches rely on controlled and complex illumination setups (Lensch et al. 2001; Debevec et al. 2000): this limits their application in the context of complex scanning projects (big artifacts and on-the-field acquisitions, such as those performed in museums).

A less accurate, but more robust, solution is the direct use of images to as a source of colour information for 3D surfaces. In these cases, the apparent colour value, as sampled in digital photographs, is mapped on the digital object's surface by registering these photographs onto the 3D model (by estimating the camera parameters), and then applying an inverse projection. In addition to other important issues (cited in sections 5 and 6), such as the image registration and how to store colour information, there are numerous difficulties in selecting the correct colour when multiple candidates are in different images. Some of these are: how to deal with discontinuities caused by colour differences between photos that cover adjacent areas, and how to reduce illumination-related artifacts, e.g., shadows, highlights, and peculiar BRDF effects.

To solve these problems, one group of methods selects, for each part of the surface, a portion of a representative image following a specific criterion - in most cases, the orthogonality between the surface and the view direction (Callieri et al. 2002; Bannai et al. 2004). However, artifacts caused by the lack of consistency between overlapping images are visible on the borders between surface areas that receive colour from different images. These can be partially removed by working on the border between two images (Callieri et al. 2002; Bannai et al. 2004).

Another group of methods "blends" the contribution of all the images by assigning a weight to each one or to each input pixel (this value expresses the "quality" of its contribution),

and selecting the final surface colour as the weighted average of the input data, as in (Pulli et al. 1998). The weight is usually a combination of various quality metrics (Bernardini et al. 2001; Rankov 2005). In particular, (Callieri et al. 2008) presented a flexible weighting system that can be extended in order to accommodate additional metrics.

Another important step in the colour management is the creation of a parametrization for the 3D model; this is still an active research topic, and a large number of papers on surface parameterization have been recently published. A good survey of this field can be found in (Sheffer and Praun 2006).

3. The Tool

MeshLab (Cignoni et al. 2008) is an open source tool for the visualization and processing of 3D models. It is oriented to the management and processing of large, unstructured triangular meshes and pointclouds, and it provides a set of tools for measuring, checking, cleaning, healing, inspecting, rendering and converting 3D meshes. MeshLab is freely available, distributed under the GPL licensing scheme and it is available for all the major platforms (Windows, MacOS, Linux).

Born as a university project, MeshLab has steadily grown in features and usability, reaching more than 200.000 downloads in this year. MeshLab is used by hundreds of research groups and industries, and by thousands of 3D hobbyists. Featuring various state-of-the-art 3D processing algorithms (often implemented by their academic authors), it represents a solid and free alternative to commercial tools for 3D scanning data management.

Processing 3D scanned data is not a simple task; most scanners do have a bundle software able to manage the data acquisition and manage simple to medium -sized projects. These tools, however, only work with a specific device. Processing of large and complex digitization projects, or mixing data coming from different devices does, in general, require specific commercial software tools, available on the market. MeshLab does implement fully the 3D scanning and colour pipeline, is able to process a large amount of data, does accept 3D data coming from different devices and it is readily available to users.

The design of MeshLab is based on a modular structure: around some core structures, able to manage high resolution 3D and provide basic rendering functions, all the functionalities of the tool are implemented as individual operations, each one independent from the others.

MeshLab relies on a wide variety of mesh processing functionalities, which are exposed as a large set of self-contained filters that take as an input one or more meshes and some user specified parameters. MeshLab offers around two hundred different filtering operations, categorized into menus according to a few keywords (remeshing, cleaning, sampling, texture, colour processing, quad mesh processing, point clouds, etc). The simple functional structure of filters is implemented through a plug-in framework that hides all the GUI coding allowing for a very simple development of additional plug-ins by new developers.

This “Swiss Army Knife” approach is also one of the most powerful features of MeshLab. Providing many different self-contained instruments, in many cases with multiple alternative filters that do the same operation, makes possible for the user to build a custom sequence of editing operations to solve specific problems of specific datasets.

While having a single-push button tool may be easier for beginners, a fully configurable and flexible one is absolutely necessary for people working on very diverse kind of objects, where each scanning project is different from

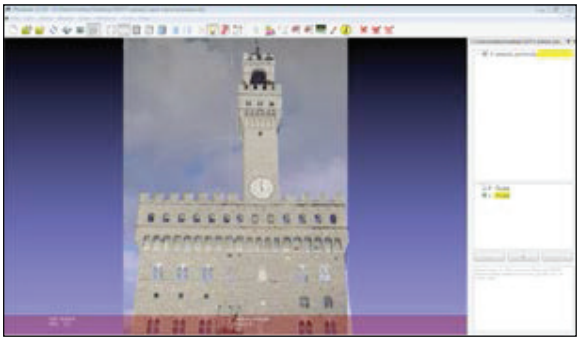

***Figure 1**. Alignment of a photo to a 3D geometry.*

the previous one, as often happens in the cultural heritage field.

4. Raster Layers

In order to support the management of photographic information, a new data type has been added to the MeshLab: the raster layers. Mimicking what is done for 3D meshes (contained in the geometry layers), the system now supports the loading of a series of images.

These layers do contain a raster image, plus the camera parameters needed to establish a correspondence between the 3D geometry and the 2D image. These parameters describe the position and orientation (extrinsic parameters) and internals of the camera, like sensor size, lens distortion and focal length (intrinsic parameters) at the moment of the shot. By obtaining these parameters it is possible to reconstruct the perspective projection that created the photo. This opens up two possibilities: being able to see the 3D scene through the same camera that took the shot (thus, exploring the photographic dataset spatially), and project back the colour information onto the 3D model (to generate colour mapping).

5. Photographic Alignment

If the photos have not been acquired by the 3D scanner (thus, being automatically registered with the 3D information), some processing steps are needed to link them to the geometry, in order to obtain a set of calibrated images, and to eventually project the colour information on the geometry.

This first step is the alignment of the images on the 3D model: this is obtained by estimating the camera parameters associated to each image. This stage may also be called Image Registration or Camera Calibration and Orientation.

This is not a simple task, since the estimation of the parameters, which is a geometrical and mathematical task, is a badly conditioned problem. Some of the parameters are mathematically linked, and it is easy to obtain a camera calibration which is not completely correct, ending in a local minima.

Since there is no simple way to check the correctness of the alignment, a visual feedback must be given to the user in order to assess the quality of the recovered data.

The implementation of photographic alignment inside MeshLab has been designed in order to cope with these problems.

The most robust approaches to image alignment are based on the setting of correspondences between the 3D model and the image (Franken et al. 2005). Yet, this point picking operation may be difficult, depending on the characteristics of the 3D object. Hence, a different approach, based on Mutual Information (Corsini et al. 2009) was applied. This is based on the calculation of a statistical measure of correlation between the image and a rendering of the model.

The Mutual Information approach has been implemented as an additional feature of MeshLab. Using an easy to use visualization, it possible to have a direct feedback about the quality of the alignment. This alignment strategy works on triangulated 3D models but also on point clouds, making it quite versatile.

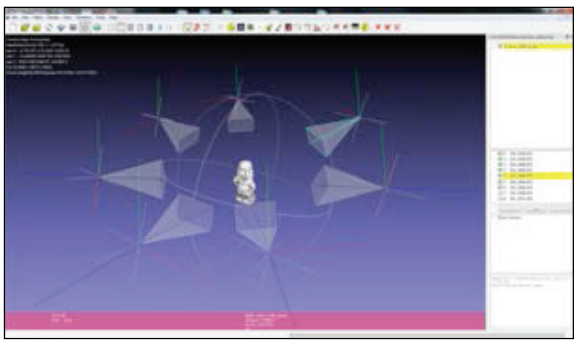

***Figure 2**. Photos aligned to the 3D geometry, displayed in their spatial position, with the associated camera parameters.*

Another alignment method, able to combine user-picked reference points and the Mutual Information is being added, and will be released in a future version of the tool. This addition will make the alignment of difficult datasets easier, at the price of a more complex user interaction.

Photographs aligned in this way may be spatially-explored directly in the 3D space, instead of browsing a folder on the disk, looking at the photos like see-through transparencies suspended in space or projected one by one onto the 3D surface.

An interesting possibility offered by this strategy is to use unconventional images, like photos with annotations, historical photos, hand drawings or sketches or even near-visible lighting photos (ultraviolet, multispectral, infrared, thermography, etc.). This possibility of spatial exploration of the geo-referenced photographic set is a powerful tool to effectively browse a collection, and has multiple uses in the CH domain.

However, in many applications, it is necessary to have colour information mapped onto geometry. With this aim, MeshLab offers different colour mapping tools to better cope with the different needs of the various datasets. By using the colour data from the calibrated images, it is possible to generate detailed, artifact free 'per-vertex' colour encoding, to fill an existing texture parametrization or to generate entirely new texture mapping, driven by the photographic coverage.

6. Colour Storage

The first problem in colour mapping is where to store the colour information on the 3D object.

While the management of 3D geometry is more or less standard among the various software for the 3D creation, editing and rendering, the support of colour information is still quite different from one tool to another. In addition, different file formats support different methods of colour storage, adding complexity to the task.

The two main methods of colour storage are per-vertex (each vertex of the mesh has an associated RGB value) or texture mapping (3d object has a parametrization, which associate a pixel in a 2 image to each point on the 3D surface, and the colour is stored in an image, called texture map).

Per-vertex encoding works well for highly dense models, such as the ones produced by 3D scanning; this because the colour detail is limited by the resolution of the geometry. It is a compact and effective way to store colour, it is simple to use and works with most 3D modelling and rendering tools.

On the other hand, texture map is a more standard solution, it decouples the resolution of the geometry from the resolution of the colour detail and it is supported (with small variations) by all 3D software.

While the texture approach seems more usable and standard in the field of 3D graphics, it requires the 3D geometry to have a parametrization (a correspondence between a point on the 3D surface and a point on the 2D plane). This requirement is difficult to fulfill in the case of digital models coming from

***Figure 3**. An example of raw geometry, and of the increased realism obtained through photographic colour mapping.*

3D scanning. These models, because of their geometric complexity and their unstructured nature (they are, basically, just a set of unorganized triangles, often with topological problems), are quite difficult to parametrize, imposing serious limitations on the texture mapping approach, especially for complex 3D models (more than 1 million triangles).

To solve this issue, MeshLab implements a series of algorithms able to generate a texture parametrization for the input 3D models. MeshLab can generate a quite trivial parametrization (each triangle packed on its own in the texture space), usable on simple objects, or an extremely advanced global iso-parametrization, with optimal properties in terms of low distortion and continuity of mapping (see (Pietroni et al. 2010) for more details).

A third parametrization method relies on the use of calibrated images to map a 3D surface to a 2D plane. It works by subdividing the mesh in parts, each one that projects correctly one of the input photos: each of these subparts is then flattened on the texture space by applying the projection associated to the corresponding photographic camera (more details in Section 8).

Finally, a general-purpose, automatic mesh flattening tool has been recently added, able to work on arbitrary 3D models, producing surface parametrizations that are well suited for texture mapping.

Again, the strategy has been to include multiple operations to compute the same kind of data, in order to cope with the specific characteristics of the projects the user has to work with. Every one of the parametrization tools has strengths and weakness, and the user may choose the one that better suits thier needs, without affecting the other stages of the workflow.

7. Weighted Blending

Given a 3D surface and a set of calibrated images, it is possible to project colour information from the images to the surface. However, the hardest problem is that for each point on the surface there are many photos that are possible contributors. Due to lighting inconsistencies , unstable camera settings and sensor noise, the colour from the different sources is never the same, and it is not easy to choose which colour map onto the surface.

While a simple blending would blur the final colour; a better solution is to intelligently blend all contributions according to their quality. This will ensure consistency of colour and preserve the detail of information present in the input photos.

The MeshLab filter implements an improved version of the weighted blending method described in (Callieri et al. 2008), with a more efficient and fast computation, a smaller memory footprint and some tweaking in the

***Figure 4**. Colour mapped using weighted blending with per-vertex encoding.*

weighting functions.

The method can calculate a colour for any point over the surface, by estimating the most correct colour as a weighted mean, which takes in account various quality metrics. Each pixel of each of the input images has a quality value associated to it: this quality value is calculated using multiple metrics, like the distance of the camera to the sampled points (closer is better), the viewing angle (when the camera is more orthogonal to the surface, it is better) and the proximity of the pixel to critical points on the photo (border pixels are bad, and so are pixels close to depth discontinuities). This meaningful and smooth weighting system ensures that weighted blending produces continuous and detailed colour mapping.

This colour mapping technique does work to both generate per-vertex colour (by evaluating the colour blending for each vertex) or to fill an existing parametrization (evaluating the colour blending for each texel). This makes the method flexible, enabling the user to choose the most appropriate output format. This method produces coherent colour, since all redundancy between images is effectively used. However, since blending is involved, a small loss in higher frequency colour details is unavoidable. The method scales well with the size of dataset, both in terms of number of input images, and in terms of the complexity of the 3D model to be mapped.

For these reasons, this method is better suited for datasets composed of a large number of images, and/or complex datasets. In our experience, we were able to map 100+ images over 3D models of 10-20 millions triangles.

8. Photo Stitching

An alternative method to map colour on a triangulated model is to build a texture map, stitching together parts of input photos.

Using the geometry and the aligned photos, it is possible to find the photograph that gives the best colour information for each part of the model. A texture map can be assembled by stitching together sections of the photos, according to this partition.

To this aim, MeshLab does implement the texturing method described in (Callieri et al. 2002), adding various optimizations and new features, like a better quality evaluation when assigning the photos to each part of the mesh, a more stable end efficient texture packing, and a more effective strategy to uniform colour.

The first step is to create a parametrization for the 3D model. The model is analyzed in order to determine which photograph best captures each triangle of the 3D surface. The “best” photograph for each part of the model is determined, again, considering the metrics described for the colour blending process (distance, angle and position in image space). In this way, the model is divided in various patches; pieces of the surface that are well represented in one of the aligned photos.

Then, each patch is projected on the corresponding image (using the camera parameters recovered during photo alignment), and the chosen part of the photo is cut and pasted onto the final texture. A final step aims at smoothing out the lighting inconsistences (which are always present, across sets of photographs), by comparing the borders

***Figure 5**. Colour mapped in a texture using Photo Stitching.*

between patches, and propagating this lighting correction across the image.

The same filter can also be used simply to generate the photo-driven parametrization of a 3D model, which can then later be filled using the weighted blending or by transferring colour information from a per-vertex encoding or another texture parametrization. This method has the advantage of producing sharp colour mapping (since there is no blending between the various photos). However, the method does not scale well with the size of the input dataset: too many photos produce a very fragmented texture, bad for storage, compression and probably with artifacts on borders; a too complex 3D model (more than 2 million triangles) may cause the algorithm to fail. Considering all these facts, this method is more suited to suitations in which there are few, good pictures, and the size of the model is not too large.

9. Colour Editing

MeshLab implements a series of filters to apply colour correction to the whole model, these corrections are similar to the ones found in many image editing tools, and are quite useful to fine-tune the mapped colour. Using simple parameters it is possible to edit the gamma, saturation, contrast, levels, white balance, hue and brightness.

As mentioned above, colour management does change a lot from one software to another. To be able to exchange data between different software tools, conversion is often required from per-vertex colour encoding to texture mapping, texture re-parametrization, and colour data transfer across different models and textures.

MeshLab does provide different filters which can be used to obtain such functionality; by exploiting these operations it is possible to build an effective pipeline of software tools.

Finally, MeshLab includes and implementation of a “painting” filter. This can be used to apply colour to the 3D surface using a paint-like interface. This filter implements basic paintbrush settings, with transparency and hardness, colour noise, clone tool, eyedropper and colour smoothing. It is simple to use for anyone with a little experience with photographic editing tools.

The interface is quite simple, when compared to more complete tools like Z-Brush or DeepPaint 4D, but it is able to work on very large unstructured 3D meshes of many millions of triangles, which is definitely helpful in the CH domain.

This paint feature may be useful in many different situations, like correcting small problems of colour mapping (remove aliasing/ ghosting, correct shadows of highlights, clone out errors in photos projections), annotating a 3D mesh (paint areas, drawing on the 3D surface), or even in helping art historians to reconstruct possible original colour of a damaged artifact.

The reconstruction of a faithful representation of the actual colour of an artifact is often just a first step towards the understanding of the original appearance of the object. Almost always, the current state of an archaeological find is degraded: to propose possible reconstruction of the original colour of

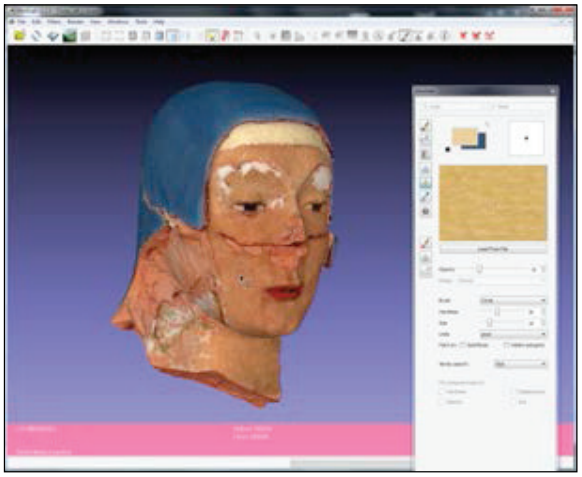

***Figure 6**. The MeshLab painting interface, used in creating a possible reconstruction of the original painted appearance of a statue.*

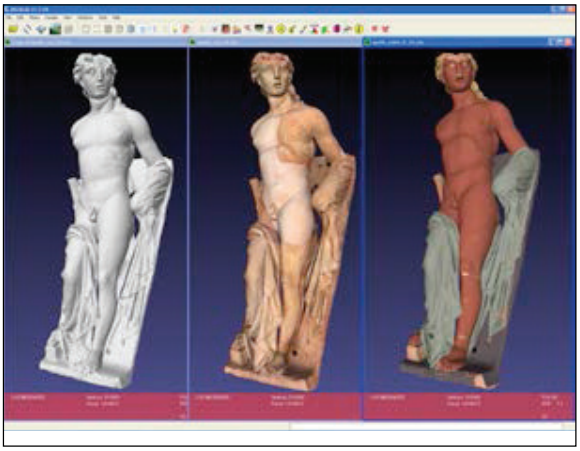

***Figure 7**. An example of mapping of the actual colour and proposed reconstruction of original decoration, a statue of the Luni Temple. On left: raw geometry; Middle: actual colour; Right: reconstructed colour.*

the object is a normal activity for archaeologists and art historians. This task, however, is generally carried out using photographs or drawings of the artifact, while it would be possible to do it directly on the 3D surface.

The Surface Painting tools provided by MeshLab may be used to colour 3D surfaces using paint-like interface. By using this editing tool, it was possible to produce such proposed colour reconstructions, directly on high-resolution 3D geometries.

10. Conclusions

We have described here the new functionalities of the MeshLab platform for the management of colour information for high-resolution 3D models.

MeshLab isa complete open tool for the management of colour information, when working on 3D scanned digital models. Being a open source tool, it is an extremely valuable instrument for the CH field, not because it is free, but also because it gives the user full access to the implementation details (ensuring a good knowledge of data provenance), and supports the use of open data formats.

Our belief is that, in order to offer a usable tool to users, it is important not only to provide a single, integrated tool with all necessary functionalities, but also to make the various functionalities as configurable and customizable as possible. The availability of multiple alternative ways to perform the same step ensures that the user will be able to choose the best solution to cope with the specific characteristics of their projects.

Plans for the future development of the tool will follow the same model of providing “multi-purpose” tool that has been at the core of the philosophy of the project. We will include in the tool other alternative functions to cope with the various steps of the colour processing; image alignment, parametrization and colour mapping. We aim to offer the user a powerful arsenal of tools, sufficient to cover the diverse situations that arise in the processing of 3D models for the CH field.

We will then devote some effort to minimizing the colour mapping artifacts due to small misalignments, and to make the tool even more scalable with respect to large datasets.

References

Bannai, N., A. Agathos, and R. B Fisher. 2004. "Fusing multiple colour images for texturing models." In *Proceedings 2nd International Symposium on 3D Data Processing Visualization and Transmission 2004 3DPVT 2004,* 558-565. Washington: IEEE Computer Society.

Bernardini, F., I. M. Martin, and H. Rushmeier. 2001. "High-quality texture reconstruction from multiple scans." In *IEEE Transactions on Visualization and Computer Graphics*. Washington: IEEE Educational Activities Department.

Brunie, L., S. Lavallée, and R. Szeliski. 1992. "Using Force Fields Derived from 3D Distance Maps for Inferring the Attitude of a 3D Rigid Object." In *Proceedings of the European Conference on Computer Vision*, edited by G. Sandini 588:670-675. London: Springer Verlag.

Callieri, M., P. Cignoni, M. Corsini, and R. Scopigno. 2008. "Masked Photo Blending: mapping dense photographic dataset on high-resolution sampled 3D models." *Computers and Graphics* 32: 464-473.

Callieri, M., P. Cignoni, and R. Scopigno. 2002. "Reconstructing textured meshes from multiple range + rgb maps. State of the Art." In *7th International Fall Workshop on Vision Modeling and Visualization*, edited by Günther Greiner, 419-426. IOS Press.

Cignoni, P., M. Callieri, M. Corsini, M. Dellepiane, F. Ganovelli, and G. Ranzuglia. 2008. "MeshLab : an Open-Source Mesh Processing Tool." In *Sixth Eurographics Italian Chapter Conference*. Geneva: Eurographics Society.

Corsini, M., M. Dellepiane, F. Ponchio, and R. Scopigno. 2009. "Image-to-Geometry Registration: a Mutual Information Method exploiting Illumination-related Geometric Properties." *Computer Graphics Forum* 7 (28): 1755-1764.

Debevec, P., T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. 2000. "Acquiring the Reflectance Field of a Human Face. Ed. Kurt Akeley." In *SIGGRAPH 2000 Proceedings of the 27th annual conference on Computer graphics and interactive techniques,* 145-156, New York: ACM Press/Addison-Wesley Publishing Co.

Dorsey, J., H. E. Rushmeier, and F. X. Sillion. 2006. "Digital modeling of the appearance of materials." *ACM SIGGRAPH 2006 Courses on SIGGRAPH 06*: 1. Accessed 2011. http://portal.acm.org/citation.cfm?doid=1185657.1185721.

Franken, T., M. Dellepiane, F. Ganovelli, P. Cignoni, C. Montani, and R. Scopigno. 2005. "Minimizing user intervention in registering 2D images to 3D models." *The Visual Computer* 21 (8-10): 619-628.

Ikeuchi, K., T. Oishi, J. Takamatsu, R. Sagawa, A. Nakazawa, R. Kurazume, K. Nishino, M. Kamakura, and Y. Okamoto. 2007. "The Great Buddha Project: Digitally Archiving, Restoring, and Analyzing Cultural Heritage Objects." *International Journal of Computer Vision* 75 (1): 189-208.

Lensch, H. P. A., W. Heidrich, and H. P. Seidel. 2000. "Automated Texture Registration and Stitching for Real World Models." In *Proceedings of the Eighth Pacific Conference on Computer Graphics and Applications*, 317-452. Washington: IEEE Computer Society.

Lensch, H., J. Kautz, M. Goesele, W. Heidrich, and H.-P. Seidel. 2001. "Image-Based Reconstruction of Spatially Varying Materials." In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques,* 103-114. London: Springer-Verlag.

Pietroni, N., M. Tarini, and P. Cignoni. 2010. "Almost Isometric Mesh Parameterization through Abstract Domains." *IEEE Transactions on Visualization and Computer Graphics* 4 (16): 621-635.

Pulli, K., H. Abi-Rached, T. Duchamp, L. G Shapiro, and W. Stuetzle. 1998. "Acquisition and Visualization of Coloured 3D Objects." In *Proceedings of Fourteenth International Conference Pattern Recognition,* 11-15. Washington: IEEE Computer Society.

Rankov, V. 2005. "An Algorithm for image stitching and blending." In *Proceedings of SPIE* 5701 *Three-Dimensional and Multidimensional Microscopy: Image Acquisition and Processing XII*, edited by J.-A. Conchello, C. J. Cogswell and T. Wilson, 190-199. San Jose: SPIE.

Sheffer, A., and E. Praun. 2006. “Mesh parameterization methods and their applications.” *Foundation and Trends in Computer Graphics and Vision* 2 (2): 105–171.

Wolberg, G., and S. Zokai. 2006. “Multiview Geometry for Texture Mapping 2D Images Onto 3D Range Data.” In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Volume 2 CVPR06* 2, no. c, 2293-2300. Washington: IEEE Computer Society.