4th International Symposium on Virtual Reality, Archaeology and Intelligent Cultural Heritage (2003)
D. Arnold, A. Chalmers,F. Niccolucci (Editors)

Visualisation of Highly Textured Surfaces

Sabry F. El-Hakim[1], Lorenzo Gonzo[2], Michel Picard[1], Stefano Girardi[2], Andrea Simoni[2], Eric Paquet[1], Herna Viktor[3], Claus Brenner[4]

[1] Visual Information Technology (VIT), National Research Council, Ottawa, Canada
[2] Center for the Technological and Scientific Research, ITC-irst, Trento, Italy
[3]School of Information Technology and Engineering (SITE), University of Ottawa, Canada
[4]Institute for Cartography and Geoinformatics, University of Hannover, Germany

Abstract
We address the main issues in real-time visualization and interactive manipulation of highly textured surfaces. For applications such as cultural heritage, achieving this will give a vista to scientists, conservationists, historians, and visitors for looking at and studying important frescoed surfaces in virtual reality. Since photo-realism is of utmost importance, it became apparent that many problems related to geometric and radiometric distortions must be solved. As a case study, we present results of modelling and visualisation of "Cycle of the Month", one of the major masterpieces of international Gothic art which is located in the Aquila tower in Buonconsiglio castle, Trento, Italy.

Keywords:
Visualisation. Rendering, Real-time, Texture, Photo-realism

1. Introduction

The ultimate goal of this project is to create extremely realistic visual experiences. This interprets to two conceptions: a photo-realistic walkthrough movie and a 3D model that allows the user interactive visualisation and manipulation. For our case study, the "Cycle of the Months" frescos, texture is the most important part of the model and any reduction in quality can affect scene comprehension. In addition, user interactivity is important to the understanding of the scene, so a drop in frame rate is unacceptable. In this first phase of the project, we selected the most appropriate technology and identified the main factors affecting the visual quality of the models and the performance of interactive visualization. When possible, we will use existing software and hardware to accomplish our objectives. However, problems that have no ready solution will be tackled with new algorithms and specially developed tools.

The first step is to create a geometrically-correct 3D model of the room. We considered three alternatives for 3D scene reconstruction:

1. Image-based rendering (IBR): Although this does not include creating a 3D geometric model [Kang, 1999], it may be considered since the room is simple and has no other objects but the walls and the ceiling. This requires taking overlapping images from the middle of the room. IBR is basically restricted to static scenes with static illumination, though some papers make suggestions on how to work around this problem. A second problem associated with certain techniques is that the movement of the viewpoint is confined to particular locations. In addition, without a geometric model there is the possibility of perspective distortion.

2. Image-based modelling: These techniques are suitable for regular geometric surfaces like architecture and monuments [Debevec et al, 1996, El-Hakim, 2002]. Photogrammetric methods can achieve high geometric accuracy, albeit without capturing all fine geometric details. The geometry of the room is a typical candidate for this approach. The same images used to construct the geometry are also used for the texture.

3. Range-based modelling: These, mainly laser scanning, are capable of capturing most geometric details, but the accuracy varies notably from one scanner to another. Having a scanner that can achieve reasonable accuracy for our room size may be difficult since most commercial scanners are either designed for long ranges (over 10 meters) or close ranges (less than 3 meters). Besides, the walls are mostly flat thus there are no fine details we need to capture by a scanner. Regardless of which scanner to use, we should separate acquiring geometry from acquiring texture. Scanners with integrated texture or colour acquisition will not provide the quality required for the detailed frescos in a room of this size. Separate high-resolution digital images should be taken and properly calibrated and registered with the geometric data [Beraldin et al, 2002].

We collected the data with digital cameras and a laser scanner however a detailed comparison between all methods using this data will be the subject of a future paper. In this paper, we focus on the image-based modelling approach as the most appropriate technique. We first provide an overview of the main issues associated with the modelling and real-time rendering of highly textured real environments, and the main techniques currently used to address these challenges. We then give some details on the approach we employ to achieve photo-realism and interactive visualisation. Results of the "Cycle of the Months" modelling and visualisation will be presented as a case study.

2. Issues in realism and visualisation

We divide the problems into those affecting visual quality or photo-realism (photo-realism means no difference between the rendered model and a photograph taken from the same viewpoint), and those affecting manipulation or interactive visualization (visualization here will always mean interactive visualization). Obviously the purpose of visualization is to provide smooth navigation through the model. This should be at a rate of at least 20 frames-per-second otherwise human brain can detect latency or jitter, also loss of interactivity may occur. For most cases with current hardware, it is usually impossible to achieve both photo-realism and smooth navigation without some compromise. In fact, for most detailed models, the state of the technology is such that neither is fully achievable yet.

2.1. Photo-realism

The issue of acquiring and constructing high-quality texture maps has received less attention than the issue of capturing high-quality geometry [Bernardini et al, 2002]. Traditional texture mapping techniques, which are simply draping static imagery over geometry, are not sufficient to represent reality. In the following sections, we will discuss factors affecting photo-realism and possible solutions to reduce their effect.

2.1.1. Geometric distortions: Due to the highly detailed contents of the frescos, it is crucial to preserve every line or edge and all minute painting details. Such textures on the models can be visibly affected by any small geometric errors. Sources of these errors can be divided into three categories:

1. Mapping between triangle plane and image plane.

2. Camera calibration and image registration.

3. The assumptions made in surface reconstruction and the simplification of shapes by triangles.

The correct mapping between the triangle plane and the image plane is given by a projective transform. Since most viewers do not use this transform, distortion arises especially for large triangles. For example the projection of a straight line inside the triangle may become crooked. In addition, after the correct mapping is computed and the texture is warped, the image must be re-sampled or filtered. The easiest texture filtering method is point sampling, wherein the texture value nearest the desired sample point is used. But in general, this results in strong aliasing artifacts. To get rid of those artifacts, sophisticated filtering methods are needed such as space variant filters whose shape varies as they move across the image.

When cameras with standard lenses are used, lens distortion corrections must be applied else geometric distortions, like line discontinuity, will be visible at common edges of adjacent triangles mapped from different images. It is therefore important that full camera calibration is applied and that it remains valid for all images by keeping camera settings constant or applying self-calibration. Accurate image registration, mainly photogrammetric bundle adjustment, must also be implemented to minimise geometric distortions.

Too large deviations of the triangle mesh from the underlying true object surface give rise to geometric errors. Since a triangle represents a plane, applying too large triangles on even slightly curved surfaces will results in features near the edges projected on the wrong surface or disappear altogether. This has been witnessed frequently with the frescos and required adding more 3D points to the surfaces at appropriately selected locations.

2.1.2. Radiometric distortions: These usually result along common edges of adjacent triangles mapped from different images. The main reasons for distortions are:

1. Different sensed brightness due to different camera positions and/or change in illumination.

2. Non-linearity of the image response function.

A typical system captures a collection of images containing particular invariant lighting conditions at the time of imaging. Variation in the sensed brightness between images will result when taking images from different locations and at varying ambient lighting. When these images are stitched together, discontinuity artifacts are usually visible.

The digitised brightness value for a pixel is not a measure of scene radiance (radiance is the amount of light energy reflected from a surface in all directions). There is an unknown non-linear mapping between the pixel value and the scene radiance. This non-linearity is largest near the saturated part of the response curve. Knowing the response function allows merging images taken at different exposure setting, different angles, or different imaging devices. The inverse of the response

function is used to recover the scene radiance, up to a scale [Debevec and Malik, 1997]. The response function can be applied to any image taken by the imaging system, therefore, we can determine the function for a given camera in a controlled environment as a calibration process then apply it to the project images.

Blending methods to create seamless texture maps have been developed [Bernardini et al 2001, Pulli et al, 1998, Rocchini et al, 2002, Wang et al, 2001]. Those techniques use the weighted average of pixel values of triangles in the overlapping sections between two adjacent textures. For example, Pulli et al, 1998, used best 3 views to calculate new texture image. The colours of the compatible rays are averaged together via a weighting scheme that uses three different weights: directional weight, sampling quality weight, and feathering weight. The task of the directional weight is to favour rays originating from views whose viewing direction is close to that of the virtual camera. Not only should the weight increase as the current viewing direction moves closer, but the other views' weights should decrease. The sampling quality weight directly reflects how well a ray samples the surface. Their algorithm assigns to each ray/pixel of each input image a weight that is defined as the cosine of the angle between the local surface normal and the direction from the surface point to the sensor. The feathering weight is used mostly to hide artifacts due to differences in lighting among the input images. Guidon, 1997, achieved seamless radiometric continuity by radiometric normalization. He combines scene-wide statistical normalization (matching scene grey-level means and variances) with local operations such as feathering or blending in overlapped regions. Blending techniques will not always work in our application since some triangles can be too large to have textures from several images.

When the lighting in the virtual environment where the texture map is used differs from the lighting under which the texture map was captured the resulting rendering will appear unrealistic without light-dependent texture mapping [Gelb et al, 2001]. However, this method adds considerably to rendering processing time since the system repeatedly computes light directions using triangles normal.

In the planner walls even when proper lighting is implemented, the rendering may look too smooth and unrealistically flat. An improvement can be achieved by applying bump mapping. This is done by introducing small variations in surface normals thus adding roughness and more realistic look to the lighted surface [Rushmeier 1997], without the need to model each wrinkle as a separate element.

2.1.3. Aliasing, jagged lines, and other effects: When textured models are viewed from a long distance, aliasing, moiré effect, jagged lines, and other visible abnormalities may result. This is because when the surface is far away, a pixel on that surface may be associated with several texels from the original texture map. MIP-maps (MIP stands for 'multum in parvo'), a sequence of textures each of which is progressively lower resolution (by a factor of 2) of the original image, is commonly used to solve this problem [Williams, 1983]. The most appropriate texture is selected depending on the polygon size or distance. Bilinear or tri-linear filtering (to remove aliasing) and texture memory caching are also needed. Since real-time re-sampling and filtering can be computationally expensive, the required filtered representations of the original texture may be pre-computed and stored. This will increase the total size of texture by a maximum of one third. Although most recent hardware supports MIP-mapping operations, the actual managing and set up of caching scheme is up to the application.

2.1.4. Dynamic Range: The dynamic range of a scene (the ratio between its brightest and darkest parts) is usually much higher than the inherit limit in the digital image dynamic range (8-bits per colour channel). This results in the flattening of the response curve (saturation) at the bright areas and the inability to resolve details in the dark areas. A high dynamic range (HDR) image can be assembled from multiple low dynamic range images on the scene taken at different exposure settings [Debevec and Malik, 1997].

2.2. Interactive visualisation

Due to the fact that this project focuses on high quality textures, the texture size could end up being quite large. The rendering algorithm should be capable of delivering images at interactive frame rates, even for very large textures. We now discuss the main factors affecting the performance of a rendering system.

2.2.1. Hardware: Specifications affecting processing speed are: number of geometric transformations that can be performed per second; the number of triangles that can be rendered with texture per second; the available main and texture memory; and the disk and network access speed (bandwidth). Memory limitations become the main issue for large textures. Some of these hardware specifications will affect only start up time. For example significant computations are initially needed in order for the rendering software to build an internal representation of the scene graph. Also high bandwidth is crucial to load the model and texture in memory at reasonable time at start up.

2.2.2. Size of the geometric model: Most existing techniques for real-time rendering [Baxter et al, 2002, Cohen-Or et al, 2003] focus on the geometry (reducing

the number of triangles) rather than the texture. For example, hierarchical levels of detail (LOD), where objects far away from the viewpoint are rendered with pre-sampled lower resolution representations, is a standard technique that is supported by most scene graph libraries. No further details will be given here since typically in image-based modelling it is not the geometry that affects real-time rendering but the texture size when using high-resolution images.

2.2.3. Texture size: For a project like this, the total texture size will likely be several hundreds of mega bytes. If the on-board texture memory is less than the active texture (the texture that must be in texture memory), swapping will occur resulting in noticeable performance degradation. Fortunately it is rare that all the textures are needed for one frame. To take advantage of this, two techniques are suitable in our case: view frustum culling, which skips rendering and loading into memory parts of the model that are outside the view frustum; and MIP-mapping, which was originally developed to reduce aliasing as discussed in section 2.1. If all the texture fit into texture memory then there is usually no problem, else the viewing application must work out a texture caching routine that decide on which texture to keep or move into memory and which to move to RAM, or even to the system virtual memory [Cline and Egbert, 1998 and Dumont et al, 2001]. To make texture caching more efficient it is recommended to divide the scene into smaller uniform groups, each with a small texture with width and height to be a power of two. This makes it possible to control the file size and the rendering speed more precisely.

Another effective performance enhancement technique for both geometry and texture is occlusion culling, which skips objects that are occluded by other objects. This is not needed in our room model since there are no occluding objects.

3. Modelling and rendering procedure

Here we summarise the modelling and rendering approaches used for this project as well as the employed software tools. We generally perform the following steps:

- Create geometrically accurate models with photogrammetric modelling.
- Divide the model into optimum, or efficient size, groups of triangles.
- Select best image for each group.
- Compute texture coordinates of vertices using internal and external camera parameters.
- Compute radiance images of all images from the calibrated response curve.
- Re-sample and filter pixels within each triangles based on projective transformation between the triangle plane and the image plane.
- Apply radiometric corrections to the textures.
- Create MIP maps.
- Create bump maps.
- Render with efficient rendering software.

Some of these operations have already been described above or are self-explanatory. Those requiring more details are given in the following sections.

3.1. Image-Based Modelling

The image-based modelling approach used for this project is the semi-automatic technique described in El-Hakim, 2002, and partially implemented in the ShapeCapture© commercial software [www.shapecapture.com]. Version 4 has all the tools we needed and was used to create each individual model. The bundle adjustment in the software was done with free network, and model scales were determined from a small number of linear measurements (such as window dimensions) collected while taking the images.

3.2. Texture Mapping and Texture Corrections

Some corrections to texture maps, mentioned later in this section, are done with specially developed software. We also use HDR Shop from Paul Debevec [www.debevec.org] to determine the response function of the digital camera.

Since the room shape is rather simple, and all the geometry is properly modelled, there is no need for view-dependent texture mapping. Also since the room lighting dynamic range is not high (except at the windows, however we are not interested in the scene through the windows), we decided to use standard images digitised at 10-bits per colour with the digital camera, or 14-bits with a scanner, rather than high dynamic range images. In the future, the effect of using HDR images will be studied.

3.2.1. Selecting most appropriate image to texture: With sufficient image overlap, the texture of a triangle can be obtained from a number of different images. A reasonable choice is to select the image in which the triangle appears largest and taken along the normal. However, this may result in having too many different images assigned to adjacent triangles. One method to repair this is to locally re-assign triangle

patches to images. In this step, the initial assignment is changed based on the image assignment of the adjacent triangles and the image area covered by the triangle in alternative images. In effect, local re-assignment generates larger groups of triangles mapped from the same image and eliminates isolated triangle texture mappings. Thus, the number of triangle edges where adjacent triangles are mapped from different images is reduced (Figure 1). Another method of texture selection to reduce texture discontinuities is to use texture blending. In blending, the mapping algorithm does not try to find the best image for each triangle but rather computes the texture from all images the triangle appears in by forming a weighted average. While blending is an algorithmically simple approach that diminishes geometric and radiometric discontinuities, it must be noted that it usually introduces a detectable blurring effect. It has to be decided from case to case if global and local gray-value adaptation, which produce sharper texture maps but might show geometric artifacts at adjacent triangles, or texture blending, which reduces artifacts at adjacent triangles but tends to blur the textures, is the better choice.

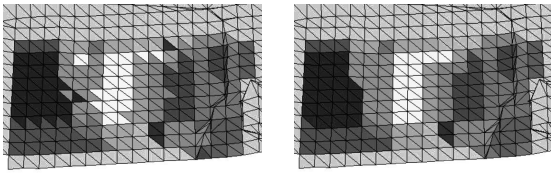

Figure 1: *Triangle shades in the mesh correspond to images from which the texture is obtained. Left: before local re-assignment. Right: after local re-assignment.*

3.2.2. Proper Geometric Fit: The employed method defines a local texel coordinate system for each 3D triangle. The texel size, in object coordinates, can be set to the desired resolution. Projective transformation between the plane of the triangle and image plane is determined and used to re-sample the texture within each triangle. This is followed by low pass filter to remove high frequency noise introduced by this re-sampling. As mentioned in section 2.1.1 full lens distortion [3 radial and 2 decentric parameters, e.g. Faig, 1975] is used for all texture mapping transformations.

3.2.3. Radiometric Differences: Our approach for radiometric texture corrections has been originally developed in El-Hakim et al, 1998. We address this problem by a correction method termed "global grey-value adaptation" and another correction on a per-triangle basis, termed "local grey-value adaptation". Here, grey value means colour channel value (red, green, or blue). The global grey-value adaptation estimates grey-value offsets between images. The grey-value differences along the border of adjacent regions of triangle sets are minimized by least-squares adjustment. The adjustment is much like in the case of a geodetic height network. Here the observed height differences are replaced with grey-value differences along region borders. Additional weighted observations of type $h_i = 0$ ensure non-singularity and prevent the offset from drifting away across larger scenes. The local grey-value adaptation modifies the grey-values of each triangle to ensure smooth transitions to all adjacent triangles. However, this is not straightforward since if we observe offset o_1 along triangle edge e_1 and o_2 along e_2 it is unclear how to correct the grey-values in the vicinity of the triangle vertex where e_1 and e_2 intersect. Thus, we have adopted a technique based on iterative procedure. In order to force a gradual change to grey-values within a triangle, we fit a plane to the grey-value offsets observed at the triangle borders. The plane parameters are determined by a least-squares adjustment that minimizes these differences. After correcting the grey-values according to the plane parameters, this process is iterated several times. Usually, there are no discernible changes after a few iterations.

Another method to reduce texture discontinuities is to use texture blending instead of the adaptations. In this case, the mapping algorithm does not try to find the best image for each triangle but rather computes the texture from all images the triangle appears in using a weighted average. While blending is an algorithmically simple approach that reduces geometric as well as radiometric discontinuities, it may introduce a detectable blurring effect. It has to be decided from case to case if global and local grey-value adaptation, which produces sharper texture maps but does not completely correct artifacts at adjacent triangles, or texture blending, which better reduces artifacts at adjacent triangles but tends to blur the textures, is the better choice.

3.3. Rendering

In order to achieve the desired performance, a special software for interactive visualisation, named “Il Teatro Virtuale©” has been developed at VIT [Paquet and Peters, 2002] using scene graph tools. Those tools are data structures used to hierarchically organize and manage the contents of spatially oriented scene data. They offer a high-level alternative to low-level graphics rendering APIs such as OpenGL. Examples of scene graph supported languages are VRML, Java 3D, MPEG-4, and of course SGI Open Inventor which is the earliest of those. Our PC-based loader is developed with Java 3D which gives complete control over the rendering process (unlike other languages). Details are given next.

4. Distributed virtual environments for visualisation

Visualisation is of tremendous importance for heritage applications. For instance, in order to study and analyse a virtualised site, like the room of the Cycle of

the Months in the Aquila tower in Buonconsiglio Castle, an adapted virtual environment is required. Most of the solutions that are currently proposed are characterized by their complexity, cost and size. This is due to the fact that low-end visualisation systems usually present poor performances that are incompatible with most applications. A mobile, high-performance, cost effective virtual environment for visualisation is proposed. A conceptual and technical description is provided. The design is considered from various aspects: photorealism, the scene's size scalability, the display's size scalability, resolution and refresh rate; the user's visual comfort and intellectual property protection.

In order to implement our virtual environment, we describe a system based on client-server architecture in which each computer is responsible for performing the rendering for a determined projector. The synchronization of the various displays is obtained by continuously sending requests from the clients to the server, in order to update their information about the input devices for which the server is responsible.

Let us considerer the system from a server perspective. The synchronization is performed within the so-called event loop. The event loop is the loop in which the input events are collected from various input devices. Among them are the mouse, trackers, haptic sensors and so on. The events from the input device are collected at constant rate and converted into virtual environments coordinates. Let us refer to them the transformation. Those coordinates or transformation correspond to the virtual coordinates utilised in the virtual scene and depends on various calibration parameters, including the dimensions of the visual display and the size of the scene. The transformation are then compressed and put in a data structure. Upon completion of the previous task, the structure is sent to the server, which makes it available to the clients. Each time there is a request from one of the clients, the server sends the structure to the corresponding client. The server has a multithread architecture: a thread is associated to each client. It has to be noticed that the events loop is totally independent from the rendering loop. The events loop affects the rate at which the navigation and manipulation information can be updated, but not the rate at which the graphical calculations are effectively performed.

Now let us consider the system from the client perspective. At the commencement of each events loop, the client sends a request to the server in order to obtain the transformation. After the request has been sent, the events loop is paused. One should remember that the server manages the input events. Upon reception of the data structure from the server, the structure is uncompressed, the client events' loop is resumed and the rendering is performed with the updated information.

Let us take a closer look at synchronization. In order to have acceptable visual performances, the refresh rate must be over a certain rate. This rate is function of the retinal persistence and has a value of around 30 frames/sec. Higher rates improve the visual quality and the overall usability of the virtual environment. Below the minimum rate, the display might not be comfortable and cause the well-known VR-sickness, which is similar from many aspects to travelling sickness. Sockets' performances determine the time involves in telecommunication. Since the client-server architecture is based on the TCP/IP protocol, the time involved in a request is about 5 msec. All clients are synchronized relatively to the server, which means that the relative delay between them is around 10 msec. That means that it is possible to render around 100 frames/sec within the proposed virtual environment. In practice, the graphic load and the performances of the graphic card limit the frame rate. The graphic load is tributary of the number of polygons composing the scenes, the number of groups of polygons and the number and dimensions of the texture maps. But because of its distributed architecture, our system spread the graphical load between the computers. Each computer renders a subset of the overall scene, which means that better performances can be achieved both from the resolution and frame rate point of view.

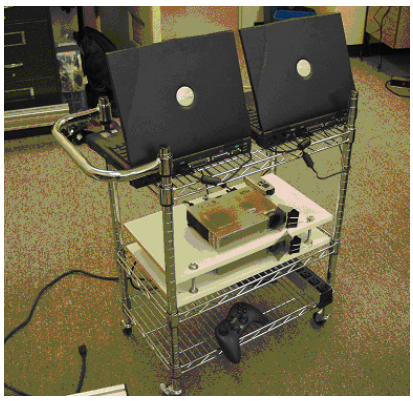

Figure 2: *Set-up for our mobile virtual environment. From top to bottom: the two video projectors with their polarized filters and the two laptops. The computers are interconnected with a twisted wire.*

Our stereo virtual environment is based on passive polarization. Two circularly polarized video projectors simultaneously project the left and right eye view. The left eye view is encoded with a circular left polarization while the right eye view is encoded with a circular right polarization. In order to experience stereo vision, users must wear polarized glasses. Circularly polarized filters performances do not degrade with head lateral inclination, which is why it has been selected.

Our projectors are based on Texas Instrument's ™ Digital Micromirror Device or DMD [www.dlp.com]. The DMD is a semiconductor light switch made of thousands of microscopic mirrors. DMD projectors preserve polarization and thus have been selected for our system. In order to obtain depth perception, the output of the two projectors must be projected on a surface that preserves the polarization state. For front-projection, a standard metallic screen is utilised. For back-projection, a specialised screen is required but is commercially available. Front-projection is a cheaper option but back-projection offers more possibilities both from the visualisation and interaction point of view.

A transformation is applied to the client VE coordinates in order to emulate the disparity between the left and the right eye view. Because our virtual environment distributes the graphical load between two computers, high refresh rate and resolution can be achieved. Instead of rendering alternatively the right and the left eye view as in an active stereo system, our virtual environment renders them simultaneously on two separate computers. In addition, views are projected simultaneously on the visual, which means that the bandwith is almost quadruple. The whole system has been implemented and deployed: the projection system and the visual processing units can be seen in figure 2.

5. Results: Walkthrough movie of the "Cycle of the Month"

One of the objectives of the project is to create a photo-realistic walkthrough movie of the "Cycle of the Months" frescos. In order to portray the room in the proper context, the movie also includes the outside view of the Aquila tower and the entrance to the room. Therefore, we created the following 3D models:

The outside walls of the tower including part of the long walkway leading to the room.

- The entrance to the room from the walkway.
- The walls of the room.
- The ceiling of the room.

5.1. Design considerations and data collection

One of the decisions that had to be made is whether to use film camera then digitise the film or directly use a digital camera. Although the film produces better quality and dynamic range than digital camera, it still needs to be digitised with a scanner and thus degradations and non-linearity are introduced. We used both: 5 Mega-pixels (1920x2560 pixels) digital camera; and a metric film camera and digitised the images at 2256x2900 pixels resolution. The results shown here are with digital camera only. Comparison between the two, particularly regarding the texture quality, is in progress.

One of the restrictions was the size of the room [7.75 m x 5.95 m x 5.35 m (H)], which resulted in a small coverage per image. This required taking 16 images to cover the walls and 8 for the ceiling. This could take a rather long time during which lighting conditions may change and result in inconsistent illumination. For the outer tower model and the entrance to the room through the walkway, 5 and 2 images were taken respectively. This resulted in a total image size of 472 Mega-bytes. After cropping the images to the size of the triangle groups, this was reduced to about one third. The average texture resolution was 2 mm x2 mm area of the wall per pixel. Figure 3 shows a top view of the room with camera locations for wall images (white solid circles) and ceiling images (grey solid squares).

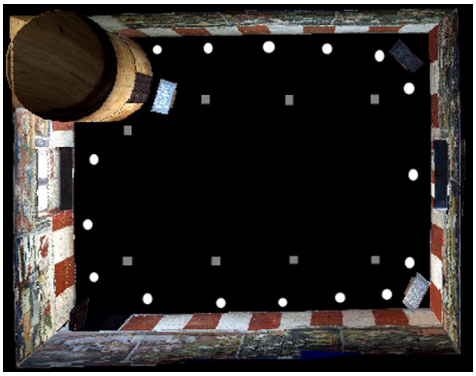

Figure 3: *Room top view with camera locations.*

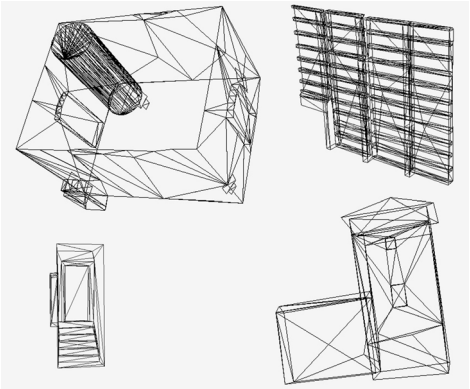

Figure 4: *Wire-frame models of the different parts (clockwise: walls, ceiling, tower, and entrance)*

5.2. Geometric modelling and basic texturing results

Results of the geometric modelling are shown in figure 4 in wire-frame. Mesh size was small (10,000 triangle), but texture size was about 160 Mega-bytes.

After texture mapping, sometimes we needed to reconstruct the model with extra points to account for small deviations from plane for some walls. For example

in the corner shown in figure 5, we had to increase the number of points near the intersection between the two walls. When only using a small number of points and assuming planes in areas without points, the lines along the corner did not project on the correct wall and lines that span the two walls had visible discontinuity. This shows that any small geometrical error will be visible in this class of models with frescoed walls. In most other types of architecture, where walls have uniform colours, such errors may be tolerated.

Figure 5: *Corner where geometric accuracy is critical.*

Without applying any texture correction, we notice in the corner where one wall texture was taken from the first image and the other wall texture was taken from the last image (the 16th) that there is a visible illumination difference (figure 6). This is because the difference in time between the two images was about 30 minutes, which was enough for the lighting conditions to change even though we used a flash. Corrections to this problem was done with the technique presented in section 3.2.3.

Figure 6: *Lighting variations, first and last image*

5.3. Creation Of the movie

The three interior models (the walls, the ceiling, and the entrance) were registered in the same coordinate system using common points. The outside model of the tower did not need to be registered with the others since in the movie there is no direct continuation between them (the movie fades from the outside scenes into the inside scenes). Images of the tower showing the surroundings were also used. Everything was put together in 3ds Max© and Adobe Premier© software. A few trials were necessary in order to get the proper viewing angle of the virtual camera, the path of the camera, and the rendering speed. Not all the radiometric texture corrections mentioned in section 2 were necessary for the movie (they will all be needed for the interactive model). This is because many of the artefacts were not visible from the virtual camera viewing range used for the movie (figure 7). However, all the geometric corrections had to be applied.

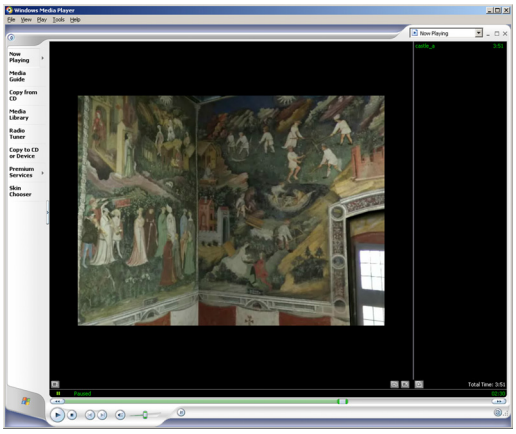

Figure 7: *A snapshot from the movie.*

6. Conclusions and future work

Although current results from image-based modelling alone are promising, the use of laser scanning combined with high-resolution texture maps is being implemented. We would like to investigate whether the scanner can capture finer details, such as small curvatures and non-straight edges, and whether this gives more realistic experience. The current model works in real-time (30 frames / sec.) on a 2.4-GH PC with 128 MB texture memory and 1GB RAM using our rendering software. However, in this phase we have not used HDR images or image-based lighting techniques. In the future these techniques will be applied and their effect on realism and performance will be examined.

7. Acknowledgements

This project is funded by the “Provinicia Autonoma di Trento”. The Castello del Buonconsiglio gave us access to the room during non-visiting hours.

References

1. Baxter III, W.V., Sud, A., Govindaraju, N.K., Manocha, D., 2002. GigaWalk: Interactive walkthrough of complex environments. University of North Carolina at Chapel Hill Technical Report TR02-013.

2. Beraldin, J.-A., Picard, M., El-Hakim, S., Godin, G., Latouche, C., Valzano, V., Bandiera, A., 2002. Exploring a Byzantine crypt through a high-resolution texture mapped 3D model: combining range data and photogrammetry. In Proc. ISPRS/CIPA Int. Workshop Scanning for Cultural Heritage Recording. Corfu, Greece, pp. 65-72.

3. Bernardini, F., Martin, I. M., Rushmeier, H., 2001. High-quality texture reconstruction from multiple scans. *IEEE Trans. Visualization and Computer Graphics*, 7(4), pp. 318-332.

4. Cline, D., Egbert, P.K., 1998. Interactive display of very large textures. Proc. IEEE Visualization'98, pp. 343-350.

5. Cohen-Or, D., Chrysanthou, Y., Silva, C.T., Durand, F., 2003. A survey of visibility for walkthrough applications. *IEEE Trans. Visualization and Computer Graphics*, to appear.

6. Debevec, P., C.J. Taylor, J. Malik, 1996. Modelling and rendering architecture from photographs: A hybrid geometry and image-based approach. SIGGRAPH'96, pp. 11–20.

7. Debevec, P.E., Malik, J., 1997. Recovering high dynamic range radiance maps from Photographs. Proc. SIGGRAPH'97, pp. 369-378.

8. Digital Light Processing http://www.dlp.com

9. Dumont, R., Pellacini, F., Ferwerda, J.A., 2001. A perceptually-based texture caching algorithm for hardware-based rendering. Proc. 12th Eurographics Workshop on Rendering, pp. 246-256.

10. El-Hakim, S.F., Brenner, C., Roth, G., 1998. A multi-sensor approach to creating accurate virtual environments, *ISPRS J. for Photogrammetry & Remote Sensing*, 53(6), pp. 379-391.

11. El-Hakim, S.F., 2002. Semi-automatic 3D reconstruction of occluded and unmarked surfaces from widely separated views. Proc. ISPSRS Comm. V Sym., Corfu, Greece, pp. 143-148.

12. Faig, W., 1975, Calibration of close-range photogrammetric systems. *Photogrammetric Engineering & Remote Sensing*, 41(12), pp. 1479-1486.

13. Gelb, D., Malzbender, T., Wu, K., 2001, Light-dependent texture mapping. HP Labs Tech Report: HPL-98-131(R.1).

14. Guindon, B., 1997. Assessing the radiometric fidelity of high resolution satellite image mosaics. *ISPRS J. Photogrammetry & Remote Sensing*. 52(5), October, pp. 229-243.

15. Java 2 / Java 3D / JCE: http://www.javasoft.com

16. Kang, S.B., 1999. Survey of image-based rendering techniques. SPIE Vol. 3641, Videometrics VI, pp. 2-16.

17. Multiple authors, 2000. Special Issue on Large Wall Displays, *IEEE Computer Graphics and Applications*, 20 (4).

18. Paquet, E., Peters, S., 2002. Collaborative Virtual Environments Infrastructure for E-Business. Proc. Int. Conference on Infrastructure for e-Business, e-Education, e-Science and e-Medicine on the Internet - SSGRRw02, January 21-27, L'Aquila, Italy, CD.

19. Pulli, K., Abi-Rached, H., Duchamp, T., Shapiro, L.G., Stuetzle, W., 1998. Acquisition and visualization of colored 3-D objects. Proc. Int. Conf. Pattern Recognition, pp. 99-108.

20. Rocchini. C., Cignoni, P., Montani, C., Scopigno, R., 2002. Acquiring, stitching and blending diffuse appearance attributes on 3D models. *The Visual Computer*, 18, pp. 186-204.

21. Rushmeier, H., Taubin, G., Gueziec, A., 1997. Applying shape from lighting variation to bump map capture. Proc. Eurographics Rendering Workshop, pp. 35-44.

22. Singhal, S., et al., 1999. Networked Virtual Environments: Design and Implementation, Addison-Wesley Pub Co.

23. Wang, L., Kang, S.B., Szeliski, R., Shum, H.-Y., 2001. Optimal texture map reconstruction from multiple views, Proc. Computer Vision and Pattern Recognition (CVPR01).

24. Williams, L., 1983. Pyramidal Parametrics. *Computer Graphics (Proc. SIGGRAPH '83*), volume 17, July, pp. 1–11.