INDIGO

Crisis Management Solutions

I4.4.1 – STATE OF THE ART REPORT ON MASSIVE GEOSPATIAL 3D DATASETS

Contractual Delivery Date: 08/2010	**Actual Delivery Date**: 09/2010
Nature: Report	**Version**: 1.0

Confidential Deliverable

Abstract

This deliverable relates to the presentation and comparative evaluation of the enabling technology for the efficient rendering of geo-spatial 3D data on commodity and distributed platforms, with a special emphasis on 3D technologies used to support tools for visual help in navigation and location awareness tasks. This document is intended to be included as annex in the INDIGO specification document.

Preparation Slip

	Name	Company	Date	Signature
From	E. Gobbetti	CRS4	27/08/2010	
Approved by				
For delivery				

Document Log			
Issue	Date	Comment	Author
V1.0	27/08/2010	First version	P. Cignoni (CNR) F. Ganovelli (CNR) E. Gobbetti (CRS4) F. Marton (CRS4) R. Pintus (CRS4) R. Scopigno (CNR)

Document Change Record		
Issue	Item	Reason for Change

Content

CONTENT **3**
1 EXECUTIVE SUMMARY **4**
1.1 Document Purpose 4
1.2 Document Structure 4
2 INTRODUCTION **5**
3 ACQUIRING 3D SHAPE AND COLOR INFORMATION: A SHORT OVERVIEW **6**
3.1 Data acquisition 6
3.1.1 Passive systems 6
3.1.2 Active systems 7
3.2 Modeling: generating dense 3D models 9
3.3 Modeling: Structure from motion for coarse 3D with registered images 14
3.4 Recommendations 15
4 NAVIGATING THROUGH DENSE 3D DATASETS: TECHNICAL STRATEGIES FOR MASSIVE MODEL RENDERING **16**
4.1 Visibility Culling 16
4.1.1 Object space subdivision 16
4.1.2 From-point visibility culling 17
4.1.3 From-region visibility culling 18
4.2 Simplification and level of details 19
4.2.1 Geometric simplification 19
4.2.2 Level of detail 20
4.3 Alternate rendering primitives 21
4.3.1 Higher order primitives 21
4.3.2 Sample based representations 22
4.3.3 Image based approaches 23
4.4 Recommendations 24
5 NAVIGATING WITHIN SPARSE DATASETS: PHOTOGRAPHS EMBEDDED IN 3D SPACE **25**
5.1 Photo Tourism 25
5.1.1 Transition between photographs 25
5.1.2 Selection of photographs 26
5.1.3 3D visualizations 26
5.2 PhotoCloud 27
5.3 Visualizing large sets of photographs 28
5.4 Considerations 29
5.5 Recommendations 29
6 CONCLUSIONS AND GENERAL RECOMMENDATIONS **30**
7 BIBLIOGRAPHY **31**

1 Executive Summary

1.1 Document Purpose

This Interim Report has been prepared in partial fulfilment of Internal Deliverable 4.4.1 required for completion of Task 4.4 (Research and Technology Monitoring) of the EU Seventh Framework Program INDIGO - Innovative Training and Decision Support for Emergency Operations (ICT-242341-INDIGO). The report relates to the presentation and comparative evaluation of the enabling technology for the efficient rendering of geo-spatial 3D data on commodity and distributed platforms, with a special emphasis technologies used to support tools for visual help in navigation and location awareness tasks. This document is intended to be included as annex in the INDIGO specification document.. This document is intended to be included as annex in the INDIGO specification document.

Following the project design guidelines, (see project contract, Annex I), this document will possibly be iteratively refined during project evolution to reflect the improved understanding of end-user needs and their technological implications.

1.2 Document Structure

The document is divided into the following main sections:

- Section 2 provides a general introduction to the document;
- Section 3 provides a very rapid overview of the technologies than can be used for acquiring 3D and Visual Information on a particular environment;
- Section 4 focuses on general technical strategies for massive model rendering, with a particular emphasis on methods for reducing the complexity of scenes on a frame by frame basis. The main techniques used to achieve this goal consists of: (a) rendering only the polygons that are determined visible (*Visibility Culling*); (b) using geometric approximations of the original model with lower polygon count (*Levels of Detail*); (c) using alternate representations in place of polygons (*Sample-based representations* and *Higher-order primitives*).
- Section 5 rapidly presents strategies that can be employed for navigating within sparse datasets created from photographs. Since the encoding of images is already standardized, the focus of this section will be on the design of navigation techniques.
- Section 6 concludes and provides general recommendations for the project.

The document concludes with a bibliography of cited reference works.

2 Introduction

The INDIGO project aims to research, develop and validate an innovative system integrating the latest advances in Virtual Reality and Simulation in order to homogenise and enhance both the operational preparedness and the management of an actual complex crisis.

One of the options that will be researched is the helping of navigation/location awareness in complex environments through the exploitation of 3D or near-3D data. The basic idea behind the approach is that 3D data of some form (shapes of buildings/environments) is already widely available, and will become more and more common in the future due to the improvement and reduction in cost of 3D acquisition technologies. With these information, that can go from complete 3D reconstruction of sites, e.g., as those acquired by aerial or terrestrial 3D laser scanners, to 3D calibrated photographs, that can be acquired by digital cameras, it will be possible to present to users an easy to understand depiction of a natural/built environment. This information can then be used in applications to create navigation tools that will complement traditional 2D maps in a number of tasks.

In this report, we provide a short overview of the technology behind 3D acquisition and rendering, with a special emphasis on technologies applicable to large scale sites. We first briefly review technologies for dense and coarse acquisition of sites.

We then provide a general overview of massive model rendering strategies for dense datasets, with a special emphasis on data reduction aspects**.** All systems dealing with massive models require the integration of techniques for reducing the working set by filtering out as efficiently as possible the data that is not contributing to a particular image. This goal is achieved by employing appropriate data structures and algorithms for visibility or detail culling, as well as by choosing alternate graphics primitive representations.

We then provide a short overview of alternate technologies that enable navigation in a site only coarsely reconstructed by a set of photos.

Finally, we provide some recommendations for the INDIGO project.

3 Acquiring 3D shape and color information: a short overview

3.1 Data acquisition

Automatic 3D reconstruction technologies have evolved significantly in the last decade [Bla], and an overview of 3D scanning technologies is presented in Curless and Seitz [CurSei]. It is beyond the scope of the report to completely cover the subject. Here we mention only two main alternatives: **passive systems** and **active systems**.

3.1.1 Passive systems

Passive system do not have any emitter component, but only rely on cameras (or video cameras). One of the main advantages is thus the simplicity (and often low cost) of the acquisition equipment (in the simplest case, just a digital camera).

Even though alternative exists (e.g. silhouette carving), most passive systems are strictly related to the concept of stereo-matching. Most people are familiar with the concept of optical triangulation, because the human vision system with its two eyes is based on a stereoscopic set-up. A technical realization might be to replace our eyes by digital cameras and an image processing system to correlate the stereoscopic images. However, there are very different objectives between the requirements of a technical vision system and the human one:

- Humans have an extraordinary system for pattern recognition and feature extraction.
- Technical systems do not even begin to compare with the complexity and versatility of the human visual system.

On the other hand, our visual system is only a qualitative one with a strong subjective component. Technical vision systems, on the other hand, should guarantee a quantitative and objective recording of our surroundings. Moreover, metrology requires high accuracy, reliability and repeatability. Therefore, homologous points in the stereoscopic images must be correlated with sub-pixel accuracy, typically in the range of 1/10 of a pixel.

Stereo matching methods work by considering correspondences in distinct images taken from cameras placed with known extrinsic (position, orientation) and intrinsic parameters and then recovering the 3D position of the point that projects onto the corresponding points by triangulation. In the case of multiple cameras (or multiple images taken with the same camera), camera extrinsic and intrinsic parameters can also be estimated using feature extraction, matching, and Structure from Motion techniques from computer vision. Once these parameters are known, (multiview) stereo matching can be used to compute the shape of the imaged models. The problem can be seen in term of computing a *disparity map* i.e. a 2-dimensional vector *d* that maps a point in a reference image to its displaced position into a second image. Given the value of the disparity map at a generic point *(x,y)*, its norm is inversely proportional to the distance of the 3D point that projected to *(x,y)*, which therefore can be easily recovered. A *dense* disparity map, i.e. such that *d* is known for almost all the pixels, entirely defines the 3D surface and it's the goal of any stereo matching algorithm. The concept can be extended to multi-view systems, using more than 2 images and matching several cameras at once. The *cost* of a disparity map is a metric to assess how much the pixels are mapped in pixels with similar intensities (the *sum of squared differences* and the *normalized cross-correlation* are popular choice). In these terms, the problem is posed as finding the disparity map with minimum cost. The degrees of freedom in defining the cost and the algorithm gave raise of a countless number of algorithms (for a comparative survey the reader can refer to [SchSze04].

Optical Passive methods are nowadays a viable choice for large scale acquisition of some kind of models. In the framework of the EU Project Epoch [EU FP6 IST NoE 507382], a fully unattended web service has been made available were the user can upload a number of uncalibrated images [VerVan] (i.e. where neither the position nor the intrisic parameters of the camera are known). Passive optical methods have also used for scanning 3D environments, i.e., from airborne sensors. A state-of-the-art system for reconstruction of human habitats from aerial images is described in [Leberl et al. 2010].

On the other hand, dense 3D reconstruction using passive systems tend to fail in areas where disparity computation is difficult, i.e., on untextured images. As we will see later, however, even though passive system cannot always substitute active systems for the dense reconstruction of a site, the portions of them that compute extrinsic (and eventually intrinsic) parameters from images, can be exploited for coarse reconstruction that can be used for image based navigation (see Section 3.3)

3.1.2 Active systems

Active systems try to overcome the problems of completely passive systems by emitting some kind of radiation or light and detecting its reflection in order to probe an object or environment. Possible types of emissions used include light, ultrasound or x-ray.

The most commonly used systems for acquiring large scale sites are time-of-flight range scanners based on lasers. The technology is often referred as “LiDAR” (Light Detection And Ranging).

The advantage of LiDARs is that they are capable of operating over very long distances, and are more accurate than, e.g., radar, since they operate at very short wavelengths and use coherent beams. Time-of-flight refers to the time a pulse of light takes to cover the distance from the light emitter to the surface and return to the receiver (located at the same position). Time-of-flight scanners consist of a laser light emitter and a receiver located in the same point. The emitter sends a light pulse which bounces off the surface of the object and returns to the receiver and the distance emitter-surface-receiver, i.e. twice the distance emitter-surface) is found as light-speed/time-of-flight.

Therefore a time-of-flight scanner essentially works by probing the object's surface sample by sample. The direction of light pulse determines which point of the surface (if any) will be found. The emitter can be controlled either by rotating the range finder itself, or by using a system of rotating mirrors.

The TOF systems are generally used to sample large-scale artefacts (such as architectures), are characterized by a very wide working volume (from a few meters to a few hundred meters for terrestrial scanners, up to kilometres for aerial ones), medium sampling resolution (usually space is sampled by taking one or a few samples for a squared centimeter), medium accuracy considering the wide operation space (terrestrial commercial systems guarantee error in the order of 0.5 centimeter), and long acquisition times (sampling speed is on the order of a few thousand samples per second, thus a single high-resolution range map can take up to 20-30 minutes). More recent hybrid systems, which add a phase shift control to the usual TOF approach, support much faster sampling speed (on the order of a hundred thousand samples per seconds) with approximately the same accuracy and are, consequently, getting a major share of the TOF market.

TOF technology is also being used in off-the-shelf hardware for small scale real time acquisition, as for example the SwissRanger (http://www.mesa-imaging.ch/index.php) which is camera with an array of 176x144 emitter/sensor able to scan in the range of 1 to 5 meters at 54 FPS.

In addition, it should be noted that engineering solutions exist that allow operating time of flight scanners in a variety of situations, including static terrestrial scan, terrestrial scans from moving vehicles, and airborne scans from airplanes.

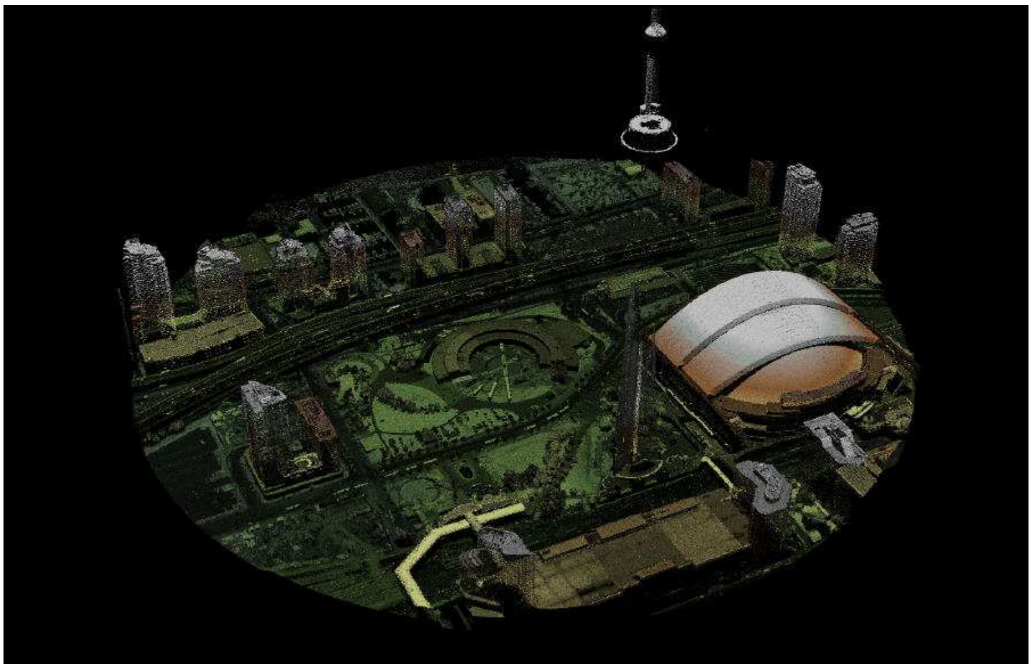
Figure 1. Downtown Toronto reconstructed using an airborne time of flight laser scanner plus high resolution registered color images. Data courtesy of Airborne 1 corporation.

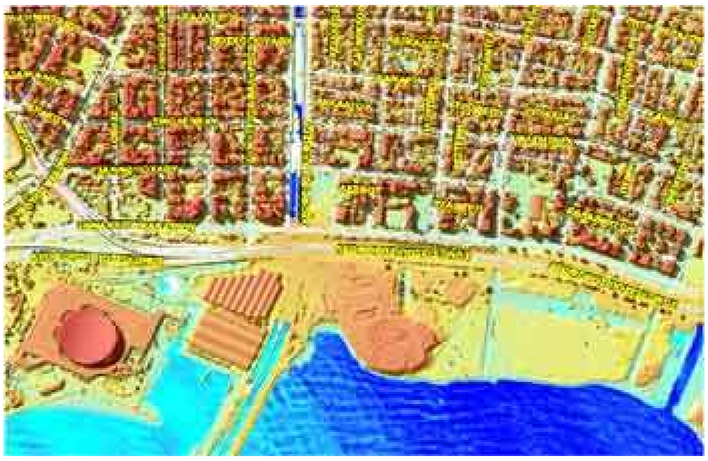
Figure 2. LiDAR reconstruction of Venice with an elevation color map. Image courtesy of BLOM CGR.

Time of flight scanners are increasingly being used to densely reconstruct parts of our environment. Some public administrations use airborne flights to monitor important parts of their environment. Airborne LIDAR sensors are used by companies in the Remote Sensing field to create point clouds of the earth ground for further processing (e.g. used in forestry). A common format for saving these points (with parameters like x, y, return, intensity, elevation) is the LAS file format. Landmarks and sensitive building are also often being mapped by both terrestrial and airborne scanners. Finally, industries routinely use scanners to acquire sensitive parts of their factories (e.g., complex buildings and pipes).

It is important to note for the INDIGO project that public repositories of LiDAR datasets exists, and are already being used in the conjunction with catastrophic events. For instance, the public site www.opentopography.org hosts a variety of datasets, including LiDAR data collected between January 21st and January 27th, 2010, in response to the January 12th magnitude 7.0 Haiti earthquake (see).

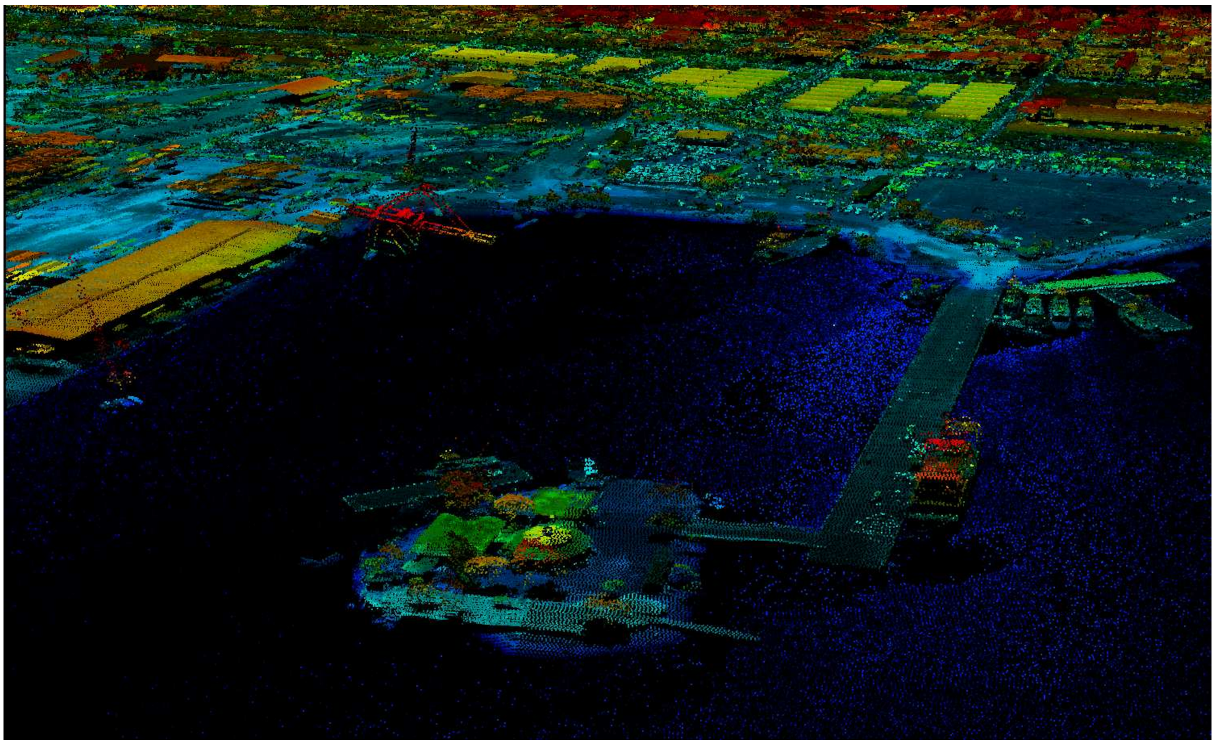

Figure 3. LiDAR point cloud data for the Port-au-Prince waterfront. Data collected between January 21st and January 27th, 2010, in response to the January 12th magnitude 7.0 earthquake. Image courtesy of Global Facility for Disaster Recovery and Recovery (GFDRR) and OpenTopography.

3.2 Modeling: generating dense 3D models

A fully automatic reconstruction pipeline for generating possibly coloured dense 3D models is not applicable in all cases. Obtaining measurable dense 3D models from acquisition typically requires the applications of a series of separate operations, that can be considered independent from the specific measurement technique taken. The main steps that must be followed in a scanning campaign are inspection, design, acquisition, alignment, editing, merge, texturing and final model building.

Object or site inspection is useful typically in order to choice the best technique to acquire the geometry, considering the final model specifications in terms of resolution, accuracy and data size, and the constraints as the budget, the time and generally the complexity of the entire acquisition pipeline related to the object. Depending on the chosen acquisition method, the acquisition design step aims at defining a precise acquisition strategy, trying to minimize the acquisition time, the data acquired for each range map and the effort in the successive steps (mainly the alignment process). The point clouds resulting from the acquisition of sites typically exceed the millions of points, and most often can exceed the billions of samples for a single scanning campaign. This size heavily affects the classes of techniques used to manage this data in the following steps. Common algorithms are not able to manage such massive data in acceptable time, so particular solutions are needed, as multi-resolution approaches for visualization and/or out-of-core techniques to filter and processing the data.

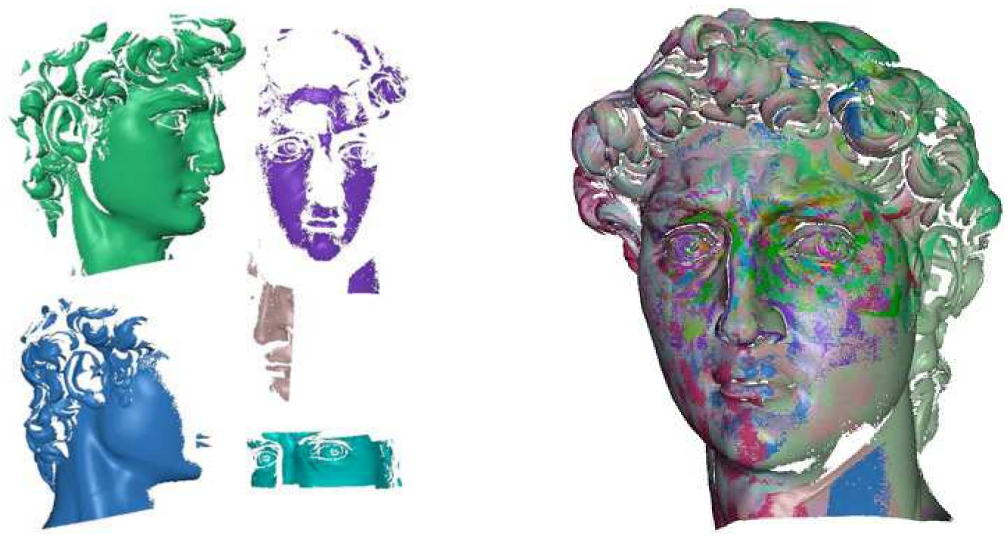

Figure 4. The alignment step brings non-referenced range-maps (left) and put them together in a single common reference frame (right).

In general, the result of a scanning campaign is a series of range maps taken from different points of view. A registration is needed in order to represent all points under the same coordinate system (Fig.1). The alignment task converts all the acquired range maps in a common reference system. This process is usually partially manual and partially automatic. The user has to find an initial, raw registration between any pair of overlapping range maps; the alignment tool then uses this approximate placement to compute a very accurate registration of the two point clouds (local registration). The precise pair-wise alignment is usually performed by adopting the Iterated Closest Point (IPC) algorithm [BesMcK, CheMed], or variations of the same. This pairwise registration process (repeated on all pairs of adjacent and overlapping range maps) is then used to automatically build a global registration of all the meshes, and this last alignment is enforced among all the maps in order to move everything in a unique reference system [Pul]. Range map alignment has been the focus of a stream of recent papers. Many possible methods can be adopted, for instance semi or completely automatic, but the most used approach remains the coarse manual alignment followed by an ICP (iterative closest point) rigid [Besl et al. 1992] or non-rigid [Brown B. and Rusinkiewicz S. 2007] fine registration. We cite here only one single result [GelMit], where interested reader can find further references. Figure 5 shows a set of range maps roughly aligned (top) renderer in synthetic colour and the result of fine alignement (bottom). Quality of the alignment is crucial for the overall quality of the merged model, since errors introduced in alignment may be the cause of wrong interpolation while we merge the range maps. Since small errors could add one to the other, the resulting global error can be macroscopic. An example of 3D acquisition where this potential effect has been monitored and accurately measured is the acquisition of Donatello's Maddalena [GuiBer]. A methodology for quality control is proposed in that paper, where potential incorrect alignement produced by ICP registration are monitored and corrected by means of data coming from close-range digital photogrammetry. With the possible exception of texturing, the alignment phase is usually the most time-consuming phase of the entire 3D scanning pipeline, due to the substantial user contribution required by current commercial systems and the large number of scans sampled in real scanning campaigns. The initial placement is heavily user-assisted in most of the commercial and academic systems (requiring the interactive selection and manipulation of the range maps). Moreover, this kernel action has to be repeated for all the possible overlapping range map pairs (i.e. in average 8 times the number of range maps). If the set of range maps is thus composed by hundreds of elements, then the user has a very complex task to perform: for each range map, find which are the partially-overlapping ones; given this set of overlapping range maps, determine which one to consider in pair-wise alignment (either all of them or a subset); finally, process all those pair-wise initial alignments. If not assisted, this becomes the major bottleneck of the entire process: a slow, boring activity that is also really crucial for the accuracy of the overall results (since an inaccurate alignment may lead to very poor results after the range maps merging). An improved management of large set of range maps (from 100 up to 1000) can be obtained by both providing a hierarchical organization of the data (range maps divided into groups, with atomic alignment operations applied to an entire group rather than to the single scan) and by using a multiresolution representation of the raw data to increase efficiency of the interactive rendering/manipulation process over the range maps. Moreover, since the standard approach (user-assisted selection of each overlapping pair and creation of the correspondent alignment arc) becomes very impractical on large set of range maps, tools for the automatic setup of most of the required alignment actions have to be provided.

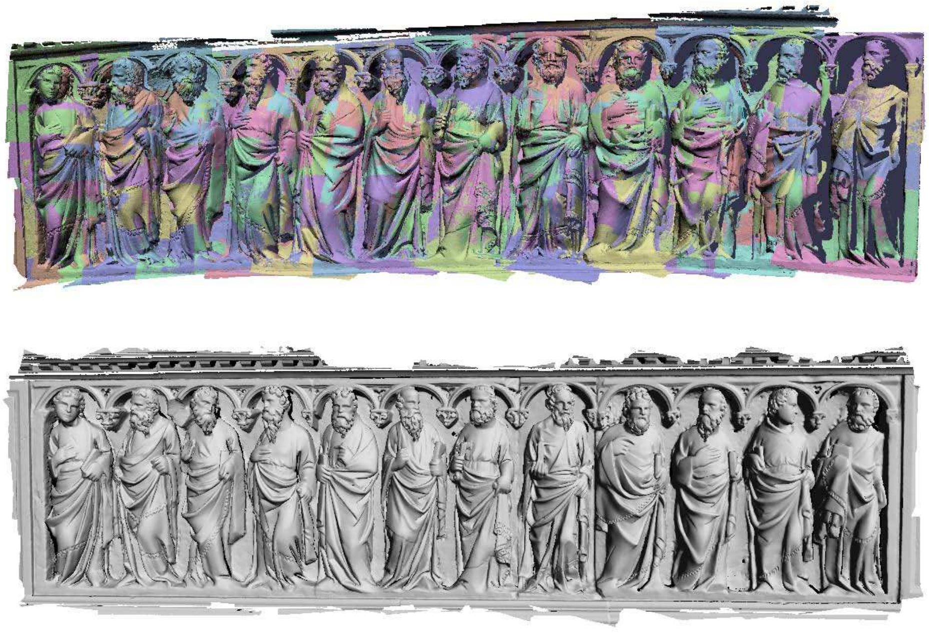

Figure 5. The coarse alignment obtained over a set of range maps representing a bas-relief (top) and t he final model (bottom) obtained after automatic completion of all overlapping-pairs

After each scan is consistent with each others, an editing/cleaning pass is applied to remove undesired points. This always involves a major user intervention. The cleaned data contains all the acquired and aligned range maps with a lot of overlapping parts. When the required result is not a point cloud, but a triangulated surface model, a triangulation procedure has to be applied to merge overlapping parts and produces point connectivity [Cuccuru et al. 2009]. If the final model is a point cloud related techniques are used to perform a regularization step to remove noise and different point resolutions in overlapping regions.

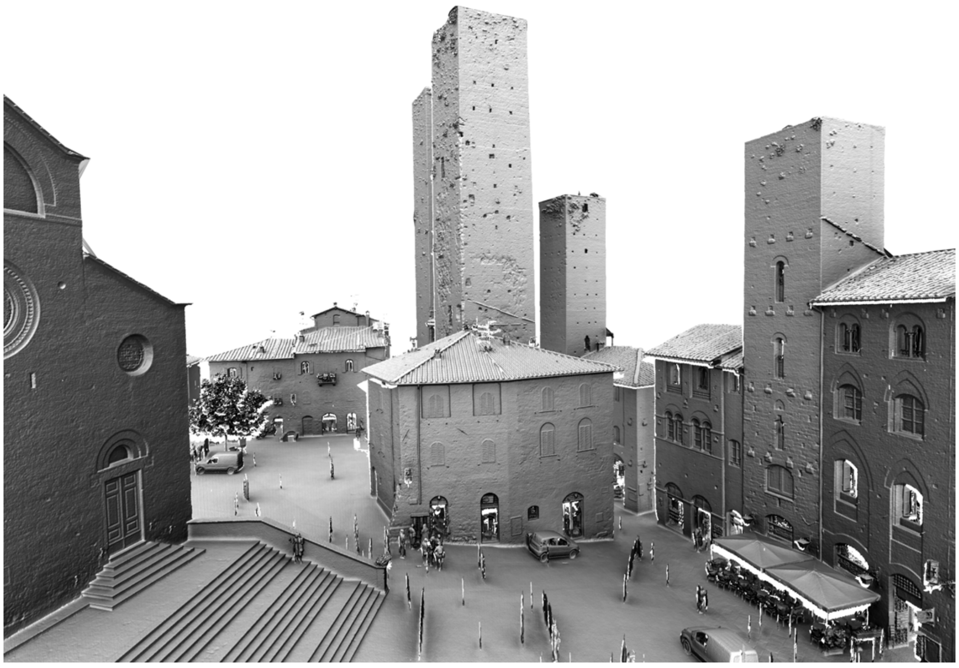

Figure 6. Domesquare at S. Gimignano (Italy) acquired using a terrestrial time of flight laser scanner. Cleaned-up model after alignment, cleaning and merging. Rendering using directional lighting and ambient occlusion. Image courtesy of ISTI-CNR.

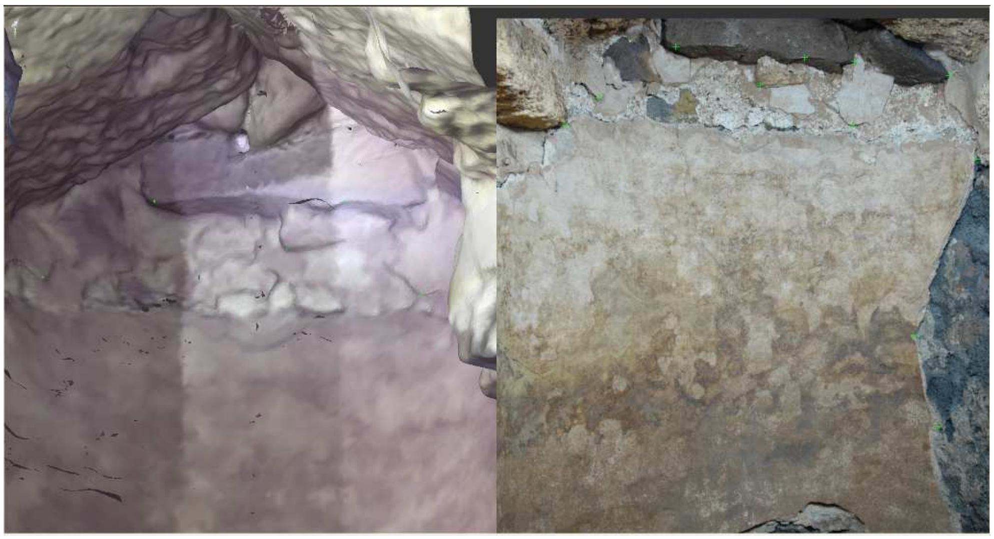

Figure 7. Differences between the 3D model of a Roman site with the color signal acquired by the laser scanner (left) and the color mapped from images taken with a 10Mp camera (right).

At this point in the pipeline one obtains a geometric, measurable data; typically the quality of the color signal acquired by laser scanners is not high enough for visualization, so a texturing process is employed to map color on the geometry data (Figure 7). The color information is most often acquired using an off the shelf camera (nowadays a 10Mp is good enough for most purposes) and is mapped using a two step procedure; the first is a manual intervention in which the user has to find a minimum number of correspondences between the 3D model (3D points on the geometry) and each image (2D points). Then the software computes intrinsic and extrinsic camera parameters and project the color from the camera point of view on the model surface.

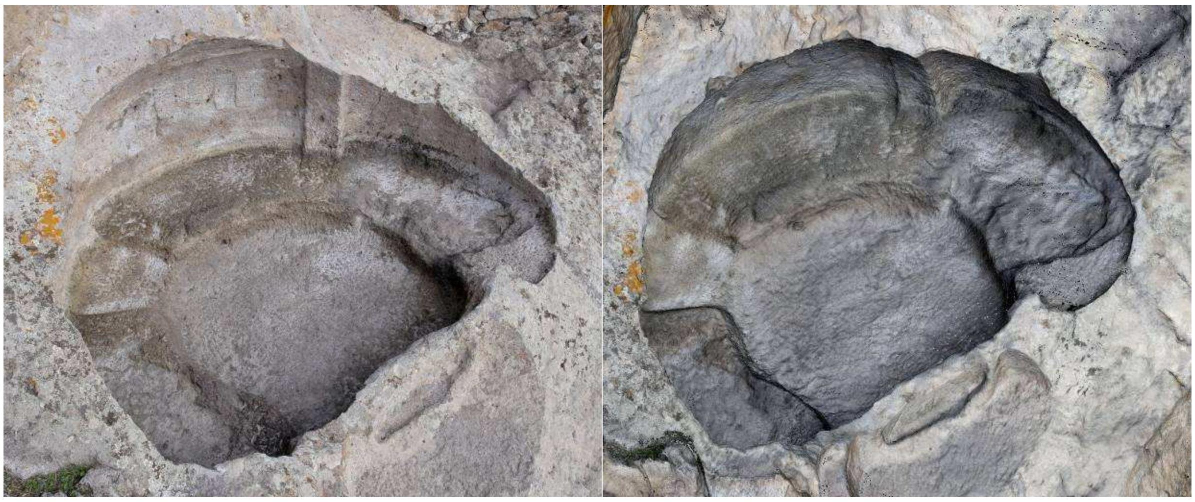

Figure 8. Comparison of a real photo of an archaeological site (left) and the 3D real-time, remote, rendering of its digital model, a point cloud of about 8M points size, obtained from time-of-flight laser scanner acquisition and color texture mapping (right).

With a fast visibility pass implemented in GPU a fast texture projection on a point cloud is feasible and efficient. Overlapping texture can be blended in many ways; a clever solution, applied to triangle meshes but general in terms of blending insights, is [Callieri et al. 2008], that use some masking function to blend colors depending on normals, depth, image borders and model silhouettes. Finally, the model with geometry and color is stored in a particular multi-resolution data structure in order to be locally or remotely available for visualization/navigation (Figure 8).

As we can see, a complete scanning pipeline for creating high quality photorealistic models is quite involved and still requires considerable manual intervention. However, if the goal is to produce measurable engineering quality models, it is often sufficient to work with (possibly filtered) aligned point clouds, thus skipping the most labor intensive portions of the pipeline.

3.3 Modeling: Structure from motion for coarse 3D with registered images

The technologies described above intend to create consistent and measurable 3D models from acquired data. A related approach is the one of image based modelling (IBM) and rendering (IBR). In this case, the problem is that of synthesizing new views of a scene from a set of input photographs. A limited version of this approach, tuned for the constrained environment of moving along street maps, has having enormous success in products such as Google StreetView or Microsoft Bing Maps. For more general environments, a lot of work has been carried out for new view synthesis starting from sets of photographs (e.g., Chen and Williams 1993; McMillan and Bishop 1995; Gortler et al. 1996; Levoy and Hanrahan 1996; Seitz and Dyer 1996; Aliaga et al. 2003; Zitnick et al. 2004; Buehler et al. 2001). Providing photo-realistic views of the world from all viewpoints has however proven very difficult. Thus, in the recent past more constrained versions of these techniques have emerged as intuitive user interfaces for browsing through collection of photographs. The most prominent example in this area is the PhotoTourism system [Snavely 2006], which differs from previous approaches in that it only reconstructs a very sparse 3D model of the world, since its emphasis is more on creating smooth 3D transitions between photographs rather than interactively visualizing a 3D world. The backbone of this kind of work is a robust Structure from Motion (SfM) approach that reconstructs 3D camera parameters, and sparse point geometry for large datasets [Snavely 2007]. Since the method provides only relative positioning between cameras, a global alignment step to a map or a reference 3D model has to be applied to find the rotation, translation, and global scale that aligns the reconstructed photographs to a global reference.

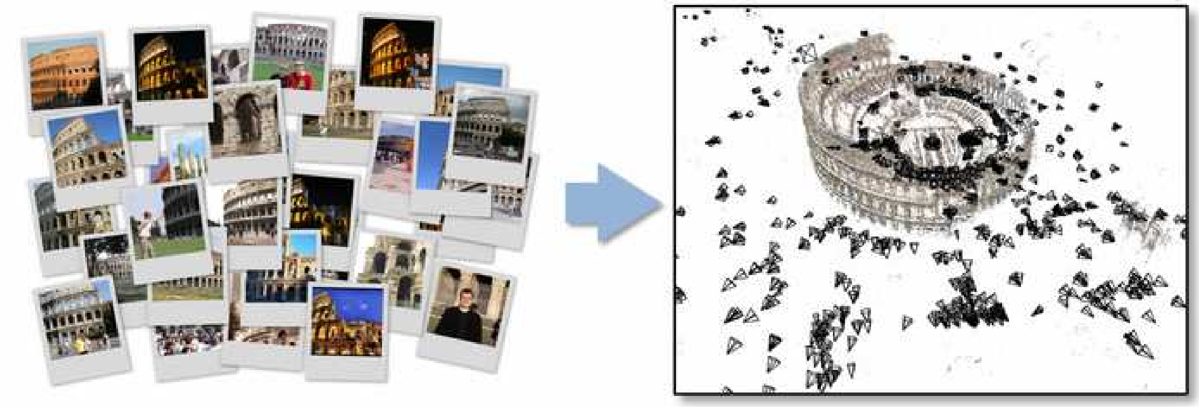

Figure 9. Bundler: Structure from Motion for Unordered Image Collections: Colosseum test case. Image Courtesy of Noah Snavely, U Washington.

Unordered image collections can be exploited as interesting means to navigate in a 3D space. They cannot always substitute a consolidated 3D scan, e.g., when precise measuring is required, but can provide a natural user interface for exploring complex environments. One major advantage of the method is the relative low cost and time required for a site acquisition. Integration with scanned data seems interesting for cases when more dense and precise reconstructions are required.

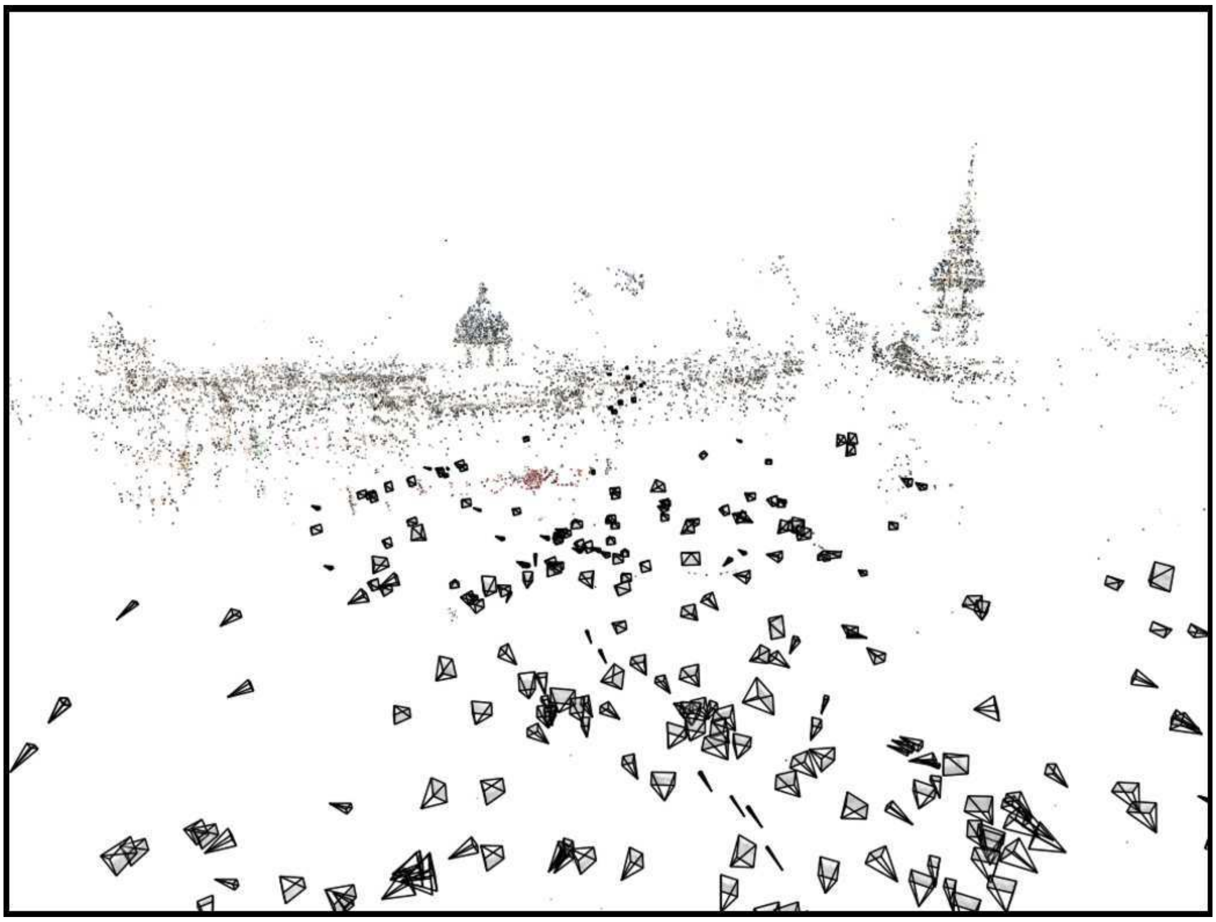

Figure 10. Bundler: Structure from Motion for Unordered Image Collections: Trafalgar Square test case. Data Courtesy of Noah Snavely, U Washington. Note that the reconstruction is sparse.

3.4 Recommendations

The objective of using 3D data in the INDIGO project is to provide simple to understand visualizations of a particular environment. These reconstructions will be created to provide means to recognize a particular area both for training and for actual operations. The tools are meant to complement traditional 2D maps and to provide additional possibility. We can think of exploiting different types of sources for creating these reconstructions:

- Dense 3D models, probably created through LiDAR or similar technologies, that provide measurable reconstructions of environments. These models can be exploited for all operations that require some sort of measurement in 3D.
- Image collections embedded in 3D space, probably created with SfM pipelines, that provide sparse but natural depictions of sites. These models can be exploited for all operations that require location recognition.

4 Navigating through dense 3D datasets: technical strategies for massive model rendering

The dense reconstruction pipeline discussed in the previous chapter result in models that accurate, measurable, but also very large.

Current graphics hardware cannot render massive models, consisting of hundreds of millions of samples/polygons at interactive frame rates using brute force techniques. This requires techniques to reduce the complexity of scenes on a frame by frame basis. The main techniques used to achieve this goal consists of: (a) rendering only the polygons that are determined visible (*Visibility Culling*); (b) using geometric approximations of the original model with lower polygon count (*Levels of Detail*); (c) using alternate representations in place of polygons (*Sample-based representations* and *Higher-order primitives*). By combining these techniques together in an adaptive system, it is possible to create a renderer whose runtime and memory footprint is proportional to the generated output complexity, not to the total model complexity. In the following, we will briefly review the state-of-the-art in the area, on the basis of the survey published in [Gobbetti et al. 2008].

4.1 Visibility Culling

Determining which surface is visible for each pixel is the at core of rendering applications. Visible surface determination techniques are essentially methods for solving a sorting problem, i.e., determining which parts of the model are closer to the viewer. The many proposed solutions vary in the order in which the sort is performed and how the problem is subdivided to make it more tractable. At the broad level, practically only two classes of algorithms are used today when dealing with massive models: (a) rasterization with z-buffering, an object order approach that determines the visible surface by streaming through scene polygons, rasterizing them, and projecting them to screen while maintaining the depth of each pixel, and (b) ray tracing, an image space approach that determines visible surfaces by computing ray intersections for each pixel.

From the memory management point-of-view, rasterization offers in principle more code and data cache coherency, because switching primitives and rendering attributes occurs much less frequently and most operations work object-by-object basis on data residing in local memory. This explains the success of rasterization hardware for supporting real-time rendering of small scenes. The situation is more complex for massive models. In their most basic form, both rasterization and raytracing techniques are limited to linear time complexity in the number of scene primitives. In order to enable rendering in sub-linear time complexity, spatial index structures and visibility culling techniques must be applied to limit the number of polygons being sent to the graphics pipeline and/or checked for ray intersection. Even though the ray tracing and rasterization fields have independently developed their own approaches in the past, the underlying issues faced when dealing with massive models are somewhat similar, and state-of-the-art systems are converging towards applying similar solutions.

4.1.1 Object space subdivision

Visibility culling methods are typically realized with the help of a so-called *spatial index*, a spatial data structure that organizes the geometric primitives in the 3D space. There are two major approaches, *spatial partitioning* and *bounding volume hierarchies* (BVHs). Bounding volume hierarchies conceptually organize geometric primitives in a bottom-up manner by hierarchically grouping bounding volumes of objects, while spatial partitioning schemes subdivide the scene in a top-down manner into a hierarchy of disjoint cells that contains the entire scene. Quite a number of spatial partitioning schemes have been proposed in the past. *Hierarchical grids*, *octrees*, *kd-trees* are the most popular methods, and *kd-trees* are usually considered the option of choice for massive models. More details can, e.g., be found in [Samet 2006]. Even though the concepts of classic spatial indexes are simple and well understood, constructing them for massive models requires specialized methods that have a low computational complexity and coherent access patterns to avoid I/O thrashing, while at the same time providing optimized space partitions for visibility queries.

In the case of kd-trees, a de-facto standard for obtaining optimized subdivision is to minimize the cost model for ray-object intersections called *Surface Area Heuristics* (SAH) [Goldsmith and Salmon 1987, MacDonald and Booth 1990, Havran 2000]. This heuristics assumes a uniform distribution of rays with no occlusion, and makes it simple to estimate the probability to traverse the different branches of the hierarchy by comparing the surface areas of the bounding boxes of the various nodes. An approximately optimal kd-tree can be computed by minimizing the expected ray tracing cost by performing a top-down greedy optimization. Such a technique, which recursively splits the model by choosing the minimum cost split plane at each step, is impractical for massive models. This is because there are too many possible splitting planes and finding the best split plane requires to sort triangles according to these planes. For these reasons, many authors have proposed simplified techniques for faster tree construction (e.g., [Popov et al. 2006, Hunt et al. 2006, Shevtsov et al. 2007]). These methods share a common set of concepts. First of all, they only consider axis aligned bounding boxes of objects instead of individual triangles for building the hierarchy. Second, they do not test all potential split planes, but only use *K* heuristically selected equidistantly spaced planes. Third, the SAH for each of these is computed in a single streaming pass. This approach greatly reduces the number of plane evaluations (K bin planes instead of $O(N)$ triangle bounds planes) but also avoids any sorting. Splitting can thus be done simply with two $O(N)$ passes. The full hierarchy can thus be constructed with $O(N\log N)$ operations. Moreover, and most importantly for massive models, all operations are performed in a streaming fashion, with good memory locality and minimal needs for in-core memory. The main drawback of these methods is the need to select up-front a small set of candidate planes. A more elaborate solution, which avoids binning and considers triangle splitting is presented in [Wald and Havran 18-20].

Bounding volume hierarchies are not generally used for large static environments but for (smaller) dynamic environments for which the hierarchy is either given up-front at modeling time, e.g., by associating bounding volumes to objects in a kinematic hierarchy, or recomputed dynamically as objects move. The research focus more on how to update a hierarchy after object motion. Reasonably fast $O(N\log N)$ algorithms for rebuilding BVHs are presented in [Wald et al. 2007, Lauterbach et al. 2006, Wald 2007], while $O(N)$ methods for refitting an already existing bounding volume hierarchy are presented in [Havran et al. 2006, Woop et al. 2006, Wächter and Keller 2006]. Hybrid methods combining refitting and construction techniques are also available [Lauterbach et al. 2006, Yoon et al. 2007].

4.1.2 From-point visibility culling

From-point algorithms are the basis of all interactive viewing applications, since they attempt to determine at each frame which parts of the scene are visible from the current viewpoint. Visible surface determination (or hidden-surface removal) is the most basic from-point algorithm and can be considered as the final stage of every rendering pipe-line. Before actual visible surface determinations takes place, visibility culling methods must be employed to filter out data at the coarse level. View-frustum and backface culling are simple but effective from-point operations that can be optimized using spatial data structures. A common choice is to combine a hierarchy of bounding spheres, axis-aligned bounding boxes, or kd-trees with cones of normals [Rusinkiewicz and Levoy 2000a, Cignoni et al. 2004, Gobbetti and Marton 2004], and to perform hierarchical view-frustum and backface culling during a top-down scene traversal. Processing stops whenever a model portion is proved out-of-view or backfacing. Such traversal schemes can also be adapted for implementing occlusion culling, which stops traversal in case of occlusion.

In the case of ray-tracing, early traversal termination in case of occlusion is simple to implement by a front-to-back traversal of the spatial index: since visibility is evaluated independently for each ray, once a hit-point has been found, it is certain that geometric primitives behind are not visible for that specific ray direction. An important optimization that is widely applied in state-of-the-art real-time ray tracing systems is to simultaneously trace bundles of rays called *packets* [Wald et al. 2001]. First, working on packets allows use of SIMD vector operations of modern CPUs to perform parallel traversal and intersection of multiple rays. Second, packets enable deferred shading, i.e., it is not necessary to switch between intersection and shading routines for every single ray, thus amortizing memory accesses, function calls, etc. Third, it is possible to avoid traversal steps and intersection calculations based on the bounds of ray packets and makes better use of both object and scanline coherence. This idea of accelerating raytracing by working on groups of rays is also exploited in frustum traversal methods [Reshetov et al. 2005].

When using rasterization, the decision about when traversal of the spatial index can be stopped can also be made in image space by exploiting the Z-buffer, and the most recent algorithms exploit graphics hardware for this purpose. For occlusion culling during front-to-back scene traversal, bounding volumes are simply tested for visibility against the current Z-buffer using the *occlusion query* functionality to determine whether to continue traversal. It should be noted that, although the query itself is processed quickly using the rasterization power of the GPU, its result is not available immediately due to the delay between issuing the query and its actual processing by the graphics pipeline. A naive application of occlusion queries can actually decrease the overall application performance due the associated CPU stalls and GPU starvation and introduce additional end-to-end latency in the application. For this reason, modern methods exploit spatial and temporal coherence to schedule the issuing of queries [Govindaraju et al. 2003, Bittner et al. 2004, Yoon et al. 2004, Heyer et al. 2005, Klosowski and Silva 2001]. The central idea of these methods is to issue multiple queries for independent scene parts and to avoid repeated visibility tests of interior nodes by exploiting the coherence of visibility classification. It is interesting to note that hierarchical front-to-back rasterization in combination with occlusion culling can be interpreted as a form of beam- or frustum tracing, which demonstrates the convergence of ray-tracing and rasterization research in the massive model rendering domain.

4.1.3 From-region visibility culling

From-point algorithms perform visibility culling at each frame by exploiting object space-subdivision. For static scenes, it is tempting to pre-compute visibility information for scene regions, to speed up run-time visibility processing. This is the domain of *from-region* algorithms, which precompute a *potentially visible set* (PVS) for cells of a fixed subdivision of the scene partitioning the view-space. During rendering, only the primitives in the PVS of the cell where the observer is currently located, are rendered, potentially leading to large savings in rendering time. For complex scenes, however, these savings are more theoretical than practically achievable with current technology. While exact visibility from a single viewpoint can be calculated using visible surface determination methods and accelerated using space partitioning structures, computing the PVS for a region is much harder. Excellent algorithms for computing exact visibility from a region in space exist for general scenes do exist [Durand 1999, Duguet and Drettakis 2002, Nirenstein et al. 2002, Bittner 2002, Haumont et al. 2005, Mora et al. 2005]. However, their running time and memory costs make them unsuitable for massive models. For this reason, many authors have concentrated on “conservative” techniques, i.e., techniques that simplify computation by, hopefully slightly, over-estimating the PVS to include some objects that are actually not visible and to never exclude unoccluded objects. In reality, the problem turns out to also be very hard to solve, and there are practically no published provably conservative techniques for general environments. Rather, known techniques are restricted to particular types of scenes. Examples of constraints are the limitations to architectural building interiors [Airey et al. 1990, Teller and Sequin 1991], 2.5D visibility for terrains and urban scenes [Wonka et al. 2000, Bittner et al. 2001, Koltun et al. 2001], volumetric occluders [Schaufler et al. 2000] or large occluders close to the view cell [Durand et al. 2000, Andújar et al. 2000, Leyvand et al. 2003]. For general scenes, non conservative sampling based solutions, that compute from-region solutions combining results from from-point queries, have recently emerged as a practical approach, due to their robustness and ease of implementation. Nirenstein and Blake [Nirenstein and Blake 2004] proposed an approach which uses rasterization hardware for sampling visibility. An approach that focuses, instead on harnessing a fast ray-tracing kernel has been recently presented by Wonka et al. [Wonka et al. 2006].

Storing and transmitting the computed PVS is also an important problem. There is an obvious trade-off between the quality of the PVS estimation on one hand and memory consumption and precomputation time on the other hand. Smaller view cells improve the quality of PVS computation, simultaneously increasing the number of view cells that need to be precomputed. In addition to requiring large precomputation times, having a large number of view cells can result in extremely large storage requirements for storing all PVSs, as well as in large bandwidth requirements for communicating the PVSs to the rendering engine, which is an important drawback for massive scenes.

In general, from-point techniques are more robust and easy to integrate in a system, since they require less storage and less preprocessing time and resources. There are, however, situations in which a from-region algorithm is appropriate and can provide considerable advantages: this is the case, for instance, of a number of videogames, in which the scene is modeled only once and can be constructed to make region selection easy. A good from-region algorithm for general or massive models with reasonable preprocessing cost and good storage optimization remains an open issue.

4.2 Simplification and level of details

Relying on efficient visibility determination alone is not sufficient to ensure interactive performance for highly complex scenes with a lot of small scale details. In such cases, many visible modeling primitives may only project to a single pixel or sub-pixel. In order to bound the amount of data required for a given frame, a filtered representation of details must thus be available. Computing such a representation from highly detailed models, and efficiently extracting the required detail from this representation at rendering time is the goal of simplification and level of detail techniques.

4.2.1 Geometric simplification

Geometric simplification is a well studied subject, and a number of high quality automatic simplification techniques have been developed [Luebke 2001]. Optimal approximation of a surface, in terms of computing the minimal number of triangle primitives that would satisfy some approximation error metric, is known to be NP-Hard [Agarwal and Suri. 1994], and hence most research has focused on developing heuristic methods.

At the broadest level, simplification methods may be grouped into *global strategies* that are applied to the input mesh as a whole, and *local strategies* that iteratively simplify the mesh by the repeated application of some local operator. Local strategies are by far the most common simplification approaches, mainly because of their efficiency and robustness. The wide majority of the simplification methods iteratively simplifies an input mesh by sequences of vertex removals or edge contractions. In most current systems, simplification is performed in an iterative greedy fashion, which maintains a sorted list of candidate operations and applies the operation associated to the minimal simplification error at each step. Unfortunately, a direct implementation of this approach is not well-suited to work on massive meshes, since maintaining a priority queue of possible operations results in a memory consumption proportional to the size of the original mesh, a clearly untenable situation for extremely large models. Even if this obstacle could be overcome by using out-of-core data structures, the order of contraction operations could exhibit little locality in terms of memory accesses, with detrimental effects on algorithm performance.

The two main solutions that have been proposed for these problem are *streaming simplification* methods and *mesh partitioning* methods. The key insight behind streaming simplification [Wu and Kobbelt 2003, Isenburg et al. 2003] is to keep input and output data in streams that document, for example, when all triangles around a vertex or all points in a particular spatial region have arrived with "finalization tags". For simplification, an in-core buffer is filled and simplified and output is generated as soon as enough data is available.

Mesh partitioning methods, instead, are based on iterative simplification of mesh regions. Several authors [Hoppe 1998, Prince 2000] have proposed methods in which a mesh is segmented so that each piece fits in the main memory. While this solution is conceptually appealing, the segmenting and rejoining operations are expensive, and make this approach less attractive for very large meshes. A major drawback of these methods is that region boundaries remain unsimplified until the very last simplification step, with leading to quality and scalability problems. OEMM [Cignoni et al. 2003a] avoids the region boundary problem by exploiting a out-of-core octree-based data structure that maintains relationships between blocks and thus supports simplification of block boundaries. Another efficient technique for avoiding boundary locking has been proposed by [Cignoni et al. 2004, Cignoni et al. 2005]. In these approaches, the mesh is spatially partitioned using hierarchical volumetric subdivision schemes that create conforming volumetric meshes that support local refinement and coarsening operations.

Streaming simplification approaches lead in general to simpler and faster solutions because of their inherent I/O efficiency. On the other hand, mesh partitioning approaches are more general, can produce very high quality results, and have also the capability of producing continuous LOD representations.

4.2.2 Level of detail

A *level-of-detail* (LOD) model is a compact description of multiple representations of a single shape and is the key element for providing the necessary degrees of freedom to achieve run-time adaptivity. LOD models can be classified as *discrete*, *progressive*, or *continuous*. Discrete models simply consist of ordered sequences of distinct representations of a shape and only support switching among representations. Progressive models consist of a coarse shape representation and of a sequence of modifications (e.g., edge splits) supporting incremental refinement. Continuous models improve over progressive models by fully supporting selective refinement, i.e., the extraction of representations with a LOD that can vary in different parts of the representation. Continuous representations can be changed on a virtually continuous scale.

A general framework for managing continuous LOD models is the multi-triangulation technique [De Floriani et al. 1998], which is based on the idea of encoding the partial order describing mutual dependencies between updates as a directed acyclic graph (DAG). In the DAG, nodes represent mesh updates (removals and/or insertions of triangles that change the representation of a mesh region), and arcs represent dependency relations among updates.

Most of the continuous LOD models can be expressed in this framework, and many variations have been proposed. Up until recently, however, the vast majority of view-dependent level-of-detail methods were all based on multi-resolution structures where LOD decisions are taken at the triangle/vertex primitive level. This kind of approach involves a constant CPU workload for each triangle and makes detail selection the bottleneck in the entire rendering process. This problem is exacerbated in rasterization approaches, because of the increasing CPU/GPU performance gap.

To overcome the detail selection bottleneck and to fully exploit the capabilities of current hardware, it is necessary to select and send batches of geometric primitives to be rendered using only a few CPU instructions. To this end, various GPU oriented multi-resolution structures have been recently proposed. The methods are based on the idea of moving the granularity of the representation from triangles to triangle patches [Cignoni et al. 2004, Yoon et al. 2004]. Thus, instead of working directly at the triangle level, the models are first partitioned into blocks containing many triangles, and, then, a multi-resolution structure is constructed among partitions. By carefully choosing appropriate subdivision structures for the partitioning and managing boundary constraints, hole-free adaptive models can be constructed. The benefit of these approaches is that the needed per-triangle workload to extract a multi-resolution model is reduced by orders of magnitude. The small patches can be preprocessed and optimized off line for a more efficient rendering, and highly efficient retained mode graphics calls can be exploited for caching the current adaptive model in video memory. Recent work has shown that the vast performance increase in CPU/GPU communication results in greatly improved frame rates [Cignoni et al. 2004, Yoon et al. 2004, Cignoni et al. 2005]. Similar structures have been presented for 2D domains and have been used for terrain visualization [Cignoni et al. 2003b, Cignoni et al. 2003c], streaming [Bettio et al. 2007] and compression. The success of these coarse level approaches indicates the increasing importance of memory/bandwidth management issues in real-time rendering applications. Even though coarsening multiresolution granularity reduces the model flexibility and requires more triangles to achieve a given accuracy, the overall efficiency of the system is dramatically increased rather than reduced, since rendering time does not depend linearly on triangle count anymore. Instead, rendering time is strongly influenced by how the triangles are organized in memory and sent to the graphics card.

4.3 Alternate rendering primitives

So far, we have concentrated on methods centered around efficient procedures for simplifying triangle meshes, arranging details in a multiresolution structure, and efficiently extracting them at run-time to realize adaptive rendering. The complexity of the rendering operation can be also reduced by switching to representations other than triangle meshes. Representations other than polygons offer significant potential for massive models visualization.

On one hand, important model classes, such as CAD models, are well described in terms of higher order geometric primitives. One might thus consider directly rendering them instead of resorting to precomputed tessellations. The potential advantages of such an approach include a reduction of needed memory and the ability to generate smooth views at high magnification levels. On the other hand, in conventional polygon-based computer graphics, models have become so complex that for most views the projection of polygons may be smaller than one pixel in the final image. As a result, many researchers have been investigating alternate, mostly sample-based, scene representations. These representations use sets of points, voxels, or images to accelerate the rendering

4.3.1 Higher order primitives

Both raster-based and ray-tracing rendering approaches work directly and efficiently with low order primitives. High order primitives are generally tessellated into triangles or into intermediate forms in a preprocessing step. Direct rendering of the high order primitives is generally too slow to sustain interactive performance even for relatively small dataset sizes. There have been efforts to integrate high order primitives into the rendering pipeline since the early 1970’s [Goldstein 1981]. This kind of work has progressed substantially, and recent approaches are using programming techniques on GPU hardware to increase performance. In particular, a number of authors have focused on devising efficient methods for raycasting quadrics, cubics, and quartics on the GPU [de Toledo and Levi 2004, Loop and Blinn 2006, Tarini et al. 2006, Sigg et al. 2006, de Toledo et al. 2007], and [Krishnamurthy et al. 2007] introduced a method for direct evaluation of NURBS surfaces on the GPU. Even when using specialized hardware, however, current systems do not match the performance of rendering from precomputed meshes. Since the performance of programmable graphics systems continues to grow, it is reasonable to expect that in the near future moderately complex models could be rendered in real-time. This is particularly important for interactive modeling applications, where manipulation of the original parametric data is important. It should also be noted that using higher-order primitives alone does not fully solve the scalability problem of massive model renderers, since at low magnification levels complex models still contain a large number primitives. The definition of a multiresolution representation above the primitive level is therefore required to support view-dependent rendering.

4.3.2 Sample based representations

Sample-based representations occupy the opposite end of the spectrum from higher order representations. They exploit discrete sampling methods to represent complex models with sets of samples, typically points or voxels.

A point-based geometry representation can be considered a discrete sampling of a continuous surface, resulting in 3D positions , optionally with associated normal vectors or auxiliary surface properties, e.g., colors or other material properties. One of the major benefits of such a modeling approach is its simplicity. There is no need to explicitly manage and maintain mesh connectivity during both preprocessing and rendering. On the other hand, the lack of connectivity makes it hard to reconstruct continuous (i.e., smooth and hole-free) images from such a discrete set of surface samples. Holes and gaps in-between the samples can be closed by image-space reconstruction techniques [Grossman and Dally 1998] or by object-space resampling.

The techniques from the latter category dynamically adjust the sampling rate so that the density of projected points meets the pixel resolution, which can be done both for rasterization and ray tracing approaches. Since this depends on the current viewing parameters, the resampling has to be done dynamically for each frame, and multi-resolution hierarchies or specialized procedural resamplers are exploited for this purpose. Examples are bounding sphere hierarchies [Rusinkiewicz and Levoy 2000b], dynamic sampling of procedural geometries [Stamminger and Drettakis 2001], the randomized Z-buffer [Wand et al. 2001], and the rendering of moving least squares (MLS) surfaces [Alexa et al. 2001].

As for polygonal multi-resolution rendering, amortization over a large number of primitives is essential to maximize rendering speed on current rasterization architectures. The highest performance is currently obtained by coarse-grained approaches. Coarse grained refinement for point clouds was introduced by the Layered Point Cloud multiresolution approach [Gobbetti and Marton 2004], a method that creates a coarse hierarchy over the samples of the datasets simply by reordering and clustering them into point clouds of approximately constant size arranged in a binary tree.

Even though these coarse grained techniques improve rendering speed over classic point render of over one order of magnitude, current point based techniques are competitive in terms of rendering performance with triangle mesh ones only if one uses simple unblended disks for point cloud rendering, which limits their ability to correctly treat texture and transparency and makes them more prone to produce aliasing artifacts.

Overall, peak performance of high quality techniques based on sophisticated point splatting is currently inferior to the performance of corresponding triangle rasterization and raytracing approaches, due to the additional overhead of sample blending. This situation might change in the near future, as novel architectures for hardware-accelerated rendering primitives are currently being introduced [Weyrich et al. 2007]. Moreover, these technologies have the advantage that they can be used directly to aligned point clouds, and do not require that a surface reconstruction step is applied in the scanning pipeline.

Sample based representations have been traditionally used to describe surfaces with oriented points. More recently, they have been used to model the appearance of small volumetric portions of the environment, which offers advantages in models with very complex geometry. In the Far Voxels approach [Gobbetti and Marton 2005], LODs are generated by discretizing spatial regions into cubical voxels. Each voxel contains a compact direction dependent approximation of the appearance of the associated volumetric sub-part of the model when viewed from a distance. The approximation is constructed by a visibility aware algorithm that fits parametric shaders to samples obtained by casting rays against the full resolution dataset. The voxels are rendered using a point primitives interpreted by GPU shaders. The approach proved to be effective in cases where pure geometric simplification remains hard to apply. These cases appear in very complex models, where the visual appearance of an object depends on resolving the ordering and mutual occlusion of even very close-by surfaces, potentially with different shading properties. For such complex models, visibility preprocessing and model simplification are strictly coupled. A similar approach to model simplification is also applicable to ray tracing [Yoon et al. 2006, Dietrich et al. 2006].

4.3.3 Image based approaches

In the geometry-based rendering approach, the visible component of the world is the union of two elements: the geometric description of the objects and the color and lighting conditions. A different approach is to consider the world as an collection of 2D images, one for each position, orientation and possibly time. The goal of *image-based rendering* (IBR) is to generate images by directly resampling such an image collection given the view parameters [McMillan and Bishop 1995], without the need of a full three-dimensional reconstruction. This approach has the theoretical advantage of decoupling rendering complexity from (geometric) scene complexity. In practice, however, a fully IBR approach is typically impractical, due to the sheer amount of data required for a full dense encoding of a complex model as a set of images. Full-scene image base rendering also loses the scene graph structure necessary for interactive selection. Restricted solutions have thus been proposed. The underlying idea behind all approaches is either to reduce the problem dimensionality by imposing constraints on viewer motion, or to compensate the aliasing effect by using additional geometric information.

Since no compensation of aliasing effects is possible without additional geometric information, either the sampling must be very dense [Gortler et al. 1996, Levoy and Hanrahan 1996], which is not practical for large scenes, or the possible viewer motion must be restricted, e.g., spherical or cylindrical panorama systems [Lippman 1980, Chen 1995].

When the viewer's motion is unrestricted, some geometric information must be employed in addition to images. In the last decade, a set of successful hybrid techniques have been proposed to accelerate the rendering of portions of a complex scene. The techniques replace complex geometry with textures in well-defined cases. In most cases, the basic idea is to use a geometry-based approach for near objects, and switch to a radically different image-based representation, called an *impostor*, for distant objects that have small, slowly changing on-screen projections. Successful examples include *portal textures* [Aliaga and Lastra 1997], *textured depth meshes* [Sillion et al. 1997, Wilson and Manocha 2003], *layered environment maps* [Jeschke et al. 2002, Jeschke and Wimmer 2002], *layered depth images* [Shade et al. 1998, Wimmer et al. 2001]. These techniques, introduced a decade ago, are enjoying a renewed interest, because of the evolution of graphics hardware, which is more and more programmable and oriented toward massively parallel rasterization. These techniques will be investigated more in depth in the section related work of urban models rendering.

A number of specialized hardware accelerated techniques, often based on GPU raycasting, have been introduced (e.g., *relief mapping* [Oliveira et al. 2000], and various forms of *view-dependent displacement mapping* [Wang et al. 2003, Wang et al. 2004, Baboud and Décoret 2006, Policarpo et al. 2005]). These methods have already demonstrated their applicability to massive model rendering systems [Wilson and Manocha 2003, Aliaga et al. 1999]. This kind of techniques is well suited to urban environments, as we will see in the next chapter.

The evolution of the methods illustrates the convergence of rasterization and raytracing approaches, and the appeal of simple image based representation that enable the powerful

GPU rasterization architecture, and more in general, streaming architectures, to process geometry in addition to images.

4.4 Recommendations

In the INDIGO project, point-based rendering seems the most appropriate approach for dense models, since it can be applied both to “raw” acquired data and polished ones. The implemented technology should strive to provide rapid construction of a multiresolution rendering database from already aligned raw data (e.g., in the las format), measurement possibilities (picking of 3D points), fast rendering rates and creation of understandable images from raw datasets (e.g, by using illustrative techniques).

5 Navigating within sparse datasets: photographs embedded in 3D space

In recent years, computer vision techniques such as structure from motion and model-based reconstruction have gained traction in the computer graphics field under the name of image-based modelling (IBM), i.e., the process of creating three dimensional models from a collection of input images. One particular application of IBM has been the creation of large scale architectural models (Debevec et al., 1996; Dick et al. 2004; Teller et al. 2003). There are also several ongoing academic and commercial projects focused on large-scale urban scene reconstruction. These efforts include the 4D Cities project (Schindler et al. 2007), which aims to create a spatialtemporal model of Atlanta from historical photographs; the Stanford CityBlock Project (Román et al. 2004), which uses video of city blocks to create multi-perspective strip images; and the UrbanScape project of Akbarzadeh et al. (2006). The V-CITY European project is also an ongoing effort that includes components for automatic city modelling from photographs (in that case, oblique aerial photographs, plus, eventually, some ground images). All these efforts are amenable to customized image based rendering techniques, such as those reviewed in the previous chapter. Detailed modeling from photographs of general environments remains however a hard tasks. A viable alternative to modelling from photographs, is to consider that the reconstruction from photograph is only coarse, and to focus on navigating between sparse photographs rather than on extracting a photorealistic consistent model from them.

In the following, we review some of the related visualization and navigation techniques.

5.1 Photo Tourism

Photo Tourism [Snavely 2006] was the first system to implement the navigation of a 3D scene by moving from photograph to photograph with only a very rough point based representation of the actual 3D geometry. In other terms, the user may view the 3D scene only from the positions from which the photographs were taken (see Figure 11).

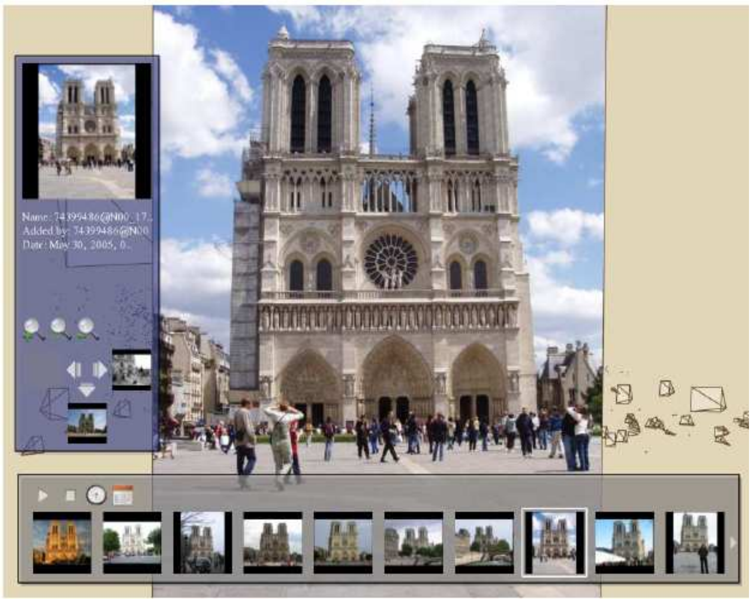

Figure 11. A snapshot of *Photo Tourism* [Snavely 2006]

5.1.1 Transition between photographs

To keep the user's sense of location in the scene, the transition from a photograph to the next is done by interpolating the point of view and the direction of view linearly. The image to render during the transition is a combination of the two obtained by projecting photographs in a common plane (computed during the building of the dataset). In this transition the image may look out of focus and distorted, but it serves its purpose. If the two photographs are very far away, i.e. if there is not a reasonable common plane, the system applies a simple fading.

5.1.2 Selection of photographs

In the Photo Tourism graphics interface the photographs are represented as small pyramids with their apex in the shooting position and the user may select the photograph to visualize by clicking on its pyramid. Alternatively, the user may select a region of interest of the picture currently visualized and the system automatically looks for the best photograph corresponding to the selection.

In order to support common movements of the point of view such as panning or zooming out, a graph is build at preprocessing and used to bind the movement of the point of view view with the proper photograph, without the need of explicitly select it.

One more feature of Photo Tourism is a bar showing iconized version of the photographs similar to the one currently visualized. This feature is useful to furtherly distinguish the photographs that are close in term of position but different in terms of quality, beautifulness, light conditions etc.

Photo Tourism was developed by Microsoft as a Java Applet. A more sophisticated version has been later developed for MS Silver Light (http://photosynth.net/)

5.1.3 3D visualizations

Presenting a general 3D overview of a set of sparsely reconstructed photographs is challenging. One of the proposals of the Photo Tourism systems is to select only the photographs for which dominant planar surfaces exists and to display them in 3D using non-photorealistic means. The dominant planar surfaces are found by robustly fitting planes to the set of 3D points associated to each image, and to accept them only if they succeed in fitting a large enough portion of the points. The corresponding image is then displayed as a semitransparent 3D rectangle with washed out colors.

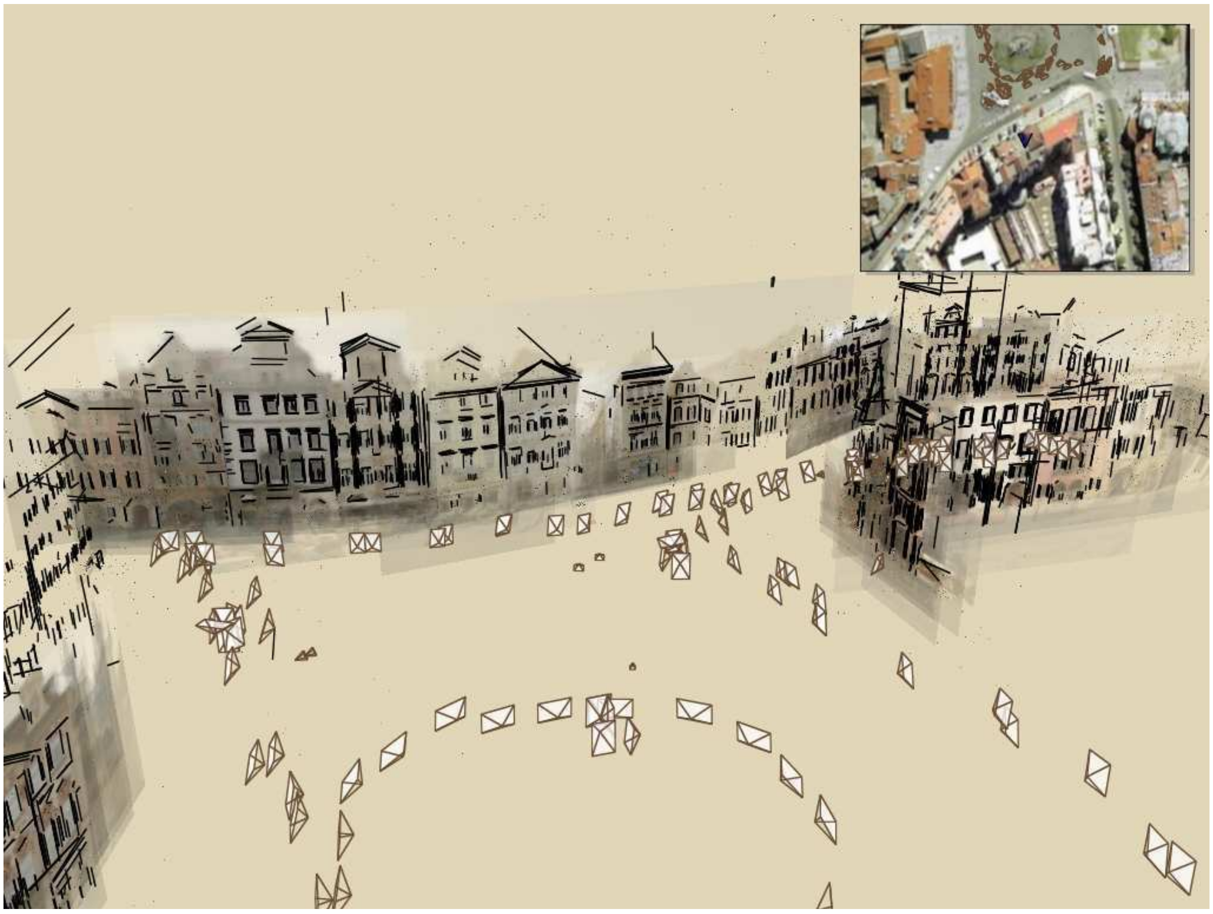

Figure 12. A view looking down on the Prague dataset, rendered in a non-photorealistic style (Snavely 2008)

5.2 PhotoCloud

An academic tool, PhotoCloud (see Figure 13), implemented the same principles as Photo Tourism improving some visualization aspects. An important one is the hierarchical representation of photographs' to avoid indistinct visualization in scene with high density of photographs. This is especially common in the touristic sites, where many photographs are taken almost from the very same place.

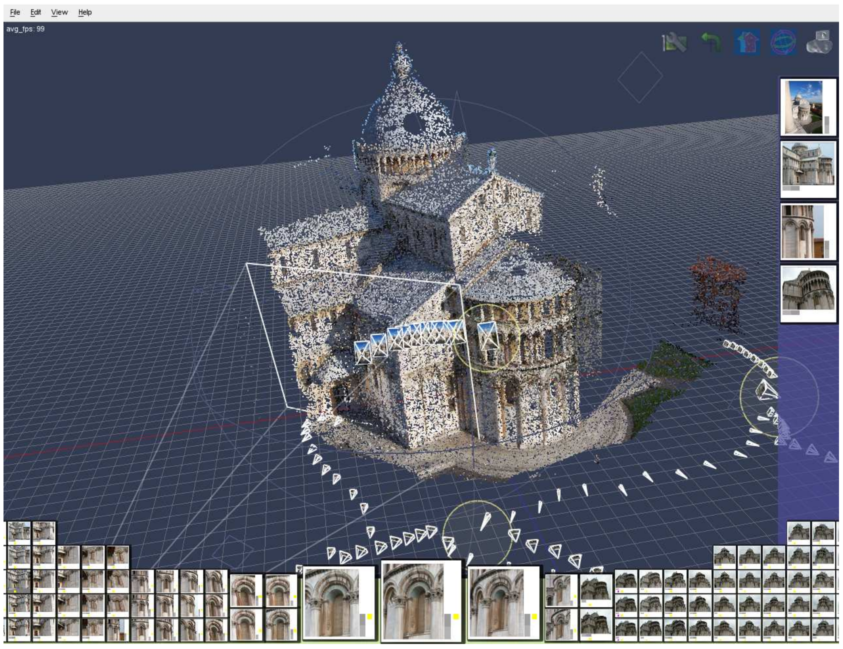

Figure 13. A snapshot of PhotoCloud

In order to avoid confused visualization of many superimposed pyramids, PhotoCloud recursively clusters group of a nearby photographs and pick up a representative one that will be shown when the user if face from the cluster. When approaching to a cluster, the pyramids of the photographs in the cluster are renderer in a position between the one of the representative and their actual position, so producing a smooth transition without popping artifacts.

5.3 Visualizing large sets of photographs

Handling large sets of photographs is also a visualization challenge: PhotoCloud improved the way to represent iconized version of the pictures by distributing them on the bottom and sides of the screen on the base of a distance criterion. A recent approach [BriTar] allows to browse large set of images that are dynamically rearranged, rescaled and reshaped to cover a given region of screen of any shape by using anisotropic Voronoi diagram to assign a region of screen to each picture and smoothly clipping the picture with the border of such region (see an example in Figure 14).

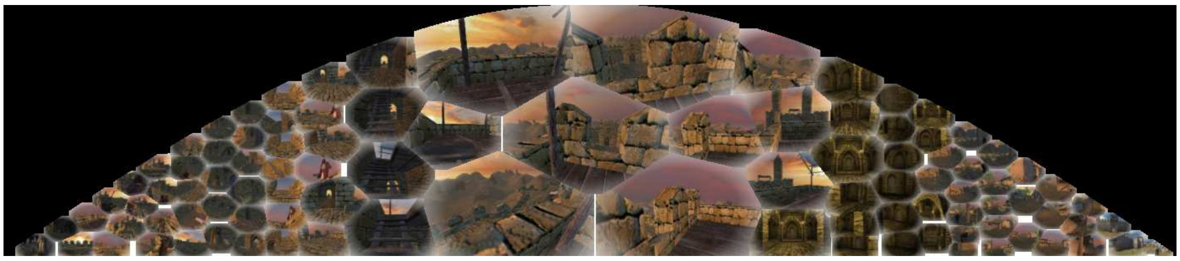

Figure 14. Photographs arranged in a curved shape with anisotropic Voronoi diagrams [BriTar]

The well known Google Street View operates on a similar principle, with the main difference that the scene is not a set of photographs but a set of panoramas (computed by eleven high resolution photographs simultaneously taken by a omni-directional camera) and the metaphor for browsing the scene is simpler and more constrained: the user can only move along the next spot on the street. Furthermore, because of the highly constrained positioning of the view, there is little or no 3D geometry involved.

5.4 Considerations

The idea of rendering 3D content by using 2D images dates well back before than Photo Tourism, as mentioned in the previous chapters. The novel aspect introduced by Photo Tourism and follow ups is that images are used only to enrich or improve a 3D description of the scene but they are the scene.

The advantages of this approach are manifold:

- the construction of the scene requires cheaper hardware and less computational effort
- the user interface is generally more friendly than with a general 3D representation, also thanks to the limited number of viewing positions, i.e. moving in a more *discrete* setting
- in a remote setting (for example over the Internet) sending images is easier that sending geometry. Although this specific issue may change in the future, at the present progressive transmission of color images is much more efficient and standardized than transmission of 3D geometry

These approaches work on the assumption that the user only want to *view* the 3D representation, but not do spatial operations with it. If it not so, it should be considered that:

- without a full 3D geometry, it is difficult if not impossible to perform precise measurements on the object
- consequently, simulations on the physical scene cannot be reliably done
- if the real scene changes, to update the set of photographs it may become extremely difficult because the ambiguous situations that may happen.

5.5 Recommendations

Supporting Image collections embedded in 3D space requires the deployment of **specialized** technology for 3D image collection rendering. The goal here should not be on providing photo-realistic views of the world from all viewpoints, but, rather, on creating smooth 3D transitions between photographs and on visually providing information on the pose relative to each image. The approach should not be seen as a substitution of 3D geometry visualization, but as a useful complement.

6 Conclusions and general recommendations

We summarize here the main conclusions and general recommendations for the project.

The objective of using 3D data in the INDIGO project is to provide simple to understand visualizations of a particular environment. These reconstructions will be created to provide means to recognize a particular area both for training and for actual operations. We can think of exploiting different types of sources for creating these reconstructions:

- **Dense 3D models**, such as those created through LiDAR or similar technologies, that provide measurable reconstructions of environments. These models can be exploited for all operations that require some sort of measurement in 3D. As the goal of INDIGO is not photorealistic reconstruction, the project should be able to support the simplest form of aligned data (unordered aligned point clouds) and should not force the creation of complex merged models.
- **Image collections embedded in 3D space**, such as those created with SfM pipelines followed by global alignment steps, that provide sparse but natural depictions of sites. These models can be exploited for all operations that require location recognition. They can augment dense 3D models for improving visualization from specific points of views, or replace them (when possible) in applications that do not require precise measurement.

Both kinds of datasets will be increasingly available in the future and are relatively easy to acquire (at least at a small scale). The focus of INDIGO will be on the creation of a real-time interactive system for navigation between these 3D datasets. Whenever possible, existing technology will be exploited for the acquisition and reconstruction steps.

Supporting dense 3D models within an interactive system requires the deployment of **specialized technology for real-time massive 3D model rendering**. In the INDIGO project, direct point-based rendering seems the most appropriate approach for dense models, since it can be applied both to “raw” acquired data and polished ones. The implemented technology should strive to provide rapid construction of a multiresolution rendering database from already aligned raw data (e.g., in the *las* format), measurement possibilities (picking of 3D points), fast rendering rates and creation of understandable images from raw datasets (e.g, by using illustrative techniques).

Supporting Image collections embedded in 3D space requires the deployment of **specialized technology for 3D image collection rendering**. The goal here should not be on providing photo-realistic views of the world from all viewpoints, but, rather, on creating smooth 3D transitions between photographs and on visually providing information on the pose relative to each image.

As far as hardware resources required, it should be noted that these applications, similarly to many 3D multimedia ones, require non trivial resources. Dense point clouds require hardware rendering rates of millions of points per seconds, and 3D embedded photographs require high performance texturing. Both require good quantities of local memory for data caching. Fortunately, the current hardware trends, driven by the gaming industry, have lead to availability of high performance commodity desktop and portable solutions. The goal for the project should be to support interactive performance on high end desktop and portable platforms, such as PCs/laptops with latest generation graphics boards and, at least, broadband (ADSL) connectivity.

7 Bibliography

[AdeBer] Computational Models of Visual Processing, The Plenoptic Function and the Elements of Early Vision, Adelson, E. H., and J. R. Bergen, The MIT Press, Cambridge, Mass. 1991

[Agarwal and Suri. 1994] AGARWAL, P. K., AND SURI., S. 1994. Surface approximation and geometric partitions. In *Proc. 5th ACM-SIAM Sympos. Discrete Algorithms*, 24? 33.

[Aiger et al. 2008] Aiger, D., Mitra, N. J., and Cohen-Or, D., "4-points Congruent Sets for Robust Surface Registration". *ACM Transactions on Graphics*, Vol 27, N. 3, 2008.

[Airey et al. 1990] AIREY, J. M., ROHLF, J. H., AND BROOKS, JR., F. P. 1990. Towards image realism with interactive update rates in complex virtual building environments. *Computer Graphics (1990 Symposium on Interactive 3D Graphics) 24*, 2 (Mar.), 41–50.

[Alexa et al. 2001] ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, C. T. 2001. Point Set Surfaces. In *Proceedings of IEEE Visualization 2001*, 21–28.

[Aliaga and Lastra 1997] ALIAGA, D. G., AND LASTRA, A. A. 1997. Architectural Walkthroughs Using Portal Textures. In *Proceedings of IEEE Visualization 1997*, 355–362.

[Aliaga et al. 1999] ALIAGA, D. G., COHEN, J., WILSON, A., BAKER, E., ZHANG, H., ERIKSON, C., HOFF III, K. E., HUDSON, T., STÜRZLINGER, W., BASTOS, R., WHITTON, M. C., BROOKS JR., F. P., AND MANOCHA, D. 1999. MMR: An Interactive Massive Model Rendering System using Geometric and Image-Based Acceleration. In *SI3D '99: Proceedings of the 1999 Symposium on Interactive 3D Graphics*, 199–206.

[Alliez and Gotsman 2005]ALLIEZ, P., AND GOTSMAN, C. 2005. *Recent advances in compression of 3D meshes*. Springer, 3–26.

[Andujar et al. 2007] Carlos Andújar, Javier Boo, Pere Brunet, Marta Fairén González, Isabel Navazo, Pere-Pau Vázquez, Alvar Vinacua: Omni-directional Relief Impostors. Comput. Graph. Forum 26(3): 553-560 (2007)

[Andujar et al. 2008] ANDÚJAR, C.,BRUNET P. Relief Impostor Selection for Large Scale Urban Rendering IEEE Virtual Reality Workshop on Virtual Citiscapes: Key Research Issues in Modeling Large-Scale Immersive Urban Environments, 2008

[Andújar et al. 2000] ANDÚJAR, C., SAONA-VÁZQUEZ, C., AND NAVAZO, I. 2000. Lod visibility culling and occluder synthesis. *Computer-Aided Design 32*, 13 (Oct.), 773–783.

[Baboud and Décoret 2006] BABOUD, L., AND DÉCORET, X. 2006. Rendering geometry with relief textures. In *Graphics Interface*, Canadian Human-Computer Communications Society, C. Gutwin and S. Mann, Eds., 195–201.

[Besl et al. 1992] Besl, P.J., McKay, H.D., "A method for registration of 3-D shapes". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1992

[BesMcK] P. J. Besl and N. D. McKay, A method for registration of 3-D shapes, IEEE Transactions on Pattern Analysis and machine Intelligence, vol. 14, pp. 239–258, Feb. 1992.

[Bettio et al. 2007] BETTIO, F., GOBBETTI, E., MARTON, F., AND PINTORE, G. 2007. High-quality networked terrain rendering from compressed bitstreams. In *Proc. ACM Web3D International Symposium*, New York, NY, USA, ACM Press, 37–44.

[Bittner 2002] BITTNER, J. 2002. *Hierarchical Techniques for Visibility Computations*. Ph.d. thesis, Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague.

[Bittner et al. 2001] BITTNER, J., WONKA, P., AND WIMMER, M. 2001. Visibility preprocessing for urban scenes using line space subdivision. In *PG '01: Proceedings of the 9th Pacific Conference on Computer Graphics and Applications*, IEEE Computer Society, Washington, DC, USA, 276.

[Bittner et al. 2004] BITTNER, J., WIMMER, M., PIRINGER, H., AND PURGATHOFER, W. 2004. Coherent hierarchical culling: Hardware occlusion queries made useful. *Computer Graphics Forum 23*, 3, 615–624.

[Bla.etal] BLAIS, F., TAYLOR, J., COURNOYER, L., PICARD, M., BORGEAT, L., DICAIRE, L.-G., RIOUX, M., BERALDIN, J.-A., GODIN, G., LAHNANIER, C., AND AITKEN, G. Ultra-high resolution imaging at 50m using a portable xyz-rgb color laser scanner. In International Workshop on Recording, Modeling and Visualization of Cultural Heritage 2005.

[Bla] Blais, F. A review of 20 years of range sensor development. In Proceedings of SPIEIS&T Electronic Imaging. Vol. 5013. 62–76, 2003

[Borgeat et al 05] BORGEAT L., GODIN G., BLAIS F., MASSICOTTE P., LAHANIER C.: Gold: interactive display of huge colored and textured models. *ACM Trans. Graph. 24*, 3 (2005), 869–877.

[Bre] REUCKMANN B.: Bildverarbeitung und optische Messtechnik in der industriellen Praxis B
Franzis-Verlag, 1993.

[BriTar] Paolo Brivio, Marco Tarini, Paolo Cignoni, Browsing large image datasets through Voronoi diagrams, IEEE Transactions on Visualization and Computer Graphics [in press] Dec 2010

[Brown B. and Rusinkiewicz S. 2007] Brown B. and Rusinkiewicz S., “Global non-rigid alignment of 3-D scans”. *ACM SIGGRAPH 2007*, Vol 26 , N. 3, 2007

[Buchholz and Dollner 2005] BUCHHOLZ H., DÖLLNER J.: View-dependent rendering of multiresolution texture-atlases. In IEEE Visualization (2005), p. 28.

[Callieri et al. 2008] Callieri, M., Cignoni, P., Corsini,, and Scopino, R., “Masked Photo Blending: mapping dense photographic dataset on high-resolution 3D models”. *Computer & Graphics*, Vol. 32, N. 4, pp 464-473, 2008.

[CheMed] Y. Chen and G. Medioni, Object modelling by registration of multiple range images, International Journal of Image and Vision Computing, vol. 10, pp. 145–155, Apr. 1992.

[Chen 1995] CHEN, S. E. 1995. Quicktime VR - an image-based approach to virtual environment navigation. In *SIGGRAPH 95 Conference Proceedings*, Addison Wesley, R. Cook, Ed., Annual Conference Series, ACM SIGGRAPH, 29–38. held in Los Angeles, California, 06-11 August 1995.

[Choe et al. 2004] CHOE, S., KIM, J., LEE, H., LEE, S., AND SEIDEL, H.-P. 2004. Mesh compression with random accessibility. In *Israel-Korea Bi-National conf.*

[Cignoni et al. 2003a] CIGNONI, P., MONTANI, C., ROCCHINI, C., AND SCOPIGNO, R. 2003. External memory management and simplification of huge meshes. *IEEE Transactions on Visualization and Computer Graphics 9*, 525–337.

[Cignoni et al. 2003b] CIGNONI, P., GANOVELLI, F., GOBBETTI, E., MARTON, F., PONCHIO, F., AND SCOPIGNO, R. 2003. BDAM – batched dynamic adaptive meshes for high performance terrain visualization. *Computer Graphics Forum 22*, 3 (September), 505–514. Proc. Eurographics 2003.

[Cignoni et al. 2003c] CIGNONI, P., GANOVELLI, F., GOBBETTI, E., MARTON, F., PONCHIO, F., AND SCOPIGNO, R. 2003. Planet–sized batched dynamic adaptive meshes (p-bdam). In *Proceedings IEEE Visualization*, IEEE Computer Society Press, Conference held in Seattle, WA, USA, 147–155.

[Cignoni et al. 2004] CIGNONI, P., GANOVELLI, F., GOBBETTI, E., MARTON, F., PONCHIO, F., AND SCOPIGNO, R. 2004. Adaptive tetrapuzzles: efficient out-of-core construction and visualization of gigantic multiresolution polygonal models. *ACM Transactions on Graphics 23*, 3 (Aug.), 796–803.

[Cignoni et al. 2005] CIGNONI, P., GANOVELLI, F., GOBBETTI, E., MARTON, F., PONCHIO, F., AND SCOPIGNO, R. 2005. Batched multi triangulation. In *Proceedings IEEE Visualization*, IEEE Computer Society Press, Conference held in Minneapolis, MI, USA, 207–214.

[Cignoni et al. 2007] CIGNONI, P., DI BENEDETTO, M., GANOVELLI, F., GOBBETTI, E., MARTON, F., AND SCOPIGNO, R. 2007. Ray-Casted BlockMaps for Large Urban Models Visualization. In *Computer Graphics Forum (Proceedings of Eurographics)*.

[Cline and Egbert 1998] CLINE D., EGBERT P. K.: Interactive display of very large textures. In *IEEE Visualization* (1998), pp. 343–350.

[Cook 1984] COOK R. L.: Shade trees. *Computer Graphics 18-3* (July 1984), 223–231.

[Cre] CREATH K. Phase-measurement interferometry techniques. Progress in Optics 26 (1988), 349–393.

[Cuccuru et al. 2009] Cuccuru, G., Gobbetti, E., Marton, F., Pajarola, R. and Pintus, R., “Fast low-memory streaming MLS reconstruction of point-sampled surfaces”. *Graphics Interface*, pp. 15-22, 2009.

[CurSei] CURLESS, B. AND SEITZ. 3D photography., S. ACM SIGGRAPH Course Notes, Course 19. 2000

[Darsa et al. 1997] DARSA L., SILVA B. C., VARSHNEY A.: Navigating static environments using image-space simplification and morphing. In *SI3D* (1997), pp. 25–34, 182.

[De Floriani et al. 1998] DE FLORIANI, L., MAGILLO, P., AND PUPPO, E. 1998. Efficient Implementation of Multi-Triangulations. In *Proceedings of IEEE Visualization 1998*, 43–50.

[de Toledo and Levi 2004] DE TOLEDO, R., AND LEVI, B. 2004. Extending the graphic pipeline with new GPU-accelerated primitives. In *Proc. 24th gOcad Meeting*.

[de Toledo et al. 2007] DE TOLEDO, R., LEVY, B., AND PAUL, J.-C. 2007. Iterative methods for visualization of implicit surfaces on gpu. In *ISVC, International Symposium on Visual Computing*, Springer, Lake Tahoe, Nevada/California, Lecture Notes in Computer Science.

[Décoret et al. 1999] DÉCORET X., SILLION F., SCHAUFLER G., DORSEY J.: Multi-layered impostors for accelerated rendering. *Computer Graphics Forum 18*, 3 (Sept. 1999), 61–73. ISSN 1067-7055.

[Deering 1995] DEERING, M. F. 1995. Geometry compression. In *ACM SIGGRAPH*, 13–20.

[Dietrich et al. 2006] DIETRICH, A., SCHMITTLER, J., AND SLUSALLEK, P. 2006. World-space sample caching for efficient ray tracing of highly complex scenes. Tech. Rep. TR-2006-01, Computer Graphics Group, Saarland University.

[Donnelly 2005] DONNELLY W.: *GPU Gems 2*. Addison-Wesley, 2005, ch. Per-Pixel Displacement Mapping with Distance Functions, pp. 123–136.

[Duguet and Drettakis 2002] DUGUET, F., AND DRETTAKIS, G. 2002. Robust epsilon visibility. In *Proceedings of ACM SIGGRAPH 2002*, ACM Press / ACM SIGGRAPH, J. Hughes, Ed.

[Durand 1999] DURAND, F. 1999. *3D Visibility: Analytical study and Applications*. PhD thesis, Universite Joseph Fourier, Grenoble, France.

[Durand et al. 2000] DURAND, F., DRETTAKIS, G., THOLLOT, J., AND PUECH, C. 2000. Conservative visibility preprocessing using extended projections. In *SIGGRAPH 00 Conference Proceedings*, 239–248.

[GelMit] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration , in In M. Desbrun and H.
Pottmann, editors, Eurographics Association, ISBN 3-905673-24-X., pp. 197–206, 2005.

[Gobbetti and Marton 2004] GOBBETTI, E., AND MARTON, F. 2004. Layered point clouds: a simple and efficient multiresolution structure for distributing and rendering gigantic point-sampled models. *Computers & Graphics 28*, 6 (Dec.), 815–826.

[Gobbetti and Marton 2005] GOBBETTI, E., AND MARTON, F. 2005. Far Voxels – a multiresolution framework for interactive rendering of huge complex 3d models on commodity graphics platforms. *ACM Transactions on Graphics 24*, 3, 878–885.

[Gobbetti et al. 2006] GOBBETTI, E., MARTON, F., CIGNONI, P., DI BENEDETTO, M., AND GANOVELLI, F. 2006. C-BDAM – compressed batched dynamic adaptive meshes for terrain rendering. *Computer Graphics Forum 25*, 3 (September), 333–342. Proc. Eurographics 2006.

[Gobbetti et al. 2008] GOBBETTI, E., YOON, S.-E., KASIK, D. 2008. Technical Strategies for Massive Model Rendering. Proc. SPM 2008.

[Goldsmith and Salmon 1987] GOLDSMITH, J., AND SALMON, J. 1987. Automatic creation of object hierarchies for ray tracing. *IEEE Comput. Graph. Appl. 7*, 5, 14–20.

[Goldstein 1981] GOLDSTEIN, R. 1981. Defining the bounding edges of a synthavision solid model. In *18th Conference on Design Automation*, 457–461.

[Gortler et al. 1996] GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R., AND COHEN, M. F. 1996. The lumigraph. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, 43–54.

[Gotsman et al. 2002] GOTSMAN, C., GUMHOLD, S., AND KOBBELT, L. 2002. *Simplification and Compression of 3D Meshes*. Springer, 319–361.

[Govindaraju et al. 2003] GOVINDARAJU, N. K., SUD, A., YOON, S.-E., AND MANOCHA, D. 2003. Interactive visibility culling in complex environments using occlusion-switches. In *2003 ACM Symposium on Interactive 3D Graphics*, 103–112.

[Grossman and Dally 1998] GROSSMAN, J., AND DALLY, W. J. 1998. Point Sample Rendering. In *Rendering Techniques 1998 (Proceedings of the Eurographics Workshop on Rendering)*, 181–192.

[Gu et al. 2002] GU, X., GORTLER, S. J., AND HOPPE, H. 2002. Geometry Images. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 335–361.

[GuiBer] G. Guidi, J.-A. Beraldin, and C. Atzeni. High-accuracy 3D modeling of cultural heritage: the digitizing of donatello's Maddalena. IEEE Transactions on Image Processing, vol. 13, no. 3, pp. 370 – 380, 2004.

[Haumont et al. 2005] HAUMONT, D., MAKINEN, O., AND NIRENSTEIN, S. 2005. A low dimensional framework for exact polygon-to-polygon occlusion queries. In *Rendering Techniques*, Eurographics Association, O. Deussen, A. Keller, K. Bala, P. Dutr? , D. W. Fellner, and S. N. Spencer, Eds., 211–222.

[Havran 2000] HAVRAN, V. 2000. *Heuristic Ray Shooting Algorithms*. Ph.d. thesis, Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague.

[Havran et al. 2006] HAVRAN, V., HERZOG, R., AND SEIDEL, H.-P. 2006. On the fast construction of spatial data structures for ray tracing. In *Proceedings of IEEE Symposium on Interactive Ray Tracing 2006*, 71–80.

[HeuSin] Heung-Yeung Shum, Sing Bing Kang, Shing-Chow Chan. Survey of image-based representations and compression techniques, Microsoft Res. Asia, Beijing, China Circuits and Systems for Video Technology, IEEE Transactions on Volume:13, Nov. 2003 Issue:11, pp. 1020 - 103

[Heyer et al. 2005] HEYER, M., PFÜTZER, S., AND BRÜDERLIN, B. 2005. Visualization Server for Very Large Virual Reality Scenes. In *4. Paderborner Workshop Augmented & Virtual Reality in der Produktentstehung*.

[Hoppe 1998] HOPPE, H. 1998. Smooth view-dependent level-of-detail control and its aplications to terrain rendering. In *IEEE Visualization '98 Conf.*, 35–42.

[Hoppe 1999] HOPPE, H. 1999. Optimization of mesh locality for transparent vertex caching. *ACM SIGGRAPH*, 269–276.

[Hunt et al. 2006] HUNT, W., MARK, W. R., AND STOLL, G. 2006. Fast kd-tree construction with an adaptive error-bounded heuristic. In *2006 IEEE Symposium on Interactive Ray Tracing*, IEEE.

[Isenburg et al. 2003] ISENBURG, M., LINDSTROM, P., GUMHOLD, S., AND SNOEYINK, J. 2003. Large Mesh Simplification using Processing Sequences. In *Proceedings of IEEE Visualization 2003*, 465–472.

[Jeschke and Wimmer 2002] JESCHKE, S., AND WIMMER, M. 2002. Textured depth meshes for realtime rendering of arbitrary scenes. In *Proceedings of the 13th Eurographics Workshop on Rendering (RENDERING TECHNIQUES-02)*, Eurographics Association, Aire-la-Ville, Switzerland, S. Gibson and P. Debevec, Eds., 181–190.

[Jeschke et al. 2002] JESCHKE, S., WIMMER, M., AND SCHUMANN, H. 2002. Layered environment-map impostors for arbitrary scenes. In *Graphics Interface*, 1–8.

[Klosowski and Silva 2001] KLOSOWSKI, J. T., AND SILVA, C. T. 2001. Efficient conservative visibility culling using the prioritized-layered projection algorithm. *IEEE Transactions on Visualization and Computer Graphics 7*, 4, 365–379.

[Koltun et al. 2001] KOLTUN, V., CHRYSANTHOU, Y., AND COHEN-OR, D. 2001. Hardware-accelerated from-region visibility using a dual ray space. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, Springer-Verlag, London, UK, 205–216.

[Krishnamurthy et al. 2007] KRISHNAMURTHY, A., KHARDEKAR, R., AND MCMAINS, S. 2007. Direct evaluation of nurbs curves and surfaces on the gpu. In *SPM '07: Proceedings of the 2007 ACM symposium on Solid and physical modeling*, ACM, New York, NY, USA, 329–334.

[Lauterbach et al. 2006] LAUTERBACH, C., YOON, S.-E., TUFT, D., AND MANOCHA, D. 2006. Rt-deform: Interactive ray tracing of dynamic scenes using bvhs. In *Proceedings of IEEE Symposium on Interactive Ray Tracing 2006*.

[Lauterbach et al. 2007] LAUTERBACH, C., YOON, S.-E., AND MANOCHA, D. 2007. Ray-strips: A compact mesh representation for interactive ray tracing. In *IEEE/EG Symposium on Interactive Ray Tracing*, 19–26.

[Leberl et al. 2010] Leberl Franz, Bischof Horst, Pock Thomas, Irschara Arnold , Kluckner Stefan, Aerial Computer Vision for a 3D Virtual Habitat. IEEE Computer, 2010.

[Levoy and Hanrahan 1996] LEVOY, M., AND HANRAHAN, P. 1996. Light field rendering. In *SIGGRAPH 96 Conference Proceedings*, 31–42.

[Leyvand et al. 2003] LEYVAND, T., SORKINE, O., AND COHEN-OR, D. 2003. Ray space factorization for from-region visibility. *ACM Transactions on Graphics 22*, 3 (July), 595–604.

[Lippman 1980] LIPPMAN, A. 1980. Movie-maps: An application of the optical videodisc to computer graphics. *Computer Graphics (SIGGRAPH ? 80 Proceedings) 14*, 3 (July), 32? 42.

[Loop and Blinn 2006] LOOP, C., AND BLINN, J. 2006. Real-time gpu rendering of piecewise algebraic surfaces. *ACM Transactions on Graphics 25*, 3 (July), 664–670.

[Luebke 2001] LUEBKE, D. P. 2001. A Developer's Survey of Polygonal Simplification Algorithms. *IEEE Computer Graphics and Applications 21*, 3, 24–35.

[MacDonald and Booth 1990] MACDONALD, J. D., AND BOOTH, K. S. 1990. Heuristics for ray tracing using space subdivision. *Visual Computer*.

[Maciel and Shirley 1995] MACIEL P. W. C., SHIRLEY P.: Visual navigation of large environments using textured clusters. In *SI3D* (1995), pp. 95–102, 211.

[Mal] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. IEEE Trans. on Patt. Anal. and Mach. Intel. 11, 7 (1989), 674–693.

[Manuel and Oliveira 2005] MANUEL M.OLIVEIRA F. P.: *An Efficient Representation for Surface Details*. Tech. Rep. RP 351, Universidade Federal do Rio Grande, January 2005.

[McMillan and Bishop 1995] MCMILLAN, L., AND BISHOP, G. 1995. Plenoptic Modeling: An Image-Based Rendering System. In *ACM Computer Graphics (Proceedings of ACM SIGGRAPH)*, 39–46.

[Mora et al. 2005] MORA, F., AVENEAU, L., AND MÉRIAUX, M. 2005. Coherent and exact polygon-to-polygon visibility. In *Proc. WSCG*, 87–94.

[Nirenstein and Blake 2004] NIRENSTEIN, S., AND BLAKE, E. 2004. Hardware accelerated visibility preprocessing using adaptive sampling. In *Rendering Techniques 2004: 15th Eurographics Workshop on Rendering*, 207–216.

[Nirenstein et al. 2002] NIRENSTEIN, S., BLAKE, E., AND GAIN, J. 2002. Exact from-region visibility culling. In *EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering*, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 191–202.

[Oliveira et al. 2000] OLIVEIRA, M. M., BISHOP, G., AND MCALLISTER, D. 2000. Relief Texture Mapping. In *ACM Computer Graphics (Proceedings of ACM SIGGRAPH)*, 359–368.

[Patterson et al. 1997] PATTERSON, D., ANDERSON, T., CARDWELL, N., FROMM, R., KIMBERLY, KEATON, CHRISTOFOROSKAZYRAKIS, THOMAS, R., AND YELLICK, K. 1997. A case for intelligent ram. *IEEE Micro.*.

[Pingi et al. 2005] Pingi, P., Fasano, A., Cignoni, P., Montani, C., and Scopigno, R., "Exploiting the scanning sequence for automatic registration of large sets of range maps". *Computer Graphics Forum*, N. 3, Vol. 24, pp. 517-526, 2005.

[Policarpo et al. 2005] POLICARPO, F., OLIVEIRA, M. M., AND COMBA, J. L. D. 2005. Real-time relief mapping on arbitrary polygonal surfaces. *ACM Trans. Graph 24*, 3, 935.

[Popov et al. 2006] POPOV, S., GÜNTHER, J., SEIDEL, H.-P., AND SLUSALLEK, P. 2006. Experiences with streaming construction of SAH KD-trees. In *Proceedings of the 2006 IEEE Symposium on Interactive Ray Tracing*, 89–94.

[Prince 2000] PRINCE, C. 2000. *Progressive Meshes for Large Models of Arbitrary Topology*. Master's thesis, Department of Computer Science and Engineering, University of Washington, Seattle.

[Pul] K. Pulli, Multiview registration for large datasets. in Proc 2nd Int.l Conf. on 3D Digital Imaging and Modeling, pp. 160–168, IEEE, 1999.

[Reshetov et al. 2005] RESHETOV, A., SOUPIKOV, A., AND HURLEY, J. 2005. Multi-Level Ray Tracing Algorithm. In *ACM Transaction of Graphics (Proceedings of ACM SIGGRAPH)*, 1176–1185.

[Ruemmler and Wilkes 1994] RUEMMLER, C., AND WILKES, J. 1994. An introduction to disk drive modeling. *IEEE Computer*.

[Rusinkiewicz and Levoy 2000a] RUSINKIEWICZ, S., AND LEVOY, M. 2000. Qsplat: A multiresolution point rendering system for large meshes. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, 343–352.

[Rusinkiewicz and Levoy 2000b] RUSINKIEWICZ, S., AND LEVOY, M. 2000. QSplat: A Multiresolution Point Rendering System for Large Meshes. In *Computer Graphics (Proceedings of ACM SIGGRAPH)*, 343–352.

[Sagan 1994] SAGAN, H. 1994. *Space-Filling Curves*. Springer-Verlag.

[Samet 2006] SAMET, H., Ed. 2006. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann.

[Schaufler et al. 2000] SCHAUFLER, G., J.DORSEY, DECORET, X., AND SILLION, F. 2000. Conservative volumetric visibility with occluder fusion. In *SIGGRAPH 00 Conference Proceedings*, 229–238.

[SchSze04] S A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms Charstein D., Szeliski R., International Journal of Computer Vision, pp. 7-42, Volume 47, n. 1-3

[Shade et al. 1998] SHADE, J., GORTLER, S., HE, L., AND SZELISKI, R. 1998. Layered Depth Images. In *Computer Graphics (Proceedings of ACM SIGGRAPH)*, 231–242.

[Shevtsov et al. 2007] SHEVTSOV, MAXIM, SOUPIKOV, ALEXEI, KAPUSTIN, AND ALEXANDER. 2007. Highly parallel fast kd-tree construction for interactive ray tracing of dynamic scenes. *Computer Graphics Forum 26*, 3 (September), 395–404.

[Sigg et al. 2006]SIGG, C., WEYRICH, T., BOTSCH, M., AND GROSS, M. 2006. Gpu-based ray-casting of quadratic surfaces. In *Symposium on Point - Based Graphics 2006*, 59–66.

[Sillion et al. 1997] SILLION, F., DRETTAKIS, G., AND BODELET, B. 1997. Efficient Impostor Manipulation for Real-Time Visualization of Urban Scenery. In *Computer Graphics Forum (Proceedings of Eurographics)*, 207–218.

[Snavely 2006] Noah Snavely, Steven M. Seitz, Richard Szeliski. Photo Tourism: Exploring image collections in 3D. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006), 2006.

[Snavely 2007] Noah Snavely, Steven M. Seitz, Richard Szeliski. Modeling the World from Internet Photo Collections. International Journal of Computer Vision, 2007.

[Stamminger and Drettakis 2001] STAMMINGER, M., AND DRETTAKIS, G. 2001. Interactive Sampling and Rendering for Complex and Procedural Geometry. In *Proceedings of the Eurographics Workshop on Rendering Techniques*, 151–162.

[Tanner et al. 1998] TANNER C. C., MIGDAL C. J., JONES M. T.: Theclipmap: A virtual mipmap. In *SIGGRAPH* (1998), pp. 151–158.

[Tarini et al. 2006] TARINI, M., CIGNONI, P., AND MONTANI, C. 2006. Ambient occlusion and edge cueing for enhancing real time molecular visualization. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (Sept./Oct.), 1237–1244.

[Teller and Sequin 1991] TELLER, S., AND SEQUIN, C. 1991. Visibility preprocessing for interative walkthroughs. *Computer Graphics (SIGGRAPH 91 Proceedings) 25*, 4 (July), 61–69.

[van Emde Boas 1977] VAN EMDE BOAS, P. 1977. Preserving order in a forest in less than logarithmic time and linear space. *Inf. Process. Lett.*.

[VerVan] Maarten Vergauwen and Luc Van Gool. Web-based 3D reconstruction service, Machine Vision and Applications (2006) 17:411–426

[Wächter and Keller 2006] WÄCHTER, C., AND KELLER, A. 2006. Instant ray tracing: The bounding interval hierarchy. In *Proceedings of the Eurographics Symposium on Rendering*, 139–149.

[Wald 2007] WALD, I. 2007. On fast construction of sah based bounding volume hierarchies. In *Proceedings of the 2007 Eurographics/IEEE Symposium on Interactive Ray Tracing*.

[Wald and Havran 18-20] WALD, I., AND HAVRAN, V. 18-20. On building fast kd-trees for ray tracing, and on doing that in o(n log n). In *Proceedings of IEEE Symposium on Interactive Ray Tracing 2006*, 61–69.

[Wald et al. 2001] WALD, I., SLUSALLEK, P., BENTHIN, C., AND WAGNER, M. 2001. Interactive rendering with coherent ray tracing. *Computer Graphics Forum 20*, 3, 153–164.

[Wald et al. 2007] WALD, I., BOULOS, S., AND SHIRLEY, P. 2007. Ray tracing deformable scenes using dynamic bounding volume hierarchies. *ACM Transactions on Graphics 26*, 1, 6.1–6.10.

[Wand et al. 2001] WAND, M., FISCHER, M., PETER, I., AUF DER HEIDE, F. M., AND STRAßER, W. 2001. The Randomized z-Buffer Algorithm: Interactive Rendering of Highly Complex Scenes. In *Computer Graphics (Proceedings of ACM SIGGRAPH)*, 361–370.

[Wang et al. 2003] WANG, L., WANG, X., TONG, X., LIN, S., HU, S.-M., GUO, B., AND SHUM, H.-Y. 2003. View-Dependent Displacement Mapping. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 334–339.

[Wang et al. 2004] WANG, X., TONG, X., LIN, S., HU, S., GUO, B., AND SHUM, H.-Y. 2004. Generalized displacement maps. In *Proceedings of the 2004 Eurographics Symposium on Rendering*, Eurographics Association, D. Fellner and S. Spencer, Eds., 227–234.

[Weyrich et al. 2007] WEYRICH, T., FLAIG, C., HEINZLE, S., MALL, S., AILA, T., ROHRER, K., FASNACHT, D., FELBER, N., OETIKER, S., KAESLIN, H., BOTSCH, M., AND GROSS, M. 2007. A Hardware Architecture for Surface Splatting. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, 90.

[Wilson and Manocha 2003] WILSON, A., AND MANOCHA, D. 2003. Simplifying Complex Environments Using Incremental Textured Depth Meshes. In *ACM Transactions on Graphics (Proceedings of ACM SIGGRPAH)*, 678–688.

[Wimmer et al. 2001] WIMMER, M., WONKA, P., AND SILLION, F., 2001. Point-based impostors for real-time visualization, May 29.

[Wonka et al. 2000] WONKA, P., WIMMER, M., AND SCHMALSTIEG, D. 2000. Visibility preprocessing with occluder fusion for urban walkthroughs. In *11th Eurographics Workshop on Rendering*, 71–82.

[Wonka et al. 2006] WONKA, P., WIMMER, M., ZHOU, K., MAIERHOFER, S., HESINA, G., AND RESHETOV, A. 2006. Guided visibility sampling. *ACM Transactions on Graphics 25*, 3 (July), 494–502.

[Woop et al. 2006] WOOP, S., MARMITT, G., AND SLUSALLEK, P. 2006. B-KD Trees for Hardware Accelerated Ray Tracing of Dynamic Scenes. In *Proceedings of Graphics Hardware*.

[Wu and Kobbelt 2003] WU, J., AND KOBBELT, L. 2003. A stream algorithm for the decimation of massive meshes. In *Proc. Graphics Interface*, 185–192.

[Yoon and Lindstrom 2006] YOON, S.-E., AND LINDSTROM, P. 2006. Mesh layouts for block-based caches. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization) 12*, 5.

[Yoon and Lindstrom 2007] YOON, S.-E., AND LINDSTROM, P. 2007. Random-accessible compressed triangle meshes. *IEEE Transactions on Visualization and Computer Graphics (Proceedings Visualization)*.

[Yoon and Manocha 2006]YOON, S.-E., AND MANOCHA, D. 2006. Cache-efficient layouts of bounding volume hierarchies. *Computer Graphics Forum (Eurographics) 25*, 507–516.

[Yoon et al. 2004] YOON, S.-E., SALOMON, B., GAYLE, R., AND MANOCHA, D. 2004. Quick-VDR: Interactive View-Dependent Rendering of Massive Models. In *Proceedings of IEEE Visualization 2004*, 131–138.

[Yoon et al. 2005] YOON, S.-E., LINDSTROM, P., PASCUCCI, V., AND MANOCHA, D. 2005. Cache-Oblivious Mesh Layouts. *Proc. of ACM SIGGRAPH*.

[Yoon et al. 2006] YOON, S.-E., LAUTERBACH, C., AND MANOCHA, D. 2006. R-LODs: Fast LOD-Based Ray Tracing of Massive Models. *The Visual Computer 22*, 9–11, 772–784.

[Yoon et al. 2007] YOON, S., CURTIS, S., AND MANOCHA, D. 2007. Ray tracing dynamic scenes using selective restructuring. *Proc. of Eurographics Symposium on Rendering*.