New methods and tools for 3D-modeling of large scale outdoor scenes using range and color images

Alejandro J. Troccoli

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2007

ABSTRACT

New methods and tools for 3D-modeling of large scale outdoor scenes using range and color images

Alejandro J. Troccoli

Systems for the creation of photorealistic models using range scans and digital photographs are becoming increasingly popular in a wide range of fields, from reverse engineering to cultural heritage preservation. These systems employ a range finder to acquire the geometry information and a digital camera to measure color detail. But bringing together a set of range scans and color images to produce an accurate and usable model is still an area of research with many unsolved problems.

In this dissertation we present new tools and methods for creating digital models from range and color images, with emphasis in large-scale outdoor scenes. First, we address the problem of range and color image registration. In this area, we introduce a semi-automatic tool for range and color image registration that makes use of line-features to solve for the position and orientation of the digital camera. This allows us to efficiently register images of urban scenery. Secondly, we present a registration technique that uses the shadows cast by the sun as cues find the correct camera pose, which we have successfully applied in the creation of a digital model of an archaeological excavation in Monte Polizzo, Sicily.

We also address the problem of how to build seamless integrated texture maps from images that were taken under different illumination conditions. To achieve this we present two different solutions. The first one is to align all the images to the same illumination. For this, we have developed a technique that computes a relighting operator over the area of overlap of a pair of images, which we then use to relight the entire image. Our proposed method can handle images with shadows and can effectively remove the shadows from the

image, if required. The second technique uses the ratio of two images to factor out the diffuse reflectance of an image from its illumination. We achieve this without any light measuring device. By computing the actual reflectance we remove from the images any effects of the illumination, which then allows us to create new renderings under novel illumination conditions.

Contents

List of Figures

List of Tables

Acknowledgements

First of all, I am very grateful to my advisor, Peter Allen, for his unconditional support and guidance throughout these years. His confidence in me has been invaluable and kept me moving forward. I also would like to thank my other committee members, Shree Nayar, Ravi Ramamoorthi, Ioannis Stamos and Michael Grossberg, for their advice and support, and for setting an example on how to conduct high quality scientific research. Their works have been a source of inspiration.

As a member of the robotics lab, it has been a great pleasure to work with smart and enthusiastic people. I would like to thank Atanas Georgiev, Paul Blaer, Andrew Miller, Matei Ciocarlie, Corey Goldfeder and Benjamin Smith for the amount of time we spent together, the technical and not so technical discussions, and their invaluable suggestions. In particular, special thanks go to Atanas and Paul, who know every little detail about the systems in the lab and were always available to give a hand.

I am grateful to my friends Agustin Gravano, Sebastian Enrique, Hrvoje Benko, Eddie Ishak, Gabor Blasko, Sinem Guven, Kshitiz Garg, Rahul Swaminathan, Vanessa Frias-Martinez, Fabiola Brusciotti, Nikolaos Egglezos, Angelos Stavrou and Konstantinos Kardaras, for they have directly or indirectly helped made this work possible.

The confidence deposited by my parents and siblings in me throughout these years has always been a source of inspiration. Special thanks go to my father Osvaldo, my mother Pochi, my brother Pablo and my sister Carla. Finally, me deepest gratitude goes to my

wife Carimer, whose patience, advice and support have been invaluable.

To Pepuna.

Chapter 1

Introduction

To create a visual reproduction of the world that surrounds us has always been a problem that attracted a wide variety of people: from those interested purely in the magnificence of color, light and forms; to those mostly concerned with matter, atoms and photons. For many centuries, recreating the world in stone or paper has been an art in itself; but today, with the advent of computers and different kind of color and range sensors, creating accurate visualizations of the world is a major problem of interest to the sciences. In fact, the task of building digital geometric and photometric models of our surroundings has become a major area of research in the computer graphics, vision, and robotics communities. Ultimately, the goal is to build mathematical models and numerical representations that can explain why objects look like they do. However, understanding visual appearance to the point of being able to build these models is a daunting task, because ultimately, what we perceive is the result of complex interactions between light, object geometry, surface reflectance and sensor response. Only by reasoning about the effects of each of these interactions will we be able to build meaningful models and representations. The problem extends beyond its theoretical importance, since there exists a wide array of practical applications that demand highly accurate geometric and photometric models: Virtual Reality, Digital Cinematography and

Animation, Tele-Presence, and Cultural Heritage Preservation, are some of them, to name a few.

The development of highly precise range sensors, multi-million pixel cameras and personal computers with commodity 3D graphics hardware, has simplified significantly the task of acquiring digital models. Nevertheless, putting this raw input data together into a usable form is still an area with unsolved problems. Much progress has been made recently in automating the acquisition and modeling of small objects in controlled laboratory environments [Lensch *et al.*, 2003, Bernardini *et al.*, 2002]. Outside controlled environments, however, the digital modeling task becomes more difficult and new challenges arise.

In this dissertation we present new methods and tools for modeling large scale outdoor scenes. In these area, two major applications that have captured much attention are urban modeling and cultural heritage preservation. Research in the urban modeling is mostly directed to developing complete systems that can acquire and process large city data in a fast, systematic way, and with minimum human interaction. Since modeling a city requires covering vast areas with different types of sensors [Früh, 2002], much emphasis is given to speed of acquisition and data processing, but less attention is paid to detail preservation and completeness. In contrast, cultural heritage preservation, is more geared towards detail acquisition, accuracy and completeness. Projects in Cultural Heritage Preservation include the modeling of statues, archaeological sites, and historic buildings [Ikeuchi *et al.*, 2003] [Beraldin *et al.*, 2002], [Allen *et al.*, 2001], [Levoy *et al.*, 2000]. These models are intended to serve art historians and archaeologists, for various purposes including research, digital archiving and teaching.

The set of tools and methods that we present in this dissertation were motivated by real-world interdisciplinary projects in which we have collaborated with people in art history, archaeology, and other disciplines of Computer Science. The first application is city modeling, for which we are building a mobile robot platform to autonomously explore and acquire a 3D model. The main goal is to build geometrically accurate and photometrically correct models of complex outdoor urban environments. In addition, we have been collab-

orating with Prof. Stephen Murray to build a 3D model of the cathedral of St. Pierre at Beauvais, France [Allen *et al.*, 2003]. This cathedral is the tallest in Western Europe and has suffered from partial collapses more than once. A digital 3D model can provide insightful information on its current state, allowing to identify potential structural problems. It can also serve as a teaching aid, giving both faculty and students the possibility of taking a virtual tour. Also with Prof. Murray, we have worked in modeling a set of Romanesque Churches of the Bourbonnais region, in France. The 3D models we have built enable researchers to make shape comparisons, take measurements, and hypothesize on the stages taken during the construction process, and the evolution of the buildings through time. Lastly, we have worked with archaeologists to build a model of an active archaeological site, the Acropolis of Monte Polizzo, in Sicily [Troccoli and Allen, 2004, Allen *et al.*, 2004]. Dense 3D modeling using range scans is an invaluable tool in archaeology for both, analysis and documentation. An excavation of an archaeological site is in itself a destructive process. A layer with materials from a certain age or period needs to be removed to uncover other layers. By using 3D range scans combined with images, we can capture an accurate snapshots of the site.

In this dissertation we address some of the challenges that we have found make the modeling of large-scale outdoor scenes a difficult problem. We introduce new methods and tools that help reduce the gap between the raw input data and a complete integrated photorealistic models. The examples that we show are taken from each of the projects in which we were involved.

1.1 Problem statement

The process of building a digital model begins with the acquisition of a set of range and digital color images. Range images consist of 3D measurements in the form of (x, y, z) coordinates. Color images measure the amount of reflected light in three separate color bands. Together, these images provide an accurate description of the geometry and appearance of

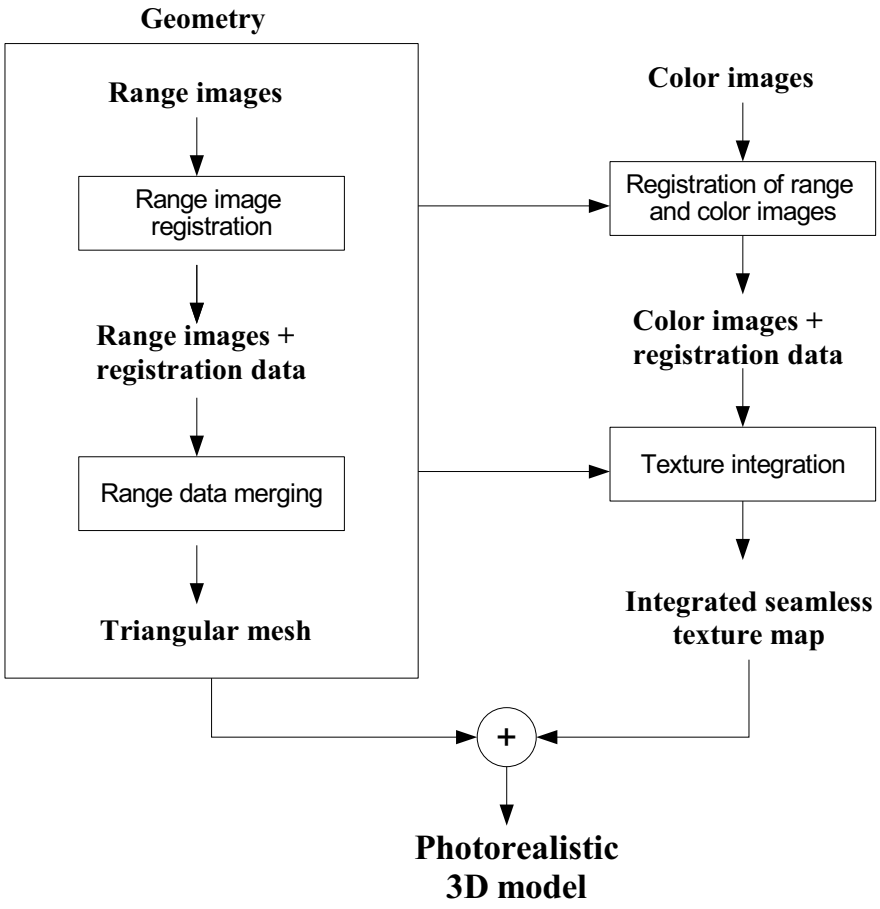

Figure 1.1: The digital modeling pipeline

an object.

Figure 1.1 shows the steps required to obtain a photorealistic 3D model from the acquired raw data. The range images are registered into the same coordinate system and merged into a usable representation, typically a triangular mesh. The color images are aligned to the geometry and an integrated into a seamless texture map. Finally, the geometry and texture maps are combined to produce a photorealistic 3D model. In an ideal scenario, a completely automated system would take the range and color images, move through the steps described above, and produce the final model without any user intervention. Currently, such a system

is not available because the underlying problems that must be solved to complete each of the tasks in the modeling process are difficult to automate. The major challenges at each stage are:

Range image registration. Range image registration is solved by finding correspondences between a pair of range images. Sometimes these correspondences can be established by placing special objects in the scene that act as fiducial marks. But in the more general case, these correspondences are established by a human and used to compute a coarse alignment that is later refined using an iterative algorithm (e.g. [Besl and McKay, 1992]). Automatic feature extraction and matching is a challenging problem.

Range and color image registration. The registration of range and color images is solved by finding correspondences between the range data and the image data, and solving for the parameters of a projection model that describes how the range data is mapped to the image. Again, automatic feature extraction and matching is a major challenge.

Integration of color images into seamless texture maps. When working in environments when one has no control over the illumination, the creation of seamless texture maps becomes a difficult problem if the illumination changes throughout the acquisition process. A change in illumination will change visual appearance of an object, hence integrating multiple images that were acquired under variable illumination will introduce seams in the model.

Each challenge mentioned above represents a major research problem. Despite all the attention these problems have received from the research community, they still remain mostly unsolved. These are important problems that need to be addressed to enable modeling of large-scale environments. Some of these problems are intractable in the general case, but solutions for specific application domains are achievable.

In this dissertation, we introduce new methods and tools for *range and color image registration* and for the *integration of color images into seamless texture maps*, with special

emphasis on urban modeling and the creation of models of historic buildings and archaeological excavations. These domains share some common properties: they are generally of large-scale and are set in outdoor environments, in which we have no control over the illumination. Hence, one of the major problems we look into is how to create integrated texture maps from images acquired under different illuminations.

1.2 Contributions

Our main contributions are:

1. **A semi-automatic method for the registration of range and color images of architectural scenes.** Our algorithm exploits the abundance of parallel line-features in architectural environments to solve the registration problem. Details are given in **chapter 3**.

2. **An algorithm for the registration of range and color images that uses the shadows cast by the sun.** For application domains where the parallel line features are not common, we developed a registration method that uses the shadows cast by the sun to find the position of the camera. Details are given in **chapter 4**.

3. **A relighting algorithm for the creation of seamless texture maps.** One possible way of creating seamless texture maps from images acquired under different lighting conditions is to bring all images to the same illumination by means of a relighting operation. In **chapter 5** we introduce an algorithm that computes a relighting operator from the region of overlap of two images. This operator can then be applied to the entire image to change its illumination. Our method works in the presence of cast shadows, and can remove the shadows from the images.

4. **An algorithm for computing illumination-free texture maps.** This algorithm, which is presented in **chapter 6**, separates the shading from the texture to compute a diffuse reflectance map (a.k.a. albedo map). We built our algorithm on the observation

that the ratio of two images of a diffuse object under different illuminations does not depend on the object's texture. From the ratio image we recover a parametric model of the illumination for each of the two images and then compute a diffuse reflectance map of the scene. This reflectance map can then be used to create renderings of the object under novel illumination conditions. In contrast to previous work, our algorithm does not require measurements of the scene illumination.

Chapter 2

Related work

There has been significant research in algorithms for building digital models using range and color images, conducted by researchers working in computer vision, computer graphics and robotics. In this chapter we present relevant related work, with particular emphasis in the areas of registration of range and color images, and the creation of integrated seamless integrated texture maps.

2.1 Range and color image registration

The registration of range and color images is a problem that is closely related to **camera calibration** and **pose estimation**. The projection of a 3D point by a camera into a 2D image is described by a mapping that depends on the position and orientation of the camera, and on its internal parameters, which define a projection model. Camera calibration is the problem of measuring the internal parameters of a camera, which for perspective cameras are: the principal point, focal length and lens distortion coefficients, plus the camera position and orientation. Pose estimation is the problem of estimating these last two for a camera with known internal parameters. The solution to these problems is generally achieved by computing a set of corresponding 3D and 2D features. Typically, these features are in the form of points and lines, but recent algorithms have employed other types of features, such as

silhouettes [Lensch *et al.*, 2001], for example. In addition, there exist registration algorithms that do not use features. Instead, these algorithms are based on intensity comparison between a rendering of the model and the actual image, searching the space of camera parameters for a point that maximizes the mutual dependence between the two images.

2.1.1 Feature-based algorithms

Feature-based algorithms are extensively used in camera calibration. Typical camera calibration algorithms [Tsai, 1987], [Heikkila and Silven, 1997], [Zhang, 2000], [Bouguet, 2001] use point and line correspondences. Since an accurate calibration requires a large number of correspondences and high-quality measurements, most methods use photographs of a specially designed pattern (typically a planar checkerboard pattern) from which the 3D and 2D features can easily be estimated. Depending on the imaging model, the number of correspondences needed for a calibration varies. The least number of points required for solving the camera calibration without lens distortion parameters is six, and the solution is found using the direct linear transform (DLT) algorithm [Hartley and Zisserman, 2000]. Solving for the lens distortion coefficients requires a large number of correspondences distributed all over the image for accurate results.

In the context of 3D photorealistic modeling using range and color images, camera calibration and pose estimation methods work on 3D and 2D feature sets that are extracted from the range and color images. Feature extraction is followed by feature matching, and the resulting set of correspondences is used to compute the camera parameters and pose. In some instances, the camera internal calibration can be performed before-hand. Then, the registration problem reduces to a camera pose problem with known internal parameters, contributing to robustness and reducing the number of required feature matches.

Some modeling systems completely avoid the camera pose problem by design. For example, certain laser-stripe scanners can capture both texture and geometry with the same camera and hence produce images that are already aligned to the geometry [Pulli *et al.*, 1997, Bernardini *et al.*, 2002] (although in some cases different 3D views will have to be aligned

together). Other system designers fix the camera rigidly to the range sensor and run a calibration only once in a laboratory environment using specially designed fiducial points or landmarks of known geometry. A system with this characteristics was developed for the Digital Michelangelo project of Levoy et al. [Levoy *et al.*, 2000]. Pre-calibration, when possible, has the advantage of avoiding any user driven post-processing at the expense of sacrificing flexibility, because the two sensors are then co-located in space and time. Having the two sensors fixed in a relative position can be problematic when these have very different characteristics (imaging range, resolution and field of view). Additionally, both color and range images must be acquired at the same time, with no possibility of re-imaging if for example, the illumination conditions are not good. Finally, appropriate measures must be taken to guarantee that the calibration remains constant during the whole scanning process.

The limitations of the fixed-arrangement can be overcome when the camera is allowed to be freely placed in space and time. In this case, however, each color image will need to be registered. Different kind of registration algorithms have been proposed for different types of applications. For example, in acquiring models of small objects (the size of a vase), Rocchini *et al.* (1999) employ a two step process: first a user manually selects corresponding points which are used to compute the camera projection; and later in a second phase, a local search procedure uses a correlation algorithm to find a better registration and re-compute the camera parameters. Lensch *et al.* (2001) present an automatic method for image registration based on silhouette matching, where the contour of a rendered version of the object is matched against the silhouette of the object in the image. Their algorithm does not require any user intervention, but their method is limited to cases where a single image completely captures the object.

Some range finders return, in addition to the 3D coordinates of each measurement, a reflected intensity value. The returned reflectance values can be put together into a 2D reflectance image, and used for registration. Ikeuchi *et al.* (2003) propose a registration algorithm that finds edges in the reflectance image and match these with edges in the color image.

In urban modeling, [Stamos and Allen, 2001] Stamos and Allen present an automatic registration feature based-method for image registration. First, 2D lines are extracted from the image by edge detection and line fitting. These lines are clustered according to their intersection points and the major vanishing points are computed. Using three orthogonal vanishing points, the internal camera parameters are estimated. Then, 3D line features are extracted from the range images by segmenting these into planar regions and computing the intersection of the segmented planes. The camera orientation is computed from matching parallel line directions. Finally, the camera position is found by grouping 2D and 3D lines into rectangles and running a RANSAC [Fischler and Bolles, 1987] based matching procedure. This method is limited to very specific settings with three main orthogonal scene directions and quadrangular features. More recently, the requirement that three orthogonal scene directions be present was removed in [Liu and Stamos, 2005]. In this new version of the camera pose algorithm, Liu and Stamos group 3D and 2D lines into higher order features (3D parallelepipeds and 2D rectangles) to efficiently search the space of 3D and 2D feature matches.

2.1.2 Intensity-based algorithms

Intensity-based algorithms originated in the medical image community for multi-modal registration of images acquired by different sensors (MR images and CT scans, for example). These algorithms measure the statistical dependence between two images using metrics from information theory. In the context of range and intensity images, these algorithms compare the intensity image with a rendering of the range image from a given camera pose and compute their mutual dependence. The goal is to search the six dimensional space of camera positions and orientations to find a global minimum or maximum of this metric. The similarity metrics used in these automatic registration algorithm are based on the chi-squared measure of dependence or the mutual information criterion. [Maes *et al.*, 1997, Hantak *et al.*, 2004]. In the generation of the renderings of the 3D model, the model is colored with the reflectance values produced by the range sensor, which can be correlated

with the texture of the scene. The robustness of these methods depend on the choice of the optimization algorithm and the initial estimate provided. Since these information-based metrics can show a number of local minima, gradient-descent methods might fail to converge to a global minimum if the initial estimate is not good enough. At the expense of a more costly search, simulated annealing (SA) has been used to avoid the local-minimum problem. An analysis of the performance of different metrics and optimization techniques is given in [Hantak and Lastra, 2006]. The authors show that in order to achieve a good registration using gradient descent methods, a very good initial estimate of the camera position is needed. Otherwise, the optimization invariable converges to a local minimum.

2.2 Seamless texture map integration

The last step in the 3D digital modeling pipeline is the creation of a seamless texture map. The input to this task is a set of color images, their camera parameters and the geometry of the scene. The output is a seamless texture map that can be used to produce photo-realistic renderings. The challenge is to combine all the input images in such a way that the final result looks seamless. This can be a difficult problem for several reasons. First, the images are taken from different points of view, which means they show different degrees of foreshortening and view-dependent effects. In addition, the illumination and camera parameters might have changed from one shot to the next, resulting in images with color mismatches.

Previous work in this area can be divided into three different categories:

1. Systems that assume that all images were acquired under constant illumination. These systems generate the final rendering by selecting an optimal weighting of the available images according to different factors.

2. Systems that do not make the constant illumination assumption and use color correction or relighting to obtain seamless models.

3. Systems that use the available images to run an inverse rendering algorithm and compute surface reflectance, obtaining and illumination-free texture representation that can be used to create new renderings under any illumination condition.

In the remaining of this section, we explore the solutions that have been proposed in each of the above categories.

2.2.1 Optimal image blending under constant illumination

Systems in this category assume that the available images have been taken under constant illumination. Then, when creating a seamless texture map, only the following factors have to be taken into account: foreshortening, view-dependent effects, and image resolution. Hence, it makes sense that most of the systems in this category create the final texture map using a weighted interpolation of the input views. The weights are fixed according to: a) the angle between the viewing direction and the camera's principal axis, b) the angle between the surface normal and the camera's principal axis, c) the pixel to surface ratio, d) the distance of the pixels to the field of view boundary. Debevec *et al.* (1996) use a weighted average of two images based on the angle between the current viewing direction and the direction to the camera, a.k.a. view dependent texture mapping (VDTM). Pulli *et al.* (1997) combine weights that depend on the viewing direction , the normal to the surface and the field of view. Buehler *et al.* (2001) follow a similar approach but they also take resolution into account, so that images that have a good pixel to area ratio are given more importance. Finally, Wang *et al.* (2001) introduce an optimal texture reconstruction method based on a signal-processing approach.

2.2.2 Color correction and relighting

When the illumination or camera parameters are different across the images, a blending algorithm will not produce seamless results. For these cases, two different solutions have been used: a) to apply a color transform, b) to relight the input images to a consistent illumination.

The color transform approach is good for fixing camera parameters and small changes in the illumination, such as chromatic or intensity changes. An example in this direction is the work of [Agathos and Fisher, 2003]. To obtain a seamless texture map, a global 3x3 color correction matrix is computed. The matrix is computed from pixel intensity constraints obtained from the overlap region of the two images, taking care to discard pixels in shadow or highlight areas by thresholding. Once the matrix is computed, one of the images is color-corrected. In [Bannai *et al.*, 2004], this pairwise color correction scheme is generalized to multiple input views. It is unclear the extent to which these methods can handle illumination changes, since the results that are presented in [Agathos and Fisher, 2003] are for small convex objects captured under laboratory conditions.

On the other hand, Beauchesne and Roy (2003) take the relighting path and compute a relighting operator that they apply to a pair of input images to create a new set of images with consistent lighting. For convex Lambertian objects, they observe that the ratio of pixels corresponding to surface points with the same normal should be constant (provided the non-linear effects of the camera gain had been removed). On the area of overlap of the two images they compute the "ratio lighting". Then, to relight pixels in the non-overlapping region, they lookup the pre-computed ratio for the corresponding surface normal and multiply it by the pixel intensity. To extrapolate for unseen normals they use a filtering mechanism based on a Gaussian kernel.

2.2.3 Inverse rendering

We consider now inverse-rendering techniques and their uses in the creation of seamless texture maps. Inverse-rendering techniques are image-based methods that revert the image information process to solve for some scene unknown, such as illumination or surface reflectance. If we can solve for a scene's reflectance properties from a collection of images, then we can create a seamless integrated texture-map that is relightable, i.e. that we can use to render a model under different illumination conditions. In the remaining of this section we describe the theory behind inverse rendering and related works.

When a camera photographs a scene, it measures the intensity of light reflected from the scene towards it. Light reflected from a scene point $\mathbf{x}$ depends on a combination of factors. It depends on the illumination reaching the surface, which is the combination of light from light sources and light reflected by other scene points, and it also depends on how a surface reflects light. In mathematical terms, the light reflected by a point $\mathbf{x}$ in the direction (θ_o, ϕ_o) expressed in local coordinates with respect to the surface normal is given by:

$$B(\mathbf{x}, \theta_o, \phi_o) = \int_{\Omega_i} L(\mathbf{x}, \theta_i, \phi_i) f_r(\mathbf{x}, \theta_i, \phi_i, \theta_o, \phi_o) \cos\theta_i d\omega_i \tag{2.1}$$

where L is the incident illumination and f_r is the bidirectional-reflectance distribution function (BRDF) which describes how a surface reflects light. A BRDF that varies over the surface of an object is referred to as a spatially varying bidirectional-reflectance distribution function (SBRDF). For now, we will use the terms BRDF and SBRDF interchangeably to denote surface reflectance, but we will make a special note when these do not mean the same.

For simplicity, the reflected light equation (2.1) does not account for the effects of subsurface scattering, which we will not address here. Certain materials, such as marble and skin, show some degree of translucency. This means that light scatters inside the material before being absorbed or leaving the material at a different point. These effects are modeled in the work of [Jensen *et al.*, 2001]. In addition, we also omitted from the above equation (2.1) a dependence on wavelength. Materials will reflect different wavelengths in a different way, hence the BRDF is wavelength-dependent. In practice, this dependence is handled by replicating the reflected light equation once for each of the red, green and blue (RGB) components.

To justify inverse-rendering, a relationship between the radiance reflected by a scene point and the recorded value of that point in a photograph (image irradiance) must be established. These two quantities are related, and the fundamental relationship, which is

derived in detail in [Horn, 1986], is described by the following equation:

$$E = B(\mathbf{x})\frac{\pi}{4}\left(\frac{d}{f}\right)^2 \cos^4\alpha. \tag{2.2}$$

In (2.2) above, E is the image irradiance, d is the diameter of the lens and f its focal length. The factor of proportionality includes the inverse of the square of the effective $f-number$ and a term that falls-off with the cosine to the fourth power of the angle the incident ray makes with the optical axis of the camera. For images that cover a narrow angle, this term is not important; indeed, vignetting effects due to multiple lens apertures aligned along the optical axis might cause more serious attenuation of brightness. In addition, some cameras add a non-linear transform in the imaging process to obtain an image that can be displayed on a computer screen and compress the range of measured intensity values. This transform is called the camera's *response function*. For equation (2.2) to hold, the effects of the camera response must be removed from the images.

Now that we have established that image irradiance is proportional to scene radiance, we can enumerate different problems that arise in the context of inverse-rendering given one or more photographs of the scene. Solving for the illumination given a photograph and the BRDF is a problem that [Marschner, 1998] in his PhD dissertation calls *inverse-lighting*. On the other hand, if the illumination and the scene geometry are known, one can solve for the BRDF. This problem is called *image-based reflectometry*.

In the context of 3D scene modeling, inverse-lighting and image-based reflectometry are problems of significant importance because they take image-based techniques a step further than simple view-dependent weighted average, enabling a wider range of applications. However, recovering and representing spatially-varying reflectance and/or illumination is a difficult problem, it involves high-dimensional functions and not always well-posed. For these reasons and to make equation (2.1) tractable, it is common practice to relax certain conditions. First, it is typical and sometimes reasonable to assume that sources of illumination are far away from the scene, therefore making the illumination field the same

for all scene points. In addition, with the exception of the works of [Yu *et al.*, 1999, Debevec *et al.*, 2004], the indirect illumination contributions are usually dropped out, either by assuming the scene consists of a single convex object or by ignoring interreflections. Under these assumptions, the reflected light equation becomes:

$$B(\mathbf{x}, \theta_o, \phi_o) = \int_{\Omega_i} V(\mathbf{x}, \theta_i, \phi_i) L(\theta_i, \phi_i) f_r(\mathbf{x}, \theta_i, \phi_i, \theta_o, \phi_o) \cos\theta_i d\omega_i, \tag{2.3}$$

where the illumination does no longer depend on the scene point, and $V$is a visibility term that accounts for occlusions to the distant light sources. Inverse rendering techniques take as input an image and the scene geometry, plus if known, either the BRDF or the illumination, and solve for the missing elements of the above equation.

Inverse rendering taxonomy

To put previous work in context, we will use the taxonomy introduced by Ramamoorthi and Hanrahan (2001b), based on which of the three quantities -lighting, BRDF and texture- are known. This taxonomy is derived from the following simplified version of equation (2.3), in which spatially varying reflectance is decoupled into a texture component and a BRDF component:

$$B(\mathbf{x}, \theta_o, \phi_o) = \int_{\Omega_i} L(\theta_i, \phi_i) T(\mathbf{x}) f_r(\theta_i, \phi_i, \theta_o, \phi_o) \cos\theta_i d\omega_i. \tag{2.4}$$

Here, T denotes a single texture that spatially modulates the BRDF. In practice, one will need separate textures for the diffuse and specular components.

Because it is unpractical to represent the BRDF by enumeration of all its possible values, researchers have looked for efficient and more compact representations that make the image-based reflectometry process more tractable. For isotropic materials, the BRDF can be reduced to a 3D function. However, even for a 3D function, the number of measurements required to completely acquire the BRDF of an object is large. Further simplifications can be done if one can assume that the reflectance on a surface point follows a certain

parametric model that depends on a small number of parameters. The simplest of such models is the Lambertian reflectance model, which states that light is scattered equally in all outgoing directions and hence the BRDF is a constant. Other physically based models are Oren and Nayar's model for rough diffuse objects [Oren and Nayar, 1994], and Cook and Torrance's model for materials with specularities [Cook and Torrance, 1982]. In physically based models, each parameters is associated with a physical property of the material. In contrast, there exist empirical models of reflectance that were proposed to serve for fitting reflectance data. In this category are Ward's model for anisotropic reflection [Ward, 1992], and Lafortune's model for modeling specularities at different reflection angles [Lafortune *et al.*, 1997]. Most image-based reflectometry applications assume the BRDF takes the form of one of these parametric models. There are a few exceptions, however. In their signal-processing framework for inverse rendering, Ramamoorthi and Hanrahan (2001b) introduce the theory and necessary conditions needed to recover a frequency-space representation of the BRDF and also show how frequency-space representations can be combined with parametric models. In their data driven approach, Matusik *et al.* (2003) acquire a very dense set of samples to represent the entire BRDF.

2.2.3.1 Known illumination

A large number of image-based reflectometry techniques are based on known or measured illumination. Sato *et al.* (1997) introduce a system to estimate spatially varying diffuse plus specular reflectance. Since measuring a specular lobe requires a significant number of samples, they assume the specular component to be homogenous on the whole surface of the object. Marschner (1998) measures Lambertian reflectance at each surface point, computing a weighted average of all data from all the images that see the point. The images are acquired under a point light source that is rigidly attached to the camera. Debevec *et al.* (2000) acquire the reflectance field of a human face. For this, they built a light stage, a two-axis of rotation device that allows them to densely sample the space of light directions. From the acquired set of images, they find spatially varying parameters of

a modified version of the microfacet model of Torrance and Sparrow (1967). Rocchini *et al.*, Bernardini *et al.* (2002, 2001) use images captured under controlled illumination to obtain high-quality albedo maps and surface normals using photometric stereo, which are then mapped to a previously scanned object. Going beyond Lambertian reflection, Ramamoorthi and Hanrahan (2001b) describe algorithms to recover spherical harmonic BRDF coefficients and a simplified 4-parameter Torrence-Sparrow microfacet BRDF. An algorithm is also presented to recover textured BRDFs by allowing the parameters of the microfacet BRDF to vary spatially. Lensch *et al.* (2003) computes spatially varying reflectance of objects consisting of different materials. Their method fits the data to a Lafortune representation of the BRDF and clusters similar materials together, generating a basis of BRDFs per material. Spatial variation within a single material is captured by projecting the measured data at each surface point to the BRDF basis of the corresponding material. In their setting, an object is illuminated by a point-source, and the direction of the light is recovered from the image of an arrangement of specular spheres that is placed in the scene.

Extending image-based reflectometry to outdoor scenes is a difficult problem. To begin with, there is no control over the illumination of the scene (unless images are acquired at night time when the moon is not visible). During the day, the sources of illumination are the sun and sky. To solve for reflectance of outdoor scenes, two main approaches have been taken. The first approach uses a parametric model for sun and sky light. The work of Love (1997) falls in the first category; keeping the camera fixed, Love acquires a set of images of a flat sample at different times of the day. Then, using a sky and sun model, Love solves for a parametric BRDF. A second approach is to measure the incident illumination using a special device. In the work of Yu and Malik (1998), the illumination is measured using photographs of the sky and the surrounding environments. Any missing regions of the sky are filled by finding the parameters of a sky model from the image data. For the BRDF a dichromatic diffuse plus specular model is used, where the diffuse component varies over the surface and the specular is constant for every surface patch. More recently, Debevec *et al.* (2004) introduced a novel lighting measurement apparatus that can record

the high dynamic range of both, sunlit and cloudy environments, using a set of specular and diffuse calibrated spheres. Their proposed method estimates spatially varying diffuse surface reflectance using an iterative inverse global illumination technique.

2.2.3.2 Unknown illumination

To measure texture or the BRDF under unknown illumination is a difficult problem which requires to solve for both, the lighting and the surface reflectance, simultaneously. This turns out to be a factorization operation, in which the image measurements are factored into its reflectance and illumination components. This factorization is tractable under certain conditions. For the case of an object consists of a single homogenous material, [Ikeuchi and Sato, 1991] recover the position of a a point-source and the parameters of a dichromatic diffuse and specular model of an object using range and brightness images. Ramamoorthi and Hanrahan (2001b) recover a microfacet BRDF model of curved surfaces under unknown complex illumination. Since the theory predicts that for low-frequency lighting the estimation of the surface roughness is ill-conditioned, their method requires that a single directional light source be present in addition to any low frequency illumination. In the more general case of textured surfaces, the factorization problem has a non-ambiguous solution only when the texture has high-frequency components. Under this assumption, Oh *et al.* (2001) use bilateral filtering to factor texture variations from low frequency illumination effects in their proposed photo-editing system. But for low frequency textures, lighting and texture can only be factored using active methods or making assumptions about their expected characteristics.

2.3 Work in context

We conclude this chapter by placing our work in context and comparing it to previous systems for modeling from range and intensity images. We consider the systems of:

1. Levoy *et al.* (2000) used for the modeling Michelangelo's David and other statues;

2. Bernardini *et al.* (2002) for modeling Michelangelo's Pieta;

3. Rocchini *et al.* (2002) for acquiring, stitching and blending diffuse appearance attributes;

4. Lensch *et al.* (2003) for acquiring spatial varying appearance and geometric detail;

5. Yu and Malik (1998) for acquiring photometric details of architectural scenes from photographs;

6. Debevec *et al.* (2004) for measuring surface reflectance under complex natural illumination.

7. Liu and Stamos (2005), Liu *et al.* (2006) for the automatic registration of range and image data in urban settings.

In the comparisons that follow, we consider two main characteristics: how the other systems solve the texture registration problem and how they integrate the color images into a seamless texture map. Table 2.1 compares the techniques used for texture registration. First we consider the camera arrangement, and we distinguish between the systems that allow for free camera placement and systems that fix the camera with respect to the range sensor. For those cases in which the camera's position is unconstrained, we indicate the registration technique used. Table 2.2 compares existing modeling systems with respect to reflectance estimation. We show the assumptions placed on the illumination model and the type of reflectance attributes that are recovered. For the illumination model, we consider the illumination type, making a distinction between those methods that use point light sources and those that allow for more complex illumination environments. We also indicate if the illumination is calibrated using a special device, or solved from the images; and if the model takes interreflections and cast shadows into account. For the reflectance model, we note if the system solves for diffuse appearance only or both, diffuse and specular parameters. Our work differs from previous work in the following areas:

Large scale outdoor scenes. Most of the previously listed systems are designed for the acquisition of small objects in lab controlled environments. The exceptions are the systems of [Yu and Malik, 1998], [Debevec *et al.*, 2004] and [Liu *et al.*, 2006]. The work of [Yu and Malik, 1998] builds the 3D model manually from the images and does not use range data at all. Hence, it is unlikely that it will scale to large environments.

Texture registration. Our shadow-based registration tool is the first of its kind, as far as we know. Shadows have been used extensively in other areas of computer vision but not for texture registration. With respect to our semi-automatic tool for registration of images of architectural scenes, it is inspired on the work of [Stamos and Allen, 2001]. Our tool can handle more general conditions not limited to scenes with three orthogonal vanishing points. However, the most recent works of [Liu and Stamos, 2005] and [Liu *et al.*, 2006] have produced tools that can handle more general cases and register images using a combination of feature matching and structure from motion that can achieve robust registrations.

Seamless texture map integration. We propose two solutions for the generation of seamless texture maps: a relighting approach and a factorization technique which decouples illumination from texture. The main distinguishing feature of these two techniques is that they do not require a special device to measure the illumination of the scene, as required in the works of [Yu and Malik, 1998] and [Debevec *et al.*, 2004]. Our relighting approach is similar to the work of [Beauchesne and Roy, 2003]. We extend this work to outdoor scenes with shadows, and show to it is possible to obtain shadow-free images. The factorization technique uses a ratio of images to solve for the illumination component. Though our main motivation was the modeling of outdoor scenes, the factorization technique can handle different settings, as described in chapter 6. We are able to solve the factorization any ambiguity, by assuming diffuse Lambertian like surfaces and specific illumination models.

	Camera	
	Arrangement	Registration
Levoy *et al.* (2000)	fixed	pre-calibration
Rocchini *et al.* (2002)	free placement	manual + optimization
Bernardini *et al.* (2002)	fixed	pre-calibration
Lensch *et al.* (2003)	free placement	silhouettes
Yu and Malik (1998)	free placement	manual
Debevec *et al.* (2004)	free placement	semi-automatic
Liu and Stamos (2005), Liu *et al.* (2006)	free placement	automatic
Ours	free placement	semi-automatic/shadows

Table 2.1: This table compares existing modeling systems with respect to sensor arrangement and texture registration.

	Illumination			Reflectance
	Model	Irref	Shdws	Model
Levoy *et al.* (2000)	point/calibrated	no	no	diffuse
Rocchini *et al.* (2002)	point/calibrated	no	no	diffuse
Bernardini *et al.* (2002)	point/calibrated	no	no	diffuse
Lensch *et al.* (2003)	point/calibrated	no	no	diff. + spec.
Yu and Malik (1998)	natural/calibrated	no	yes	diff. + spec.
Debevec *et al.* (2004)	natural/calibrated	yes	yes	diffuse
Liu and Stamos (2005), Liu *et al.* (2006)	not applicable			
Ours	natural/relighting or solve parametric	no	yes	diffuse

Table 2.2: This table compares existing modeling systems with respect to reflectance estimation. We show the assumptions on the illumination model and the type of reflectance attributes that are solved for. For the illumination model, we consider its type (calibrated point light source, natural illumination), and if it considers interreflections and shadows. For the reflectance model, we note if the system solves for diffuse appearance only or both, diffuse and specular parameters.

Chapter 3

Range and intensity image registration using points and lines.

In this chapter we address the problem of image and range data registration using point and line features. Registration is a required step for all 3D modeling applications wanting to achieve photorealistic renderings from photographs and range data. To bring the range and image data into registration is to find a mapping between 3D points in the world with 2D points in an image. In general, this mapping is described by a camera model plus the camera position and orientation in the world, and is computed from either feature correspondences or maximizing a metric of mutual information. In chapter 2 we reviewed different methods for automatic and semi-automatic registration. In this chapter and the next one, we describe a set of tools that we have developed for this purpose. Our main goal is to provide a set of registration tools that reduce user-intervention and achieve high-quality registrations. For this purpose, we have developed three tools that require different degrees of user-intervention and can be applicable in modeling of outdoor scenes:

Point-and-click. Our first tool is a point and click interface for users to select point correspondences and solve for the camera parameters.

Line-based. The second tool uses a semi-automatic registration algorithm that uses line-

features extracted from range and image data to find the correct camera pose.

Shadow-based. The third tool uses the shadows cast by the sun as clues for the finding the correct camera pose. We developed this tool for modeling archaeological sites.

In this chapter we describe the algorithms we use in the point-and-click and line-based registration tools. Next, we review the camera model and algorithms used in these tools. The shadow-based registration algorithm is described in detail in chapter 4.

3.1 Camera model

A camera model describes how a camera maps 3D world points to the 2D image plane. A detailed mathematical analysis of different camera models used in computer vision can be found in Hartley and Zisserman (2000). The simplest of all models is the pinhole camera model. Under this model, a point in 3D space $\mathbf{X_{world}}$ is mapped to the point $\mathbf{x_{cam}}$, which is the point of intersection between the line joining $\mathbf{X_{world}}$ with the camera center $\mathbf{C}$ and the image plane, as shown in figure 3.1. The line from the camera center to the image plane is the *principal axis*, which intersects the image plane at the *principal point*. When image and world points are represented by homogeneous vectors, this projection can be written as a linear mapping between using matrix notation:

$$\mathbf{x_{cam}}^T = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \mathbf{X_{world}}^T. \tag{3.1}$$

Here f is the distance between the camera center and the image plane, typically referred to as *focal length* and p_x and p_y are the image coordinates of the principal point. This projection model has three parameters f, p_x, p_y and assumes the center of the camera is the origin of the world coordinate system. This is not usually the case, points in the world will be expressed in world coordinates and points in the image, in image coordinates, generally measured in pixels. Since both of these coordinate systems are Euclidean frames, there

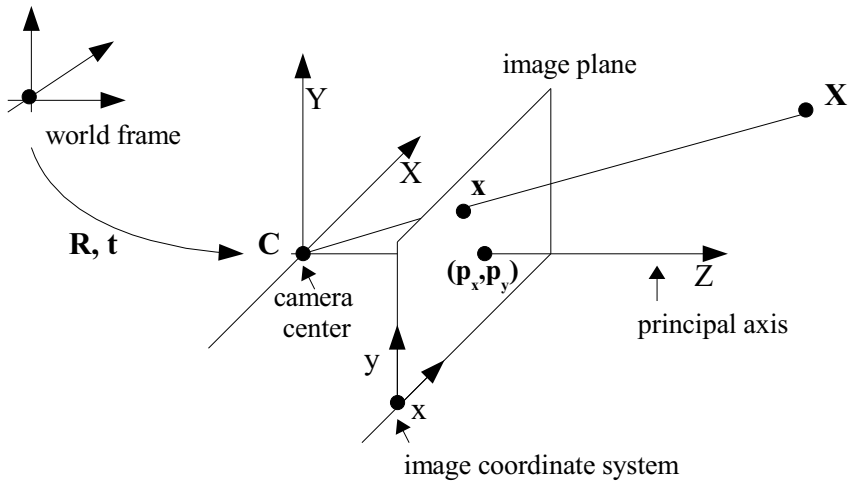

Figure 3.1: Image formation showing the relationship between a world point $\mathbf{X}$, its projection point $\mathbf{x}$ and the camera's position and orientation in the world.

exists a rotation and translation that brings the two frames into alignment, plus a scaling factor between distance units in the world and pixel units. Thus, the projection of world points to image coordinates can now be written as:

$$\mathbf{x_{cam}} = \boldsymbol{K}[\ \boldsymbol{R}\ |\ \mathbf{t}\ ]\mathbf{X_{world}}, \tag{3.2}$$

where $\boldsymbol{R}$ is a rotation matrix that describes the orientation of the camera with respect to the world coordinate frame, $\mathbf{t}$ is a translation vector that indicates the camera position in the world, and $\boldsymbol{K}$ is a 3x3 matrix of the form:

$$\boldsymbol{K} = \begin{bmatrix} f_x & s & x_0 \\ & f_y & y_0 \\ & & 1 \end{bmatrix}. \tag{3.3}$$

In (3.3), f_x and f_y represent the focal length of the camera in terms of pixel dimensions in the x and y directions respectively, $(x_0,\ y_0)$ are the coordinates of the the camera's principal point in pixels, and s is a skew parameter that measures the orthogonality of the

x and y camera axes. This parameter is zero for most cameras. The matrix $\boldsymbol{K}$ is known as the *camera calibration matrix* containing the camera *intrinsic* parameters, and the rotation and translation ($\boldsymbol{R}$,$\mathbf{t}$) are the camera's *extrinsic parameters*. The intrinsic and extrinsic parameters can be combined into a 3x4 camera projection matrix of the form:

$$\boldsymbol{P} = \boldsymbol{K} \left[\, \boldsymbol{R} \mid \mathbf{t} \right], \tag{3.4}$$

which for a matrix $\boldsymbol{K}$ of the form (3.3) describes a **finite projective camera**. A finite projective camera has 11 degrees of freedom.

This linear model of projection is an ideal model; for real lenses, the linear assumption does not hold, due to lens distortion. The most common effect is that of radial distortion, which is modeled as a mapping between points in the image plane as:

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(\tilde{r}) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \tag{3.5}$$

where $(\tilde{x}, \tilde{y})$ is the ideal image position which obeys the linear projection model, (x_d, y_d) is the actual image position after distortion, $\tilde{r}$ is the radius from the center for radial distortion (which is sometimes assumed to be equal to the camera's principal point), and $L(\tilde{r})$ is the distortion factor that depends on the radius $\tilde{r}$ alone. The function $L(\tilde{r})$ is generally defined as a polynomial on $\tilde{r}$:

$$L(\tilde{r}) = 1 + \kappa_1 \tilde{r} + \kappa_2 \tilde{r}^2 + \kappa_3 \tilde{r}^3 \ldots \tag{3.6}$$

Here, κ_i are the distortion coefficients, which belong to the set of the intrinsic camera parameters. The effects of radial distortion can be removed from an image by means of a non-linear image warp. The resulting image is the one that would have been obtained under a perfect linear camera.

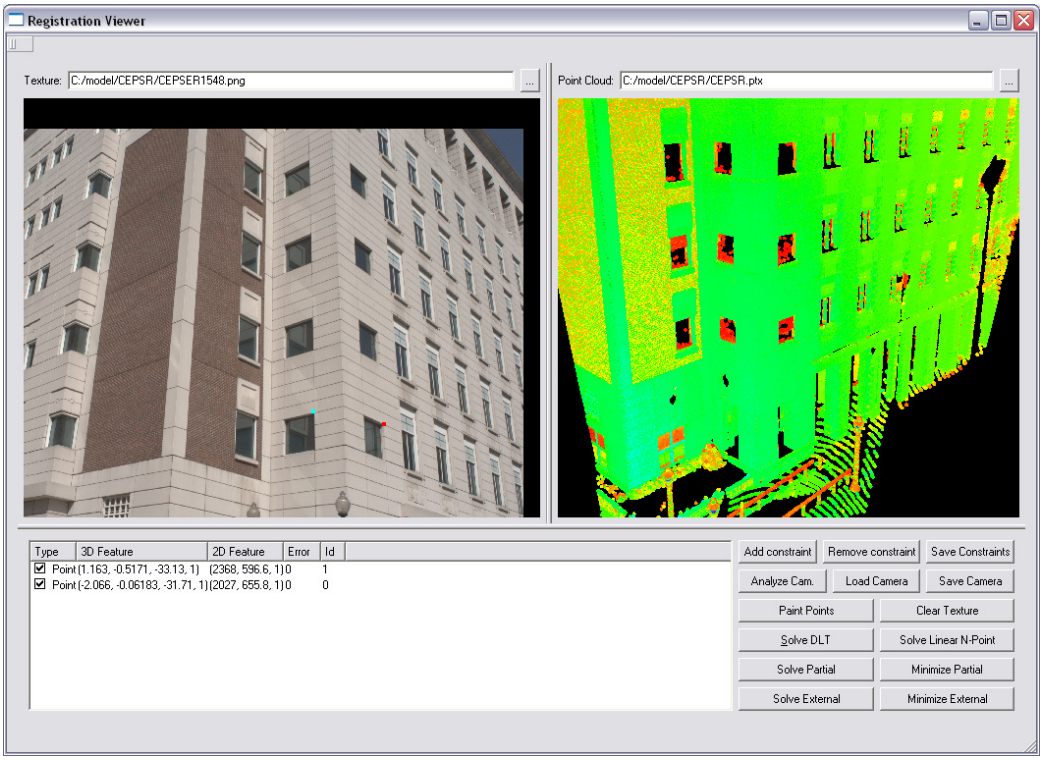

Figure 3.2: Point-and-click user interface for image registration. The user is presented with a view of both, the range and color images, selects a set of corresponding points and solves for the camera parameters.

3.2 Finding the camera parameters from point correspondences

Given a set of corresponding world and image points, it is the goal of the registration tool to find the camera parameters that best fits the provided data. In this section we describe the main algorithms and procedures we have implemented to solve for the camera calibration using point correspondences, and which we have implemented into our point-and-click interface shown in figure 3.2. Correspondences are selected manually by a user, which is given a view of both, the color image and the range data. Once the correspondences are selected, the user is given the option to solve for both the camera intrinsic and extrinsic parameters, or load a set of previously computed intrinsic parameters and compute the camera pose (rotation and translation) only. We now outline a solution to the problem

of finding both the intrinsic and extrinsic parameters simultaneously, and then present a solution to the problem of pose estimation for a camera with a known calibration matrix $\boldsymbol{K}$. Most of the techniques we implemented in our tools work on the assumption of a pure pinhole camera, with no radial distortion. However, for those methods in which the camera intrinsics are assumed known, the radial distortion coefficients are computed as part of the camera intrinsic parameters, and the effects of distortion are removed from the images.

3.2.1 Solving for the camera intrinsic and extrinsic parameters simultaneously

Given a set of corresponding point pairs $(\mathbf{X_i}, \mathbf{x_i})$, where $\mathbf{X_i}$ is a 3D point and $\mathbf{x_i}$ an image point, the direct linear transform (DLT) is a linear algorithm that computes the 3x4 projection matrix $\boldsymbol{P}$ of (3.4). Assuming both $\mathbf{X_i}$ and $\mathbf{x_i}$ are given in homogenous coordinates the DLT algorithm finds the projection matrix $\boldsymbol{P}$ that best fits the equations $\boldsymbol{P}\mathbf{X_i} = \mathbf{x_i}$. The equality relationship between vectors $\boldsymbol{P}\mathbf{X_i}$ and $\mathbf{x_i}$ is the equality between homogeneous vectors. Although algebraically $\boldsymbol{P}\mathbf{X_i}$ and $\mathbf{x_i}$ might not be equal, they can still represent the same point. However, two homogeneous vectors that represent the same point satisfy the condition that the cross product of the two will be zero. From this property, the following linear constraints on $\boldsymbol{P}$ can be obtained:

$$\begin{bmatrix} \mathbf{0}^T & -w_i\mathbf{X_i}^T & y_i\mathbf{X_i}^T \\ w_i\mathbf{X_i}^T & \mathbf{0}^T & -x_i\mathbf{X_i}^T \end{bmatrix} = \begin{pmatrix} \boldsymbol{P}^1 \\ \boldsymbol{P}^2 \\ \boldsymbol{P}^3 \end{pmatrix} = \mathbf{0}, \tag{3.7}$$

where $\boldsymbol{P}^i$ is a row of $\boldsymbol{P}$. From a set of n point correspondences a $2n*12$ matrix $\boldsymbol{A}$ is obtained by stacking up the equations (3.7). Then, the matrix $\boldsymbol{P}$ can be found by solving the linear system $\boldsymbol{A}\mathbf{p} = \mathbf{0}$. At least 6 point correspondences are required to obtain a solution. Due to noise in the data, typically an over-determined system with $n > 6$ is solved. In this case, one seeks the solution the minimizes $\|\boldsymbol{A}\mathbf{p}\|$ subject to $\|p\| = 1$, which is given by the singular vector of $\boldsymbol{A}$ corresponding to its smallest singular value. Solving for $\boldsymbol{P}$ using singular value

decomposition gives the solution that minimizes the algebraic error. A more meaningful metric to minimize is the geometric error, defined over the set of correspondences as:

$$\sum_i d(\mathbf{x_i}, \boldsymbol{P}\mathbf{X_i})^2. \tag{3.8}$$

Here, d is the Euclidean distance. To minimize the geometric error requires to solve a non-linear least-squares problem using an iterative gradient descent method such as Levenberg Marquardt. In this case, the solution obtained by the DLT algorithm can be used as the starting point for the minimization.

The matrix $\boldsymbol{P}$ obtained from the previous steps is a general projective matrix. In some cases, it might be more desirable to restrict the camera by setting conditions on the camera parameters. For instance, we might want to set the skew s to zero and enforce the condition that pixels are square by requiring $f_x = f_y$. These conditions can be enforced in the iterative minimization of the geometric error. As before, an initial solution is computed using the DLT algorithm. One can also introduce in the iterative minimization the radial distortion coefficients. But in practice, solving for the radial distortion parameters from manually selected point correspondences might not yield optimal results and requires a large number of points.

3.2.2 Solving for the pose of a camera with known intrinsics

In many situations, the camera intrinsic parameters can be found from a set of images of a calibration object, such as a planar checkerboard [Bouguet, 2001]. A calibration object facilitates feature extraction and matching. Typically, features computed on images of a calibration object are accurate and abundant. For this reason, using a calibration object is generally more accurate than computing the camera parameters from manually selected range and image correspondences, because of the larger number of points involved in the optimization. In addition, the radial distortion parameters can be accurately computed.

When the camera intrinsics are known, one needs only to solve for the camera orientation

and position to completely register the range and color images. The solution to this problem is found by iteratively minimizing the geometric error (3.8), fixing the camera intrinsic parameters and optimizing over the camera pose. Still, this requires a set of corresponding features plus a good initial estimate of the camera pose for the initialization of the nonlinear minimizer. Ideally, one would like to obtain this set of initial parameters from as few correspondences as possible using a closed form or linear method. We use for this purpose the linear algorithm of Ansar and Daniilidis (2003), which guarantees to return a unique solution for a minimum of 4 point correspondences.

Given $n \geq 4$ point correspondences, the algorithm of [Ansar and Daniilidis, 2003] finds the depth of each point by enforcing the rigidity of the distance between all possible combination of points. The solution is obtained by linearizing a set of quadratic equations and solving for the depths. After the depths of have been computed, the camera position and orientation can be found by solving a 3D to 3D point registration problem using the technique of Horn (1987).

We now summarize the steps required to solve for the position and orientation of the camera for the case were the intrinsic parameters are known or have been pre-computed:

1. Remove the effects of radial distortion (if any) from the image to register.
2. Select at least 4 point correspondences between the intensity and range image.
3. Solve for the camera pose using the algorithm of [Ansar and Daniilidis, 2003].
4. Minimize the geometric error (3.8) using a gradient descent technique such as Levenberg-Marquardt using the camera pose computed in the previous step as initial estimate.

We have implemented these steps in our point-and-click tool. The only disadvantage of this approach is that, once the camera parameters have been solved, the camera's focal distance is fixed for the entire acquisition process. However, we have found this not to be a problem when imaging distant objects such as buildings.

3.3 Semi-automatic registration based on line features

The point-based registration algorithm presented in the previous section requires a user to select correspondences. Although the number of correspondences required for an accurate registration is in the order of five or six when the camera intrinsics are known, finding and selecting easily identifiable point features takes time. For this reason, automating this matching process is an important area of research. In this section we present a semi-automatic tool for image registration that uses line features to find the orientation and position of the camera with known intrinsic parameters. The main motivation for developing such a tool comes from the fact that lines are abundant features in architectural scenes and can be robustly extracted from 3D point clouds and from images, making them good features to use for solving the image registration problem. In addition, line features in buildings can be typically clustered into sets of parallel lines that share the same orientation. Based on these observations, our tool solves for the camera pose in two stages: in the first stage the camera orientation is automatically computed by matching corresponding clusters of parallel lines; in the second stage, a user drags a rendering of the correctly oriented 3D model over the image to a position in which the rendering and the image are in close proximity, after which a line-matching search is started to find at least three line correspondences to compute the camera position. This technique is similar to the algorithm presented by Stamos and Allen (2001) for automatic image registration. The algorithm of Stamos and Allen extracts 2D and 3D line features, clusters these into sets of parallel lines, computes the camera intrinsic parameters from three orthogonal vanishing points, finds the camera rotation by matching clusters of parallel lines and, in its final step, groups the line segments into 2D and 3D rectangles that are matched using RANSAC to compute the final camera position. Our method overcomes some of the scene restrictions in [Stamos, 2001], since it does not require three orthogonal vanishing points nor the presence of rectangular features.

The complete registration procedure takes the following steps:

1. Extraction of two feature sets L_{3D} and L_{2D} of 3D and 2D line segments from the

range and image data sets.

2. Grouping of the elements in L_{3D} and L_{2D} into clusters of parallel 3D and converging 2D lines.

3. Solving for the rotation from two corresponding 3D and 2D clusters.

4. Manual positioning of the rotated model over the image so that the rendered model and the image are in close agreement.

5. Searching for corresponding lines and computing the translation vector $\mathbf{t}$.

We now explain in detail each of the above steps.

3.3.1 3D line extraction.

Line features are computed from range images using the algorithm of Stamos and Allen (2002). Lines are obtained from the intersection and boundaries of planar regions. First, a planar segmentation algorithm labels all range points according to two possible cases: either a point is locally planar, in which case it is labeled with a plane identifier and a surface normal; or the point is not locally planar and ignored in subsequent steps. Each extracted planar region is tested for proximity with each of the other planar regions, and when the proximity test succeeds a line is computed from the intersection of the two planes in which these regions are embedded. In addition, an additional set of line segments is obtained from the boundary of each of the planar regions.

3.3.2 2D line extraction.

Line segments in the 2D image are found using an edge detector followed by a line fitting procedure. Our tool finds edges using a Canny edge filter and fit lines to the edges using orthogonal regression.

3.3.3 3D line clustering

The aim of the 3D feature clustering step is to partition the set of 3D lines L_{3D} into subsets L_{3D}^{i} such that each of the lines in these subsets are parallel with respect to each other. For this, we use the same technique of [Stamos and Allen, 2001], that employ a nearest neighbor clustering algorithm. Initially, each 3D line defines its own cluster. Then, the algorithm iteratively searches for the two closest clusters based on the angle of the average line direction in each, merges them into a single one and updates the average line direction. Termination occurs when the distance between the two closest clusters is greater than a given threshold.

3.3.4 2D line clustering

Partitioning the 2D line segments in L_{2D} into sets that correspond to parallel lines in the scene is a task that differs from 3D line clustering. A set of parallel lines in the scene is projected by a projective camera to a set of lines that intersect in a common point, called the *vanishing point* (VP). Hence, the clustering of 2D lines is analogous to finding the vanishing points of the image and the corresponding set of supporting lines. From all possible VPs, we are only interested in finding the major ones, those that define the main directions of lines in the scene.

To find the VPs we compute the pairwise intersection of all possible line segments and create a 2D histogram of intersections. Since each vanishing point can be associated to a 3D direction (i.e. a point in the Gaussian sphere), we parametrize the 2D histogram of intersections over a 2D representation of the unit sphere. To create this histogram, we take every pair of line segments l_i and l_j and find the image coordinates of the point v where the infinite extension of these lines intersect. Such a point v, when projected back to the 3D world using the camera's intrinsic parameters, results in a 3D line L_v, with direction vector $\mathbf{d}$. It is this vector $\mathbf{d}$ that represents the direction of all parallel lines associated with the vanishing point v. Also, $\mathbf{d}$ represents a point in the Gaussian sphere, and casts a vote in a particular bin of our 2D histogram. Once all possible line pairs have been examined, the

peaks on the 2D histogram represent the main VPs and directions of the lines in the image. The result is a partition of L_{2D} into subsets L^i_{2D} of lines converging to a major VP.

3.3.5 Finding the camera orientation

To find the relative orientation of the camera's coordinates with respect to the world's coordinate frame we use the closed form solution of Horn (1987). Using this technique, the relative orientation can be solved from two pairs of matching directions, which in terms of our framework is equivalent to finding two correspondences between the clusters of parallel 3D lines L^i_{3D} and the clusters of converging 2D lines L^i_{2D}.

By taking advantage of properties that are typical of architectural scenes we can easily find two matching clusters. First, we note that it is generally easy to identify the vertical direction, both in the 3D and 2D clusters. To find the cluster corresponding to the vertical direction in L^i_{3D} we compute the angle the direction vector of each cluster subtends with the world's up direction; then, the cluster whose direction vector subtends the smallest angle is the most likely to be the cluster of vertical lines. Similarly, to find the vertical direction in the image, we assume the image is correctly oriented (otherwise it can be rotated before hand) and find the vanishing point that subtends the smallest angle with the vertical direction in image coordinates, where the angle is measured between the image up direction and the vector that joins the VP with the image center. After the vertical directions have been matched, the second scene constraint to exploit is orthogonality: most of the remaining lines in architectural scenes will be perpendicular to the up direction. Hence we look for clusters in L^i_{3D} and L^i_{2D} that are perpendicular to the vertical clusters. Usually, there will be either one or two of these clusters. If there is only one, the rotation can be solved for. If there are two orthogonal clusters, then there will be an ambiguity in the rotation that can not be solved automatically. In this case, the tool computes a set of plausible rotations and the user selects the correct one.

3.3.6 Computing the translation

The final step in the registration process is to compute the camera position. At this point, the camera orientation is known and two sets of corresponding parallel lines have been matched correctly. What remains now is to find at least three corresponding lines. Since we have already knowledge of two matching clusters of 3D and 2D lines, the search space is reduced considerably, because we only need to conduct the search within the corresponding clusters; i.e. vertical lines will only be matched against vertical lines, and the same applies to horizontal ones. Still, if the number of lines in each of the clusters is large, an exhaustive search is not possible. For this reason, we require a user to move the a rendering of the model over the image until the rendering and the image are in close proximity. Then, we set the camera position to the point of view of the rendered model and instead of running an exhaustive search, we run a closest-line search as follows:

1. Select the first pair of corresponding 2D and 3D line clusters L^0_{2D} and L^0_{3D}.

2. For each 2D line in L^0_{2D}, compute the plane that is obtained back-projecting the line to the 3D world.

3. Iterate over all lines in L^0_{3D} and find the 3D line segment that is closest to the plane. The distance from the line to the plane is computed as the average distance between the two line end-points and the plane. If the distance of the closest line is greater than a given threshold then discard the matching.

4. Repeat for the second pair of corresponding 2D and 3D line clusters L^1_{2D} and L^1_{3D}.

The result of this closest line search is a list of 3D-2D line pairs (l^i_{3D}, l^i_{2D}). Now to solve for the camera position, we employ the non-linear algorithm presented in [Kumar and Hanson, 1994], which enforces the condition that the two end-points (p^i_0, p^i_1) of a 3D line l^i_{3D} must lie in the plane that is formed by back-projecting the corresponding image line l^i_{2D}. This condition can be written as an optimization function over the set of all line correspondences:

$$E_t = \sum_{i=0}^{n} \sum_{j=0}^{1} \rho \left(\mathbf{n_i} \cdot (\boldsymbol{R} \cdot p_j^i + \mathbf{t}) \right) \tag{3.9}$$

where $\mathbf{n_i}$ is the normal to the plane formed by back-projecting l_{2D}^i, $\boldsymbol{R}$ is the rotation matrix, and ρ is a weighting function that adds robustness to the estimation by weighting down outliers. [Kumar and Hanson, 1994] use the following function proposed by Mosteller and Tukey:

$$\rho(u) = \begin{cases} u^2/2 + u^4/2a^2 + u^6/6a^4 & \text{if } |u| \leq a \\ a^2/6 & \text{otherwise.} \end{cases} \tag{3.10}$$

This function bounds the influence of outliers to a fixed value.

3.3.7 Registration examples and results

We show now results of using our line-based registration tool on images of **CEPSR Hall** and **Pupin Hall**, two buildings in the Columbia University campus. In figures 3.3 and 3.4 we show the registration process, step by step, for each of the buildings:

1. The color image and the camera parameters are loaded (top left picture).
2. The point cloud is loaded and displayed on top of the image (top right picture).
3. The rotation is solved for and the point cloud is displayed correctly oriented (bottom left picture).
4. The user drags the rendering of the point cloud until it closely matches the image (bottom right picture).

After these steps the user clicks on the button labeled *Compute translation*, which triggers the closest line search, after which the final camera position is computed. In figure 3.5 we show a rendering of the 3D and 2D extracted lines of CEPSR Hall after the registration has been completed. The 3D lines are colored green and the 2D lines are shown in red. Observe how our algorithm brings the matching lines into correspondence.

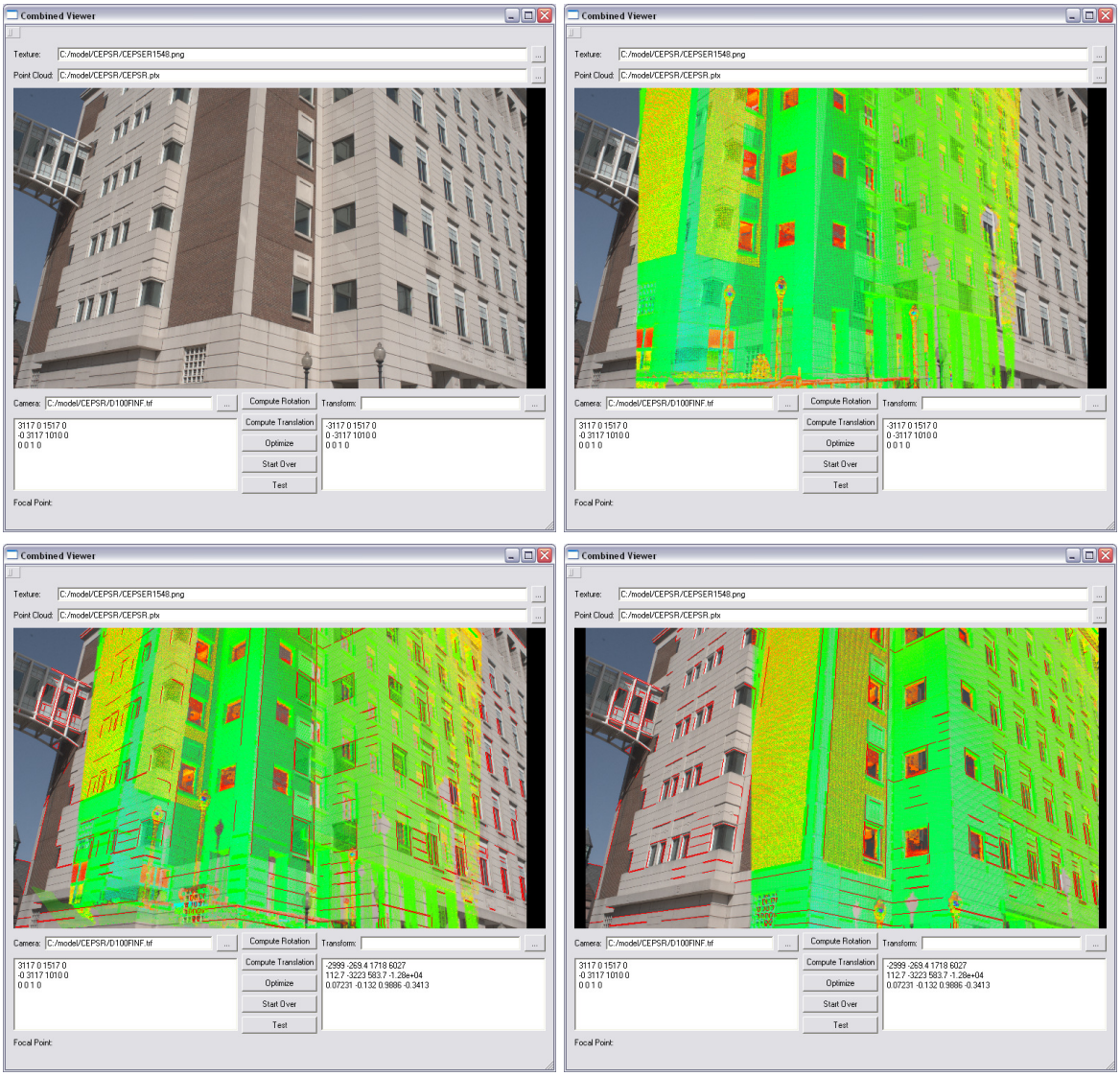

Figure 3.3: Steps in the semi-automatic line based registration. *Top left:* The image and the camera parameters have been loaded. *Top right:* The point cloud has been loaded and is shown over the image. *Bottom left:* After automatically computing the rotation, the point cloud is shown correctly oriented. *Bottom right:* The user dragged the model over the image until the rendering of the point cloud closely matches the image.

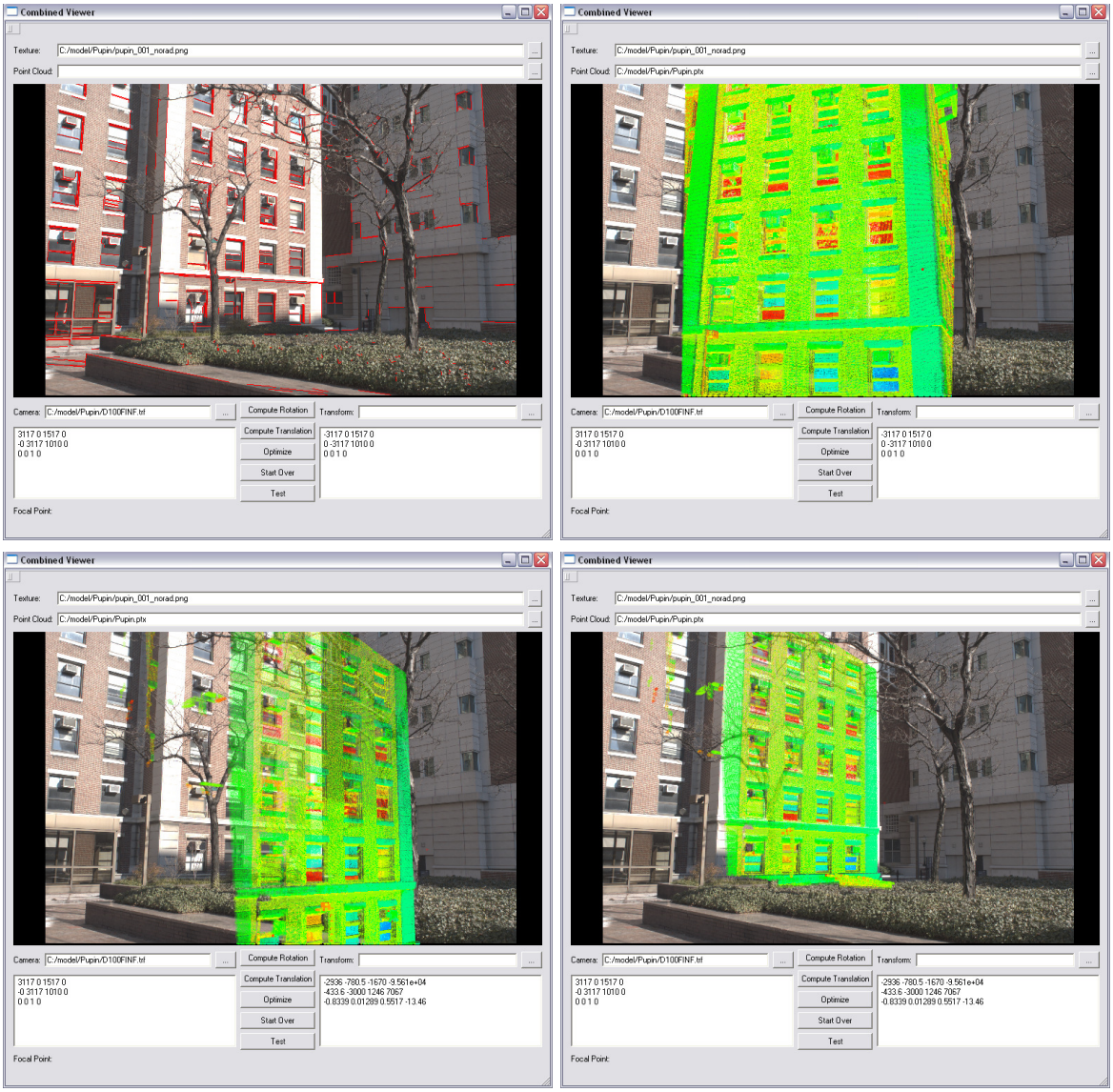

Figure 3.4: Steps in the semi-automatic line based registration of Pupin Hall. *Top left:* The image and the camera parameters have been loaded. *Top right:* The point cloud has been loaded and is shown over the image. *Bottom left:* After automatically computing the rotation, the point cloud is shown correctly oriented. *Bottom right:* The user dragged the model over the image until the rendering of the point cloud closely matches the image.

		Rotation parameters			Camera position			Rep. Error
Image	Tool	θ_x	θ_y	θ_z	C_x	C_y	C_z	
CEPSR	Point	-0.128	-0.075	-0.016	-4.67	-0.51	-2.70	2.00
CEPSR	Line	-0.133	-0.072	-0.012	-4.74	-0.62	-2.91	2.92
Pupin	Point	0.027	0.984	-0.238	-28.87	-0.81	-4.35	1.40
Pupin	Line	0.023	0.986	-0.24	-28.97	-0.87	-4.36	2.39

Table 3.1: Image registration results. This table shows the computed camera rotation and position for the images of CEPSR and Pupin Halls using the point-and-click and the semi-automatic line based method. Each row shows the result obtained from a given image and tool. The rotation parameters are given in radians, using Euler angles, and the camera position is shown in meters. The last column shows the reprojection error in pixels of the set of manually selected correspondences used with the point and click method. The test images are 3008 pixels wide by 2020 pixels high.

For a quantitative measure of the quality of the resulting registrations, we compared the results of the line-based method with the results obtained using the point-and-click tool, which are shown in table 3.1. Each row in the table shows the camera position and orientation obtained from a given image and registration tool. The camera orientation is shown as a set of Euler angles in radians, and the camera position as a 3D vector in meters. Finally, the last column shows the average reprojection error, in pixels, for a set of point correspondences that we manually selected using the point-and-click tool. We computed one set of correspondences for the image of CEPSR Hall and another set for the image of Pupin Hall. We then projected the 3D points to the image using the camera settings obtained by our tools and computed the average reprojection error. Given the large size of the images we used (3008 x 2020 pixels), the reprojection errors reported are small. Comparing the point-and-click tool against the line-based registration, we noticed that the reprojection error is slightly larger for line-based registration, but not significantly larger. Both tools perform almost equally well.

3.3.8 Summary of line-based registration

The line-based registration tool simplifies the registration process significantly, without requiring manual selection of features. Even though a user is still required to provide an initial estimate of the camera position, the time required for this step is much less than the

time required in the manual selection of correspondences.

There are still some areas for improvement. The current algorithm sensible to outliers generated during the closest-line search, biasing the results of the computed camera position. Even when using the robust technique of [Kumar and Hanson, 1994], we have found that the number of outliers can be significant if the estimate of the camera position is not good enough. However, since the user immediately receives visual feedback after the registration process is finished, a better initial estimate can be provided and the camera position can be re-computed. These incorrect matches are mostly caused by lines that are present in one data set and missing in the other. In the closest-line search step, the algorithm tries to find a closest-match for every backprojected 2D line. But, if there does not exist a true corresponding line in the 3D data, which might happen, and there is another line that is within the threshold distance, an incorrect matching pair is generated. Hence, there is a trade-off between the search distance threshold and accuracy of the initial position estimate. The search distance threshold can be lowered, potentially reducing the number of mismatches. However, this would require the user to provide a better initial estimate. For more robust results and at the expense of a higher computation time, the registration process could fully operate automatically if we used a RANSAC [Fischler and Bolles, 1987] style approach to find a correct set of line matches.

3.4 Conclusions

In this chapter we have presented two different tools for the registration of range and intensity images. These tools achieve the registration task with different degrees of automation and user intervention. The point-and-click tool is the most user intensive. Using line features, that can be automatically computed on the range and image data, we were able to reduce the time required for registration. Line segments proved to be good features for image registration in domain of architectural scenes. But away from man made structure, it is hard to come by sets of straight parallel lines. It is in these cases that other kind of

features, like shadows as we will discuss in the next chapter, can provide useful information for registration purposes.

As regards accuracy, one must note that the best results are still achieved by manually selecting correspondences. One of the main problems automatic registration methods must address is feature extraction and matching on two different domains: the domain of image intensities and the domain of 3D range data. Feature extraction on these domains are completely different tasks which depend on different conditions. For example, while feature extraction in intensity images will be affected by illumination conditions and cast shadows, the accuracy of feature extraction in 3D range data will depend on the sampling rate of the range finder, which can vary spatially depending on the distance of points to the sensor. Hence, working on two different domains makes the problem harder, because it is difficult to compare metrics that were obtained over a set of pixels with metrics that are obtained over a set of 3D points.

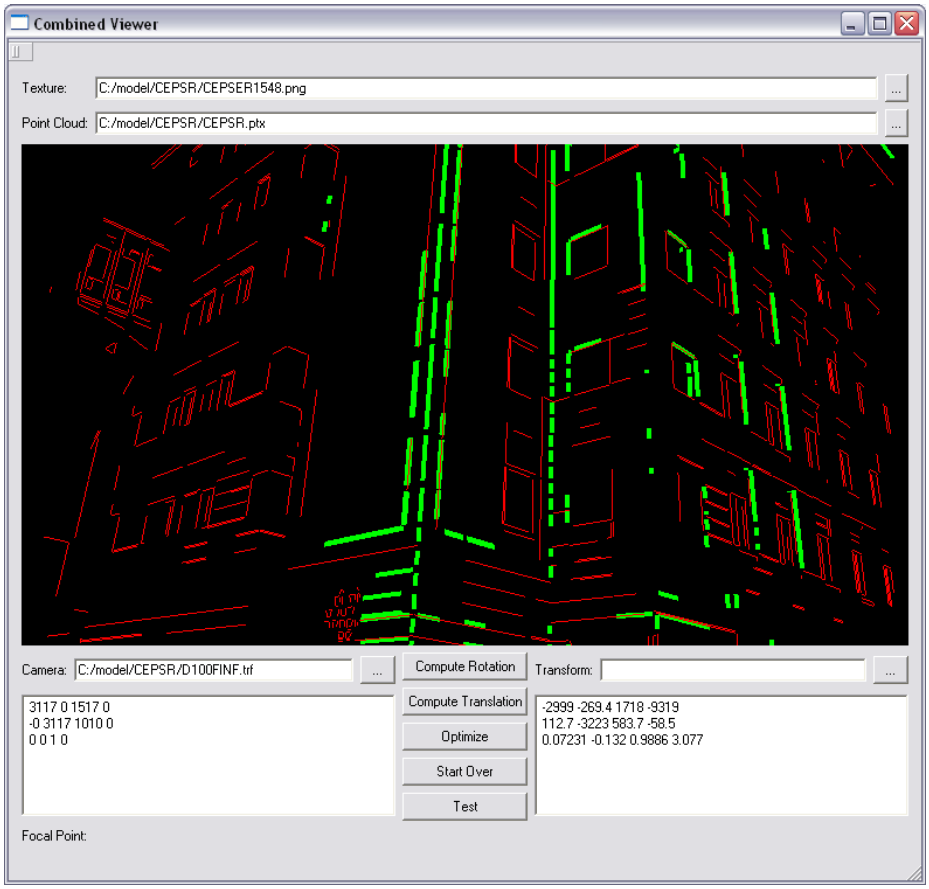

Figure 3.5: Line based registration results. After running the closest line search and computing the camera position, the 3D and 2D lines are brought into alignment. In the above picture, the 3D lines are shown in green and the 2D lines in red.

Chapter 4

Shadow-based color and range image registration

In this chapter we present an algorithm for the registration of color and range images that uses the shadows cast by the sun [Troccoli and Allen, 2004]. As shown in the previous chapter, straight lines are good features for image registration in architectural scenes; however, they are not so abundant outside man made structures. We have developed the shadow-based algorithm algorithm to construct a model of the Acropolis at Monte Polizzo, in Sicily, an archaeological site excavated by a team from the Stanford Archaeology Center. By using the shadows as cues for the registration we can overcome the inherent lack of traditional geometric features. In fact, shadows have been used in many computer vision applications. For example, when the light direction is known, shadows can reveal information about scene structure [Daum and Dudek, 1998, Yu and Chang, 2005, Kriegman and Belhumeur, 1998, Irvin and David M. McKeown, 1989]. In addition, when the geometry is known, shadows can provide information about the illumination of the scene [Sato *et al.*, 2003]. In our case, both scene structure and light source position (position of the sun) are known; we take advantage of this information and use it to compute the camera position.

Our shadow-based algorithm finds the position and orientation of a camera with known intrinsic parameters. We based our algorithm on the observation that when the correct

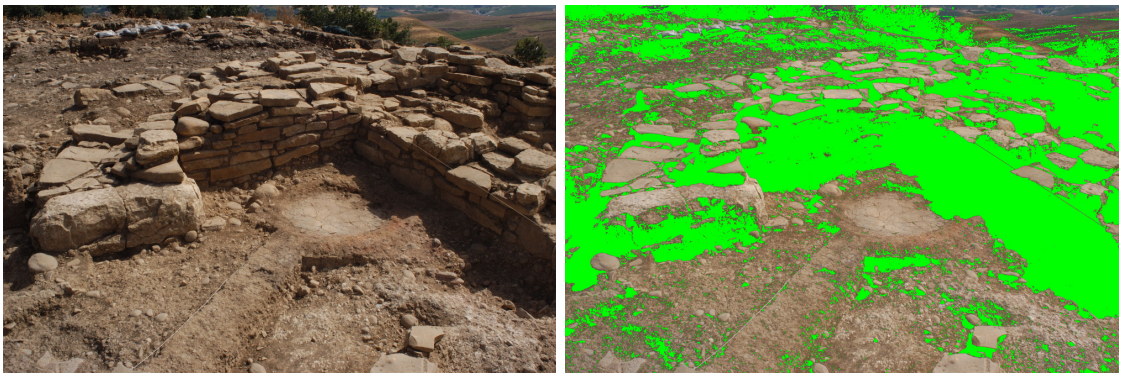

Figure 4.1: One view of the Acropolis at Monte Polizzo. The image on the right shows the shadows masked in green.

camera rotation matrix and translation vector pair ($\boldsymbol{R_f}$, $\mathbf{t_f}$) is known, an orthographic rendering of a textured version of the model viewed from the position of the sun should show no shadow pixels. However, if the texture is misaligned, shadow pixels will be visible. As an example, figure 4.1 shows one view of the Acropolis at Monte Polizzo and the detected shadows. In figure 4.2 we show two views of the model as seen from the sun, generated from correct and incorrect image registrations. It can be observed that the rendering that was generated using the correct image registration shows almost no shadow pixels (masked in green). Following this idea, we frame our solution as an optimization problem. Given an initial camera position ($\boldsymbol{R_0}$, $\mathbf{t_0}$), we search the parameter space of Euclidean transformations in the vicinity of this initial configuration for a point that minimizes a cost function whose value is proportional to the number of visible shadow pixels in the rendered model.

Since our technique is based on shadow detection and matching, the following preconditions apply: shadows should be detectable in the image; the 3D model's geolocation (latitude, longitude and orientation with respect to North must be known); and objects casting shadows should be present in the model. The last two assumptions are typically met in archaeological excavations and other 3D outdoor modeling applications. On the other hand, the requirement that shadows be present in the images might not always be satisfied. This by no means invalidates our method; it makes it one more tool available in our 3D modeling system. For images with shadows, we use the tool; for images without

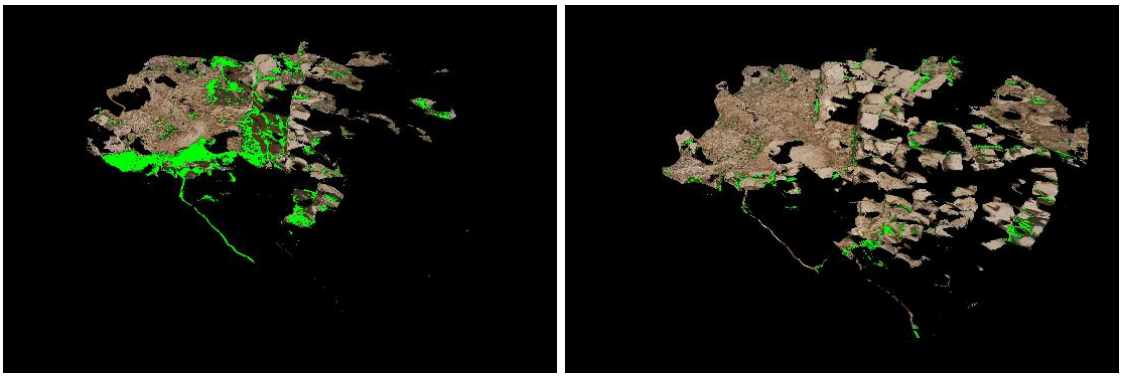

Figure 4.2: Two renderings of the model as seen from the direction of the sun. Left: when the image registration is incorrect, the number of visible shadow pixels (in green) is large. Right: the correct registration minimizes the number of visible shadow pixels.

shadows, we can fall back to a user-based manual registration tool. The main difference between these two registration tools is the amount of user interaction. Where the manual point and click registration tool is user intensive, the shadow-based method only requires minimum user interaction.

To build the 3D model of the Acropolis at Monte Polizzo we acquired a set of range and intensity images to cover the entire region of interest within the archaeological site. The range sensor used was a Cyrax 2500 time-of-flight laser scanner, which can gather 1 million points within a field of view of 40 by 40 degrees. Each point measurement consists of its 3D Cartesian coordinates (x,y,z) in the scanner's local coordinate system and a fourth value representing the amplitude of the laser light reflected back to the scanner. At each scanning position, we also acquired a photograph by placing the camera in close proximity to the scanner. Thus, the complete acquisition process interleaves a 3D sensing with 2D sensing.

The acquired range scans were aligned to the same coordinate system using fiducial markers that were placed in the scene before scanning, and that are optically designed to be recognized by the scanner. The scanner measures the position of each marker in its local coordinate system and, at the same time, we also measured the coordinates of each marker in the site's coordinate system using a total station device that was initialized from a set of geo-referenced control points. Thus, for each range scan we had a set of corresponding

points measured in both, the local scanner coordinate system and the site's coordinate system, from which we computed an alignment transformation. From this set of registered point-clouds we built a triangular mesh using the VripPack package developed by Curless and Levoy [Curless and Levoy, 1996].

After the 3D geometric model is constructed, the shadow-based image registration is completed in three steps:

1. Shadow are detected and masked in the 2D image.

2. The rendering window is set up. The position of the sun is computed and an appropriate rendering window size is selected.

3. A cost function is minimized. This is an iterative minimization process in which the model is rendered as seen from the sun and the cost function is evaluated over the rendered image.

In the remaining of this chapter we explain in detail the three main steps of the algorithm, and derive a suitable cost function for the optimization step. Finally, we present results on synthetic cases and real data.

4.1 Shadow detection in the image

As a first step in our shadow based algorithm, shadows in the input image are masked with a pre-defined color. We detect the shadow regions in the image using global thresholding on the luminance channel. Though it is known that accurate detection of cast shadows and their boundaries is a difficult task due to the complex effects of penumbrae and inter-reflections, shadows cast by the sun are significantly dark because the intensity of sun light is much greater than skylight. For this reason, we have found that shadow detection by global thresholding to produce good results in most of our test cases, except for late afternoon images. If required, other methods for shadow detection could be used [Funka-Lea and Bajcsy, 1995, Salvador *et al.*, 2004]. In any case, it is important to note that for

the algorithm to work, perfect shadow detection is not required, as long as the following conditions are met:

1. Some cast shadows are detected, but it is not necessary for all of them to be detected.
2. It is desirable that shadows due to surface normals pointing away from the sun be also detected as shadows.
3. The number of non-shadow pixels that are masked as shadow pixels should be small.

For shadow detection using global thresholding, these conditions define the ideal threshold to be the minimum luminance value of a non-shadow pixel. In our system, threshold selection is an interactive process in which a user selects an appropiate value that best meets the above conditions.

4.2 View setup

The next step after the shadows in the image have been masked is the setup of the rendering window. In this step, the size of the rendering window is chosen by the user of the system. Next, to achieve the effect of rendering the model as it would be "seen" from the sun we set up an orthographic projection with the view vector parallel to the direction of the sun rays. This direction is calculated using an astronomical formula [Reda and Andreas, 2003] that takes as inputs the time-stamp of the image and the latitude and longitude values of the site. Our system sets the view direction automatically, and lets the user translate the model within the rendering window until the desired section of the model is visible. The view setup is complete when the area of the model imaged in the photograph is fully seen in the rendering.

4.3 Cost function definition and optimization

The final step in the algorithm is the minimization of a cost function that depends on the number of visible shadow pixels in a rendering of the model. In practice, we have found

that using the number of visible shadow pixels alone as the cost function will not always allow the algorithm to converge to the correct camera position. If the optimizer selects a candidate pose in which the intersection of the camera viewing frustum and the model is small, then the rendering might show a small number of shadow pixels, not due to the selection of a better camera pose, but because the camera is looking at a much smaller region of the model. Therefore, a good cost function should penalize camera configurations in which the intersection of the camera's viewing frustum and the scene is small or empty. This can be achieved by keeping track of the number of textured pixels in the rendered model. The more pixels that are textured, the larger the area the camera is viewing. We these guidelines, we derived a suitable cost function.

Let I denote the image to be registered, M the model, and I_r a rendered image of M textured with I and camera parameters $(\theta, \phi, \omega, d_x, d_y, d_z)$. Then, we define the cost function as:

$$f(I_r) = \begin{cases} \frac{shadow\_count(I_r)}{number\_of\_pixels(I_r)} & \text{if } v(I_r) \geq t \\ 1.0 & \text{otherwise.} \end{cases} \tag{4.1}$$

where $v(I_r)$ is the count of textured pixels and t is a threshold value. This threshold forces the camera to a position that looks at the scene. In practice, we have defined this threshold as a fraction of the total number of textured pixels in the rendering produced when the camera pose is set to its initial estimate.

Since the cost function f defined in (4.1) has no analytical derivatives and typically contains several local minima, most optimization methods based on gradient descent techniques fail to converge to a good solution. For this reason, we employ a variant of simulated annealing [Ingber, 1989], which has the advantage of avoiding local minima by randomly sampling the parameter space and occasionally accepting parameters that drive the search uphill to a point of higher cost, as opposed to gradient descent methods, that only take downhill steps.

The optimization process searches the six dimensional space of camera positions and orientations. For the optimization to converge to a minimum, a suitable parametrization of

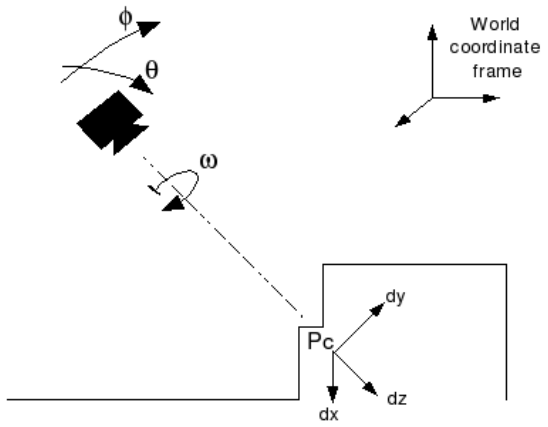

Figure 4.3: Search space parametrization. A new reference coordinate system is created by translating the camera coordinate frame to the point where the camera's optical axis intersects the scene (from the initial camera configuration). The space of rotations and translations is then defined with respect to this new coordinate system.

this space is required. Note that the trivial parametrization, using a 3-vector for the camera position and a 3-vector with the euler angles of the camera orientation, has the disadvantage that a small change in the camera orientation can produce a large displacement of scene elements within the image (specially those that are far from the camera). To overcome this problem we chose a parametrization in which the center of rotation is placed in the scene and not in the camera. Consider a camera restricted to lie on a sphere of radius r around a fixed point P_c in the model; then the space of camera configurations allowed in this case can be defined by the 2D spherical coordinates of a point in the sphere and a roll angle. To add the three remaining degrees of freedom, we allow the sphere center to translated away from P_c. This parametrization, shown in figure 4.3, describes the entire six dimensional space of camera positions and orientations. The camera pose is described by a 6-vector $(\theta, \phi, \omega, d_x, d_y, d_z)$, where θ and ϕ are the spherical coordinates of the camera position in the sphere, ω is the camera's roll angle and (d_x, d_y, d_z) is the displacement of the sphere center from P_c. We set P_c by taking the initial camera pose and tracing a ray along the camera's view direction to find the intersection between this ray and the model.

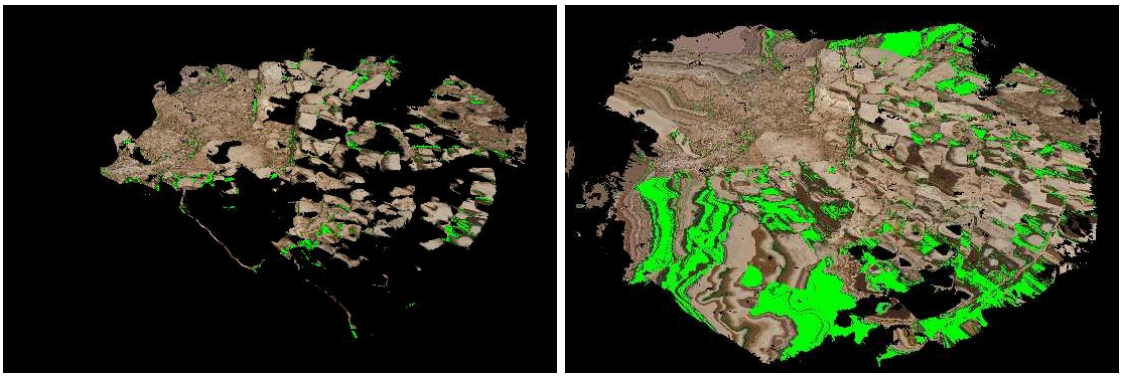

Figure 4.4: A correct rendering of the model requires occlusion detection, as shown in the image on the left hand side. Projective texture mapping alone does not yield the correct results, as seen in the right image.

At each iteration of the optimization, the minimizer provides a set of camera parameters $\mathbf{c}_k$. Using these parameters, the model is rendered and the cost function is evaluated over the generated image. To obtain a correct rendering requires occlusions to be detected and accounted for. Not every scene point is visible from the texture camera. If projective texture mapping is used, as shown in figure 4.4, these occluded points will still be textured. We can avoid this problem by applying a technique similar to shadow mapping [Segal *et al.*, 1992]. First, we render the scene from the position of the camera and make a copy of the depth buffer. Then we set the viewpoint to the position of the sun and re-render the model with shadow testing enabled so that scene points that are not visible from the camera are neither textured nor rendered.

4.4 Results and robustness analysis

We have run different sets of synthetic and real experiments to test the performance and robustness of the presented algorithm with respect to the different sources of error: 1) selection of shadow threshold, 2) resolution of 3D model (i.e accuracy of scanned geometry), 3) accuracy of the sun position. All of these three parameters can affect the final registration result.

For our simulation experiments we used one of the images of the archaeological site. To

Table 4.1: Shadow registration simulation results

Shadow threshold		30	40	60	80	140	60	60	60
Resolution (F/D)		F	F	F	F	F	D	F	F
Time offset (min)		0	0	0	0	0	0	-10	+10
Run #	Initial	(A)	(B)	(C)	(D)	(E)	(F)	(G)	(H)
1	143.5	10.3	5.4	6.0	5.1	6.9	14.1	5.6	6.0
2	101.4	5.8	5.8	4.1	5.0	7.0	9.0	6.0	5.9
3	208.3	9.2	5.8	4.5	4.9	6.8	14.5	5.9	5.8
4	125.3	10.7	5.7	4.9	5.6	6.4	16.8	5.2	5.6
5	171.2	7.8	5.0	4.9	6.0	6.1	11.3	6.0	5.3
6	81.1	6.8	5.0	4.7	5.2	7.0	14.4	5.3	6.7
7	207.4	8.8	4.7	4.9	5.7	6.2	12.4	5.0	8.5
8	77.0	10.1	5.3	5.2	4.9	6.4	11.4	5.5	5.3
9	238.4	7.7	4.8	4.8	5.0	6.3	12.7	5.8	5.1
10	180.6	7.2	5.4	4.8	5.8	6.9	13.0	5.4	4.9
Avg	**153.4**	**8.4**	**5.3**	**4.9**	**5.3**	**6.6**	**13.0**	**5.6**	**5.9**

obtain ground truth registration, we placed scanner's fiducial targets on the scene. The 3D position of these targets was measured by the range scanner with a precision higher than 3 millimeters. We manually selected these targets in the image and computed the camera pose using the linear pose estimation method of [Ansar and Daniilidis, 2003] followed by nonlinear optimization. Then, we created a sequence of camera positions by randomly perturbing the orientation of the camera by as much as ± 5 degrees in each rotation angle and the translation by ± 0.25 meters in each axis. From each of these positions, we ran our algorithm setting the visibility threshold in equation (4.1)to 60%.

Table 4.1 shows the results of 10 simulation runs. We use the reprojection error of the mesh vertices as a metric to evaluate the resulting camera: first the mesh vertices are projected to image coordinates using the ground-truth camera, then the image coordinates are computed using the camera position resulting from our algorithm. The reprojection error is the distance between these two points. Each column in table 4.1 shows the computed average reprojection error for different shadow thresholds and model resolutions. Columns (A) to (E) were obtained using a high resolution model and varying shadow thresholds (30, 40, 60, 80 and 140). On column (F) we show results for a decimated version of the

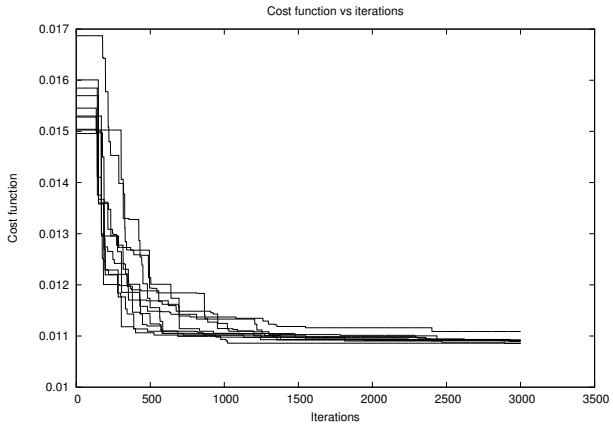

Figure 4.5: Cost function optimization. This plot shows the best cost found against the number of iterations for ten simulation runs. It can be seen how the optimization converges.

model and on columns (G) and (H) we show the simulation results for two cases in which the position of the sun had been computed from a time value that was ten minutes away from the actual time. The execution time for each simulation run was approximately 12 minutes on a Pentium IV machine. This time corresponds to 3000 minimization iterations. The progress of the iterative minimization over time can be seen in figure 4.5, which is a plot that shows cost of the the best configuration found against the number of iterations for each of the 10 simulations run in column (E). This plot shows that the minimization process driven by simulated annealing converges to a minimum.

4.4.1 Robustness against shadow threshold

By looking at each of the columns (A)-(E) in table 4.1, we can observe how the algorithm behaves with respect to the selection of the shadow threshold. The average reprojection error does not vary much when the shadow threshold is in the range [40,80], but it does increase for smaller and larger values. For a threshold value of 30, the algorithm does not perform as well because the low threshold fails to select most of the pixels corresponding to cast shadows. As a result, there are many camera configurations for which the number of shadow pixels is minimized and the algorithm performs poorly. A threshold of 40 does

detect most of the cast shadows and some attached shadows, improving the registration results. A value of 60 detects more shadows and labels some non-shadow pixels incorrectly as shadowed, but the results are not affected. Only when the shadow threshold is increased significantly to 140, where a large number of non-shadowed pixels are masked as being in shadow, is there a significant change in the performance of the algorithm. The image used in our test is large, 3008 x 2000 pixels, hence an average re-projection error of 4.9 pixels, as shown in column (C), is unnoticeable. This error corresponds to roughly 1.8 pixels for a 1024 x 768 image.

4.4.2 Robustness against geometry resolution and sun position

To evaluate the effects that errors in the geometry can introduce in our algorithm, we decimated the model reducing the number of triangles by 80 %, and performed a set of simulation runs using a threshold value of 60 in the shadow detection. The results are shown in column (F) of Table 4.1. In this case, the average reprojection error is higher than the error obtained for the same shadow threshold (column (E)) using the full resolution model. However, one advantage of using a decimated model is an increase in running speed, since less geometry has to be processed. This suggests an area of future exploration that could allow a speed increase: to run the optimization first with a highly decimated model and then refine the obtained registration with a more detailed one.

We also ran the experiments introducing a small error in the time of the day used to calculate the position of the sun. The results for a time offset of ± 10 minutes are shown in columns (G) and (H) of Table 4.1. The reprojection errors are higher in these cases, but not significantly, suggesting that a highly accurate time of the day (and hence position of the sun) is not required.

4.4.3 Results on archaeological data

We used our registration method to align the images of our Acropolis model. The model we used consisted of a 138K triangle mesh, for which we acquired twelve 3008 x 2000

pixels texture images. Using our algorithm, we were able to successfully align ten of the twelve images. In two cases, the algorithm failed to find a good camera registration and we had to fall back to manual registration. For one image the algorithm failed to find the intersection of the camera's optical axis and the scene (i.e. point P_c) because of a holes in the mesh. For a different image, the algorithm failed because the shadows in the image, which had been taken in a late afternoon, did not have enough contrast and some non-shadowed regions were incorrectly masked as shadowed. The resulting model of the Acropolist at Monte Polizzo is shown in figure 4.6. Each picture shows the model from a different view point. In addition, a short video showing an animation of the entire model www.cs.columbia.edu/∼allen/sicily.avi. In the animation, we combine the model with a cylindrical panorama to add visual context and enhance the overall visual experience.

4.5 Conclusions

We have presented an algorithm for texture registration that uses the shadows cast by the sun. We successfully applied our method to build a model of a real archaeological site. Our algorithm helps reduce the amount of user involvement in the modeling process and find high-quality results. We have identified the different sources that can introduce errors in the optimization and shown quantitative results for the performance of the algorithm under different conditions. From our simulation experiments we can conclude that the algorithm is robust to variations in the shadow detection and the sun position.

The range of applications in which one could potentially use this shadow-based algorithm is determined by three conditions. First, strong cast shadows are necessary. Second, all geometry that is casting shadows in the scene must be present in the 3D model. And finally, it is also desirable, but not strictly required, that there be geometry for every pixel in the image. This is not always the case, and sometimes there is no geometry for distant objects or the sky. In this cases, a user can mask out these regions. However, this last restriction is not intrinsic to our shadow-based method but also applies to other texture registration

techniques (e.g. silhouette based algorithms require the background to be masked out).

Finally, an area of future improvement is the execution time. The iterative optimization using simulated annealing requires a large number of iterations to converge. This execution time is determined by the number of triangles in the model and the size of the rendering window. Hence, one could probably find an increase in performance by adopting a hierarchical scheme in which the model is rendered on different window sizes and at different resolutions.

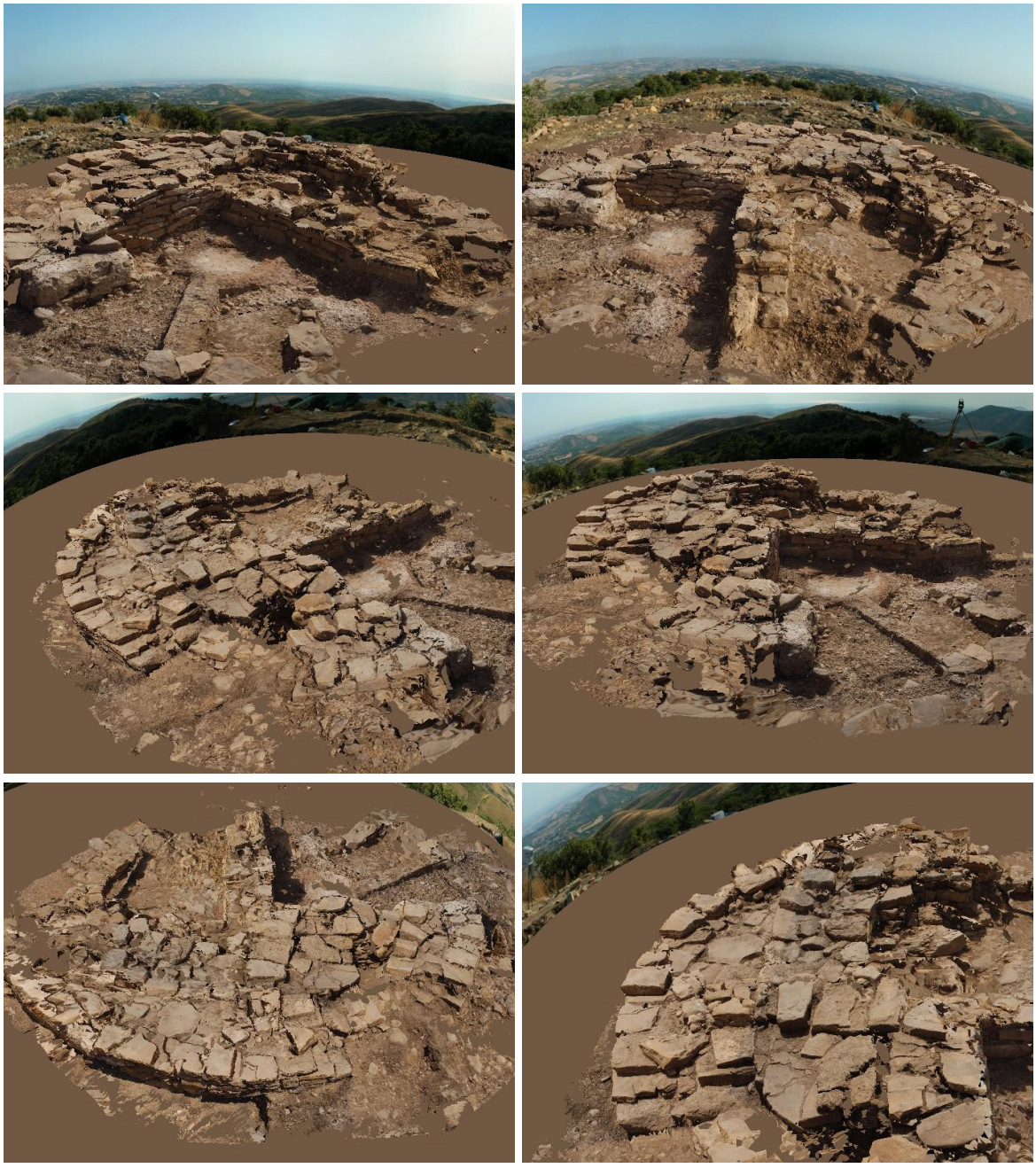

Figure 4.6: Six different views of the textured model of the Acropolis at Monte Polizzo, created using our shadow-based registration. The background is given by a panoramic mosaic.

Chapter 5

Texture relighting and de-shadowing

In this chapter we deal with the problem of texture fusion, i.e. how to combine multiple intensity images with the range data to produce photo-realistic renderings. In particular, we analyze the problem of combining intensity images of urban environments captured under different (unknown) illumination conditions by applying a combination of relighting and de-shadowing operation [Troccoli and Allen, 2005]. The relighting operation brings two images to the same illumination; while the de-shadowing operation removes any shadows that are present in the image. This approach is a significant departure from the traditional methods of range and intensity image rendering presented earlier in chapter 2 that either use a weighted average of the input images [Pulli *et al.*, 1997, Debevec *et al.*, 1996, Buehler *et al.*, 2001] or apply a global color correction matrix [Agathos and Fisher, 2003, Bannai *et al.*, 2004]. In addition, it does not require any kind of apparatus to measure the incident illumination as in [Debevec *et al.*, 2004].

The algorithm we introduce computes a relighting operator by analyzing intensity values in the area of overlap of a pair of images: a source image to be relighted, and a target image whose illumination we want to match. By applying this operator over the non-overlapping region of the source image, we transform its color intensities in such a way that they are

consistent with the colors observed in the target image. The operator we compute to perform the relighting is the ratio of intensity values per surface orientation, which under certain assumptions, is consistent over all points with the same orientation and only depends on the illumination of the scene. From this orientation-consistency property[1] it follows that if we compute the intensity ratio in the region of overlap of a pair of images, then we can relight any surface point outside that region provided there was a point with the same orientation in the region of overlap. Orientation-consistency holds for Lambertian objects under distant illumination in the absence of local illumination effects such as shadows and interreflections, and in this case, the intensity ratio turns out to be an irradiance ratio that is independent of the albedo variations over the surface of the object. Furthermore, the Lambertian BRDF assumption can be relaxed if the camera is orthographic and the viewpoint is fixed. Under these new conditions orientation-consistency will hold as long as points with the same orientation have the same BRDF.

The idea of using intensity ratios for relighting was also used in in [Marschner and Greenberg, 1997, Beauchesne and Roy, 2003]. In this dissertation we extend this concept to relighting scenes with ambient illumination and shadows cast by occlusion of a single point light source. In particular, we apply our method to relighting outdoor scenes, mostly man-made structures such as buildings, in which shadows are observed due to occlusions of the sun. In outdoor environments, the sun and the sky are the two main sources of illumination. To extend the concept of intensity ratios to handle shadows, we compute four ratios per surface normal: one that relates non-shadow irradiance in one image to non-shadow irradiance in the other image, another that relates shadow irradiance to shadow irradiance, and two additional ones for each of the shadow to non-shadow and non-shadow to shadow mappings. In the presence of shadows, orientation-consistency will hold for scenes that are mostly convex with few concavities.

[1]The term orientation-consistency is introduced in [Hertzmann and Seitz, 2003] in the context of photometric stereo with general BRDFs.

5.1 Problem definition

Given,

1. $\mathcal{G}$ the geometry of the scene.
2. $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ a set of photographs of the scene captured under illumination conditions $\mathcal{L} = \{L_1, L_2, \ldots, L_n\}$. Some of the images overlap.
3. $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ the set of camera projection matrices for each image.

Our goal is to create a textured model of $\mathcal{G}$ as illuminated by one of $L_r \in \mathcal{L}$: we want to relight all images in $\mathcal{I}$ to illumination L_r and remove, if possible, any shadows present in the images. Therefore the output is a new set of images $\mathcal{I}^r = \{I_1^r, I_2^r, \ldots, I_n^r\}$. In its most atomic form, the relighting and de-shadowing operations are applied to a pair of images at a time, and can be extended to multiple images by successive execution of pairwise operations.

For two images I_i and I_j the following three steps are required for relighting I_j to the illumination of I_i:

1. Detection of shadows in I_i and I_j.
2. Computation of the four irradiance ratio maps over the region of overlap of the two images.
3. Relighting and de-shadowing of I_j using the computed IRMs.

5.2 Theoretical background

In this section we develop the theory behind the relighting and de-shadowing algorithms. First we explore the relighting equation for scenes without shadows, then we generalize to those cases where shadows are present, and finally we present the de-shadowing algorithm.

5.2.1 The relighting equation

Recall the reflected radiance equation (2.1) introduced in chapter 2:

$$B(\mathbf{x}, \theta_o, \phi_o) = \int_{\Omega_i} L(\mathbf{x}, \theta_i, \phi_i) f_r(\mathbf{x}, \theta_i, \phi_i, \theta_o, \phi_o) \cos\theta_i d\omega_i, \tag{5.1}$$

where L is the incident illumination and f_r the BRDF at surface point $\mathbf{x}$. For Lambertian surfaces, equation (5.1) becomes:

$$B(\mathbf{x}) = \rho(\mathbf{x}) \int_{\Omega_i} L(\mathbf{x}, \theta_i, \phi_i) \cos\theta_i d\omega_i, \tag{5.2}$$

where $\rho(\mathbf{x})$ is the surface albedo at $\mathbf{x}$. For simplicity, define the irradiance at $\mathbf{x}$ as

$$E(\mathbf{x}) = \int_{\Omega_i} L(\mathbf{x}, \theta_i, \phi_i) \cos\theta_i d\omega_i. \tag{5.3}$$

Then the reflected radiance equation (5.2) turns into:

$$B(\mathbf{x}) = \rho(\mathbf{x}) E(\mathbf{x}). \tag{5.4}$$

Note that there is still a dependence on the surface position, on both the albedo ρ and the irradiance E. If we force orientation consistency to hold, under the assumptions that the illumination of the scene is distant, the effects of interreflections are negligible and there are no shadows, then equation (5.4) can be reparametrized by the surface normal at $\mathbf{x}$ which we denote as $\mathbf{n_x}$ to yield:

$$B(\mathbf{x}) = \rho(\mathbf{x}) E(\mathbf{n_x}). \tag{5.5}$$

Consider two different observations of the same surface point $\mathbf{x}$ under different and unknown illuminations L_i and L_j. The image irradiance values for these two observations are:

$$B_i(\mathbf{x}) = \rho(\mathbf{x})E_i(\mathbf{n_x}),$$
$$B_j(\mathbf{x}) = \rho(\mathbf{x})E_j(\mathbf{n_x}).$$

Taking the ratio of these two expressions we obtain:

$$R_{ij}[\mathbf{n_x}] = \frac{E_i(\mathbf{n_x})}{E_j(\mathbf{n_x})}, \tag{5.6}$$

since the albedo terms cancel out. R is an irradiance ratio, and is only dependent on the surface orientation at $\mathbf{x}$. For every surface orientation that is present in the region of overlap of two images we can define an irradiance ratio. The union of the irradiance ratios over all possible orientations defines an irradiance ratio map (IRM). It can be verified that given the image irradiance of a surface point $\mathbf{x}'$ under illumination L_j we can compute its image irradiance under illumination L_i by taking the product with the corresponding irradiance ratio value, as follows:

$$B_i(\mathbf{x}') = B_j(\mathbf{x}')R_{ij}[\mathbf{n_{x'}}]. \tag{5.7}$$

This is verified by substitution of equations (5.5) and (5.6) into (5.7). Thus, we have defined a relighting operator based on the orientation-consistency assumptions that allows us to relight an image I_j to the illumination of I_i. The only requirement is to have sufficient surface orientations in the area of overlap of I_i and I_j to be able to relight the non-overlapping region of I_j.

5.2.2 Relighting in the presence of shadows.

Architectural scenes can contain structures that cast shadows. When shadows are present, orientation-consistency does no longer hold because there can exist two points $\mathbf{x}$ and $\mathbf{x}'$ with the same surface orientation that do not have the same intensity ratio. This will

happen if in either of I_i or I_j one of these points is in shadow and the other one is not. However, it is possible to extend the relighting algorithm based on intenstiy ratios to handle shadows. For a scene illuminated by a single source plus smooth ambient illumination (e.g. sun plus sky) there exists a shadow mapping function $\mathcal{S} : \mathcal{G} \rightarrow [0, 1]$, that assigns a value of 1 to those scenes points completely shadowed, a value of 0 to those scene points that are completely lit by the source, and an intermediate value in the range $(0, 1)$ to those points that are in the penumbra regions. Momentarily ignoring penumbra regions, $\mathcal{S}$ is a binary function that partitions the scene into two sets: a set $\mathcal{G}_0$ of points lit by the source and a set $\mathcal{G}_1$ of shadowed points. In addition, in terms of surface orientations, points with the same surface orientation will also be partitioned in two sets. Therefore, when taking the ratio of two images I_i and I_j, the surface orientations in the scene will be partitioned into four different sets $\{\mathcal{G}_{00}, \mathcal{G}_{01}, \mathcal{G}_{10}, \mathcal{G}_{11}\}$ according to the values of the shadow bits $\mathcal{S}_i$ and $\mathcal{S}_j$. We can now redefine the orientation-consistency property for scenes with shadows: when orientation-consistency holds, two points with the same surface orientation and same shadow bit value will show the same intensity ratio. Under these new conditions, we can compute four different IRMs from the ratio image: $R_{ij}^{00}, R_{ij}^{01}, R_{ij}^{10}$ and R_{ij}^{11}. Orientation-consistency will hold in this generalized case with shadows for scenes with mostly diffuse or Lambertian surfaces, distant illumination and negligible effects of interreflections. The relighting equation for a point $\mathbf{x}$ becomes:

$$B_i(\mathbf{x}) = B_j(\mathbf{x}) R_{ij}^{s_i s_j}[\mathbf{n}_\mathbf{x}], \tag{5.8}$$

where s_i and s_j are the shadow bits of $\mathbf{x}$ under illuminations L_i and L_j. This expression is almost identical to equation (5.7), with the exception of the index into one of the four computed IRMs, and is well defined for binary values of s_i and s_j. The points for which s_i or s_j are not in $\{0, 1\}$ are points in the penumbra regions. Penumbra regions are transitions from shadow to non-shadow regions (or viceversa) in which the irradiance varies gradually. To deal with points in penumbra we can generalize the relighting operator for real values

of s_i and s_j in the following manner: first, we define the four base cases for binary shadow values to be equal to the measured data:

$$R_{ij}[\mathbf{n_x}, 0, 0] = R_{ij}^{00}[\mathbf{n_x}], \tag{5.9}$$

$$R_{ij}[\mathbf{n_x}, 0, 1] = R_{ij}^{01}[\mathbf{n_x}], \tag{5.10}$$

$$R_{ij}[\mathbf{n_x}, 1, 0] = R_{ij}^{10}[\mathbf{n_x}], \tag{5.11}$$

$$R_{ij}[\mathbf{n_x}, 1, 1] = R_{ij}^{11}[\mathbf{n_x}]. \tag{5.12}$$

$$\tag{5.13}$$

We recall now from the definition of the irradiance ratio, that we can write the IRMs in terms of its constituent components:

$$R_{ij}^{00}[\mathbf{n_x}] = \frac{E_i^0(\mathbf{n_x})}{E_j^0(\mathbf{n_x})} \qquad R_{ij}^{01}[\mathbf{n_x}] = \frac{E_i^0(\mathbf{n_x})}{E_j^1(\mathbf{n_x})}$$
$$R_{ij}^{10}[\mathbf{n_x}] = \frac{E_i^1(\mathbf{n_x})}{E_j^0(\mathbf{n_x})} \qquad R_{ij}^{11}[\mathbf{n_x}] = \frac{E_i^1(\mathbf{n_x})}{E_j^1(\mathbf{n_x})},$$

where E^0 and E^1 refer to non-shadow and shadow irradiance respectively. We first define the irradiance in the penumbra regions by linear interpolation of the respective shadow and non-shadow irradiance values:

$$E_i[\mathbf{n_x}, s_i] = E_i^1(\mathbf{n_x})s_i + E_i^0(\mathbf{n_x})(1 - s_i). \tag{5.14}$$

Now, we can define the relighting operator for real values of s_i and s_j as:

$$R_{ij}[\mathbf{n_x}, 0, s_j] = \frac{E_i^0(\mathbf{n_x})}{E_j^1(\mathbf{n_x})s_j + E_j^0(\mathbf{n_x})(1-s_j)} \tag{5.15}$$

$$R_{ij}[\mathbf{n_x}, 1, s_j] = \frac{E_i^1(\mathbf{n_x})}{E_j^1(\mathbf{n_x})s_j + E_j^0(\mathbf{n_x})(1-s_j)} \tag{5.16}$$

$$R_{ij}[\mathbf{n_x}, s_i, s_j] = \frac{E_i^1(\mathbf{n_x})s_i + E_i^0(\mathbf{n_x})(1-s_i)}{E_j^1(\mathbf{n_x})s_j + E_j^0(\mathbf{n_x})(1-s_j)}. \tag{5.17}$$

Finally, we write the above equations in terms of the measured data:

$$R_{ij}[\mathbf{n_x}, 0, s_j] = \frac{1}{\frac{1}{R_{ij}^{01}[\mathbf{n_x}]}s_j + \frac{1}{R_{ij}^{00}[\mathbf{n_x}]}(1-s_j)} \tag{5.18}$$

$$R_{ij}[\mathbf{n_x}, 1, s_j] = \frac{1}{\frac{1}{R_{ij}^{11}[\mathbf{n_x}]}s_j + \frac{1}{R_{ij}^{10}[\mathbf{n_x}]}(1-s_j)} \tag{5.19}$$

$$R_{ij}[\mathbf{n_x}, s_i, s_j] = R[\mathbf{n_x}, 1, s_j]s_i + R[\mathbf{n_x}, 0, s_j](1-s_i). \tag{5.20}$$

We can now state the generalized relighting equation for scenes with shadows as:

$$B_i(\mathbf{x}) = B_j(\mathbf{x})R[\mathbf{n_x}, \mathcal{S}_i(\mathbf{x}), \mathcal{S}_j(\mathbf{x})] \tag{5.21}$$

5.2.3 De-shadowing

De-shadowing is a variation of the relighting problem, in which the resulting relighted image is shadow-free. It is straight forward to convert the relighting equation (5.21) into a de-shadowing equation by requiring all surface points in $\mathcal{G}$ to be lit under the target illumination L_i. This is equivalent to re-defining $\mathcal{S}_i$ to map all points to 0. Hence, the de-shadowing equation is:

$$B_i(\mathbf{x}) = B_j(\mathbf{x})R[\mathbf{n_x}, 0, \mathcal{S}_j(\mathbf{x})] \tag{5.22}$$

We can also consider the problem of self de-shadowing, in which we remove the shadows

of an image without recurring to relighting a second image. By using the computed IRMs over a pair of images, it is possible to define a de-shadowing IRM as:

$$R_i^0 = \frac{E_i^0}{E_j^0} \times \frac{E_j^0}{E_i^1} = \frac{R_{ij}^{00}}{R_{ij}^{10}}, \tag{5.23}$$

and the self de-shadowing equation is:

$$B_i^0(\mathbf{x}) = B_i(\mathbf{x}) R_i^0[\mathbf{n_x}] \mathcal{S}_i(\mathbf{x}) + B_i(\mathbf{x})(1 - \mathcal{S}_i(\mathbf{x})). \tag{5.24}$$

Note that we can also compute the de-shadowing IRM from:

$$R_i^0 = \frac{E_i^0}{E_j^1} \times \frac{E_j^1}{E_i^1} = \frac{R_{ij}^{01}}{R_{ij}^{11}}, \tag{5.25}$$

5.2.4 Extending to multiple images

To work with multiple images, each acquired under a different illumination, we proceed chaining pairwise operations. First we select the image I_r whose illumination we will convert all other images to. Then, for every image that overlaps with I_r we can proceed as already explained. If there is an image I_j that does not overlap, then we apply a multi-step relighting. For example, if image I_i overlaps with both I_r and I_j, then we can first convert I_j to the illumination of I_i and then to the illumination of I_r, using the previously computed IRMs. The problem we might run into is that there might not be enough surface orientations in the respective areas of overlap to relight all points in the image.

If we have multiple images, but there exist a subset of these images that were all acquired under the same illumination, then we can compute the IRMs using the contributions from all images in this subset.

5.3 The relighting and de-shadowing pipeline

This section describes in detail the stages of the relighting and de-shadowing pipeline. Most of the operations in the relighting pipeline are done in image space using high-dynamic range linearized images. These are created from a set of variably exposed images. For each set of images, we take the camera's raw sensor values, which are linear with respect to the light intensity, compute the exposure ratios between the images, and create the final high-dynamic range image using a weighted average.

The inputs to our relighting operation are then: a collection of high-dynamic range images taken under different illuminations, the geometry of the scene and the camera's projection matrices. To handle variations in viewpoint in our pairwise operations, we warp one of the images to the viewpoint of the other one by back-projecting the image pixels to the geometry and projecting these to the new viewpoint.

Note that the relighting equations (5.7) and (5.21) are both defined in terms of a surface point $\mathbf{x}$. There exists, however, a correspondence between pixels in the images and surface points in the 3D world. We assume each pixel represents a single surface point. When we iterate and perform operations over all pixels in an image, we are in fact working on the corresponding 3D points and surface normals.

Figure 5.3 shows an overview of the steps involved in the relighting and de-shadowing pipeline. We first create the high dynamic range images and register the images with the geometry. We then pick a pair of images to work on, warp one of the images to the viewpoint of the other one if necessary, and find the shadow regions in the images. After this we iterate over all pixels and compute the IRMs. Finally, we perform the relighting operation. A detailed description of each of these steps follows.

5.3.1 Shadow detection

In the shadow detection stage we compute a coarse shadow map $\mathcal{S}_i$ associated to an image I_i. The shadow detection operation assigns each pixel in I_i a real value in $[0, 1]$, where 0

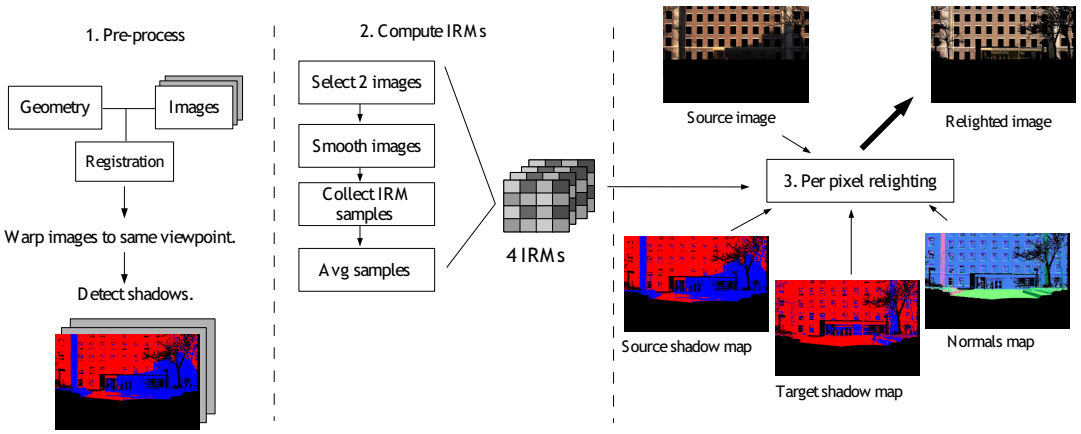

Figure 5.1: The stages of the relighting and de-shadowing pipeline

means the corresponding point in the scene is lit, 1 means it is completely shadowed, and an intermediate value means the point is in a penumbra region. It is well known that shadow detection from a single image is a difficult problem, and for this reason, many researchers perform shadow detection on image sequences or video (e.g. [Chuang *et al.*, 2003]).

In our application, knowledge of the scene geometry can help find the shadows if the position of the light source is known. Nevertheless, scanned geometry will not yield shadows that are correct to pixel level accuracy because of holes in the geometry and the effects of coarse sampling. For this reason, an image-based method for shadow detection is necessary. We employ a user-assisted method, where the user selects regions of the image and thresholds the selected region using a pair of thresholds s_0 and s_1. All pixels with a luminance value that below s_0 are marked as shadowed, all pixels with luminance above s_1 are marked as lit, and pixels in between are labeled as penumbra pixels with a value of $(l - s_0)/(s_1 - s_0)$ where l is the pixel luminance.

Before thresholding we filter the images using bilateral filtering [Tomasi and Manduchi,

1998] to remove the high frequency effects of texture. The bilateral filter is designed to average spatially close pixels that are similar to each other. This similarity condition acts as an edge-stopping function and overcomes the traditional problem of edge-blurring that is common of Gaussian filtering. The output of the bilateral filter for a pixel p is:

$$F_p = \frac{1}{k(p)} \sum_{p' \in \Omega} f(p - p') g(I_p - I_{p'}) I_{p'}, \tag{5.26}$$

where $k(p)$ is the normalization term:

$$\sum_{p' \in \Omega} f(p - p') g(I_p - I_{p'}).$$

In (5.26) above, f is a spatial domain Gaussian kernel with standard deviation σ_s and g is a range domain Gaussian kernel centered at pixel intensity I_p and standard deviation σ_r. The range kernel is easy to set-up for monochrome images, but for color images a similarity metric is needed. This is generally done by converting from RGB color space to CIE LAB or YUV space. Instead of applying a color space conversion, we use the luminance channel of the image to define the range kernel as suggested by Bennet (2006). The luminance channel combines the readings of the three R, G, and B components and will therefore be more robust to image noise.

After bilateral filtering and thresholding we obtain a shadow map. This shadow labeling need only be accurate for the shadowed and non-shadowed regions. Values assigned to penumbra regions will be ignored during the IRM data collection stage and can be later refined once the IRMs are computed, as explained in section 5.3.3.

5.3.2 Data collection and IRM computation

To compute the four IRMs we iterate over all pixels, computing the ratio of the two images. For every pixel, we look up its surface normal and shadow bits. If the pixel has been labeled as penumbra in either of the images, we ignore it. Otherwise, we use the shadow bits to establish which of the four IRMs the pixel contributes to. Each IRM is stored

in a 2D representation of the Gaussian sphere. The surface normal is used to index the corresponding entry to the current pixel. The final value stored in each IRM entry is the average of all values contributing to it. In addition, we compute one IRM for each of the three color channels.

5.3.3 Shadow map update

After the IRM data has been gathered, it is possible to update the shadow masks to find the correct values for the penumbra regions. This operation will work on all pixels in the area of overlap of the two images. The basic idea is to update the shadow map by comparing the true irradiance ratio at each pixel with the ratio obtained from the computed IRMs. Given images I_i and I_j , the current shadow masks $\mathcal{S}_i$ and $\mathcal{S}_j$, and the computed IRMs $\{R_{ij}^{00}, R_{ij}^{01}, R_{ij}^{10}, R_{ij}^{11}\}$, we update both shadow masks $\mathcal{S}_i$ and $\mathcal{S}_j$. To update $\mathcal{S}_j$ we iterate over all the pixels applying the following rules:

1. Lookup $\mathcal{S}_i(\mathbf{x})$ and $\mathcal{S}_j(\mathbf{x})$ and the surface normal $\mathbf{n_x}$.

2. If $\mathcal{S}_j(\mathbf{x})$ is either 1 (non-shadowed) or 0 (shadowed), skip the current pixel because only pixels labeled as penumbra will be updated.

3. Based on the value of $\mathcal{S}_i(\mathbf{x})$ solve for the shadow mask s_j using equation (5.13) if $\mathcal{S}_i(\mathbf{x})$ is 0, or equation (5.19) if $\mathcal{S}_i(\mathbf{x})$ is 1. If the computed value of s_j is outside the range $[0, 1]$, we clamp it to the nearest value in the range. If $\mathcal{S}_i\mathbf{x}$ is neither 0 or 1, then $\mathbf{x}$has been labeled in penumbra in the two images. In this case, which is quite unlikely, we can not update the shadow mask unless we assume one of the two masks is correct.

When we update the shadow masks we compute a new mask per color channel, as opposed of a single mask based on luminance. We have found that better results are obtained in this way. Finally, once $\mathcal{S}_j$ has been updated, the process can be repeated to update $\mathcal{S}_i$, this time using IRMs R_{ji} and the newly computed shadow map $\mathcal{S}_j$, and interchanging $\mathcal{S}_i$ and $\mathcal{S}_j$ in the steps already outlined.

5.3.4 Relighting

After the IRMs and shadow masks have been computed, the relighting or de-shadowing operations can be carried out. For every pixel p in the source image I_j, we compute the corresponding relighted pixel using equation (5.21) or the de-shadowed pixel using equation (5.22).

5.4 Results

In this section we present the results of applying the relighting and de-shadowing algorithms on three different models of buildings in the campus of Columbia University in New York City: a model of **Casa Italiana**, a model of **St. Paul's Chapel** and a model of **Pupin Hall**. The 3D range scans were acquired using a Leica HDS 3000 time-of-flight laser scanner, registered together by placing fiducial markers on the scene, and meshed using the VripPack package [Curless and Levoy, 1996].

The images used for the test on Casa Italiana are shown in Figure 5.2. These images were taken from slightly different view points and registered manually using the point and click registration tool of chapter 3. One of the images was acquired at 1:28pm and the other one at 3:22pm. Note that shadows are in different locations, and that in one case, one complete face of the building is shadowed. In figure 5.3 we show the same images, now aligned to the same viewpoint. We do the view-warp process by back-projecting the pixels to the scene, finding the distance to the camera of the corresponding surface point and projection to the viewpoint of the corresponding image. Hence, only those pixels for which we have geometry can be warped. The remaining pixels are left black, and correspond to holes in the 3D model. Figure 5.4 shows a color-coded normals map and a mask of the region used to compute the IRMs. Regions that do not correspond to diffuse surfaces, such as the windows, are masked out and ignored in computing the IRMs. The shadow masks were computed using the parameters shown in table 5.1. The table shows, for every image, the parameters we used in the bilateral filter operation and the thresholds used for shadow

detection. The thresholds are real numbers, since we are working with high-dynamic range images. After filtering and shadow detection we collected the IRM data and updated the shadow masks. The resulting shadows masks are shown in Figure 5.5. Finally, using these shadow masks and the compute IRMs, we ran the relighting and de-shadowing algorithms to turn the image that was acquired at 1:28pm to the illumination at 3:22pm. In figure 5.6 we show both, the relighted image and the de-shadowed image. Note that the relighted images preserves the shadows while the de-shadowed image has none. One important aspect to note about the de-shadowed image is that regions that were in shadow in both of the input images have been successfully de-shadowed. For better visualization, figure 5.7 shows two composite pictures with the image before and after the relighting operation. Each composite picture is divided in two regions. The left region shows the original image of Casa Italiana acquired at 3:22pm. The right region, shows the image acquired at 1:28pm. In one of the composite pictures we show the original image at 1:28pm and in the other one, the relighted image. Note how in the latter case it is hard to distinguish the boundaries between the actual and relighted image.

In a similar way we conducted the experiments on the images St. Paul's Chapel and Pupin Hall. For the tests on St Paul's Chapel, figures 5.8 shows the input images, one of which was taken at 11:22am and the other one at 12:35pm. In figure 5.9 we show the corresponding shadow masks, and in figure 5.10 the results of applying the de-shadowing operation. In this case, we completely removed the shadows from the image acquired at 12:35pm. The picture on the left of figure 5.10 shows the obtained de-shadowed image, and the picture on the right is a composite picture in which, the left half corresponds to the original image and the right-half to the de-shadowed image. Note how the shadows were completely removed.

Finally, we show the images, shadow masks and results of the experiments on Pupin Hall. Figures 5.11 shows the input images, one of them taken at 10:30am on a cold winter morning, and the other one at 2:41pm on that same day. In figure 5.12 we showed a zoomed version of the images which focuses on the region of interest in which the relighting operation

	Casa Italiana		St. Paul's		Pupin	
	Image 1	Image 2	Image 1	Image 2	Image 1	Image 2
Bilateral filter	3.0, 0.4	3.0, 0.4	3.0, 0.4	3.0, 0.2	3.0, 0.2	3.0, 0.2
Shadow detection	0.1, 0.24	0.04, 0.20	0.1, 0.15	0.08, 0.14	0.04, 0.15	0.04, 0.2

Table 5.1: Parameters used in the relighting experiments. For the bilateral filter, the spatial σ_s and range σ_d are shown. For the shadow detection, the two thresholds s_0 and s_1 are listed. See section 5.3.1 for a description of the meaning of these values.

results are more noticeable. There are trees in the scene that had to be manually removed from the images for processing. In fact, trees are in general very problematic for both, color imaging and range sensing. The shadow masks we computed are shown in figure 5.13. In figure 5.14 we show the results of relighting and de-shadowing the image taken at 2:41pm to the illumination of the morning image. The left picture shows the result of the relighting operation, and the right picture the results of relighting plus de-shadowing. Finally, in figure 5.15 we show a side-by-side comparison of the images before and after relighting. The picture in the left is composite image that shows made of the original images side by side. The picture on the right is another composite images that shows the morning image together with the relighted afternoon image. Note how the relighting algorithm correctly solved chromatic differences in the light coming from the sun, the transition from the original to the relighted image is unnoticeable.

5.5 Discussion

Our relighting and de-shadowing algorithm produces high quality results, as shown in the previous section. It is important, however, to understand the possible sources of error that can affect the quality of the final image. These possible sources of error are:

Inadequate geometry sampling. The geometric models we use are obtained using a time-of-flight laser scanner that samples the scene in a discrete manner. Using discrete sampling we are only able to reconstruct geometry variations at half the frequency of the sampling rate. Rapid variations in the geometry are lost. Since surface normals

are computed by triangulation of the measured points, the algorithm is expected to work better in scenes with smooth varying surfaces than in rapidly changing scenes.

Registration errors. These are errors that are introduced when computing the camera parameters, either intrinsic or extrinsic. Registration errors can result in the incorrect mapping between a pixel and a surface normal, and in incorrect pixel correspondences between two images. Image-geometry registration is a difficult problem, and we have tried in our experiments to reduce registration errors as much as possible. Registration errors can influence the final result more or less depending on the scene. For example, small registration errors have less impact on geometrically smooth surfaces, where a small misalignment will still map a pixel to the correct surface normal. This is the case of flat walls, for example. Also, for smooth textures, small registration errors can be tolerated. It is for this reason that we apply a bilateral filter and smooth high frequency textures before computing the IRMs: we smooth out the texture and hence reduce the effects of image misalignments. Note that this smoothing is not a theoretical requirement but a practical one to make the algorithm more robust.

Shadow detection. The shadow labels define which of the four IRMs a sample contributes to. If a shadow label is incorrectly assigned, then the sample will incorrectly be attributed to the wrong IRM. For large number of samples per surface normal, a small number of outliers will not affect the results; but if there are only a few samples for a given surface normal, then the computed IRM could contain errors.

Non-Lambertian reflectance. Most real-world surfaces are not purely Lambertian. However, some diffuse surfaces have Lambertian-like behavior for a range of viewing directions. Hence, our relighting and de-shadowing algorithms will produce good results in these cases. In architectural scenes, windows can be a problem. Windows can act like mirrors, and significantly deviate from the Lambertian assumption. For this reason we mask windows out in our experiments, completely ignoring them during the IRM computation stage.

Interreflections and spatially varying ambient occlusion. When a scene is not perfectly convex, the effects of interreflections and spatially varying ambient occlusion can affect the results of the relighting and de-shadowing algorithms. Interreflections are indirect contributions of light bouncing off a surface and reaching another one. Spatially varying ambient light occlusions are variations in the cone of sky[2](or extended distant light sources) visible by a scene point. If two scene points with the same surface normal see a different cone of sky, then the irradiance at each of these points will be different, violating the orientation-consistency assumption. Also, the effects of interreflections and spatially varying ambient light occlusion are more noticeable in shadowed regions. In our algorithm, we do handle these indirectly and in an ad-hoc manner by allowing the shadow masks to take real values in the range [0,1].

5.6 Summary and conclusions

We have developed a relighting and a de-shadowing algorithm that is applicable for mostly convex outdoor scenes. We have shown results of these algorithms on different kind of buildings: polyhedral (as in Casa Italiana and Pupin) and rounded (as was the case of St. Paul's Chapel). In the next chapter we will explore the use of the ratio of images further, and use the IRMs to compute a parametric model of the illumination of the scene. This will allow us to compute a diffuse reflectance map which we will then use to render the scene under different and novel illumination conditions.

[2]Also defined as sky-aperture in [Narasimhan and Nayar, 2001]

Figure 5.2: Two images of Casa Italiana taken at different time of the day from slightly different view points. The left image was taken 3:22pm and the right one at 1:28pm on the same day, under partly cloudy conditions. Note that the shadows are on different locations and that one face of the building is completely shadowed at 1:28pm.

Figure 5.3: The same images of Casa Italiana, now warped to same view point and zoomed to show region of interest. The right image was warped by backprojecting each pixel, computing its distance to the camera and projecting to the viewpoint of the left image. Pixels for which we have no geometry can not be warped.

Figure 5.4: Left: Color encoded normal map of Casa Italiana. Right: Region over which the IRMs were computed.

Figure 5.5: Shadow masks for the two images of Casa Italiana

Figure 5.6: Relighted and de-shadowed images of Casa Italiana. Left: Image taken at 1:28pm relighted to illumination at 3:22pm. Right: Image relighted with shadows removed.

Figure 5.7: Side by side comparison of relighting results for Casa Italiana. Left: Composite picture before relighting; the left half of the image is the original input image acquired at 3:22pm, the right half is the image taken at 1:28pm. Right: Composite picture after relighting, where the right half is now replaced with the results of relighting the image taken at 1:28pm with the illumination at 3:22pm.

Figure 5.8: Two images of St Paul's Chapel at Columbia University. The left image was taken at 11:22am and the right one at 12:35pm.

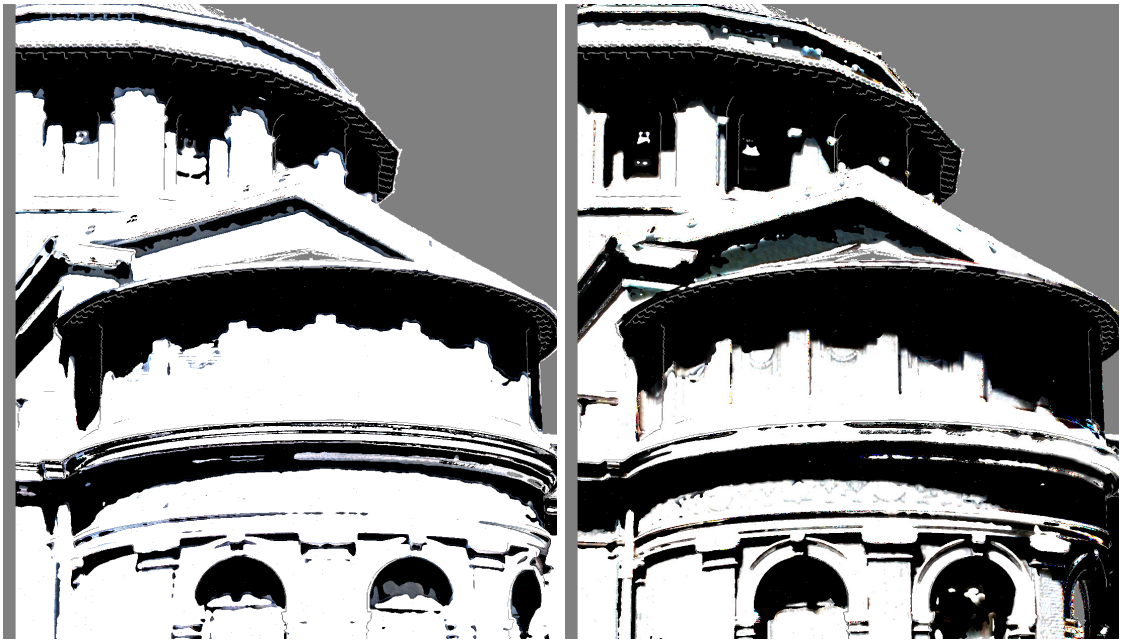

Figure 5.9: Shadow masks for St Paul's Chapel images.

Figure 5.10: De-shadowing results for the images of St Paul's Chapel. Left: de-shadowed image. Right: Composite image that shows, on the left, the original image and on the right, the de-shadowed image. Note how the shadows are successfully removed.

Figure 5.11: Two images of Pupin building.

Figure 5.12: Zoomed images of Pupin building.

Figure 5.13: Shadow masks for the images of Pupin.

Figure 5.14: Relighting and de-shadowing results for Pupin. Left: Relighting results. Right: De-shadowing results.

Figure 5.15: Pupin, side-by-side comparison of results before and after relighting. Left: Composite picture before relighting. Right: Composite picture after relighting.

Chapter 6

Illumination and texture factorization

In the previous chapter we introduced an algorithm for relighting and de-shadowing images taken under different unknown illumination conditions. We made use of the ratio image to compute a set of relighting and de-shadowing operators, which allowed us to bring two images into a consistent illumination. However, these operators do not provide any information about scene lighting or surface reflectance. In this chapter, we build on the concepts already set forth and look into the problem of illumination and texture factorization [Troccoli and Allen, 2006]. Our goal is to factor the illumination from the texture and solve for the shading of each image and the surface reflectance. In this way, we obtain an illumination-free texture map from a pair of images and the object geometry without prior recording or calibration of the incident illumination. Our assumptions are the same as before: mostly diffuse and convex scenes under distant illumination.

The method we present in this chapter falls in the category of inverse-rendering techniques, since we are measuring scene properties from images and objects of known geometry. While measuring surface reflectance of an object of known geometry under controlled and calibrated illumination has proved to produce very good results [Debevec *et al.*, 2000, Lensch *et al.*, 2003], working with unknown illumination is yet an open problem. Typically,

to handle unknown illumination it is assumed that the material properties of the object are homogeneous over the whole surface [Ikeuchi and Sato, 1991] [Ramamoorthi and Hanrahan, 2001a](i.e. the object is textureless). When dealing with textured objects, the problem of recovering both texture and illumination becomes unconstrained. As noted by Ramamoorthi and Hanrahan (2001b) in the development of a signal-processing approach for inverse rendering, lighting and texture can not be factored without resorting to active methods or making prior assumptions of their expected characteristics. Our method achieves this factorization by assuming diffuse surface reflectance, because as it has already been shown in chapter 5, the ratio of two images of a convex Lambertian object is texture-free and only depends on the incident illumination.

Before proceeding further, we shall define the most important terms we will be using in this chapter. We will frequently talk about:

Illumination maps. An illumination map is a function defined on the unit sphere that describes the intensity of the light arriving at a scene point from a given direction. In most of this chapter we will be dealing with distant illumination, assuming the illumination map of each point is the same.

Irradiance maps. An irradiance map is a function that is also defined on the unit sphere that maps a surface normal direction to incident irradiance. It is in essence, the convolution of an illumination map with the half-cosine function[1].

Albedo maps. An albedo map stores the spatially varying diffuse reflectance of the scene. The albedo of a scene point is the ratio of scattered radiance (the same in all directions) to incoming irradiance. Since albedo varies with the wavelength, we work with albedo maps defined over areas of the red, green and blue colors.

The factorization of illumination and texture is done in two steps: first we take the ratio of the input images and compute the illumination in the form of a pair of irradiance maps;

[1]The half-cosine function is the cosine function restricted to positive values.

second, we use the recovered irradiance to factor out the texture from the shading. Once we have obtained the texture in the form of an albedo map, we can render the object under new illumination conditions.

6.1 Problem definition

The definition of our problem is as follows. Given

1. $\mathcal{G}$ the geometry of an object.
2. $\mathcal{I} = \{I_1, I_2, \ldots, I_n\}$ a set of overlapping photographs of the object captured under unknown illumination conditions $\mathcal{L} = \{L_1, L_2, \ldots, L_n\}$.
3. $\mathcal{P} = \{P_1, P_2, \ldots, P_n\}$ the set of camera projection matrices that relates $\mathcal{G}$ with $\mathcal{I}$.

we want to recover the relative illumination L_i in the form of irradiance maps of each image and an albedo map of the scene. The images need not to be taken from the same viewpoint. Since we have the geometry of the scene and the projection matrices, we can warp any two overlapping images to the same viewpoint. To simplify the discussion that follows, we will only consider a single pair of images and assume that these have been warped to the same view. We consider three different illumination models: 1) an object illuminated by a point light source, 2) non-point light source illumination, 3) outdoor illumination represented as a combination of a point light source and an ambient component. The input to the illumination recovery procedure is a ratio image $R(x, y)$, which we compute by taking the quotient of the two input images, and a normals image $\mathbf{n}(x, y)$ which gives the normal for each pixel and that is generated by ray-tracing the geometry of the object.

We now give some background information about the use of ratio images and then we go into a detailed explanation of our method.

6.2 Background - The ratio image

In this section we define the ratio image and present some related work in the area of object recognition that makes use of variant form of the ratio image to address the task of recognition under variable illumination.

Under the same assumptions set forth in chapter 5, i.e. distant illumination and convex objects, the image of a diffuse Lambertian object is:

$$I(x, y) = \rho(x, y)E(\mathbf{n}(x, y)) \tag{6.1}$$

where I denotes the observed intensity at pixel (x, y), ρ denotes the albedo and E is the incident irradiance parameterized by the surface normal $\mathbf{n}$ at (x, y), and defined as the integral of the product of the incident light and the half-cosine function over the upper-hemisphere:

$$E(\mathbf{n}) = \int_{\Omega_i} L(\theta_i, \phi_i) \cos\theta_i d\omega_i. \tag{6.2}$$

When the illumination source is a distant point light source, the above expression simplifies to a dot product of the surface normal and the light direction:

$$E(\mathbf{n}) = \max(\mathbf{n} \cdot \mathbf{l}, 0), \tag{6.3}$$

where $\mathbf{l}$ is a unit vector in the direction of the light source. Given two different images I_1 and I_2 of the same object acquired from the same viewpoint, the ratio image R is defined as:

$$R(x, y) = \frac{I_1(x, y)}{I_2(x, y)} = \frac{E_1(\mathbf{n}(x, y))}{E_2(\mathbf{n}(x, y))}, \tag{6.4}$$

since the albedo terms in the numerator and denominator cancel each other. Hence, as shown in chapter 5, the ratio image is invariant to texture, and can be considered to represent the *irradiance ratio*.

A variant form of the above definition of ratio images has been used for class-based object recognition, with particular emphasis in face recognition under different illumination. Objects of the same class have the same geometry but different texture. For example, in face recognition, one can think of the class of faces as having same geometry but different texture. For this particular case, Shashua and Riklin-Raviv (2001) define the quotient image Q_{ab} of two different faces **a** and **b**:

$$Q_{ab}(x,y) = \frac{\rho_a(x,y)}{\rho_b(x,y)} \tag{6.5}$$

Under the definition above, the quotient image is the ratio of the albedos and is illumination free. The quotient image Q_{ab} can be computed from an image of face **a** and three images of face **b** illuminated by three non-collinear point light sources. To show this is possible, let I_1, I_2 and I_3 be three images of face **b** illuminated by three point light sources with direction $\mathbf{l_1}$, $\mathbf{l_2}$ and $\mathbf{l_3}$. An image of the same face under a different point light source direction can be obtained as a linear combination of these three images with coefficients x_j. Now, given an image I_a of face **a** illuminated by source with direction $\mathbf{l_a}$, the quotient image Q_{ab} can be written as:

$$\begin{aligned} Q_{ab}(x,y) &= \frac{\rho_a(x,y)}{\rho_b(x,y)} \\ &= \frac{\rho_a(x,y)\mathbf{n}(x,y)\cdot\mathbf{l_a}}{\rho_b(x,y)\mathbf{n}(x,y)\cdot\mathbf{l_a}} \\ &= \frac{I_a(x,y)}{\rho_b(x,y)\mathbf{n}(x,y)\cdot\sum_{j=1}^{3} x_j\mathbf{l_j}} \\ &= \frac{I_a(x,y)}{\sum_{j=1}^{3} x_j I_j(x,y)} \end{aligned}$$

In [Shashua and Riklin-Raviv, 2001], the images I_j are constructed from a bootstrap set of images of faces acquired under three different point light sources. The complete recognition algorithm consists of several steps: 1) compute the I_j images from the bootstrap set; 2)

given a test image I_a, find the coefficients x_j that define the position of the light source; 3) map all images in the face database to the same illumination and perform face matching. In essence, this recognition algorithm does solve for the illumination of the test image; but it is different to our algorithm various ways. First, it has a different purpose: to obtain an illumination invariant signature of the face; second, it does not make explicit use of geometry, instead it makes an implicit assumption that the geometry of faces are the same; and third, it requires a database of images to bootstrap the process. In more recente work, [Wang *et al.*, 2004] take this method a step further and generalize it to images illuminated by non-point light sources.

6.3 Methodology

We introduce now three different algorithms for illumination and texture factorization using ratio images. These algorithms address different situations that arise in practice:

1. The first situation we consider is that of an object illuminated by a distant point light sources. Given two images of this object under different point source illumination and the surface normals at each pixel, the algorithm solves for the direction of the lights and relative intensities.

2. Secondly, we solve for a more general form of illumination expressed as an expansion in terms of spherical harmonics. This algorithm can include cases that arise from illumination by area sources, or multiple point light sources.

3. Finally, we consider the case of a scene illuminated by a point source and ambient illumination. Such is the case we encounter in outdoor scenes, where the sun acts as a point source and the sky and the surrounding environment as an ambient component.

6.3.1 Point light source

When an object is illuminated by a directional point light source whose direction is described by a normalized 3-vector $\mathbf{l}_1$, the irradiance for a scene point $\mathbf{x}$ with associated normal $\mathbf{n_x}$ is:

$$E(\mathbf{n_x}) = L \max(\mathbf{n_x} \cdot \mathbf{l}_1, 0), \tag{6.6}$$

where L denotes the source intensity and $\cdot$ the vector dot product. Given two images illuminated by point sources with directions $\mathbf{l}_1$ and $\mathbf{l}_2$, respectively, the ratio image obtained is described by the following equation:

$$R(x, y) = \frac{L_1 \max(\mathbf{n}(x, y) \cdot \mathbf{l}_1, 0)}{L_2 \max(\mathbf{n}(x, y) \cdot \mathbf{l}_2, 0)}, \tag{6.7}$$

defined only for non-zero values of the denominator and numerator. Or goal is to solve for the direction of the light sources given the ratio image and the surface normals. It should be clear at this point that there will be an ambiguity in the light source intensities L_1 and L_2, since multiplying the numerator and denominator by the same constant in the above expression does not affect the final result. Hence, we can fix L_1 to unity and solve for $\mathbf{l}_1$ and $\mathbf{l}_2$ scaled by L_2. In the remaining of this section we simplify the notation and drop the (x, y) coordinates in $R(x, y)$ and $\mathbf{n}(x, y)$. Instead we use a single subindex to enumerate all pixels. Then, we can solve for the light direction and relative intensities from the following system of linear equations:

$$\begin{bmatrix} \mathbf{n}_0^T & -R_0\mathbf{n}_0^T \\ \vdots & \vdots \\ \mathbf{n}_k^T & -R_k\mathbf{n}_k^T \end{bmatrix} \begin{bmatrix} \mathbf{l}_1 \\ L_2\mathbf{l}_2 \end{bmatrix} = 0. \tag{6.8}$$

This is a linear system of the form $\boldsymbol{A}\mathbf{x} = 0$. The solution we are looking for is the one dimensional null-space of $\boldsymbol{A}$. When the dimension of the null-space of $\boldsymbol{A}$ is greater than one it will not be possible to solve uniquely for the light directions. This condition will arise

if the distribution of the imaged surface normals is small: e.g. if the scene is a flat wall. Given the null-space vector $\mathbf{x} = (x_0, x_1, x_2, x_3, x_4, x_5)$, we obtain $\mathbf{l_1}$, $\mathbf{l_2}$ and L_2 as:

$$\mathbf{l_1} = \frac{(x_0, x_1, x_2)}{||(x_0, x_1, x_2)||} \tag{6.9}$$

$$\mathbf{l_2} = \frac{(x_3, x_4, x_5)}{||(x_3, x_4, x_5)||} \tag{6.10}$$

$$L_2 = \frac{||(x_3, x_4, x_5)||}{||(x_0, x_1, x_2)||} \tag{6.11}$$

To handle color images we could treat each channel separately and solve (6.8) per channel. However, this typically yields three slightly different positions for the light source. We can obtain a more robust solution if we convert the image to luminance space and use the luminance values, instead. After we recover the direction of the light sources, the relative scale L_2 for each channel c is obtained from the original images by averaging the following expression over all pixels:

$$L_{2,c}(x, y) = \frac{\max(\mathbf{n}(x, y) \cdot \mathbf{l}_1, 0)}{R(x, y) \max(\mathbf{n}(x, y) \cdot \mathbf{l}_2, 0)}. \tag{6.12}$$

Also, note that nothing is known about the absolute chromaticity of the light sources. By fixing the intensity L_1 to the same value for the all three channels, we assume that light to be white. This chromatic ambiguity can not be solved without further assumptions or resorting to a color calibration object.

6.3.2 Generalized illumination

We can extend the previous case to a more general form of illumination. To do so, we define irradiance as an expansion in terms of spherical harmonic basis functions. Ramamoorthi and Hanrahan [Ramamoorthi and Hanrahan, 2001a] and Basri and Jacobs [Basri and Jacobs, 2003] have established that the image of a diffuse convex object under general illumination is well approximated by a low dimensional spherical harmonic expansion. Spherical harmonics

are orthonormal basis defined over the sphere. Using this framework, we can approximate the incident irradiance as:

$$E(\mathbf{n}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} A_l L_{lm} Y_{lm}(\mathbf{n}). \tag{6.13}$$

In (6.13) above, Y_{lm} are the spherical harmonic functions, L_{lm} are the spherical harmonic coefficients of the incident illumination, and A_l is a constant that represents the effects of multiplying the incident light with the half-cosine function. In other words, (6.13) is the frequency space equivalent of the integral (6.2) [Ramamoorthi and Hanrahan, 2001a]. In this context, we want to solve for L_{lm}. Since A_l decays very rapidly, a very good approximation can be obtained by limiting $l \leq 2$. A first order spherical harmonic approximation (up to $l = 1$) has a total of four terms and a second order approximation has a total nine. Before we write the expression of the irradiance ratio in spherical harmonics, we do one more notation change for clarity purposes. We replace the double-indexed Y_{lm} functions and L_{lm} coefficients by their single-index equivalent Y_s and L_s, where $s = l^2 + l + m$. Also, since we have to solve for two different illuminations L_s, we will denote these with L_{1s} and L_{2s}. Using this new notation, we can substitute (6.13) into our irradiance ratio expression to obtain:

$$R_i = \frac{\sum_{s=0}^{n} A_s L_{1s} Y_s(\mathbf{n}_i)}{\sum_{s=0}^{n} A_s L_{2s} Y_s(\mathbf{n}_i)}. \tag{6.14}$$

where $n = 4$ or $n = 9$ depending on the order of the desired approximation. We can now derive a system of linear equations similar to (6.8) on the unknown lighting coefficients L_{1s} and L_{2s}.

$$\begin{bmatrix} A_0Y_0(\mathbf{n}_0)\dots A_nY_n(\mathbf{n}_0) & -R_0A_0Y_0(\mathbf{n}_0)\dots \\ \vdots & \vdots \\ A_0Y_0(\mathbf{n}_k)\dots A_nY_n(\mathbf{n}_k) & -R_kA_0Y_0(\mathbf{n}_k)\dots \end{bmatrix} \begin{bmatrix} L_{10} \\ \vdots \\ L_{1n} \\ L_{20} \\ \vdots \\ L_{2n} \end{bmatrix} = 0. \tag{6.15}$$

The solution to (6.13) is once more the null-space of $\boldsymbol{A}$ and the coefficients L_{1s} and L_{2s} will be defined up to a scale factor. This scaling factor can be fixed by setting $L_{10} = 1$, which fixes both the relative scale and chromaticity of the illumination.

The well-conditioning of the system of equations (6.15) will depend on the distribution of surface normals. For better results and higher robustness against noise, it is possible to re-cast the problem in terms of principal components. This means replacing the spherical harmonic basis by lower dimensional orthogonal basis obtained using principal component analysis (PCA). The rationale behind this change of basis is that the principal components are vectors in the direction of greater variability (in this case due to illumination changes). Ramamoorthi (2002) derived an analytic expression for the principal components of the image of an object under all possible point light sources and showed that these are related to the spherical harmonic basis Y_s. In particular, Ramamoorthi shows that the eigenvectors obtained from PCA of the image space of an object illuminated under all possible point light sources can be well approximated as a linear combination of spherical harmonic functions up to order two. Let $\boldsymbol{V}$ be the matrix with the principal eigenvectors as columns, then there exists a matrix $\boldsymbol{U}$ such that $\boldsymbol{V} \approx \boldsymbol{YU}$, where $\boldsymbol{Y}$ is a matrix whose columns are the first nine spherical harmonics $Y_0...Y_8$. The matrix $\boldsymbol{U}$ can be computed analytically from the geometry of the object and details on how to do this are given in [Ramamoorthi, 2002]. Using the eigenvectors $\boldsymbol{V}_i$ as the new basis, we can now write the incident irradiance as:

$$E(\mathbf{n}) = \sum_{i=0}^{n} e_i \boldsymbol{V}_i(\mathbf{n}), \tag{6.16}$$

where e_i are the coefficients of the irradiance in principal components basis. The number of terms to employ in this new approximation will depend on the object geometry, but by looking at the eigenvalues associated to each vector it is possible to determine a good cut-off point. Now, we can write (6.15) as:

$$\begin{bmatrix} V_0(\mathbf{n}_0)\dots V_n(\mathbf{n}_0) & -R_0V_0(\mathbf{n}_0)\dots \\ \vdots & \vdots \\ V_0(\mathbf{n}_k)\dots V_n(\mathbf{n}_k) & -R_kV_0(\mathbf{n}_k)\dots \end{bmatrix} \begin{bmatrix} e_{10} \\ \vdots \\ e_{1n} \\ e_{20} \\ \vdots \\ e_{2n} \end{bmatrix} = 0. \tag{6.17}$$

Once we find the coefficients e_{1i} and e_{2i} we can find the corresponding L_{1s} and L_{2s} coefficients by substitution into:

$$L_{1s} = \frac{\sum_i^n \boldsymbol{U}_{si} e_{1i}}{A_s}, \qquad\qquad L_{2s} = \frac{\sum_i^n \boldsymbol{U}_{si} e_{2i}}{A_s}, \tag{6.18}$$

where $\boldsymbol{U}_s$ is the s^{th} row of $\boldsymbol{U}$. To handle color images we treat each of the RGB channels separately and solve for three sets of coefficients L_{1i} and L_{2i}. Once again, as before, there is an inherent chromatic ambiguity that we can only solve for if we have an image of a color calibration object.

6.3.3 Point plus ambient illumination

Consider now the case of outdoor illumination, where the sun is a distant point light source and the sky a hemispherical area source. We can model this situation as a sum of a point light source plus an ambient component. However, the sun is not just any directional light source. Its daily trajectory over the sky has been very well studied, and its position in the sky dome can be computed from the time of the day and the geo-location (latitude and

longitude) of the scene [Reda and Andreas, 2003]. Since these two pieces of information are easily available, the time being provided by the camera and the geolocation by a GPS unit or any of today's web-based mapping engines, we can assume the position of the sun to be known. Hence, our model of outdoor irradiance can be expressed as the combination of the sky irradiance plus a half-cosine term. To model sky irradiance, we can use a spherical harmonic expansion (or PCA expansion) as developed for the generalized illumination scenario. Then, outdoor irradiance is defined by the following equation:

$$E(\mathbf{n}) = P \max(\mathbf{n} \cdot \mathbf{s}, 0) + \sum_{s=0}^{n} L_s Y_s(\mathbf{n}). \tag{6.19}$$

Here L_s are the coefficients for the spherical harmonic expansion of sky irradiance, P is the relative intensity of the sun and $\mathbf{s}$ the sun direction. Since the ambient and direct components are modeled separately in equation (6.19), we can work with images with shadows, as we did in chapter 5. All we need is the shadow mapping function $\mathcal{S}$ defined earlier in section 5.2.2. Recall that this mapping assigns each scene point a value of 1 if the point is shadowed, a value of 0 when is lit by the sun, and intermediate value when it lies in the penumbra region. The outdoor irradiance equation (6.19) can be now be defined for a surface point $\mathbf{x}$ as:

$$E(\mathbf{x}) = \mathcal{S}(\mathbf{x}) P \max(\mathbf{n_x} \cdot \mathbf{s}, 0) + \sum_{s=0}^{n} A_s L_s Y_s(\mathbf{n_x}). \tag{6.20}$$

For two images taken at different times of the day, with the sun at directions $\mathbf{s_1}$ and $\mathbf{s_2}$ and relative intensities P_1 and P_2, the irradiance ratio of point $\mathbf{x}$ is given by:

$$R(\mathbf{x}) = \frac{\mathcal{S}_1(\mathbf{x}) P_1 \max(\mathbf{n_x} \cdot \mathbf{s_1}, 0) + \sum_{s=0}^{n} A_s L_{1s} Y_s(\mathbf{n_x})}{\mathcal{S}_2(\mathbf{x}) P_2 \max(\mathbf{n_x} \cdot \mathbf{s_2}, 0) + \sum_{s=0}^{n} A_s L_{2s} Y_s(\mathbf{n_x})}. \tag{6.21}$$

Our goal is to solve for the relative sun intensities P_1 and P_2 and the sky irradiance coefficients L_{1s} and L_{2s}. To solve for these unknown variables, we set up a system of linear equations similar to (6.15) and add the unknowns P_1 and P_2. Alternatively, we can also

work with the analytical PCA basis and solve a system of equations similar to (6.17). In any case, the same scale and chromatic ambiguities outlined earlier for the point light source and generalized illumination cases apply to this case as well. To resolve this ambiguity, we set the relative sun intensity value P_1 to one for all three channels. For robustness, we do not include point in penumbra in our system of equations.

6.3.4 Extracting the albedo map

After we have solved for the relative irradiance using one of the three models presented in the previous section, we can compute an albedo map for the scene. However, note that the chromatic and scale ambiguity in the estimated irradiance will translate to the estimation of the albedo map, which will also be defined up to scale. From the image pair I_1 and I_2 with estimated irradiance E_1 and E_2 we compute the albedos at each pixel:

$$\rho_1(x,y) = \frac{I_1(x,y)}{E_1(\mathbf{n}(x,y))} \tag{6.22}$$

$$\rho_2(x,y) = \frac{I_2(x,y)}{E_2(\mathbf{n}(x,y))} \tag{6.23}$$

$$\rho(x,y) = \frac{I_1(x,y)\rho_1(x,y) + I_2(x,y)\rho(x,y)}{I_1(x,y) + I_2(x,y)}. \tag{6.24}$$

In other words, for each pixel (x,y) we set its albedo $\rho(x,y)$ to a weighted average of the albedos we obtain from I_1 and I_2. The weights are set to the pixel intensities, so that dark pixels, which are more dominated by noise, are down-weighted.

6.4 Practical aspects

There are important aspects of the illumination and texture factorization algorithms that need to be addressed when putting these algorithms into practice:

6.4.1 Surface normal aggregation.

Regardless of selection of illumination model, each pixel in the ratio image provides one constraint to the system of equations. For medium to large images, having an equation for every single pixel is not practical due to the size of the resulting equation matrix. Instead, we can aggregate the ratios per surface normal. To do this, we can take any 2D parametrization of the sphere and compute the average normal and the average ratio for each (u, v) coordinates.

6.4.2 Weighted least-squares minimization.

The system of linear equations we have defined for each of the three illumination models is of the form $\|\boldsymbol{A}\mathbf{x}\| = 0$. In most situations, we will be dealing with the case in which there are more equations than unknowns, resulting in an over-determined system. The trivial solution $\mathbf{x} = 0$ is not of interest, we seek instead a non-zero $\mathbf{x}$. For robustness against outliers, we formulate the problem in a weighted least-squares way. This is equivalent to minimizing the $\boldsymbol{C}$-norm $\|\boldsymbol{A}\mathbf{x}\|_{\boldsymbol{C}}$, where $\boldsymbol{C}$ is a positive-definitive matrix. Typically, $\boldsymbol{C}$ is a diagonal matrix, with each element of the diagonal being the weight of a row in $\boldsymbol{A}$. The solution we are seeking is the non-zero vector $\mathbf{x}$ that minimizes:

$$\boldsymbol{A}^\top \boldsymbol{C}\boldsymbol{A}\mathbf{x} = 0, \tag{6.25}$$

which can be obtained from the SVD decomposition of $\boldsymbol{A}^\top \boldsymbol{C}\boldsymbol{A}$. In our solution, we set the elements of $\boldsymbol{C}$ to the number of pixels that contributed to a particular surface normal. After aggregating all pixels with the surface normal together, as described earlier, we can find how many contributed to each orientation. In this way, orientations that are more predominant are given higher weights.

6.4.3 Concavities and shadowing.

Most of the theory presented in this chapter was developed for convex scenes and no self-shadowing. Shadows were taken into consideration when developing the point plus ambient illumination model, but for the point light and generalized illumination models, self-shadowing can be a problem. How to address this problem depends on the chosen illumination model.

Point-light illumination. For the point-light illumination model, shadows in the object will be significantly darker than illuminated points, since the only light these shadowed regions will receive is from inter-reflections. Therefore, we discard shadowed regions by ignoring the dark pixels in the image. We implemented this technique by setting a threshold on the luminance value of the pixels. Pixels that do not meet the threshold condition are excluded in all stages of our algorithm.

Generalized illumination using SH basis. When using the generalized illumination model with either SH or PCA basis we also use thresholding of dark pixels. However, thresholding shadowed regions might not work well in all cases, depending on illumination and scene geometry. In chapter 7 we discuss a different an alternative solution.

Point plus ambient model. In our point plus ambient illumination model shadows are handled using shadow masks. These shadow masks indirectly model the effects of concavities and inter-reflections.

6.5 Factorization results

We tested the three factorization algorithms on different kinds of scenes under different types of illumination. In this section, we report the obtained results.

6.5.1 Point light source model

First, we ran the point light source on synthetic and real data imaged under point-light illumination. The images used in our tests are shown in Figure 6.1. The first two renderings are the synthetic scenes for which we used a model of a sphere and a model of the Armadillo[2] textured with a synthetic wooden pattern and rendered using a ray-tracer. We then included three objects that had been imaged under known point-light source: - a buddha, a cat and an owl[3]. The geometry and normals-map for these objects were obtained using photometric stereo. Finally, we included in our testing a set two objects that had been scanned using a Polhemus hand-held scanner[4]: a chicken and a figure of an Asian girl. The synthetic renderings and photometric stereo models are good for ground-truth comparisons, because we know the position of the light sources and do not require image registration. For the chicken and girl models, we captured several images varying the position of a point light source but leaving the viewpoint fixed. We then manually registered these images with the 3D model using the software tool described in chapter3.

We ran our point-light source estimation model of section 6.3.1 on all of the image pairs shown in Figure 6.1. Tables 6.1 and 6.2 show the ground truth and recovered light source positions and relative scales for the synthetic and photometric stereo models. For the synthetic scenes, the recovered light source directions and scaling factors shown in Table 6.1 are almost identical to the actual directions. Likewise, the computed light source directions for the buddha, cat and owl models listed in Table 6.2 are very close the ground truth data. We had no ground truth data for the chicken and Asian girl models. Nevertheless, we ran our algorithm to obtain the position of the light sources and obtain the factorization.

As a second step, we used the computed light direction to factor the input images into their corresponding texture (albedo maps) and shading (irradiance) components. The

[2]The Armadillo model was downloaded from the Stanford scanning repository.

[3]The buddha, cat and owl data sets were generously provided by Dan Goldman and Steve Seitz from the University of Washington, Seattle, WA.

[4]These models were scanned for us using the Polhemus scanner by Michael Reed of Blue Sky Studios

Figure 6.1: Objects used for testing our algorithm. Starting from the left, first come the synthetic renderings: a sphere and the Armadillo; followed by three objects with their geometry acquired using photometric stereo: the buddha, the cat and the owl; and finally two scanned objects: the chicken and the girl. Each row shows the objects with a different illumination.

Figure 6.2: Results obtained using the point light source model for the images in Figure 6.1. The top row shows the recovered albedo, the middle row shows the factored irradiance for the first illumination, and the last row the factored irradiance for the second illumination. Notice how the factorization de-couples texture from shading.

	Point source 1			Point source 2			Rel. intensity
	x	y	z	x	y	z	(R, G, B)
Actual position	-0.58	0.36	0.73	0.28	-0.28	0.92	(5.00, 10.00, 20.00)
Sphere	-0.58	0.36	0.73	0.27	-0.28	0.92	(5.03, 10.10, 20.48)
Armadillo	-0.58	0.35	0.73	0.27	-0.28	0.92	(5.11, 10.27, 20.88)

Table 6.1: Ground truth and recovered light directions and relative intensities for the synthetic images of the sphere and the Armadillo

	Point source 1			Point source 2			Rel. intensity
	x	y	z	x	y	z	(R, G, B)
Actual position	0.40	0.48	0.78	-0.32	0.49	0.92	(1.00, 1.00, 1.00)
Buddha	0.44	0.47	0.77	-0.32	0.47	0.82	(1.03, 1.03, 1.04)
Cat	0.39	0.49	0.78	-0.33	0.47	0.82	(1.09, 1.09, 1.05)
Owl	0.39	0.48	0.78	-0.31	0.44	0.84	(1.02. 1.01, 1.00)

Table 6.2: Ground truth and recovered light directions and intensities for the buddha, cat and owl models.

Figure 6.3: Results obtained using the generalized illumination model the images in Figure 6.1. The top row shows the recovered albedo, the middle row shows the factored irradiance for the first illumination, and the last row the factored irradiance for the second illumination.

Model	Method	Error 1	Error 2
Sphere	PL	$< 0.1\%$	$< 0.1\%$
	PCA 5	0.40%	0.20%
Armadillo	PL	$< 0.1\%$	$< 0.1\%$
	PCA 3	3.50%	4.30%
Buddha	PL	0.40%	0.50%
	PCA 3	0.10%	1.60%
Cat	PL	$< 0.1\%$	$< 0.1\%$
	PCA 3	4.40%	4.50%
Owl	PL	$< 0.1\%$	$< 0.1\%$
	PCA 3	3.80%	3.50%

Table 6.3: Normalized reconstruction error for the irradiance images. The first column indicates the object, the second one the method used (PL = point light, PCA n = generalized illumination using PCA of size n), and the last two columns show the normalized reconstruction error for the two irradiance images.

results are shown in Figure 6.2 - the top row shows the albedo map, and the second and third rows the irradiance maps for each of the input images. Note that, with the exception of a few minor artifacts, the albedo maps do not contain any shading effects. This is certainly true for the synthetic models: both the sphere and the armadillo albedo maps look completely flat. For the real data sets, some artifacts can be seen where the surfaces are not purely Lambertian. For example, the owl shows some specular components in the albedo map. Other artifacts are brighter spots in non-convex regions, in particular at the junction of the head and body of the cat and owl models, the junction of the arm and body in the chicken model, and the junction of the hair and face in the girl model. The convexity assumption fails here and inter-reflections influence the final result. The visible effect is a brightening of the albedo map, since the pure Lambertian model can not explain the increase in irradiance due to inter-reflections. As a final comment, the pants in the chicken model are not in the albedo map since the luminance value of those pixels falls below the shadow threshold, and hence ignored by the algorithm.

When ground truth was available, we also computed a quantitative measure of the quality of the factorization by comparing the obtained irradiance images with the irradiance images generated using the ground truth light source position. Since there is a scale ambiguity that is inherent to our method, we normalized all images before computing the error metric. This normalization was achieved by setting the norm $\| I \|^2 = \sum_{x,y} I(x,y)^2$ equal to one. Then, for a given pair of normalized ground truth image I^0 and reconstructed irradiance image I^1, we computed the relative squared error of the reconstruction[5]:

$$err(I^1, I^0) = \frac{\| I^1 - I^0 \|^2}{\| I^0 \|^2}. \tag{6.26}$$

The resulting reconstruction errors are reported as percentages in Table 6.3. It can be observed that the reconstructed irradiance images using the point light source algorithm are very accurate.

[5]The relative squared error is frequently used in the literature (e.g. [Basri and Jacobs, 2003, Frolova *et al.*, 2004]) as a metric for evaluating the goodness of image reconstructions.

6.5.2 Generalized light model

We tested the generalized light model algorithm on the same set of images we used for testing the point-light source factorization, shown in Figure 6.1. In all cases, we first analytically computed the PCA basis. We used a basis of dimension 3 for all of the models except for the sphere, for which we used a PCA basis of dimension 5. We found these dimensions empirically, by testing with different basis size. The resulting factorizations into irradiance images and albedo maps are shown in Figure 6.3 and the reconstruction errors for the irradiance images are tabulated in Table 6.3. It can be seen that the model approximates well the irradiance and produces a good factorization. The reported reconstruction errors are less than 4.5%. For the sphere, the quality of the approximation is as good as for the point-light source model. For the remaining models, the irradiance reconstruction error varies between 0.5% for the buddha to 4.50% for the cat.

6.5.3 Point plus ambient light model

To test the factorization using the point plus ambient light model we ran our algorithm on a model of the church of St. Marie, in Chappes, France, one of the many churches that we have modeled in collaboration with the **Visual Media Center** of the **Department of Art History and Archeology** at Columbia University, and on a subset of the images and models presented in chapter 5 for the relighting and de-shadowing experiments. All models were built using a Leica HDS-3000 range finder and meshed using the VripPack package of [Curless and Levoy, 1996].

Running the point plus ambient light model on outdoor scenes requires the position of the sun to be known relative to the geometry, so we first aligned the geometry with respect to the Earth's coordinate system. For this, we set up a coordinate system in which the negative Z axis points to North, the positive X axis to East and the Y axis up. The Leica HDS-3000 scanner produces scans aligned with the vertical direction, so we only had to compute a rotation around the Y axis that would align our models correctly with respect to North. We achieved this final alignment manually: we took a picture at a known

	Image 1		**Image 2**	
	Sun	Ambient	Sun	Ambient
Ground truth	1.00	0.133	0.472	0.0650
Computed	1.00	0.131	0.474	0.0652

Table 6.4: Relative illumination estimation results from two synthetic renderings under monochromatic illumination of a model of the church of St. Marie, Chappes.

time of the day, registered the image with the model using our manual registration tool, rendered the model with an orthographic projection as seen from the direction of the sun, and manually rotated the model around the Y axis until none of the shadows in the image were visible. This idea is similar to the shadow-based registration presented in chapter 3, with the exception that now we know the image registration and we want to compute the correct orientation of the model.

We first ran a test on synthetic images of the church of **St. Marie**, to test our algorithm under ideal conditions. We rendered two images illuminated by a monochrome directional light source and a constant monochrome ambient term, with no inter-reflections, shown in figures 6.5. Figure 6.4 shows a normals map of the south façade of St. Marie. The acquired geometry has regions with holes, which are seen as black patches. The ground truth and computed sun and ambient intensities are shown in table 6.4. The errors are very small, which verifies that out algorithm can correctly estimate the illumination parameters from the ratio image.

We did further testing with real images of the church of St. Marie. First, we manually registered two set of images, one set taken at 10:56am, and the other taken at 4:55pm. Each set consists of four images. We put the images together into a composite picture for each illumination, which are shown in figure 6.6. Since each set of images is acquired under constant illumination with the same camera parameters, these blended images do not show artifacts. The two composites, plus the shadows masks shown in figure 6.7 which we obtained using thresholding, the normals map, and the directions of the sun, are the inputs to our illumination and texture factorization algorithm. The resulting albedo and illumination maps are shown in figure 6.8 and 6.9, respectively. The resulting albedo map

is illumination free; the only artifacts are some regions in which the shadow boundaries are noticeable. This is due to poor shadow masking. Unfortunately, this is a limitation of our method: it requires very accurate shadow masks which are very difficult to compute. Nevertheless, the computed albedo map is good for generating new renderings under novel illumination conditions. To show this, we created a set of renderings of St Marie as the sun traverses the sky from morning to afternoon. The results, which were created using a ray tracer, are shown in figure 6.10.

Our second test was on the images of **Casa Italiana** shown in Figure 6.11 with the same shadow masks used in chapter 5. We ran the point plus ambient model using different approximations for the ambient term. Figures 6.12 and 6.13 show the resulting albedo and irradiance maps obtained using a constant term approximation for the ambient term. Figures 6.14 and 6.15 show the results of the factorization a spherical harmonic approximation of order 1 for the ambient component, and Figures 6.16 and 6.17 show the results obtained from using a 3-dimensional PCA basis. In the absence of ground truth, we restrict ourselves to a qualitative analysis of the results. Comparing the albedo maps we can see the best results were obtained from the constant term ambient approximation. The albedo map in Figure 6.12 shows almost none of the illumination effects, except for a few brightening in regions where the sampling of the scanner was not high enough to approximate correctly the geometry variations. In contrast, the results obtained using higher order ambient terms (SH order 1 and PCA basis of dimension 3) do show some artifacts for surfaces looking down. Regions with downward pointing normals only receive indirect illumination from inter-reflections, so it is very unlikely that orientation consistency will hold for those points. These higher dimensional models try to model this effects.

6.6 Conclusions and Future Work

In this chapter we have addressed the problem of illumination and texture factorization from two images of a Lambertian object of known geometry acquired under unknown illu-

mination. We have developed our solution as a two step process: first we compute the ratio image and solve for the irradiance maps of the scene under the unknown illuminations. In a second step, we divide the original images by the found irradiance maps to solve for the albedo. We have presented three different illumination models that our method can handle: point-light, generalized illumination and point plus ambient, and we have validated these with results from real and synthetic scenes. The results show that when our assumptions are met, the algorithms perform as expected. Possible sources of error are the same mentioned in the conclusions of chapter 5: inaccurate geometry due to poor sampling, errors in geometry and image registration, non-Lambertian surfaces, poor shadow mask detection and strong effects of inter-reflections. One issue that we did not address is the possibility that in solving for the irradiance maps in terms of SH or PCA expansions, the illumination map might contain negative values for some direction. Negative light, though mathematically possible, is physically impossible. [Basri and Jacobs, 2003] have already addressed this issue. In the appendix A we discuss a potential adaptation of this technique to our problem.

To conclude, the main advantage of the presented algorithms, and our motivation for their development, is the lack of over-head they introduce in the data acquisition process: just two images are needed and there is no need for a device to measure the incident light. As such, as far as we know, this is the first method introduced to solve for irradiance maps from the ratio image. But as we have seen, such flexibility can only be met by imposing the scene to meet certain conditions. The results we have presented are for simple objects under relatively simple illumination. For more complex scenes and illumination settings, our assumptions might not hold, and our method may not produce the expected results. Inverse rendering, in particular for outdoor scenes, is indeed a difficult problem, and even more elaborate techniques such as that of Debevec *et al.* (2004) that makes use of a sophisticated light-probing device, are limited to diffuse surfaces.

Figure 6.4: Normals map for the church of Saint Marie, Chappes, France.

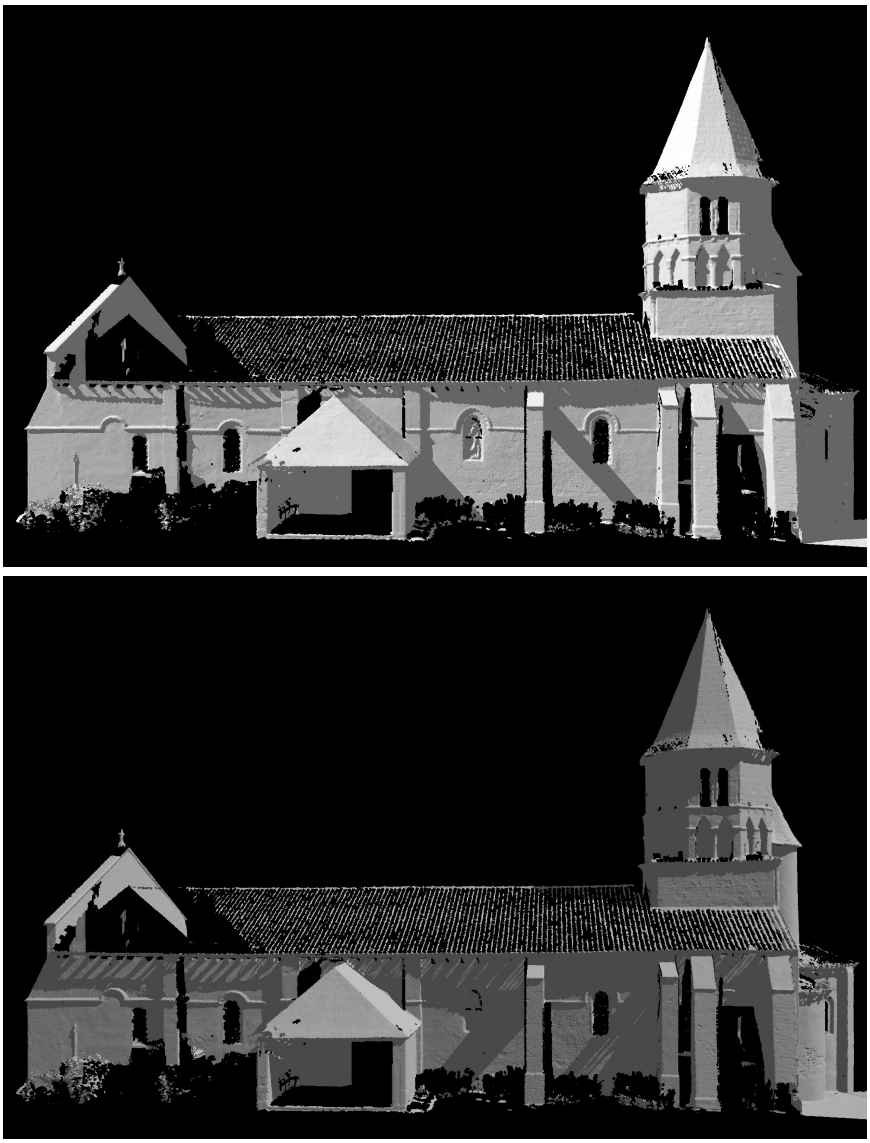

Figure 6.5: Synthetic renderings of the church of Saint Marie, Chappes, France under two different illuminations (point + ambient light models).

Figure 6.6: Real images of Saint Marie, Chappes, France. The top image is a composite made of images acquired at 10:56AM on May 26th 2005. The bottom is a composite made of images taken at 4:55PM on that same day.

Figure 6.7: Shadow masks for the input images of Saint Marie, Chappes, France.

Figure 6.8: Albedo map for the south facade of Saint Marie, Chappes, France.

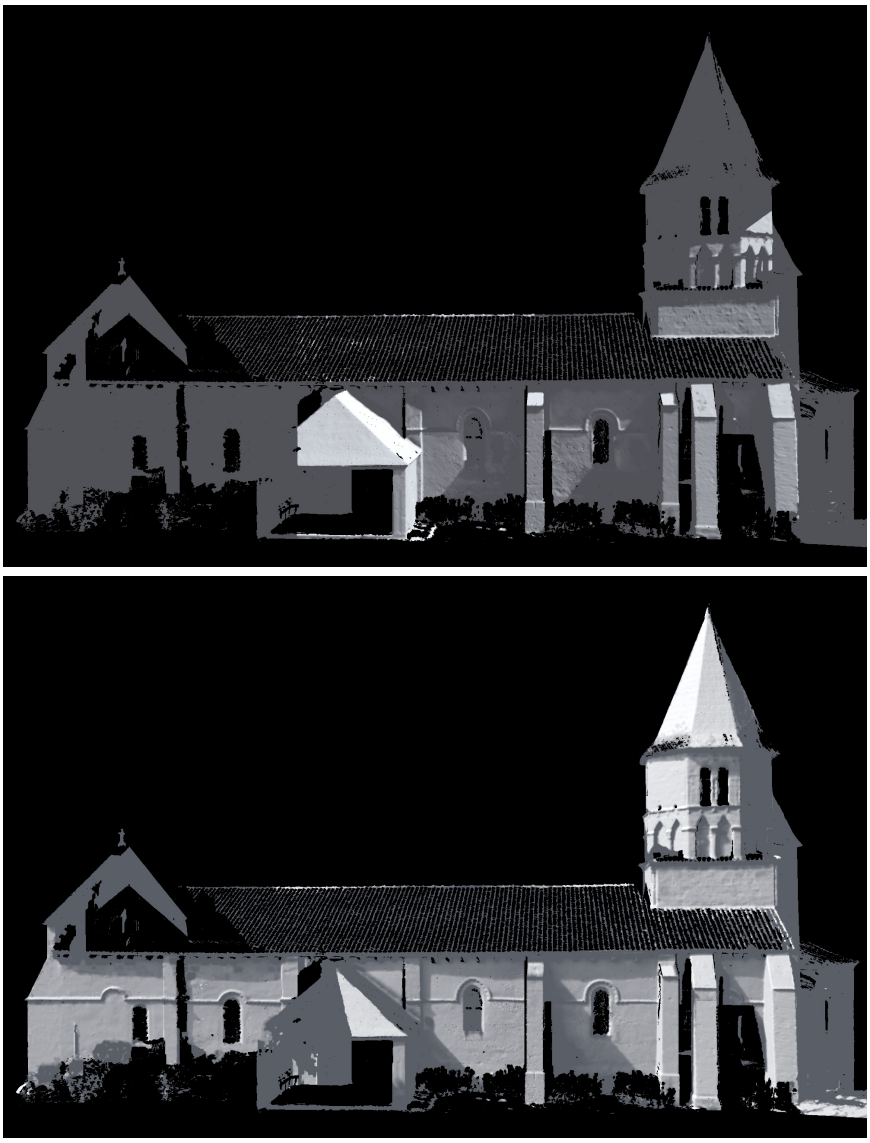

Figure 6.9: Illumination images computed for the images of Saint Marie, Chappes, France.

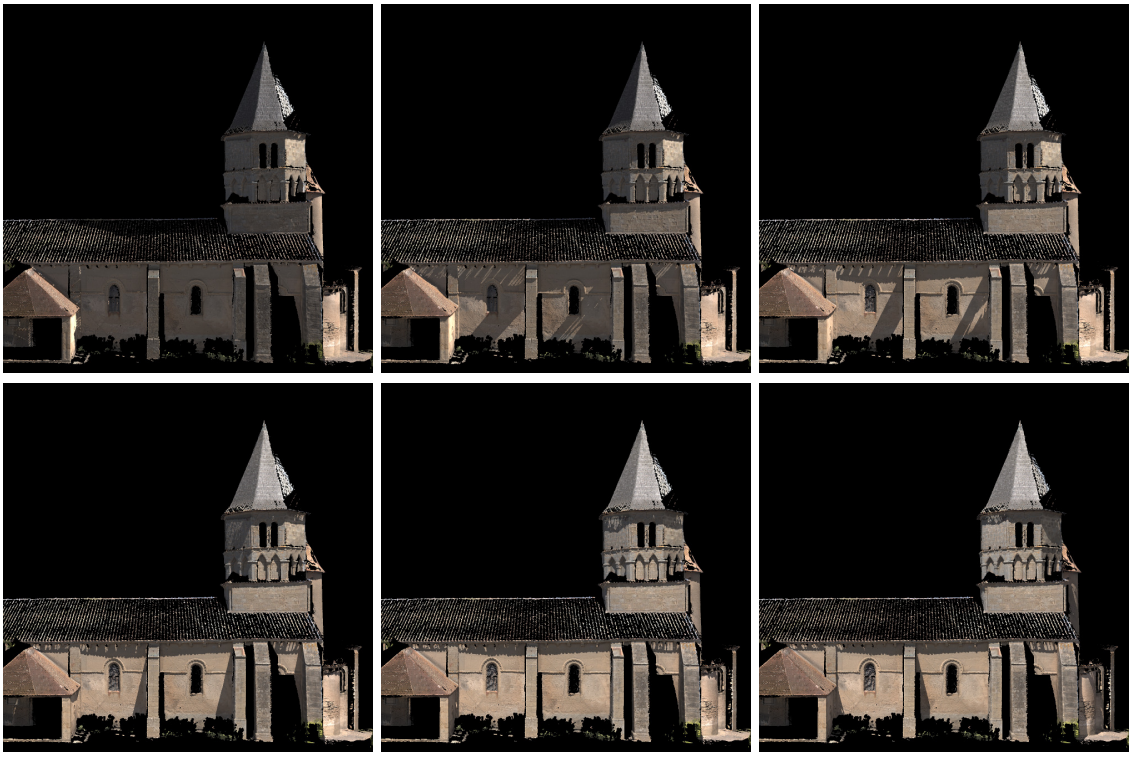

Figure 6.10: Renderings of Saint Marie at Chappes under novel illumination conditions. These renderings simulate a day-time sequence with the sun at 10am, 11am, 12pm, 1pm, 2pm and 3pm.

Figure 6.11: Two images of Casa Italiana taken at different time of the day from slightly different view points. The left image was taken 3:22pm and the right one at 1:28pm on the same day, under partly cloudy conditions.

Figure 6.12: Albedo map computed for Casa Italiana using the point plus ambient model with a constant term for the ambient component.

Figure 6.13: Irradiance maps computed for Casa Italiana using the point plus ambient model with order 0 SH for the ambient component.

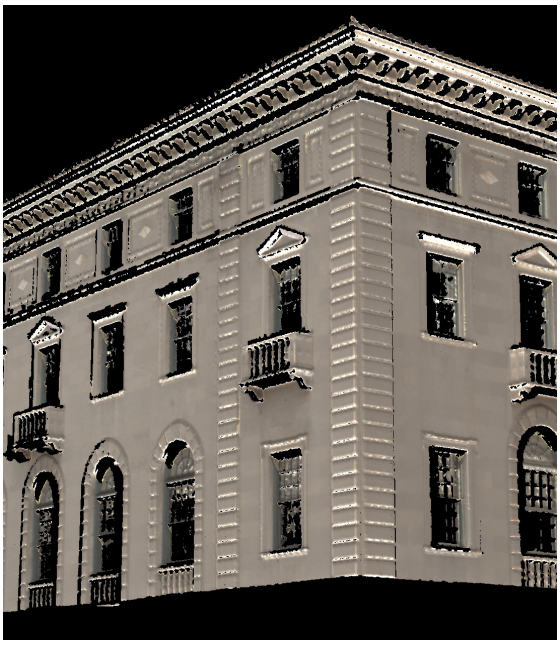

Figure 6.14: Albedo map computed for Casa Italiana using the point plus ambient model using an order 1 spherical harmonic basis for the ambient component.

Figure 6.15: Irradiance maps computed for Casa Italiana using the point plus ambient model with order 1 SH for the ambient component.

Figure 6.16: Albedo map computed for Casa Italiana using the point plus ambient model with order 3 PCA basis for the ambient component.

Figure 6.17: Irradiance maps computed for Casa Italiana using the point plus ambient model with order 3 PCA basis for the ambient component.

Chapter 7

Conclusions and Future Work

In this dissertation we have presented a set of algorithms that we have implemented into tools for photorealistic 3D modeling using dense data from a range sensor and photographs, making emphasis on the acquisition and modeling of large scale outdoor settings. This research area has become of significant importance recently due to the commercialization of fast and accurate range finders, with applications not only in reverse engineering, but also in cultural heritage conservation and digitalization. We focused on two major areas of the modeling and acquisition pipeline: the registration of range and intensity data, and the generation of seamless integrated texture maps. Below we discuss our main conclusions and directions of future work for each of these areas.

7.1 Range and intensity image registration

As we have mentioned in chapter 3, the registration of intensity images with range data is a problem that involves two very different domains: the domain of intensity values recorded by a photographic camera and the domain of 3D points recorded by a range finder. One way of bringing these two different types of data into registration is to find a set of corresponding features, and following this direction we presented a tool for manual registration based on a point-and-click interface and a semi-automatic tool for registration of images of architectural

scenes that uses extracted line 2D and 3D line features. The advantage of using geometric features, such as points, lines, or parametric curves, is that the projection of these objects into an image follows mathematical models from which one can derive an error function that can be efficiently computed from a set of correspondences. Hence, the main task any feature-based registration method has to solve is the search for corresponding features. The size of the search space can be large enough to make an exhaustive search of all possible correspondences prohibitive. In addition, the fact that the input data comes from two different domains makes it very difficult to compute a similarity metric between features.

Another approach to image registration is to produce a rendering of the 3D model and compute an intensity-based metric to find the mutual dependence (or lack of it) between the rendering and the actual image. Our shadow-based registration method falls in this category, though we measure the mutual dependence in a slightly different way, by taking a rendering as seen from the direction of the sun and counting the number of visible shadow pixels. Since a rendering of the model is required, this kind of techniques would require the camera intrinsic parameters to be known, leaving the six parameters of the camera pose to be estimated. Hence, the principal task of any registration algorithm of this kind is to search the six dimensional space of camera pose parameters for a point that globally maximizes the mutual dependence. In our shadow-based method we require an initial estimate of the camera pose, therefore we can restrict the search space to a neighborhood around this initial point making the search tractable.

It can be noted then, that whether using a feature-based method or a mutual information like approach, there is an underlying search problem that becomes the main bottleneck of any registration algorithm. This search space can be efficiently explored either by adding a human into the registration loop, or by imposing additional constraints given by a specific application domain. In our line-based registration, for example, we combine domain constraints with user-interaction to achieve real-time registration. More recently, [Liu and Stamos, 2005] developed an automatic algorithm for registration of urban scene data, that groups line features in the scene in higher-order primitives like parallelepipeds and rectan-

gles. These higher-order primitives, which are domain dependent, help reduce the size of the search space. Another example of a registration task specific to a domain is the work of [Lensch *et al.*, 2001]. Lensch *et al.* (2001) use silhouettes for registration. Silhouette based methods work well when two conditions are satisfied: first, it is required that the object of interest can be captured within a single image; second, it should be easy to separate the object from the background.

A one-size fits all automatic solution to the range and intensity image registration problem is still to be found. The point-and-click user driven tool presented in chapter 3 can fit most applications, but is far from being automatic. However, it is very well optimized and can find a solution with as little as four point correspondences when the camera intrinsic parameters are known. On the other hand, both the line-based and shadow-based tools do take a step further into full-automation, but are restricted to very specific domains. The ideal solution to the registration problem should be as simple as today's panorama stitchers (see for example the panoramic tools of the 2006 Digital Image Editor by Microsoft), which given a collection of images can build a full panorama in seconds. This will only be possible when the gap between the two domains is bridged and a generic feature descriptor that can enable the comparison of 3D and 2D features is developed. An alternative possible path of bridging this gap is to approach the problem as the simultaneous registration of multiple images with range data, applying existing structure-from-motion techniques. Once a single color image is registered with the range data, then the remaining color images can be registered with respect to the first one, using existing color image registration methods. Hence, while the first step requires registration across two domains, the second step works entirely in the domain of color images. Some progress is already being made in this direction, as shown in the recent work of [Liu *et al.*, 2006]. However, these techniques might produce limited results with images acquired under different illuminations, since most image registration techniques are based on the color constancy assumption.

Yet another way of bridging the gap between range and color data is to use the return intensity that the most time of flight laser scanner return together with a 3D measurement.

This return intensity depends on several factors, such as scanner distance and incident angle, and also on the reflectance of the surface being modeled. Finding features in the reflectance image is a problem that falls in the domain of 2D image processing, and it is likely that a metric computed on these images could be comparable with a metric computed over color images, making the matching problem easier. [Ikeuchi *et al.*, 2003] has already used the scanner's reflectance image to find edges. Still some work in this area could produce better algorithms. For instance, one could compute and match local scale invariant (SIFT) features [Lowe, 1999] on the reflectance and color images to find the registration.

To conclude, there are different ways of finding the registration between color and range images. Some are domain specific, and some are more general. In this dissertation we took the domain specific approach, and we presented an algorithm for solving the registration problem using line features, and another algorithm for finding the camera position using the shadows cast by the sun.

7.2 Generation of seamless integrated texture maps

In chapters 5 and 6 we introduced two different algorithms for generating integrated seamless texture maps using images of outdoor scenes. The biggest challenge when imaging large scale structures in outdoor settings is the variability of the illumination, which can not be controlled. This is also true for large-scale scenes in general, because outside laboratory environments it is difficult to have control on the illumination of the scene. The relighting technique we presented in chapter 5 generates an integrated texture maps by computing a relighting operator over the are of overlap of two images and bringing the images into the same illumination. One of the main contributions of our method is the handling and removal of shadows cast by the sun. In chapter 6 we took a slightly different approach and showed that it is possible to factor the irradiance from the texture using the ratio of two images. By doing so, we were able to create an integrated texture map with an illumination free representation of the spatial varying diffuse reflectance. The advantage

of this technique is that we can then use this reflectance map to generate renderings of the scene under novel illumination conditions. Our main contribution in this area is the factorization of illumination and diffuse reflectance from two images and the geometry of the scene. Previous methods that achieve similar results for outdoor scenes require light probes [Debevec *et al.*, 2004] or images of the sun and sky [Yu and Malik, 1998], in addition to the geometry of the scene.

Both, the techniques of chapter 5 and 6, require certain conditions to be met to guarantee the model assumptions are satisfied. These conditions do indeed restrict the applicability of our method to more general cases. For example, we have assumed that shadows are easily identified. This is not always the case, and thresholding methods such as the one we have employed do not always produce optimal results. Our method could be improved with better shadow detection. In fact, one could combine geometry and color information to find a segmentation into shadow and lit regions. Using ray-casting, we can use the geometry information to find a coarse location of the shadow regions. This shadow map may contain some errors because of holes in the model and coarse geometry sampling. Nevertheless, the resulting ray-casted shadows could be good enough to compute local intensity statistics of the characteristics of shadow regions, which can be combined with region growing techniques to find the actual shadows in the image, or used with the probabilistic techniques, such as that of [Wu and Tang, 2005].

In addition, we have assumed convex scenes. If the scene has concave regions, local illumination effects such as interreflections and spatially varying ambient light occlusion could adversely distort the final results. It is possible, however, to improve our algorithm by modeling spatially varying ambient light occlusion. In our current implementation we are indirectly doing this when we compute the shadow masks using two thresholds: an upper and lower threshold. But, to better model ambient occlusion, we could use the geometry to compute a visibility function for every scene point. This visibility function, which indicates which regions of the hemisphere above a point are occluded, can be computed by a ray-tracer (assuming a complete geometric model). Once the visibility function is computed,

we can select to include in the computation of the irradiance ratio maps of chapter 5 the subset of pixels whose visible hemisphere is above some threshold, guaranteeing that only the convex regions are used. The same enhancement can be applied to our factorization algorithm of chapter 6. We can solve for the illumination parameters using only those pixels that are in locally convex regions. After the illumination model parameters are solved for, the incident irradiance at each pixel can be computed taking into account the visibility function, very much like existing pre-computed radiance transfer techniques do (e.g. [Sloan *et al.*, 2002]). Also, once we have solved for the irradiance parameters, we can compute the theoretical irradiance ratio taking visibility into account and compare it with the actual measured irradiance ratio. Any difference between these two will be due to the effects of interreflections which we have so far ignored. At this point we could iteratively update our initial estimate of the albedo maps taking interreflections into account to minimize the difference between the computed and measured irradiance ratios.

Another constraint we imposed is diffuse Lambertian like behavior. This guarantees that the ratio image is texture-free and allows us to derive a simple model for relighting and for factoring irradiance and texture. It seems unlikely that this restriction can be relaxed for the general case, but under certain circumstances, we might be able to. For example, if we had a small database of materials representative of the materials in the scene, and assumed point light source illumination, we could search the space of materials and source positions until we find the parameters that best fit the observed ratio image. This will require proper segmentation and materials clustering, as in [Hertzmann and Seitz, 2005].

Yet to be explored is the use of active methods for reflectance modeling in outdoor environments. The range finder is in fact, an active device which does report a returned intensity value. In recent work [Xu *et al.*, 2006], pure diffuse reflectance of large environments is obtained by analyzing the scanner's returned intensity value in combination with color-balancing techniques on the input color images. This brings an interesting idea to speculate about: could it be possible using lasers with different wavelengths to obtain information reflectance information?.

To conclude, the creation of seamless textured maps of outdoor scenes is a difficult problem which can be addressed in different ways. In this dissertation, we have presented two different algorithms for producing seamless integrated texture maps of large diffuse outdoor scenes using the ratio of images of diffuse scenes. We have successfully applied our methods to mostly convex buildings. Still, as we have outlined in this section, there is considerable work to be done to generalize this idea to more complex cases.

7.3 Summary

In this dissertation we have presented a set of algorithms and techniques that contribute towards the automation and simplification of the 3D acquisition and modeling pipeline. We focused on two major open problems: the registration of range and intensity images and the creation of integrated texture maps from images acquired under uncontrolled illumination. We presented both, the theory behind our techniques and their application to real scenes.

Bibliography

[Agathos and Fisher, 2003] Alexander Agathos and Robert Fisher. Colour texture fusion of multiple range images. In *Proceedings of the 4th International Conference on 3D Digital Imaging and Modeling*, pages 139–146, October 2003.

[Allen *et al.*, 2001] Peter K. Allen, Ioannis Stamos, Atanas Gueorguiev, Ethan Gold, and Paul Blaer. Avenue: Automated site modeling in urban environments. In *Proceedings of the 3rd International Conference on 3D Digital Imaging and Modeling*, pages 357–364, 2001.

[Allen *et al.*, 2003] Peter K. Allen, Alejandro Troccoli, Benjamin Smith, Stephen Murray, Ioannis Stamos, and Marius Leordeanu. New methods for digital modeling of historic sites. *IEEE Comput. Graph. Appl.*, 23(6):32–41, 2003.

[Allen *et al.*, 2004] Peter Allen, Steve Feiner, Alejandro Troccoli, Hrvoje Benko, Edward Ishak, and Benjamin Smith. Seeing into the past: Creating a 3D modeling pipeline for archaeological visualization. In *Proceedings of 2nd International Symposium on 3D Data Processing, Visualization and Transmission*, September 2004.

[Ansar and Daniilidis, 2003] Adnan Ansar and Kostas Daniilidis. Linear pose estimation from points or lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):578–589, 2003.

[Bannai *et al.*, 2004] Nobuyuki Bannai, Alexander Agathos, and Robert B. Fisher. Fusing multiple color images for texturing models. In *Proceedings of 2nd International Sym-*

posium on 3D Data Processing, Visualization and Transmission, pages 558–565. IEEE Computer Society, 2004.

[Basri and Jacobs, 2003] Ronen Basri and David W. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(2):218–233, 2003.

[Beauchesne and Roy, 2003] Etienne Beauchesne and Sebastien Roy. Automatic relighting of overlapping textures of a 3D model. In *Proceedings of Computer Vision and Pattern Recognition*, 2003.

[Bennet, 2006] Eric P. Bennet. Personal communication. 2006.

[Beraldin *et al.*, 2002] J.-A. Beraldin, M. Picard, S.F. El-Hakim, Guy Godin, G. Valzano, A. Bandiera, and D. Latouche. Virtualizing a Byzantine crypt by combining high-resolution textures with laser scanner 3D data. In *Proceedings of the VMMS 2002*, pages 3–14, September 2002.

[Bernardini *et al.*, 2001] Fausto Bernardini, Ioana M. Martin, and Holly Rushmeier. High-quality texture reconstruction from multiple scans. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):318–332, 2001.

[Bernardini *et al.*, 2002] Fausto Bernardini, Holly Rushmeier, Ioana M. Martin, Joshua Mittleman, and Gabriel Taubin. Building a digital model of Michelangelo's Florentine Pietà. *IEEE Computer Graphics and Applications*, 22(1):59–67, /2002.

[Besl and McKay, 1992] Paul J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Trans. Pattern Anal. Mach. Intell.*, 14(2):239–256, 1992.

[Bouguet, 2001] Jean-Yves Bouguet. Camera calibration toolbox for Matlab. http://www.vision.caltech.edu/bouguet/calib_doc, 2001.

[Buehler *et al.*, 2001] Chris Buehler, Michael Bosse, Leonard McMillan, Steven Gortler, and Michael Cohen. Unstructured lumigraph rendering. In *Proceedings of the 28th annual*

conference on Computer graphics and interactive techniques, pages 425–432. ACM Press, 2001.

[Chuang *et al.*, 2003] Yung-Yu Chuang, Dan B Goldman, Brian Curless, David H. Salesin, and Richard Szeliski. Shadow matting and compositing. *ACM Trans. Graph.*, 22(3):494–500, 2003.

[Cook and Torrance, 1982] R. L. Cook and K. E. Torrance. A reflectance model for computer graphics. *ACM Trans. Graph.*, 1(1):7–24, 1982.

[Curless and Levoy, 1996] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM Press, 1996.

[Daum and Dudek, 1998] M. Daum and G. Dudek. On 3-d surface reconstruction using shape from shadows. In *CVPR '98: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 461, Washington, DC, USA, 1998. IEEE Computer Society.

[Debevec *et al.*, 1996] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20. ACM Press, 1996.

[Debevec *et al.*, 2000] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156. ACM Press/Addison-Wesley Publishing Co., 2000.

[Debevec *et al.*, 2004] Paul Debevec, Chris Tchou, Andrew Gardner, Tim Hawkins, Jessi Stumpfel, A. Jones, Per Einarsson, T. Lundgren, P. Martinez, and Marcos Fajardo. Estimating surface reflectance of a complex scene under natural captured illumination.

Technical report, University of Souther Californian, Institute for Creative Technologies, June 2004.

[Fischler and Bolles, 1987] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. pages 726–740, 1987.

[Frolova *et al.*, 2004] Darya Frolova, Denis Simakov, and Ronen Basri. Accuracy of spherical harmonic approximations for images of Lambertian objects under far and near lighting. In *ECCV (1)*, pages 574–587, 2004.

[Früh, 2002] Christian Früh. *Automated 3D model generation for urban environments*. PhD thesis, Universität Karlsruhe, Fak. f. Elektrotechnik und Informationstechnik, 2002.

[Funka-Lea and Bajcsy, 1995] G. Funka-Lea and R. Bajcsy. Combining color and geometry for the active, visual recognition of shadows. In *ICCV '95: Proceedings of the Fifth International Conference on Computer Vision*, page 203, Washington, DC, USA, 1995. IEEE Computer Society.

[Hantak and Lastra, 2006] Chad Hantak and Anselmo Lastra. Metrics and optimization techniques for registration of color to laser range scans. In *Proc. 3rd Int'l Symp. 3D Data Processing, Visualization, and Transmission (3DPVT 06)*. IEEE Computer Society, 2006.

[Hantak *et al.*, 2004] C. Hantak, Kok-Lim Low, A. Lastra, M. Pollefeys, and N. Williams. Automatic image alignment for 3d environment modeling. In *Proceedings of the 17th Brazilian Symposium on Computer Graphics and Image Processing, 2004.* IEEE, 2004.

[Hartley and Zisserman, 2000] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, 2000.

[Heikkila and Silven, 1997] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *CVPR '97: Proceedings of the 1997 Confer-*

ence on Computer Vision and Pattern Recognition, page 1106, Washington, DC, USA, 1997. IEEE Computer Society.

[Hertzmann and Seitz, 2003] Aaron Hertzmann and Steven M. Seitz. Shape and materials by example: A photometric stereo approach. In *CVPR '03: Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'03)*, pages 533–540, 2003.

[Hertzmann and Seitz, 2005] Aaron Hertzmann and Steven M. Seitz. Example-based photometric stereo: Shape reconstruction with general, varying brdfs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1254–1264, 2005.

[Horn, 1986] Berthold Klaus Paul Horn. *Robot vision*. MIT Press, 1986.

[Horn, 1987] B. K. P. Horn. Closed form solutions of absolute orientation using unit quaternions. *Journal of the Optical Society of America*, 4(4):629–642, April 1987.

[Ikeuchi and Sato, 1991] Katsushi Ikeuchi and Kosuke Sato. Determining reflectance properties of an object using range and brightness images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13(11):1139–1153, 1991.

[Ikeuchi *et al.*, 2003] Katsushi Ikeuchi, Atsushi Nakazawa, Ko Nishino, and Takeshi Oishi. Creating virtual buddha statues through observation. In *IEEE Workshop on Applications of Computer Vision in Architecture*, volume 1, 2003.

[Ingber, 1989] Lester Ingber. Very fast simulated re-annealing. *Mathl. Comput. Modelling*, 12(8):967–973, 1989.

[Irvin and David M. McKeown, 1989] R. Bruce Irvin and Jr. David M. McKeown. Methods for exploiting the relationship between buildings and their shadows in aerial imagery. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(6):1564–1575, December 1989.

[Jensen *et al.*, 2001] Henrik Wann Jensen, Stephen R. Marschner, Marc Levoy, and Pat Hanrahan. A practical model for subsurface light transport. In *SIGGRAPH '01: Pro-*

ceedings of the 28th annual conference on Computer graphics and interactive techniques, pages 511–518, New York, NY, USA, 2001. ACM Press.

[Kriegman and Belhumeur, 1998] David J. Kriegman and Peter N. Belhumeur. What shadows reveal about object structure. In *ECCV '98: Proceedings of the 5th European Conference on Computer Vision-Volume II*, pages 399–414, London, UK, 1998. Springer-Verlag.

[Kumar and Hanson, 1994] Rakesh Kumar and Allen R. Hanson. Robust methods for estimating pose and a sensitivity analysis. *CVGIP: Image Underst.*, 60(3):313–342, 1994.

[Lafortune *et al.*, 1997] Eric P. F. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 117–126. ACM Press/Addison-Wesley Publishing Co., 1997.

[Lensch *et al.*, 2001] Hendrik P.A. Lensch, Wolfgang Heidrich, and Hans-Peter Seidel. A silhouette-based algorithm for texture registration and stitching. *Graphical Models*, 63(4):245–262, 2001.

[Lensch *et al.*, 2003] Hendrik P. A. Lensch, Jan Kautz, Michael Goesele, Wolfgang Heidrich, and Hans-Peter Seidel. Image-based reconstruction of spatial appearance and geometric detail. *ACM Trans. Graph.*, 22(2):234–257, 2003.

[Levoy *et al.*, 2000] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital Michelangelo project: 3D scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144, 2000.

[Liu and Stamos, 2005] Lingyun Liu and Ioannis Stamos. Automatic 3D to 2D registration for the photorealistic rendering of urban scenes. In *CVPR '05: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 137–143, Washington, DC, USA, 2005. IEEE Computer Society.

[Liu *et al.*, 2006] Lingyun Liu, Gene Yu, George Wolberg, and Siavash Zokai. Multiview geometry for texture mapping 2D images onto 3D range data. In *CVPR '06: Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2293–2300, Los Alamitos, CA, USA, 2006. IEEE Computer Society.

[Love, 1997] R.C. Love. *Surface Reflection Model Estimation from Naturally Illuminated Image Sequences*. PhD thesis, Leeds, 1997.

[Lowe, 1999] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, pages 1150–1157, 1999.

[Maes *et al.*, 1997] F. Maes, A. Collignon, D. Vandermeulen, P. Suetens, and G. Marchal. Multimodality image registration by maximization of mutual information. *IEEE Transactions on Medical Imaging*, 16(2):187–198, April 1997.

[Marschner and Greenberg, 1997] S. R. Marschner and D. P. Greenberg. Inverse lighting for photography. In *Proc. 5th Color Imaging Conference*, 1997.

[Marschner, 1998] Stephen R. Marschner. *Inverse rendering for computer graphics*. PhD thesis, Cornell University, 1998.

[Matusik *et al.*, 2003] Wojciech Matusik, Hanspeter Pfister, Matt Brand, and Leonard McMillan. A data-driven reflectance model. *ACM Trans. Graph.*, 22(3):759–769, 2003.

[Narasimhan and Nayar, 2001] Srinivasa G. Narasimhan and Shree K. Nayar. Removing weather effects from monochrome images. 02:186, 2001.

[Oh *et al.*, 2001] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 433–442, New York, NY, USA, 2001. ACM Press.

[Oren and Nayar, 1994] Michael Oren and Shree K. Nayar. Generalization of Lambert's reflectance model. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 239–246. ACM Press, 1994.

[Pulli *et al.*, 1997] Kari Pulli, Michael Cohen, Tom Duchamp, Hugues Hoppe, Linda Shapiro, and Werner Stuetzle. View-based rendering: Visualizing real objects from scanned range and color data. In *Rendering Techniques '97*, pages 23–34, New York, NY, 1997. Springer Wien.

[Ramamoorthi and Hanrahan, 2001a] Ravi Ramamoorthi and Pat Hanrahan. On the relationship between radiance and irradiance: Determining the illumination from images of a convex Lambertian object. *Journal of the Optical Society of America A*, 18(10):2448–2459, October 2001.

[Ramamoorthi and Hanrahan, 2001b] Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 117–128, New York, NY, USA, 2001. ACM Press.

[Ramamoorthi, 2002] Ravi Ramamoorthi. Analytic PCA construction for theoretical analysis of lighting variability in images of a Lambertian object. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(10):1322–1333, 2002.

[Reda and Andreas, 2003] Ibrahim Reda and Afshin Andreas. Solar position algorithm for solar radiation applications. Technical report, National Renewable Enery Laboratory, Golden, Colorado, June 2003.

[Rocchini *et al.*, 1999] Claudio Rocchini, Paolo Cignomi, Claudio Montani, and Roberto Scopigno. Multiple textures stitching and blending on 3D objects. In *Rendering Techniques '99*, Eurographics, pages 119–130. Springer-Verlag Wien New York, 1999.

[Rocchini *et al.*, 2002] Claudio Rocchini, Paolo Cignoni, Claudio Montani, and Roberto Scopigno. Acquiring, stitching and blending diffuse appearance attributes on 3D models. *The Visual Computer*, 18(3):186–204, May 2002.

[Salvador *et al.*, 2004] Elena Salvador, Andrea Cavallaro, and Touradj Ebrahimi. Cast shadow segmentation using invariant color features. *Comput. Vis. Image Underst.*, 95(2):238–259, 2004.

[Sato *et al.*, 1997] Yoichi Sato, Mark D. Wheeler, and Katsushi Ikeuchi. Object shape and reflectance modeling from observation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 379–387. ACM Press/Addison-Wesley Publishing Co., 1997.

[Sato *et al.*, 2003] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Illumination from shadows. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(3):290–300, 2003.

[Segal *et al.*, 1992] Mark Segal, Carl Korobkin, Rolf van Widenfelt, Jim Foran, and Paul Haeberli. Fast shadows and lighting effects using texture mapping. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 249–252. ACM Press, 1992.

[Shashua and Riklin-Raviv, 2001] Amnon Shashua and Tammy Riklin-Raviv. The quotient image: Class-based re-rendering and recognition with varying illuminations. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(2):129–139, 2001.

[Sloan *et al.*, 2002] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *SIGGRAPH '02: Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, pages 527–536. ACM Press, 2002.

[Stamos and Allen, 2001] Ioannis Stamos and Peter K. Allen. Automatic registration of 2-D with 3-D imagery in urban environments. In *Proceedings of the 8th International*

Conference On Computer Vision (ICCV-01), pages 731–737, Los Alamitos, CA, July 9–12 2001. IEEE Computer Society.

[Stamos and Allen, 2002] Ioannis Stamos and Peter K. Allen. Geometry and texture recovery of scenes of large scale. *Comput. Vis. Image Underst.*, 88(2):94–118, 2002.

[Stamos, 2001] Ioannis Stamos. *Geometry and Texture Recovery of Scenes of Large Scale: Integration of Range and Intenisty Sensing*. PhD thesis, Department of Computer Science, Columbia University, 2001.

[Tomasi and Manduchi, 1998] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV '98: Proceedings of the Sixth International Conference on Computer Vision*, page 839, Washington, DC, USA, 1998. IEEE Computer Society.

[Torrance and Sparrow, 1967] K. E. Torrance and G. M. Sparrow. Theory for off-specular reflection from roughened surfaces. *Journal of Optical Society of America*, 57(9), 1967.

[Troccoli and Allen, 2004] Alejandro Troccoli and Peter K. Allen. A shadow based method for image to model registration. In *2nd IEEE Workshop on Image and Video Registration, (IVR 04)*, 2004.

[Troccoli and Allen, 2005] Alejandro Troccoli and Peter Allen. Relighting acquired models of outdoor scenes. In *Proceedings of 3DIM'05*, 2005.

[Troccoli and Allen, 2006] Alejandro Troccoli and Peter Allen. Illumination and texture factorization using ratio images of an object of known geometry. In *Proc. 3rd Int'l Symp. 3D Data Processing, Visualization, and Transmission (3DPVT 06)*. IEEE Computer Society, 2006.

[Tsai, 1987] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, 3:323–344, 1987.

[Wang *et al.*, 2001] Lifeng Wang, Sing Bing Kang, Richard Szeliski, and Heung-Yeung Shum. Optimal texture map reconstruction from multiple views. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 347–54, 2001.

[Wang *et al.*, 2004] Haitao Wang, Stan Z. Li, and Yangsheng Wang. Generalized quotient image. In *CVPR (2)*, pages 498–505, 2004.

[Ward, 1992] Gregory J. Ward. Measuring and modeling anisotropic reflection. In *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 265–272. ACM Press, 1992.

[Wu and Tang, 2005] Tai-Pang Wu and Chi-Keung Tang. A bayesian approach for shadow extraction from a single image. In *10th IEEE International Conference on Computer Vision (ICCV 2005), Beijing, China*, pages 480–487, 2005.

[Xu *et al.*, 2006] Chen Xu, Athinodoros Georghiades, Holly Rushmeier, and Julie Dorsey. A system for reconstructing integrated texture maps for large structures. In *Proceedings of 3rd International Symposium on 3D Data Processing, Visualization and Transmission*, 2006.

[Yu and Chang, 2005] Yizhou Yu and Johnny T. Chang. Shadow graphs and 3d texture reconstruction. *Int. J. Comput. Vision*, 62(1-2):35–60, 2005.

[Yu and Malik, 1998] Yizhou Yu and Jitendra Malik. Recovering photometric properties of architectural scenes from photographs. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 207–217. ACM Press, 1998.

[Yu *et al.*, 1999] Yizhou Yu, Paul Debevec, Jitendra Malik, and Tim Hawkins. Inverse global illumination: recovering reflectance models of real scenes from photographs. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 215–224. ACM Press/Addison-Wesley Publishing Co., 1999.

[Zhang, 2000] Zhengyou Zhang. A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334, 2000.

Appendix A

Enforcing nonnegative light

In chapter 6, we presented a technique to solve for a pair of irradiance maps from the ratio image. When we solve for the irradiance maps by finding a linear combination of basis images, be it a spherical harmonics basis or a PCA basis, we may end up obtaining a solution that is not physically possible. The linear combination of basis may contain negative values, representing negative light. This situation can arise with the generalized or the point plus ambient illumination models. We can enforce nonnegative light by using a variation of the technique introduced by Basri and Jacobs (2003), which approximate a nonnegative lighting function as a nonnegative combination of delta functions, each representing a directional source. We will first describe this technique for the SH basis, and later consider other bases. Let $\delta_{\theta_0\phi_0}$ be the impulse function defined over the domain of spherical coordinates, returning a value of 1 for (θ_0, ϕ_0) and 0 everywhere else. Consider now the projection of the delta function to the first few SH basis, which is given by

$$\delta_{\theta_0\phi_0} = \sum_{l=0}^{N} \sum_{m=-n}^{n} Y_{lm}(\theta_0, \phi_0) Y_{lm}, \tag{A.1}$$

where N is the order of the approximation. Also, the irradiance map corresponding to the convolution of the delta function with the half-cosine function is

$$E_{\theta_0\phi_0} = \sum_{l=0}^{N} \sum_{m=-n}^{n} A_l Y_{lm}(\theta_0, \phi_0) Y_{lm}. \tag{A.2}$$

A_l, previously introduced in section 6.3.2, is the frequency space transform of the half-cosine function. Now, a nonnegative lighting function $\ell(\theta_0, \phi_0)$ can be expressed as a nonnegative combination of delta functions. That is

$$\ell = \sum_{j=1}^{J} a_j \delta_{\theta_0\phi_0}, \tag{A.3}$$

for some J. We can now write ℓ as an approximation in spherical harmonic basis by replacing the delta functions with their corresponding approximations,

$$\ell = \sum_{j=1}^{J} a_j \sum_{l=0}^{N} \sum_{m=-n}^{n} Y_{lm}(\theta_0, \phi_0) Y_{lm}. \tag{A.4}$$

We can now write the irradiance map for ℓ as

$$E = \sum_{j=1}^{J} a_j \sum_{l=0}^{N} \sum_{m=-n}^{n} A_l Y_{lm}(\theta_0, \phi_0) Y_{lm}. \tag{A.5}$$

This follows from the additivity of light, i.e. the irradiance map for a combination of light sources is the sum of the individual irradiance maps, and from equation (A.2). It is now possible to express the irradiance ratio between two images as

$$R = \frac{\sum_{j=1}^{J} a_{1j} \sum_{l=0}^{N} \sum_{m=-n}^{n} A_l Y_{lm}(\theta_0, \phi_0) Y_{lm}}{\sum_{j=1}^{J} a_{2j} \sum_{l=0}^{N} \sum_{m=-n}^{n} A_l Y_{lm}(\theta_0, \phi_0) Y_{lm}}. \tag{A.6}$$

Our goal is then to solve the nonnegative least-squares problem

$$\min_a \|Aa\| \qquad \text{s.t} \qquad a > 0. \tag{A.7}$$

Here A is the matrix obtained by writing the equations as in (6.15), and a is the concatenation of the vectors a_{1j} and a_{2j}.

We can replace the SH basis used in the development of the above procedure to enforce

nonnegative light with the analytic PCA basis. Each vector of the analytic PCA basis, as computed in Ramamoorthi (2002), is approximated by a linear combination of order 2 SH basis. Then, it is straight forward to write the projection of the irradiance map for the delta function $\delta_{\theta_0\phi_0}$ in terms of the PCA basis.