AUTOMATED REGISTRATION OF 2D IMAGES WITH 3D RANGE DATA IN A PHOTOREALISTIC MODELING SYSTEM OF URBAN SCENES

by

LINGYUN LIU

A dissertation submitted to the Graduate Faculty in Computer Science in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the City University of New York

2007

UMI Number: 3283175

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

This manuscript has been read and accepted for the Graduate Faculty in

Computer Science in satisfaction

of the dissertation requirement for the degree of Doctor of Philosophy

Chair of Examining Committees: Dr. Stamos Ioannis

Signature _____

Date Approved _____

Executive Officer: Dr. Ted Brown

Signature _____

Date Approved _____

Supervisory Committee: Dr. George Wolberg

Dr. Michael D. Grossberg

Dr. Michael K. Reed

The City University of New York

Abstract

AUTOMATED REGISTRATION OF 2D IMAGES WITH 3D RANGE DATA IN A PHOTOREALISTIC MODELING SYSTEM OF URBAN SCENES

by

Lingyun Liu

Advisor: Professor Ioannis Stamos

The photorealistic modeling of large-scale scenes, such as urban structures, requires the combination of range sensing technology and 2D digital photography. In this thesis, we attack the key problem for this combination, the registration of 2D images with 3D range data, in a unique and efficient manner. A set of novel algorithms are developed and implemented in a comprehensive system. Given 3D range data and a collection of unregistered 2D images of the same scene, our system automatically finds the registration data that is required to produce the optimal texture mapping of 2D images onto the 3D range data.

Dedications

To my parents Weiwei Yang and Feipeng Liu

and my wife Ling Zhang

Acknowledgments

I would like to first express my deep gratitude to my advisor, Dr. Ioannis Stamos. He is a superb mentor one can only wish to have but also a great person. It was him who introduced me into this research field that I love and guided me through every step of the way. Without his invaluable support and confidence in me, this thesis would not be possible. I am very grateful to Dr. George Wolberg for inspiring parts of this thesis and many valuable suggestions. I would also like to thank Dr. Michael D. Grossberg for his advices on my research. Many thanks to Dr. Michael K. Reed for being a member of my thesis committees and critically reading my thesis.

I would like to thank other members in vision lab: Cecilia Chao Chen and Marius Leordeanu. It was a great pleasure to work with them. I am grateful to Siavash Zokai and Gene Yu for providing the high resolution photographs and structure-from-motion results.

Finally, special thanks to my parents and my wife. Their support and unconditional love are the source of my every accomplishment in life.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The visual reproduction of the real world has been a great desire of humans. This desire inspired people to try different ways to achieve this task, either by sculpturing in stones or painting on cloth. Today, modern technology provides us with new means to visualize the world. With the advent of computers and various kinds of color and range sensors, creating the geometric and photometric accurate models from our surroundings has become a major research area in the field of computer vision, graphics and robotics. However, the complexity of real world objects makes it one of the most difficult problems to attack. The natural visual appearance of real scenes is the result of complex interactions of object geometry, reflectance properties and lighting conditions, which makes it very hard to interpret. This complexity and difficulty is what makes this problem extremely interesting from a research point of view. Nevertheless, there is

a high demand of accurate geometric and photometric models in a wide range of practical applications: Google Earth/Microsoft Virtual Earth and Maps, Virtual Reality, Digital Cinematography, Urban Planning, Computer Games, Animation, just to name a few.

Depending on the needs of different applications, generation of photorealistic 3D models with high degree of geometric accuracy and photorealism usually requires large amounts of human interaction. It is an extremely painstaking and error-prone process. The recent development of highly precise range sensors and mega-pixel cameras makes it possible to directly capture accurate geometry and realistic visualization of real-world objects. This opens a new way to obtain digital models and makes this process significantly simplified. Under controlled laboratory environment, the automatic acquisition and modeling of small objects has been well studied and proven to be very effective [Curless and Levoy, 1996, Reed and Allen, 1999, Bernardini *et al.*, 2002, Lensch *et al.*, 2003]. However, for outdoor scenes, this task is more difficult and raises new challenges. In this thesis, we present new methods and tools that can be used for photorealistic modeling of large scale urban scenes.

A typical pipeline for a photorealistic modeling system by the fusion of range data and 2D images is shown in Fig. 1.1. The input raw data are range scans and 2D color images. The range scans are obtained by a laser scanner. Each scan contains the partial 3D geometry information of the scene

in the form of a point cloud. Each 3D point is associated with at least three values $(x, y, z)^T$, which are its Cartesian coordinates in the scanner's local coordinate system. For some scanners, the laser intensity from the returned laser beam is also provided for each point. There are four major steps in this pipeline. First, the range scans are registered and merged together in the same reference frame by the Range-to-Range registration step. As a result, one big point cloud is generated that represent the geometry of the whole scene. Then, a usable 3D geometry model, usually a polygon mesh, is extracted by the surface reconstruction step from the big point cloud. The Image-to-Range registration step is responsible for the alignment of 2D images with the 3D geometry. The registration results are required for the accurate mapping between 3D geometry and 2D images. The texture mapping step produces the final photorealistic 3D model by combining the registered color images with the geometry model. The ultimate goal of the modeling system is to take the raw data in, run through all the steps in the pipeline and produce a desirable result without any human interaction. Unfortunately, this goal has not yet been reached because each of these steps has its own obstacles to conquer.

Range-to-Range Registration The range scans are normally registered in a pair-wise manner by finding correspondences between every two scans. Major challenges include automatic feature extraction and matching, and global optimization.

Image-to-Range Registration The registration between color images and range scans is solved by finding the 2D camera pose and internal calibration based on matching features. How to automatically extract and match 2D and 3D features is the major challenge.

Surface Reconstruction There are many ways to fit surfaces to point clouds. Common major challenges are hole filling and mesh simplification while preserving geometry details.

Texture Mapping The texture mapping of a single image onto a 3D model is well studied. However, stitching multiple 2D images taken under different lighting conditions to produce a seamless, continuous texture mapping is a challenging problem.

1.1 Image-to-Range Registration

This thesis focuses on developing new methods and tools for automatic image-to-range registration, with special emphasis on urban architectural objects. We believe that the ability to automatically register 2D images captured by freely moving cameras to 3D urban models is of major importance. The solution of this problem is critical for texture-mapping 3D models of large-scale scenes. The problem we attack in this thesis can be described as follows: Given a set of 3D range scans and a set of 2D color images, find the relative

position and orientation of the camera with respect to the 3D world coordinate system. We assume that the scene of interest is urban and thus dominated by structures with strong parallelism constraints.

Our main contributions can be summarized as follows:

1. **An automatic image-to-range registration from higher-order linear features.** Our algorithm introduces new types of higher order features which are abundant in architectural environments. Novel algorithms are developed for feature extraction, feature matching and transformation computation. This system requires at least two 2D vanishing points and two 3D major directions to calibrate the camera. Therefore, this system is suitable for the urban scenes with strong parallelism and orthogonality constraints. In scenes containing multi-layer facades or non-planar structures, our system may fail to produce satisfactory results due to limitations of 3D and 2D feature extraction. The robustness of the system also depends on the quality of the 3D/2D features extraction. Sufficient and reliable features are needed to compute the optimal registration results. Details are given in Chapter 3.

2. **Integration of multiview geometry for the automatic registration between a set of color images and the 3D range data.** By registering individual color images with the 3D range data, valuable information derived by the complete set of images is not utilized. We

have thus developed a novel algorithm that merges multiview geometry with image-to-range registration methods. This allows us to align the whole set of 2D images to the 3D range data. This integration improves the robustness, accuracy and efficiency of the whole registration system. Details are given in Chapter 4.

3. **A new line-based image-to-range system** An improved system has been developed to automatically register 2D images with 3D range data. New strategies are introduced for image-to-range registration, including new linear feature extraction, feature matching and camera pose estimation. The new system utilizes 2D and 3D linear features without significant grouping and explores the whole search space of possible matches between 2D and 3D linear features. An easy-to-use interface is also developed to register individual 2D images with 3D range data at an interactive rates. In combination with multiview geometry, this thesis provides a complete solution for registering a set of 2D images with 3D range data. Our system does not assume orthogonality (as in Chapter 3). It still requires, however, the existence of strong parallelism in order to obtain at least two vanishing points and two 3D major directions. For scenes with dominate curved structures, our algorithms may fail. Although the new feature extraction improves the robustness and accuracy over our previous method (Chapter 3), high quality features

are still required for a successful registration. More details are provided in Chapter 5.

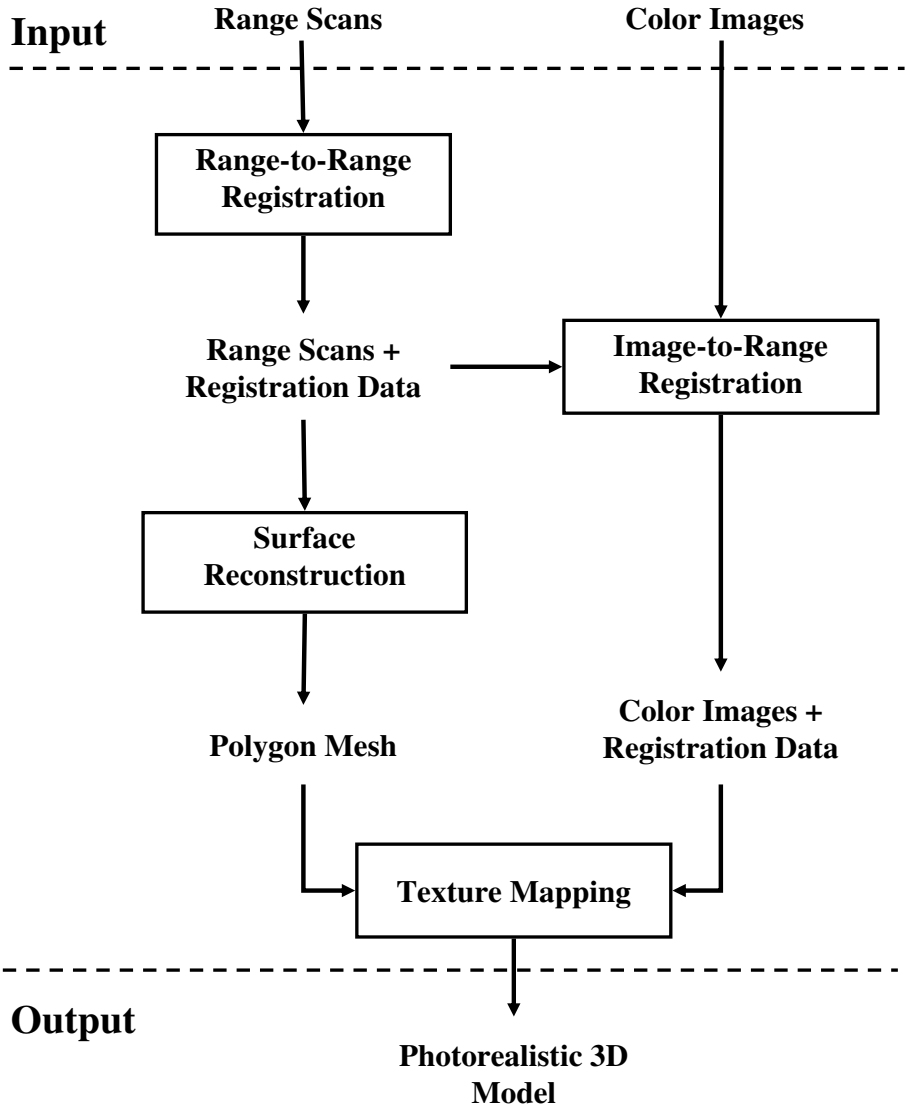

Figure 1.1: General Pipeline for Photorealistic Modeling by combining range sensing and digital photography.

Chapter 2

Related Work

This section presents work in the area of Image-to-Range registration. The problem can be formulated as follows: Given a 2D image of a scene and a 3D range model of the same scene, find the transformation that brings their local coordinate systems into alignment. This is normally achieved by first finding correspondences between the 3D model and 2D image. Then a transformation that aligns the 3D domain to the 2D domain is computed. In particular, the transformation that brings a 3D model point to its corresponding 2D image point can be computed in two steps. The first step is to transform the 3D point from the range scanner coordinate system to the 2D camera coordinate system (Fig. 3.1). This is defined by a rotation and translation (i.e. rigid transformation). Recovering this rigid transformation is referred as **camera pose estimation or external camera calibration**. The second step is to

project a 3D point already expressed in the camera coordinate system onto a 2D point (pixel) on the image plane (Fig. 3.1). This projection is defined by a matrix that depends on the focal length, principal point, aspect ratio, and pixel size. Also the lens distortion needs to be taken into account. Finding the parameters of this projection is known as the **internal camera calibration**. To solve both problems, the knowledge of a set of correspondences between 2D and 3D features is required (e.g. correspondence between 3D model points or lines and their projections on the image plane).

2.1 Camera Pose Estimation

The pose estimation problem was first formally defined by Fischler and Bolles in [Fischler and Bolles, 1981] as Location Determination Problem. It was stated as: "given an image depicting a set of landmarks with known locations, determine that point in space from which the image was obtained". The *landmarks* could be either points or lines. Their algorithm provided a robust solution based on matching 3D points in space with 2D points on image in the presence of gross correspondence errors. First, a subset of correspondences (at least 3 pairs of 2D and 3D points) was randomly selected to compute a pose of the camera. If most of the correspondences agreed with this pose, a least-square method was then used to compute the final pose estimation. Otherwise, another random subset was picked and new estimation was computed.

Three pairs of corresponding 3D and 2D points can provide the solution to the registration problem (3-point based methods). Haralick et al. [Haralick *et al.*, 1989] analyzed the 3-point based methods in different applications, such as 2D image to 2D image registration, 3D model-to-3D model registration and 2D image-to-3D model registration. In the 2D image-to-3D model registration, an iterative least-square method was presented by minimizing weighted depth errors. The weight was used to filter out outliers. In [Horaud *et al.*, 1989], Horaud et al. gave an analytic solution for the case of four non-coplanar point correspondences. Later in [Quan and Lan, 1999], Quan and Lan showed that with four or more point correspondences, it was guaranteed to obtain a unique pose solution. Some other similar methods based on point correspondences are included in [Dhome *et al.*, 1989, Dementhon and Davis, 1995, Oberkampf *et al.*, 1996].

In the case of using line correspondences, theoretically, as few as three lines are enough to compute the camera pose. However, in many cases, three corresponding lines can not give a unique solution (e.g. when the lines are parallel or coplanar on the projection plane). Therefore, more than three corresponding lines are usually needed. In [Liu *et al.*, 1990], Liu et al. proposed a method to estimate the camera pose based on either line or point correspondences. The rotation and translation were obtained separately. To compute the rotation using a linear algorithm, at least eight line or six point correspon-

dences are required, while using a nonlinear method, only three line or two point correspondences are needed. The translation was then computed by a linear method using at least three line or two point correspondences. Kumar and Hanson [Kumar and Hanson, 1994] presented a line-based approach that solved the rotation and translation simultaneously. An iterative least-square algorithm was employed to update the rotation and translation at each round by a small amount until the algorithm converged or the maximum number of iterations reached. Here, to handle the problem of outliers, a Least Median of Square method was proposed. In Christy and Horaud's work [Christy and Horaud, 1999], weak perspective and paraperspective camera models were used in a least-square method to compute the pose.

2.2 Internal Camera Calibration

A typical way to recover the internal camera parameters is using point correspondences [Tsai, 1987, Heikkila and Silven, 1997, Zhang, 2000]. The Direct Linear Transform (DLT) [Hartley and Zisserman, 2004] is a standard algorithm employed in this process. The minimum number of corresponding points required to recover the internal parameters of a camera is based on the type of the camera model. There are various camera models, such as affine, weak perspective, perspective and perspective with lens distortion. In the case of perspective camera without lens distortion, at least six corresponding points

are necessary. Generally, for the purpose of acquiring an accurate calibration, a large number of accurately located correspondences are needed. This is normally achieved by using photographs of a special designed pattern (e.g. planar checkerboard), where the corresponding 3D and 2D points can be easily and accurately obtained. Since the internal camera calibration is independent of the pose estimation (if we assume that the camera moves, but the internal parameters are not changed), it is usually done as a pre-processing step. Under certain constrains, e.g. parallelism and orthogonality, the internal camera parameters can also be recovered automatically from the photographs of real scenes [Caprile and Torre, 1990].

2.3 Automated Pose Estimation

As we have mentioned before, for the efficient photorealistic 3D modeling by the combination of range sensing technology with 2D photograph, the automatic pose estimation is essential. The problem of the automated pose estimation can be described as the follows: given a 3D model and a 2D photograph of the same scene, automatically find correspondences between 3D and 2D data and compute the relative position and orientation of the camera with respect to the 3D model. A conventional method to solve the problem involves the following steps: a) feature extraction (both 2D and 3D); b) feature matching; and c) transformation computation from correspondences. The

major challenges are how to extract meaningful features from 2D images and 3D models, and how to automatically match them. What makes this problem particular harder is that the matching happens between two distinctively different domains, 2D and 3D domains. The features extracted from these two domains are different due to different acquisition processes. 2D features depend on geometry (depth and normal discontinuities), lighting, shadows, occlusions (that cause depth discontinuities), and material properties. 3D features depend on geometry (depth and normal discontinuities) and occlusions only.

In most photorealistic modeling systems using 3D range data and 2D images, the problem of pose estimation is avoided by fixing the relative positions of two sensors (range sensor and 2D camera sensor) [VITG, 2000, Levoy *et al.*, 2000, Zhao and Shibasaki, 2001, Sequeira and Goncalves, 2002, Katsushi Ikeuchi *et al.*, 2003]. This is achieved by rigidly attaching the two sensors together. The camera calibration is done as a pre-processing step in a laboratory environment using specially designed landmarks of known geometry. Some systems use certain type of range finders that can capture both texture and geometry, and hence produce the geometry model with images already aligned [Pulli *et al.*, 1998, Bernardini *et al.*, 2002]. The obvious drawback of these methods is that they sacrifice of the flexibility of 2D photography. The 2D color images can only be acquired at the

same time and space as the 3D range data. Also, a strict scanning process has to be taken so that the relative positions between the two sensors can remain constant. In the work by Yu [Yu, 2000], a different approach was proposed. Spheres of known size and reflectance were manually placed in the scene and used as 3D markers to be detected in the image acquisition step. However, these manually-placed landmarks may destroy the textures and need to be removed afterwards. Some methods obtained the correspondences by manually selecting points or lines [Debevec *et al.*, 1996, Rocchini *et al.*, 1999]. This is actually a commonly used method in practise but very tedious. Another type of approach estimates the camera pose automatically based on an initial estimation. In [Kurazume *et al.*, 2002], the edges from the reflectance images [1] and 2D color images were used as 3D and 2D features. An iterative pose estimation algorithm was used to find the correspondences starting from an initial pose. In each iteration, a set of correspondences were established based on the pose from the previous round. Then, a new pose was computed. The algorithm only stopped when it converged or an error threshold was reached. Lensch [Lensch *et al.*, 2001] proposed an approach to compute the camera pose by matching silhouette of the rendered object with the silhouette extracted from the 2D image. A good initial estimation of the camera pose was also required. In addition, this algorithm worked

[1] reflectance image records the laser reflectance strength for each pixel. It is a natural byproduct of the range scan acquisition process.

well only when the 2D image contained the whole object. A robust method to separate the background was also needed.

As mentioned before, the challenge for a fully automated pose estimation algorithm is feature matching. A typical way to solve this problem follows the Alignment framework [Huttenlocher and Ullman, 1990]. It involves four basic steps:

1. Pick a small subset of 2D and 3D features from the feature space as a hypothetical match.

2. Based on the matched features, compute the transformation that can bring the 3D features into the camera coordinate system.

3. All 3D features are projected onto the image plane using the obtained transformation. For each projected 3D feature, find a possible 2D image feature that is a match according to a certain criterion.

4. Use the found 2D image features to verify the picked hypothetical match and decide whether to accept it or not.

This hypothesis-verification scheme has been proven to be useful in visual object recognition. In the work of [Huttenlocher and Ullman, 1990], a matching algorithm was used to check all possible correspondences between 3D and 2D points. Each hypothetical match containing a minimum set of corresponding features (i.e. three pairs of points) is verified following the Alignment

framework. An affine camera model was used as the approximation of the perspective projection. This approximation made the transformation computation linear. It has been recognized that the verification of the hypothesis is very difficult, considering the uncertainty of location and detection of both 3D and 2D features involved. In [Alter and Grimson, 1997], Alter and Grimson presented an approach to verify the hypothesis by a likelihood-ratio criterion. Instead of using a fixed-sized hypothetical matching feature set as in [Huttenlocher and Ullman, 1990], all the features were utilized to find and verify a hypothesis. Wells [William M. Wells, 1997] proposed a statistical approach to recognize a hypothesis. In this approach, the feature fluctuation was modeled by normal distribution, and hence feature matching was equivalent to a Maximum A-Posterior Probability (MAP) formulation of the problem. Two methods were presented: MAP Model Matching (MMM) and Posterior Marginal Pose Estimation (PMPE). MMM finds the correspondences based on the Alignment framework. PMPE searches for the best pose in the transformation space. For each candidate pose in a local search, posterior probability is computed based on the correspondences. The transformation-space based approach was also used by Cass in [Cass, 1997]. For each pose in the transformation space, all possible geometrically consistent matches were computed. A technique was introduced to reduce the high dimensional transformation space (8D)[2] down to a 2D transformation parameter space. This makes the

[2]The 8D transformation space involves a 2×3 rotation matrix and a 2D translation

transformation-space based method more efficient and feasible. One common attribute of the above methods is the types of features. They are either points or lines. These are primitive features that can be directly extracted from both 2D and 3D data, but normally come in a large quantity. The large search space can make the search for correspondences computationally inefficient or intractable. Outliers may have stronger support than the correct matches.

One way to reduce the number of extracted features and increase the feature reliability is by using higher-order features. In [Stamos and Allen, 2001], Stamos and Allen presented an automated pose estimation method based on matching rectangle features, which were extracted by grouping lines from both 2D and 3D data. The internal camera parameters and camera orientation were recovered by matching vanishing points with major 3D directions. A RANSAC [Fischler and Bolles, 1981] based matching procedure was used to find the correspondences between rectangles. For each sampling set of rectangles, a camera position was computed based on the matching lines from the rectangles. The final camera position was further refined based on the recomputed 2D and 3D line correspondences. This approach shows the advantage of using higher-order features, i.e. increased efficiency and robustness for the registration. However, it is limited to the cases with specific settings of three main orthogonal directions and rectangular features. Also, the RANSAC approach does not guarantee a convergence. In addition, it should be noted that

vector assuming an affine projection.

by using higher-order feature for matching, the range of applications can be narrowed. This is because the extraction of such features is usually based on the assumption of certain constraints, e.g. parallelism and orthogonality. Therefore, there is a trade-off between using higher-order feature which can decrease the search space and using low-level feature which can be applied to more general cases.

Recently, Troccoli and Allen [Troccoli and Allen, 2004] proposed a shadow-based method for pose estimation. Shadows were automatically detected from 2D color images. With an initial estimation of the camera position and known sun position, the final pose was computed by minimizing the shadow pixels in the rendered image. This method works for specific applications since it requires the knowledge of the sun position, a good initial camera pose and the self-casted shadows (i.e. the shadow must be produced by a part of the scene in the image).

In [Zhao *et al.*, 2005], Zhao presented an approach to align continuous video onto a 3D point cloud obtained from a 3D sensor. This method estimates a set of camera poses with respect to the 3D model by registering two 3D point clouds together based on the ICP algorithm [Besl and McKay, 1992]. First, an SFM/stereo algorithm produces a 3D point cloud from the video sequence. This point cloud is then registered to the 3D point cloud acquired from the range scanner by applying the ICP algorithm. The limitation of this approach

is that it shares the shortcomings of the ICP algorithm. In particular, the 3D point clouds must be manually brought close to each other to yield a good initial estimate for the ICP algorithm to work. The ICP may fail in scenes with few discontinuities, such as those replete with planar or cylindrical structures. Also, in order for the ICP algorithm to work, a very dense model from the video sequence must be generated. This means that this approach is restricted to video sequences, which limits the resolution of the 2D imagery.

Chapter 3

Automatic Image-to-Range Registration from Higher-Order Linear Features

In this chapter we deal with the problem of automatic image-to-range registration. The registration method is part of a larger system which constructs high-resolution photorealistic 3D models from unregistered 3D range scans and uncalibrated 2D color images. Our goal is to enhance the geometric model with photographic observations taken from a freely moving 2D camera by automatically recovering the camera's position and orientation with respect to the model of the scene and by automatically calibrating the camera sensor. We are attacking the stated problem under the following assumption: the 3D

scene contains 3D lines defining two major orthogonal directions, i.e. one major vertical direction and at least one major horizontal direction. This is a valid assumption that represents the large majority of scenes in urban settings. We relax the orthogonality (but not the parallelism) assumption in Chapter 5.

As mentioned before, most systems recreating photorealistic models of the environment by a combination of range and image sensing [VITG, 2000, Levoy *et al.*, 2000, Zhao and Shibasaki, 2001, Sequeira and Goncalves, 2002] solve the 3D range to 2D image registration problem by fixing the relative position and orientation of the camera with respect to the range sensor (the two sensors are rigidly attached on the same platform). The fixed-relative position approach provides a solution that has the following major limitations: **A)** The 3D-range and 2D-image captures occur at the same point in time and from the same location in space. That leads to a lack of 2D sensing flexibility, since the limitations of 3D-range sensor positioning (standoff distance, maximum distance) will cause constraints on the 2D camera placement. Also, the 2D images may need to be recaptured due to poor lighting conditions at the time of the 3D-range capture. **B)** The static arrangement of 3D and 2D sensors also means that the 2D camera can not be dynamically adjusted (by changing its focal length and position) to the requirements of each particular scene. **C)** The fixed approach can not handle the case of mapping historical photographs on the models or of mapping 2D images captured at a different instant in time

(under different lighting conditions), something that our method is able to accomplish. In summary, by fixing the relative position between the 3D-range and 2D-sensors, we sacrifice the flexibility of 2D-image capturing.

The part of the thesis presented in this chapter is a continuation on the subject of 3D range to 2D image registration [Stamos and Allen, 2001, Stamos and Allen, 2002]. Major improvements over the previous work can be summarized as follows:

- Utilization of new type of higher-order clusters of 3D and 2D features, 3D rectangular parallelepiped and 2D rectangle.
- Development of a new method for optimizing the internal camera parameters based on two orthogonal vanishing points.
- Development of a new algorithm for matching 3D with 2D features. Our algorithm is **not a probabilistic RANSAC** approach.The whole search space of all possible matching 3D and 2D features is efficiently and systematically explored.

3.1 Problem Formulation

The input to our system is a set of 3D range scans and 2D color images, capturing the same part of real 3D scenes. The relative position and orientation of each 2D camera with respect to the 3D range scans are unknown. Our

goal is to automatically recover these camera parameters. We assume that the 2D camera is a perspective pinhole camera without radial lens distortion (we assume that distortion has been corrected). The geometry of the camera model is shown in Fig. 3.1. $(x_w, y_w, z_w)^T$ are the 3D coordinates of an object

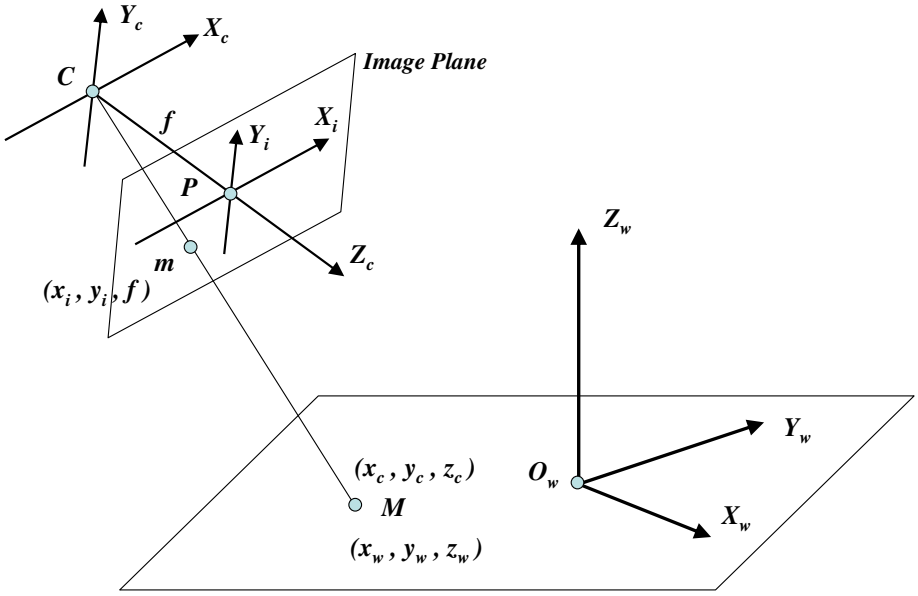

Figure 3.1: Camera Model.

point $\mathbf{M}$ in the 3D world coordinate system $[\mathbf{O}_w, \mathbf{X}_w, \mathbf{Y}_w, \mathbf{Z}_w]$. $(x_c, y_c, z_c)^T$ are the 3D coordinates of the same point $\mathbf{M}$ in the 3D camera coordinate system $[\mathbf{C}, \mathbf{X}_c, \mathbf{Y}_c, \mathbf{Z}_c]$, which is centered at the optical center $\mathbf{C}$, with the $\mathbf{Z}_c$ axis the same as the optical axis. $\mathbf{X}_i$ and $\mathbf{Y}_i$ are the axes of image coordinate system and parallel to axes $\mathbf{X}_c$ and $\mathbf{Y}_c$. The point $\mathbf{P}$ is the intersection of the optical

axis $\mathbf{Z}_c$ with the image plane, which is normally referred to as the principal point of the image. $\mathbf{m}$ is the projection of $\mathbf{M}$ on the image plane and its coordinate is $(x_i, y_i)^T$. In the camera coordinate system, $(x_i, y_i, f)^T$ is the 3D coordinate of $\mathbf{m}$, where f, the effective focal length, is the distance between the image plane and the optical center $\mathbf{C}$. The scale of the camera coordinate system is expressed in pixels and the scale of the world coordinate system in meters. For a perspective pinhole camera without lens radio distortion, the overall transformation from $(x_w, y_w, z_w)^T$ to $(x_i, y_i)^T$ can be depicted by two steps:

1. Rigid body transformation from the 3D world coordinate system $(x_w, y_w, z_w)^T$ to the camera 3D coordinate system $(x_c, y_c, z_c)^T$.

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \mathbf{R} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \mathbf{T}$$

   $\mathbf{R}$ is a 3×3 rotation matrix and $\mathbf{T}$ is a translation vector.

2. Transformation from the 3D camera coordinate $(x_c, y_c, z_c)^T$ to image coordinate $(x_i, y_i)^T$

$$\begin{bmatrix} {x'}_i \\ {y'}_i \\ w_i \end{bmatrix} = \mathcal{K} \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}, \begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} {x'}_i / w_i \\ {y'}_i / w_i \end{bmatrix}$$

where

$$\mathcal{K} = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix}$$

, which is called the camera calibration matrix [Hartley and Zisserman, 2004]. $(p_x, p_y)^T$ are the coordinates of the principal point on image plane.

$\mathbf{R}, \mathbf{T}$ and $\mathcal{K}$ are the parameters to be recovered.

In our method, the rotation and translation are solved separately. The internal camera calibration $\mathcal{K}$ and rotation $\mathbf{R}$ are computed by matching at least two vanishing points with 3D major directions. For the translation $\mathbf{T}$, higher-order 3D and 2D features are used to find the correspondences and compute the translation $\mathbf{T}$. The approach involves 3 major steps:

1. 3D feature extraction: A novel type of 3D linear features, rectangular parallelepiped, are extracted by clustering 3D lines.

2. 2D feature extraction, Internal camera calibration and Rotation Computation: The internal calibration of the camera and the rotation are recovered by matching extracted vanishing points with the major 3D axis. These parameters are used to rectify 2D lines. 2D rectangles are then extracted by grouping the rectified 2D lines.

3. Automated feature matching and translation Computation. A pair-wise algorithm is used to estimate matching 2D and 3D features. The candi-

date matches are found by evaluating an overlap criterion. The translation is computed by maximize the overlap among all candidates.

The following sections describe these steps in more details.

3.2 3D Feature Extraction

3.2.1 3D Range Data Acquisition

The first step is to acquire n range scans to adequately cover the 3D scene. Range scans are acquired by a laser-range scanner (Leica HDS 2500 [Geosystems, 2000]), an active sensor that emits eye-safe laser beams into the scene. It is capable of gathering one million 3D points at a maximum distance of 100 meters. A range scan of an urban scene is shown in Fig. 3.2(top). Each point is associated with four values $(x, y, z, l)^T$, where $(x, y, z)^T$ is its Cartesian coordinates in the scanner's local coordinate system, and l is the laser intensity of the returned laser-beam. The intensity depends on the material properties of the physical 3D surface, the distance of the point from the range sensor, and the orientation of the laser beam with respect to the local surface normal at the measured 3D point.

Each range scan is processed via an automated segmentation algorithm [Stamos and Allen, 2002]. A set of major 3D planes and a set of geometric 3D lines G_i are extracted from each scan $i = 1, \ldots, n$. The geometric 3D lines

are computed at the intersections between segmented planar regions and at the borders of the segmented planar regions [Stamos and Allen, 2002]. The range scans are registered in the same coordinate system via the automated range-range feature-based registration method which is described in [Stamos and Leordeanu, 2003, Chen and Stamos, 2005]. As a result, all range scans are registered with respect to one selected pivot scan, in the scene's coordinate system, namely S_{3D}.

3.2.2 3D Line Extraction

We use lines as the basis of our feature extraction process. This is because they are reliably localized since they are supported by a large number of measurements. In addition to the geometric lines G_i, a set of reflectance 3D lines L_i are extracted from each 3D range scan. They are produced by discontinuities of the laser intensity (Fig. 3.2 (top)). We extract 2D lines from the reflectance image using standard image processing techniques (Canny edge detector followed by orthogonal regression). The end points of each reflectance 2D line $(p_1, p_2)^T$ correspond to two 3D points $(x_1, y_1, z_1)^T$ and $(x_2, y_2, z_2)^T$ which define a geometric line in 3D space. We call this line a reflectance 3D line because it is computed based on information gathered from the reflectance image alone. Individually, the reflectance lines may not be as accurate as 3D geometry lines, but as a group, they can provide stronger support for the 3D feature extraction

than using 3D geometry lines only. The combination of all 3D geometric and reflectance lines provides a very rich representation of the acquired 3D scene. We use $\mathcal{L}^{3D}$ to represent those lines. Therefore: $\mathcal{L}^{3D} = \bigcup_i L'_i \cup G'_i$, where L'_i and G'_i are the computed reflectance lines L_i and geometric lines G_i of each scan after their transformations into the scene coordinate system S_{3D}[1].

The next step is to cluster the line set $\mathcal{L}^{3D}$. This clustering is necessary for the computation of rotation (matching of clusters with 2D vanishing points). It is also used for 3D face extraction, and 3D parallelepiped extraction. Each line in a cluster has the similar orientation as every other line in the same cluster. One large cluster of vertical 3D lines and a number of clusters of horizontal 3D lines are expected to be obtained. We call the cluster of vertical lines ${\mathcal{L}^{3D}}_v$, and the one (or more) clusters of horizontal lines ${\mathcal{L}^{3D}}_{h_1}, {\mathcal{L}^{3D}}_{h_2}, \ldots, {\mathcal{L}^{3D}}_{h_m}$. Fig. 3.2 (bottom) shows the clustered sets of 3D geometric and reflectance lines from one of our experiments.

In all the following algorithms, 3D and 2D lines or features are transformed to the coordinate system S_{base}. S_{base} serves as a common intermediate coordinate system of reference, where the horizontal features are parallel to the x-axis and the vertical features parallel to the y-axis. That property makes the implementations of feature matching and translation computation very efficient (see section 3.4). S_{base} is defined by the three orthogonal axes $\hat{\mathbf{x}} = [1, 0, 0]^T$,

[1]Note that we know the rotational and translational transformation between range scans from the range-range registration module.

$\hat{\mathbf{y}} = [0, 1, 0]^T$, $\hat{\mathbf{z}} = [0, 0, 1]^T$, and the origin which is at $[0, 0, 0]^T$.

3.2.3 3D Face Extraction

The 3D *face* extraction is necessary for the 3D parallelepipeds extraction. A 3D *face* is a set of 3D lines of two major directions, one vertical and one horizontal. Each 3D *face* can be considered as a line-representation of one planar facade of the scene. The final 3D features (parallelepipeds) are extracted only from the 3D lines in the 3D faces. Let us consider one pair of vertical and horizontal clusters $({\mathcal{L}^{3D}}_v, {\mathcal{L}^{3D}}_{h_i})$[2]. The 3D lines of this pair are aligned to the axes of S_{base} through a rotation $\mathbf{R}_{3D}(i)$: $\mathbf{R}_{3D}(i) = [\hat{\mathbf{x}}\,\hat{\mathbf{y}}\,\hat{\mathbf{z}}]^T \cdot [\mathbf{x_n}\,\mathbf{y_n}\,\mathbf{z_n}]^{-T}$, where $\mathbf{y_n}$ = direction of ${\mathcal{L}^{3D}}_v$, $\mathbf{z_n} = \mathbf{y_n} \times$ direction of ${\mathcal{L}^{3D}}_{h_i}$, and $\mathbf{x_n} = \mathbf{y_n} \times \mathbf{z_n}$. The transformed 3D lines are further clustered into major 3D *face* (Fig. 3.3 (top)). Each 3D *face* is thus defined by its base-plane and all lines that lie on it. The lines can be either vertical or horizontal but their distances from the base-plane should be smaller than a user-defined threshold $\mathcal{B}_{th}$.

3.2.4 Rectangular Parallelepiped Extraction

Our goal is to obtain 3D features from the 3D line sets that are matchable with 2D features from the 2D color images. Matching individual 3D lines with individual 2D lines is impractical due to the large size of the generated search

[2]This pair can be interactively selected by the user via a simple color-based user interface.

space. Another problem is that some 3D lines are not present in the 2D image and vice versa (eg. 2D lines that are generated by shadow discontinuities are not present in the 3D model of the scene). Therefore, we use higher level features, i.e. vertical or horizontal 3D rectangular parallelepipeds that can be matched with 2D rectangles obtained from the 2D images.

The rich set of geometric and reflectance lines in a 3D $face$ (see Sec. 3.2.3) are grouped into sets of lines which define 3D rectangular parallelepipeds [3] in space. The set of extracted parallelepipeds (Fig. 3.3 (bottom)) for a 3D $face$ is called RP. Each parallelepiped $rp \in RP$ contains clusters of nearby 3D lines. There are two types of clusters: vertical (containing lines parallel to the vertical direction) and horizontal (containing lines parallel to the horizontal direction). rp is thus defined by three attributes: 1) type: vertical or horizontal; 2) **tl** : top left vertex of rp; 3) **br** : bottom right vertex of rp. The computation of the set RP is done as follows. Initially, every line on a 3D $face$ becomes a trivial parallelepiped rp. This trivial rp is a rectangle with a fixed initial width. When projections of two rp on the 3D $face$ base-plane overlap, these two rp are merged into a bigger rp which includes all lines in them. This merging process will continue until there is no overlap between any of the remaining rp.

[3] Note that we are extracting parallelepipeds instead of rectangles (a rectangle can be viewed as a parallelepiped with zero depth). We can not expect all 3D lines to lie exactly on top of the 3D $face$ base-plane; some of the lines are produced by architectural details on each facade, or by window frames. Therefore, the extracted linear features of each 3D $face$ may lie as far as $\mathcal{B}_{th}$ meters away from the major scene facade.

3.3 2D Feature Extraction, Internal Camera Calibration, & Rotation Computation

3.3.1 Internal Camera Calibration and Rotation Computation

The internal parameters (focal length f and principal point $\mathbf{P}$ as in Fig. 3.1) of the camera sensor can be calculated from a 2D image, if the image contains at least two vanishing points (i.e. the 3D scene which the camera is viewing has at least two major scene directions). We use a previously developed robust methods to generate and cluster 2D lines from a 2D image [Stamos and Allen, 2001]. The result is a set of major vanishing points $\mathbf{V}_1, \mathbf{V}_2, \cdots, \mathbf{V}_n$. Using the methods described in [Stamos and Allen, 2001] we can compute the effective focal length and principal point (expressed in pixels) by utilizing **three** orthogonal vanishing points. In the case that the scene contains only two vanishing points, we calculate the center of projection as follows.The two vanishing points $\mathbf{V}_1$ and $\mathbf{V}_2$ are expressed in the camera coordinate system as: $\mathbf{V}_1 = [(V_1)_x, (V_1)_y, f]^T$, $\mathbf{V}_2 = [(V_2)_x, (V_2)_y, f]^T$. The angle between the directions that created the vanishing points is given by the clusters of 3D lines (in most scenarios, it's 90 degrees). If this angle is θ radians, then $\cos(\theta) = n(\mathbf{C} - \mathbf{V}_1) \cdot n(\mathbf{C} - \mathbf{V}_2)$, where $n(\mathbf{v})$ is the unit vector parallel to vector $\mathbf{v}$. The effective focal length f can be computed from the equation by

assuming an approximate principal point $\mathbf{P}$ at the center of the image. This approximation is further refined at the end of this section.

The matching of the computed vanishing points with the extracted major scene directions ${\mathcal{L}^{3D}}_v$ and ${\mathcal{L}^{3D}}_{h_i}$ can be automatically achieved by using a-priori assumptions about the position of the camera with respect to the 3D scene, or by using information about the relative size of matched clusters. Alternatively, the user can easily pick the correct matches through a color-based user-interface. The correspondence between vanishing points $\mathbf{V}_v$, $\mathbf{V}_{h_k}$ and 3D directions ${\mathcal{L}^{3D}}_v$, ${\mathcal{L}^{3D}}_{h_i}$, provides a solution to the rotation $\mathbf{R}_{2D}$ that brings the 2D features in the coordinate system S_{base} (see Sec. 3.2): $\mathbf{R}_{2D} = [\hat{\mathbf{x}}\,\hat{\mathbf{y}}\,\hat{\mathbf{z}}]^T \cdot [n(\mathbf{v}_1 \times \mathbf{v}_2)\,\mathbf{v}_1\,\mathbf{v}_2]^{-T}$, where $\mathbf{v}_1 = n(\mathbf{V}_v - \mathbf{C})$ and $\mathbf{v}_2 = n(\mathbf{v}_1 \times (\mathbf{V}_{h_k} - \mathbf{C}))$. Then all 2D lines can be rectified by applying the rotation R_{2D} about $\mathbf{C}$ and then by projecting to the image plane.

The effective focal length f and principal point $\mathbf{P}$ can be further refined as follows. By construction, the rectified 2D lines from ${\mathcal{L}^{2D}}_v$ and ${\mathcal{L}^{2D}}_{h_i}$ should be parallel to the $y-axis$ and $x-axis$ of S_{base}, respectively (Fig. 3.4 (bottom)). But if the estimation of f and $\mathbf{P}$ is not accurate, the angle between the actual direction of a rectified line and its expected direction will not be zero. This per-line computed angle is considered to be an error due to an inaccurate estimation. The sum of the errors of all lines, called $\mathcal{A}$, is used as a criterion to compute a more accurate f and $\mathbf{P}$ as follows. We consider all possible principal

points in a small spatial neighborhood around the initially computed principal point $\mathbf{P}$. Searching for the best principal point $\mathbf{P}_{best}$ is done sequentially in a spiral manner in the neighborhood of the initial $\mathbf{P}$. For each candidate principal point, an effective focal length is computed. $\mathbf{P}_{best}$ and f_{best} are the first principal point and focal length that produces an angle error $\mathcal{A}$ less than a user-defined threshold $\mathcal{A}_{th}$ (normally between 2 to 5 degrees). Then we have the camera calibration matrix $\mathcal{K}$ as ([Hartley and Zisserman, 2004]):

$$\mathcal{K} = \begin{vmatrix} f_{best} & 0 & (P_{best})_x \\ 0 & f_{best} & (P_{best})_y \\ 0 & 0 & 1 \end{vmatrix}$$

The same method is applied to extract horizontal and vertical rectangular 2D features RC as the one used to extract 3D features RP in Sec. 3.2.4, except that the extracted 2D features has a zero depth, i.e. they are rectangles instead of parallelepipeds (Fig. 3.4).

3.4 Translation Computation

In this section we present the algorithm that automatically computes the translation between the scene S_{3D} and the camera S_{camera} coordinate systems (Fig. 3.5). From the previous steps, we have the sets of 3D parallelepipeds RP and 2D rectangles RC ready to be matched in S_{base}. The rotation $\mathbf{R}$ that brings S_{3D} to S_{camera} is $\mathbf{R} = {\mathbf{R}_{2D}}^{-1} \cdot \mathbf{R}_{3D}$. The two rotational components

R_{3D} and R_{2D} were computed in Sec. 3.2 and 3.3, respectively.

In order to compute the translation between the scene and camera coordinate systems, we need to identify K 3D parallelepipeds rp (see Fig. 3.3 for an example set of 3D features) that match K 2D rectangles rc (see Fig. 3.4 for an example set of 2D features). It is important to note that we can only match horizontal 3D features with horizontal 2D features, and vertical 3D features with vertical 2D features. The problem is thus reduced to a 2D pattern matching problem. A single matched rp with a rc is not able to provide the correct translation as our experiments have shown. This is due to the uncertainty of the location and detection of both 3D and 2D features. However, two correctly matched rps with 2D rcs are able to provide us a very accurate translation as will be shown in the algorithm to follow. Thus our algorithm follows a hypothesis-verification scheme and searches through all pairs of possible matches systematically. Note that we consider all $\binom{N_h}{2} \times \binom{n_h}{2} + \binom{N_v}{2} \times \binom{n_v}{2}$ possible matched pairs (where N_h, N_v are the number of horizontal and vertical rp, respectively, and n_h, n_v are the number of horizontal and vertical rc, respectively). This is a large search space, but it can be efficiently explored as our results show (see Sec. 3.5).

Our algorithm consists of six steps (Fig. 3.5). In the first four steps, a list of candidate translations are being computed from the exploration of all hypothetical matches between pairs of 3D parallelepipeds and pairs of 2D

rectangles. The fifth step determines the grades of each candidate translations based on the number of matching 3D and 2D feature pairs produced by this translation. The candidates with grades smaller than a threshold will be eliminated. The sixth step searches in the neighborhood of each remaining candidate translation for the one that maximizes the amount of overlap. Thus, for each candidate, a final optimized translation $\mathbf{t}_{opt}$ is computed.

For better understanding of our algorithm we present the notation we are using. We are considering 3D feature (parallelepiped) pairs $\mathbf{rp} = (rp_1, rp_2) \in RP \times RP$, and 2D feature (rectangle) pairs $\mathbf{rc} = (rc_1, rc_2) \in RC \times RC$. $\mathbf{c}_{rpi}$ or $\mathbf{c}_{rci}$ is defined as the centroid of a 3D feature rp_i or 2D feature rc_i respectively, while π_{3D} is the base-plane of 3D $face$ where $\mathbf{rp}$ lie on and π_{2D} is the 2D image plane (note that these planes are parallel to the x-y plane of the S_{base} coordinate system). The projection of a 3D feature rp_i on the 2D image plane is denoted as rp_i^p.

The following describe the six steps of the translation computation algorithm in more detail. The flowchart of this algorithm is shown in Fig. 3.5.

(Step 1) Consider the next pair of 3D features $\mathbf{rp} = (rp_1, rp_2)$ to match the pair of 2D features $\mathbf{rc} = (rc_1, rc_2)$. All possible matching pairs are considered. Many of them can be discarded as shown in the following steps.

(Step 2) We assume that the 3D feature rp_1 matches the 2D feature rc_1. Then the translation vector $\mathbf{t}_{(rp_1, rc_1)}$ that brings the centroids of both

features into alignment should satisfy the following equation (see Fig. 3.6): $\mathbf{c}_{rc_1} = \mathcal{K}\,[\,\mathbf{I}\,|\,\mathbf{t}_{(rp_1,rc_1)}\,]\,\mathbf{c}_{rp_1}$, where $\mathbf{I}$ is identity matrix. $\mathbf{t}_{(rp_1,rc_1)}$ is computed as follows. First, we compute the ratio $r = \frac{len_{rp_1}}{len_{rc_1}}$, where len indicates length of the feature (horizontal or vertical length depending on feature's type). This ratio reflects the scale that should be applied to the 2D image feature (measured in pixels), so that it matches the 3D feature (measured in meters). Then, considering the line segment from the centroid of the 2D feature $\mathbf{c}_{rc_1}$ to the center of projection $\mathbf{C}$, we find the point $\mathbf{d}$ so that $\frac{\|\mathbf{C}-\mathbf{d}\|}{\|\mathbf{C}-\mathbf{c}_{rc_1}\|} = pm$, where $\|\mathbf{v}\|$ denotes the norm of vector $\mathbf{v}$. The translation vector $\mathbf{t}_{(rp_1,rc_1)}$ can now be computed as follows (Fig. 3.6): $\mathbf{t}_{(rp_1,rc_1)} = \mathbf{d} - \mathbf{c}_{rc_1}$. This translation vector translates the 3D feature rp_1 (note that rp_1 is already expressed in the common coordinate system S_{base}) into the unique position that makes its projection to the image plane π_{2D} have the following properties: A) The length of the projection of rp_1 is exactly the same as the length of rc_1, and B) The centroid of rp_1 is projected exactly on the centroid of rc_1. The estimation of this translation vector can be performed very efficiently in the coordinate system S_{base}. This is one of the factors that attributes to the efficiency of our algorithm.

The just obtained translation brings the two features rp_1 and rc_1 into alignment (the center of rp_1 is projected on the center of rc_1). A correct translation will also bring rp_2 and rc_2 into alignment if these 2 features are corresponding to each other. By applying the translation $\mathbf{t}_{(rp_1,rc_1)}$ to rp_2,

and projecting it onto π_{2D}, we produce a 2D feature called ${rp_2}^p$ (this is a 2D rectangle): ${rp_2}^p = \mathcal{F}(rp_2, \mathbf{t}_{(rp_1,rc_1)})$, where $\mathcal{F}$ is the function that projects 3D features onto π_{2D}. This is achieved by projecting two translated diagonal vertices (top left and bottom right) of the 3D feature onto π_{2D} and then generating 2D rectangle based on the two projected vertices [4] The percentage of overlap between the features ${rp_2}^p$ and rc_2 is denoted as $\mathcal{O}_{({rp_2}^p,rc_2)}$. If this overlap is larger than a user-defined threshold $\mathcal{O}_{th}$ (normally 80%), we proceed to the next step; otherwise we go back to step 1, and have the next pair of features to be matched.

(Step 3) Step 2 is repeated for the computation $\mathbf{t}_{(rp_2,rc_2)}$ (we now assume that rp_2 matches rc_2). If the overlap $\mathcal{O}_{({rp_1}^p,rc_1)}$ is larger than $\mathcal{O}_{th}$ as well, the two pairs $(\mathbf{rp}, \mathbf{rc}) \in RP^2 \times RC^2$ are considered as matching candidates and the algorithm proceeds to Step 4. Otherwise we go back to Step 1 to consider the next pair of matches.

(Step 4) The final translation $\mathbf{t}_{(\mathbf{rp},\mathbf{rc})}$ is computed by a weighted average of the component translations: $\mathbf{t}_{(\mathbf{rp},\mathbf{rc})} = w_1 \cdot \mathbf{t}_{(rp_1,rc_1)} + w_2 \cdot \mathbf{t}_{(rp_2,rp_2)}$, where the weight is the overlap area ratio (i.e. the bigger the relative overlap, the bigger the weight): $w_1 = \frac{\mathcal{O}_{({rp_2}^p,rc_2)}}{\mathcal{O}_{({rp_1}^p,rc_1)}+\mathcal{O}_{({rp_2}^p,rc_2)}}, w_2 = 1 - w_1$.

(Step 5) By repeating steps 1 through 4 on all possible pairs of 3D and 2D feature pairs $(\mathbf{rp}, \mathbf{rc}) \in RP^2 \times RC^2$, the translations of all matching candidates

[4] This is an approximation of the projection. It works well when the thickness of the 3D feature is small (< 0.5 m). 3D features with larger thickness may generate wrong projections with larger projected areas.

are computed and stored in a set $\mathcal{T}$. Each translation $\mathbf{t}_i$ in $\mathcal{T}$ is applied to all $rp \in RP$. The translated rp are then projected on π_{2D}, producing rectangles rp^p. Every pair (rp^p, rc) with overlap larger than $\mathcal{O}_{th}$ is stored in the set $\mathcal{M}_i$. The total number of the pairs in $\mathcal{M}_i$ is defined as the grade $\mathcal{G}_i$ of the translation $\mathbf{t}_i$. $\mathbf{t}_i$ in $\mathcal{T}$ are then sorted based on $\mathcal{G}_i$, and the translations with grade less than a user-defined threshold $\mathcal{G}_{th}$ are deleted from the list [5]. This increases the efficiency, without sacrificing the accuracy of the algorithm.

(Step 6) Each translation $\mathbf{t}_i$ in the list $\mathcal{T}$ can be optimized even further. Our goal is to maximize the overlap between matching pairs produced by each translation $\mathbf{t}_i$ (the set of matching features for each translation has been stored in $\mathcal{M}_i$ – see previous step). The optimization is performed in the transformation space. We consider a neighborhood $\mathcal{N}_i$ of all possible translations around $\mathbf{t}_i$. For each one of the translations in this neighborhood, the overlap between the matching features in $\mathcal{M}_i$ is calculated. The translation in $\mathcal{N}_i$ that produces the maximum overlap is chosen as our final optimum result for $\mathbf{t}_i$: $\mathbf{t}_{opt}(i) = \max_{\mathbf{t}\in\mathcal{N}_i}(\sum_{(rp,rc)\in\mathcal{M}_i} \mathcal{O}_{(\mathcal{F}(rp,\mathbf{t}),rc)})$. As a result, the user is presented with a sorted list of $\mathbf{t}_{opt}(i)$. In most cases, the translation with the highest grade (i.e. largest number of matching features) is the one that produces the best result. Otherwise, user can select the 2nd, 3rd or other listed translation (Fig. 3.10).

[5] The grade threshold $\mathcal{G}_{th}$ is computed as $\mathcal{G}_{th} = \mathcal{G}_{min} + \alpha \cdot (\mathcal{G}_{max} - \mathcal{G}_{min})$, where $\mathcal{G}_{max}$ and $\mathcal{G}_{min}$ are the maximum and minimum grades, $0.6 \leq \alpha \leq 1.0$.

This selection is extremely simple, since user can instantly assess the computed translation by visually inspecting the computed texture mapped result.

Finally, a point $\mathbf{x}_{3D}$ can be transformed from the 3D scene coordinate system to a point $\mathbf{x}_{cam}$ in the 2D camera coordinate system via:

$$\mathbf{x}_{cam} = \mathcal{K}[\mathbf{R} \,|\, \mathbf{T}\,]\, \mathbf{x}_{3D}$$

where $\mathbf{R} = {\mathbf{R}_{2D}}^{-1} \cdot \mathbf{R}_{3D}$ and $\mathbf{T} = {\mathbf{R}_{2D}}^{-1}\, \mathbf{t}_{opt}(i)$

3.5 Results

We performed experiments in three urban settings (buildings 1, 2 and 3). Building 1 is the Thomas Hunter building at Hunter College of CUNY in NYC. Building 2 is at the corner of 695 Park Avenue and 69th street in NYC. Finally, building 3 is the Shepard Hall building at CCNY in NYC. Buildings 1 and 2 are regular urban structures with many windows and large planes (as shown in Fig. 3.11), while building 3 has a more complicated structure (as shown in Fig. 3.12). A number of 3D range scans and 2D images were acquired for each building. After the range scans are registered on the same coordinate system, the 2D images are automatically calibrated and registered on the 3D model of each acquired structure. Fig. 3.7, 3.8 and 3.9 present overviews of our texture-mapping results and automatically recovered camera configurations. We have not performed any intelligent blending of overlapping 2D images, something

that can be part of the future work. Finally, detailed textured maps are shown in Figs. 3.11. Two of the automated transformations computed from building 3 required a very small correction (few pixels of translational adjustment) by a human user through our easy to use user interface.

The performance of our 3D range to 2D image registration algorithm is shown in Table 3.1. The following abbreviations are being used. FP: Number

Table 3.1: Building 1 (11 images, 12 3D scans) – Building 2 (6 images, 3 range scans) – Building 3 (6 images, 4 range scans)

FP($3D \times 2D$)	CM	G	OP(%)	RDT(%)	T
117×100	25	14	91.23	0.21	42 sec
34×54	9	13	95.67	0.49	2 sec
117×55	28	17	93.45	0.43	11 sec
117×145	8	20	92.34	0.04	85 sec
117×106	15	9	91.23	0.12	44 sec
117×44	19	11	92.05	0.37	4 sec
117×74	7	32	94.56	0.08	21 sec
117×113	20	18	87.14	0.15	55 sec
35×24	1	16	98.40	0.1	0.6 sec
117×74	14	16	93.78	0.17	18 sec
35×59	9	5	89.31	0.43	1.5 sec
52×80	4	13	92.79	0.09	4 sec
63×55	155	18	96.13	4.4	35 sec
63×46	153	22	90.17	5.2	38 sec
52×81	8	8	86.38	0.19	5 sec
52×61	4	9	93.14	0.12	3 sec
52×57	18	7	92.23	0.61	4 sec
67×60	12	8	87.62	0.29	6 sec
67×81	18	8	92.45	0.31	10 sec
67×39	18	11	94.34	0.69	3 sec
67×38	12	11	91.74	0.47	2 sec
67×71	23	7	88.16	0.48	9 sec
67×33	7	8	93.51	0.31	1 sec

of feature pairs (number of 3D features $\times$ number 2D features), CM: Number

of candidate translations in list $\mathcal{T}$ after thresholding (see sixth step of algorithm of Sec. 3.4), G: Grade of optimum translation $\mathbf{t}_{opt}$ (number of matching features), OP: Amount of overlap among matching features (see fifth step of algorithm of Sec. 3.4), RDT: Percentage of candidate translations (CM) over all possible translations generated by matches of 3D and 2D features, T: Execution time of the automated registration algorithm on a 2GHz Pentium machine. The fast execution time is based on the large reduction of candidate translations (see RDT column).

3.6 Discussion

In this chapter, we presented a novel and efficient algorithm for the 3D range to 2D image registration problem in urban scene settings. Our input is a set of 3D range scans and a set of 2D images. The range scans are abstracted into sets of 3D lines, followed by three clustering steps. As a result, sets of 3D rectangular parallelepipeds are extracted. For each 2D image, 2D rectangles are generated via vanishing point extraction, camera calibration and rectification steps. Finally, with sufficient features provided, an automated algorithm computes an optimized transformation between the 2D images and 3D range scans. This transformation is based on a match of 3D with 2D features that maximizes an overlap criterion. Our algorithm attacks the hard 3D range to 2D image registration problem in a systematic, efficient, and automatic way,

in the context of urban scenes. Images captured by a high-resolution 2D camera, which moves and adjusts freely, are mapped on a centimeter-accurate 3D model of the scene providing photorealistic renderings of high quality.

It should be noted that this system requires the existence of strong orthogonality and parallelism constraints in the scene. Without these constraints, the system may not be able to extract sufficient (at least two) reliable 2D vanishing points and 3D directions to compute the internal camera calibration and the camera rotation which brings the 2D and 3D coordinate systems into alignment. As shown in Fig. 3.13, the scenes in the images are dominated by the curved structures which are not suitable for the line-based vanishing point extraction applied in this system. Therefore, accurate vanishing points can not be extracted from these scenes. Due to the lack of major and accurate vanishing points, our algorithm will fail in this scenario. However, for the same building, some images can be registered by our system. For example, the image in Fig. 3.14 (bottom left) can be registered with 3D range data (Fig. 3.14 top left). This is because the two towers of the building are captured in the image. These two towers contain the sufficient lines for the extraction of two orthogonal vanishing points and 2D features (Fig. 3.14, bottom right). 3D features can also be extracted from these two towers (Fig. 3.14 top right). These 3D features are of two major directions, one vertical (red) and one horizontal (green). With sufficient 3D/2D features, the system can register the 2D image

with 3D range data. Another limitation of the system is that it has to extract 3D and 2D faces. That means that scenes with various layers of planar facades, or without clear major facades can not be handled. For example, as in Fig. 3.15, the scene of a part of the Grand Central Station, contains multiple layers of facades, such as cylinders, back walls, ticket-selling stand. During 3D face extraction, all these multiple layers will be combined into one face, which has a large thickness. As a result, the 3D features extracted from the combined 3D face will have deep depths as well. When projecting these thick 3D features on the 2D image plane to match with 2D features, the incorrectly larger projections (see Steps 2 & 3 of translation computation algorithm) will be created due to the thickness of 3D features. Eventually, these incorrect larger projections can cause wrong matches with 2D features. In Chapter 5, we present a registration method that avoids the face extraction step and is thus suitable for more general scenes (such as the one seen in Fig. 3.15).

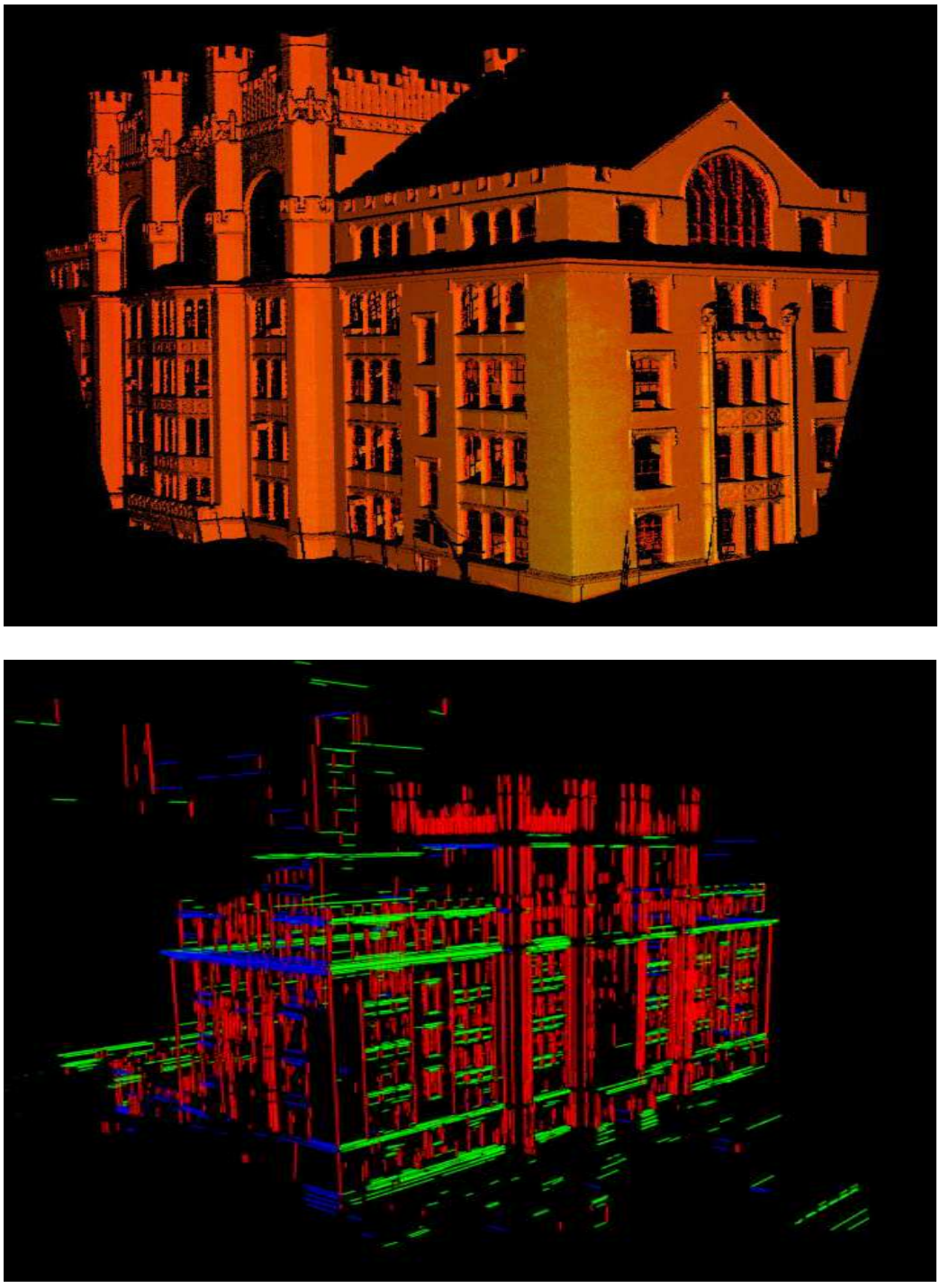

Figure 3.2: Example of a range scan of Thomas Hunter building at Hunter College of CUNY (building 1) (top) and the extracted 3D lines (bottom).

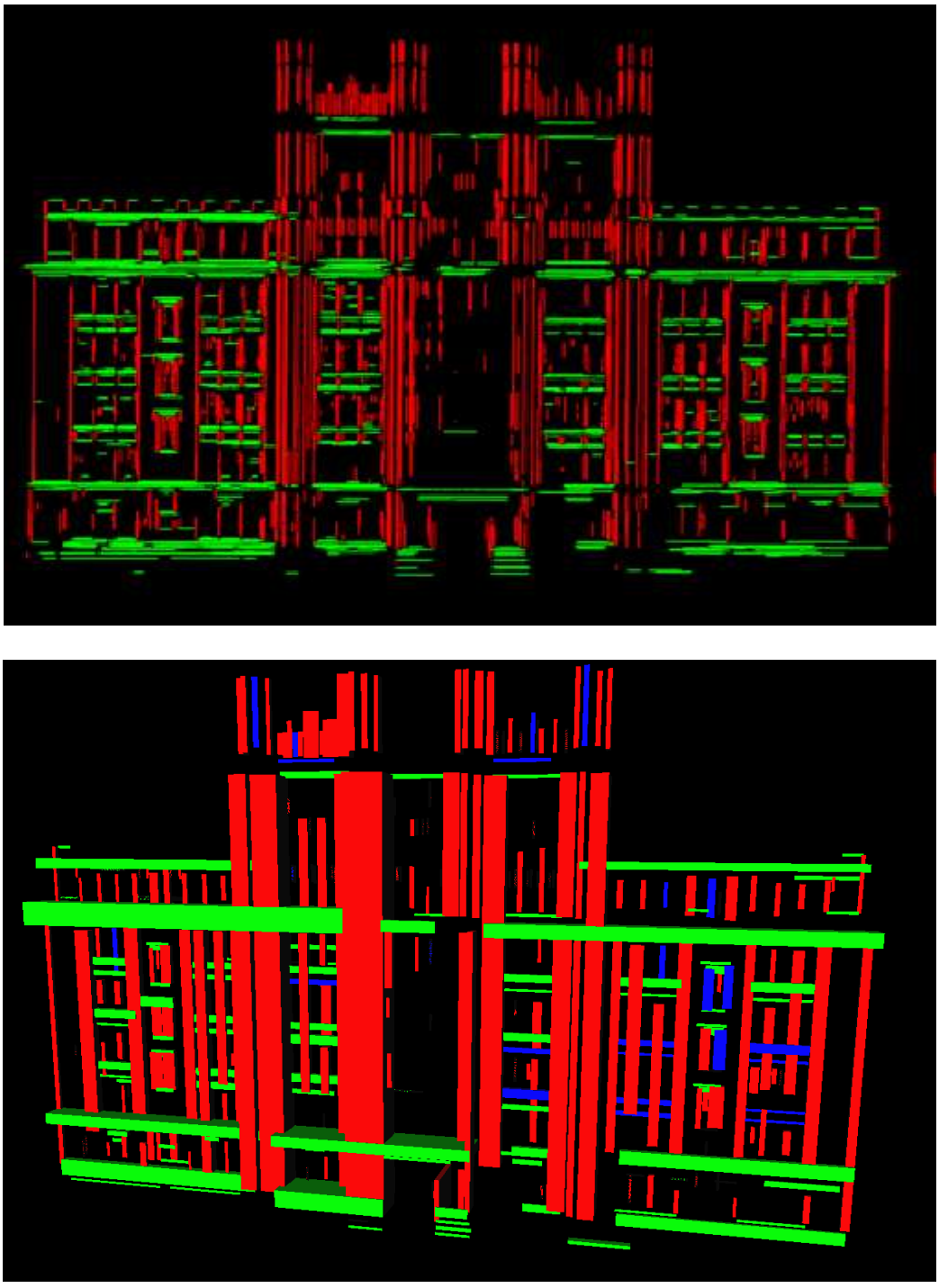

Figure 3.3: Thomas Hunter building at Hunter College of CUNY (building 1): examples of a 3D face (top) and 3D features (bottom).

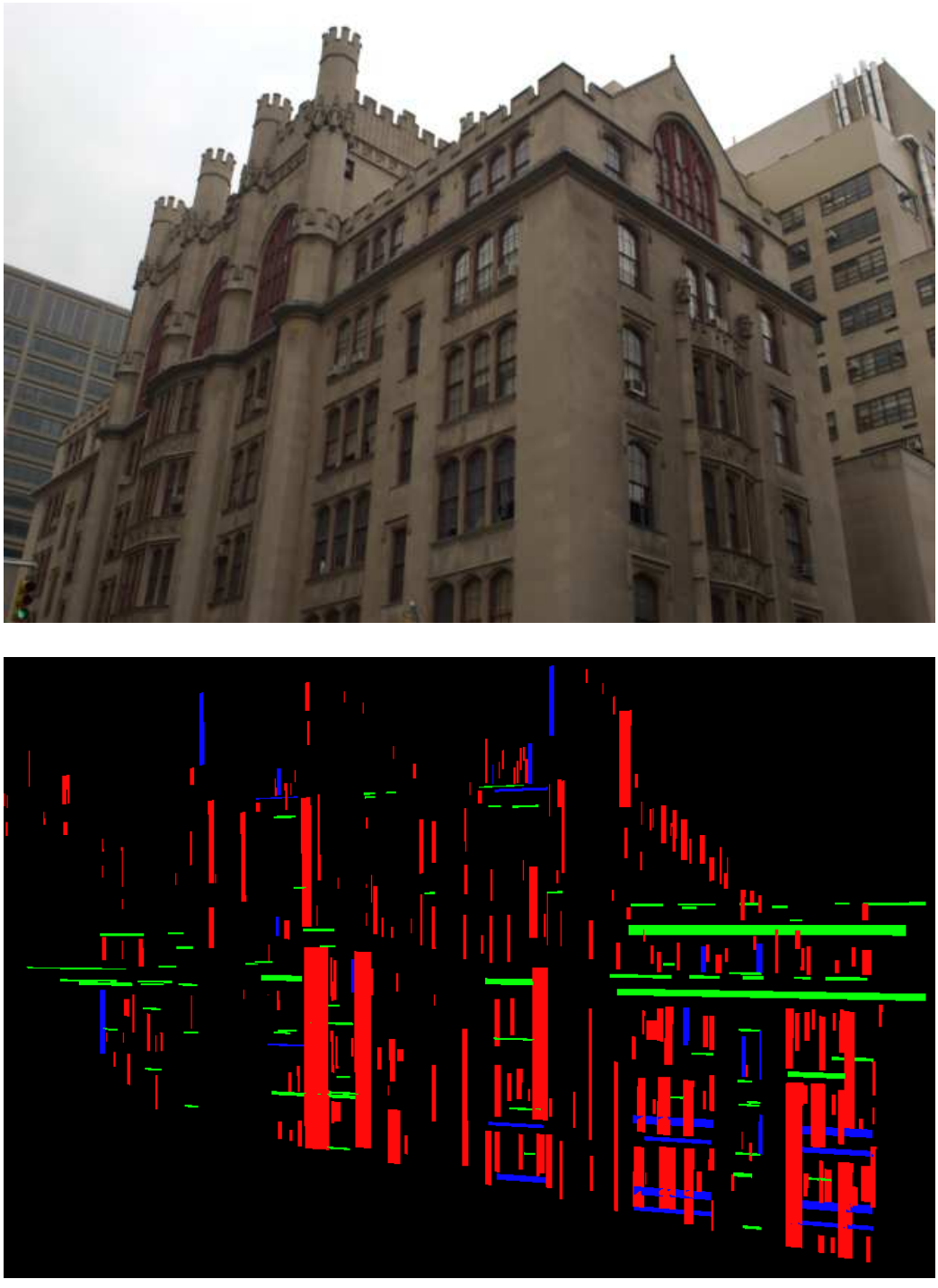

Figure 3.4: Thomas Hunter building at Hunter College of CUNY (building 1): examples of a 2D color image (top) and extracted 2D features (bottom).

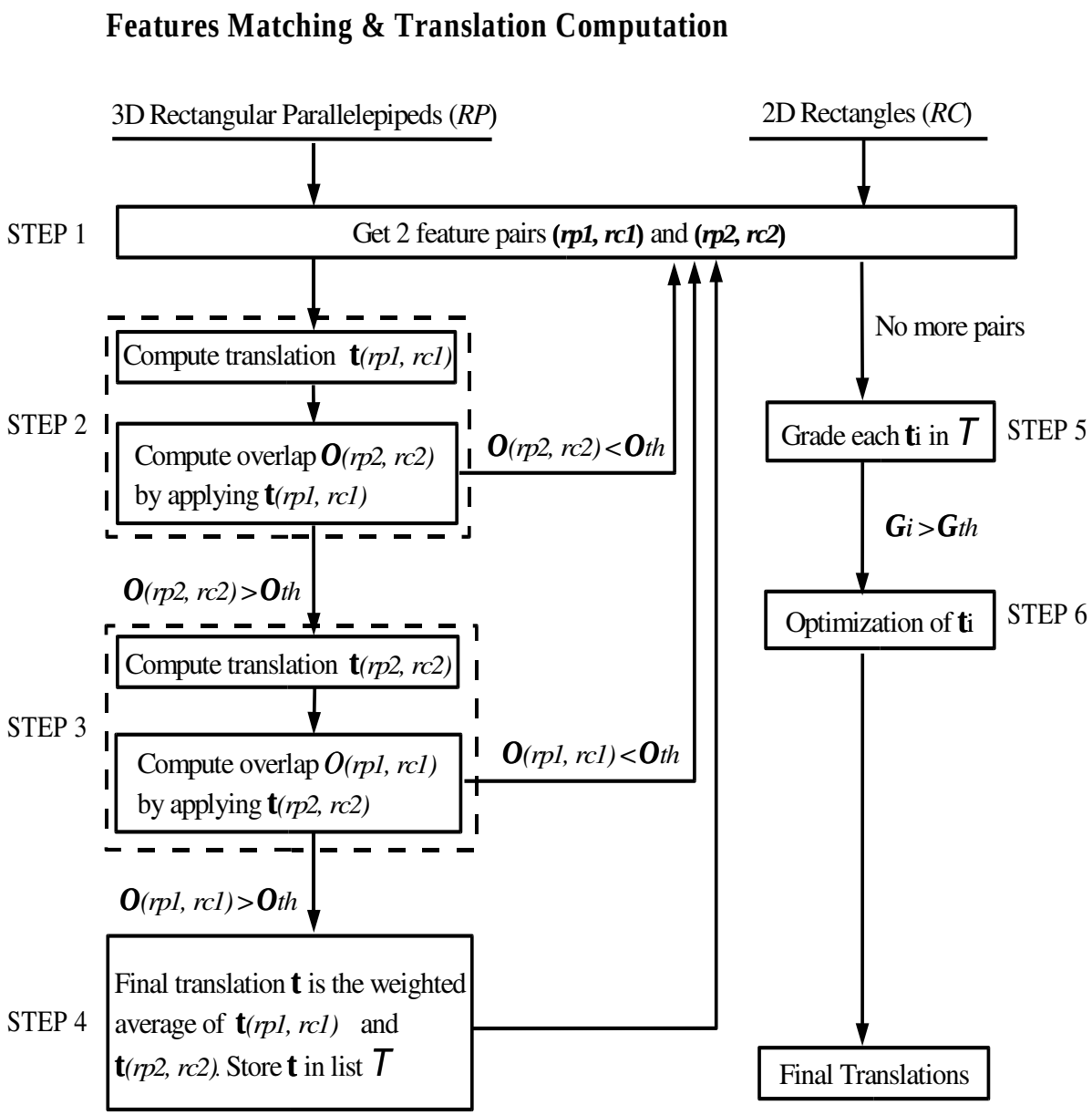

Figure 3.5: Translation computation algorithm outline.

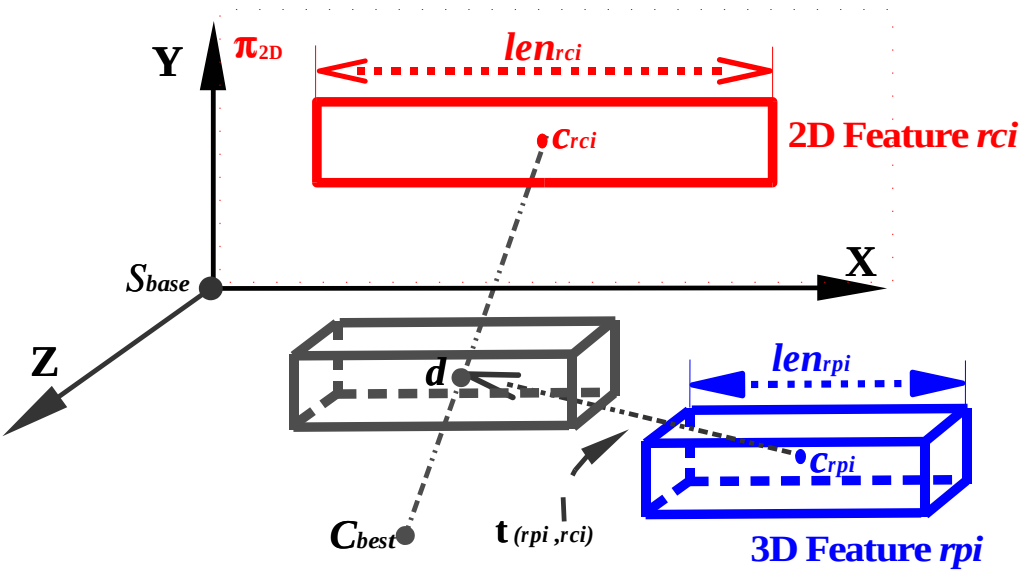

Figure 3.6: Translation computation.

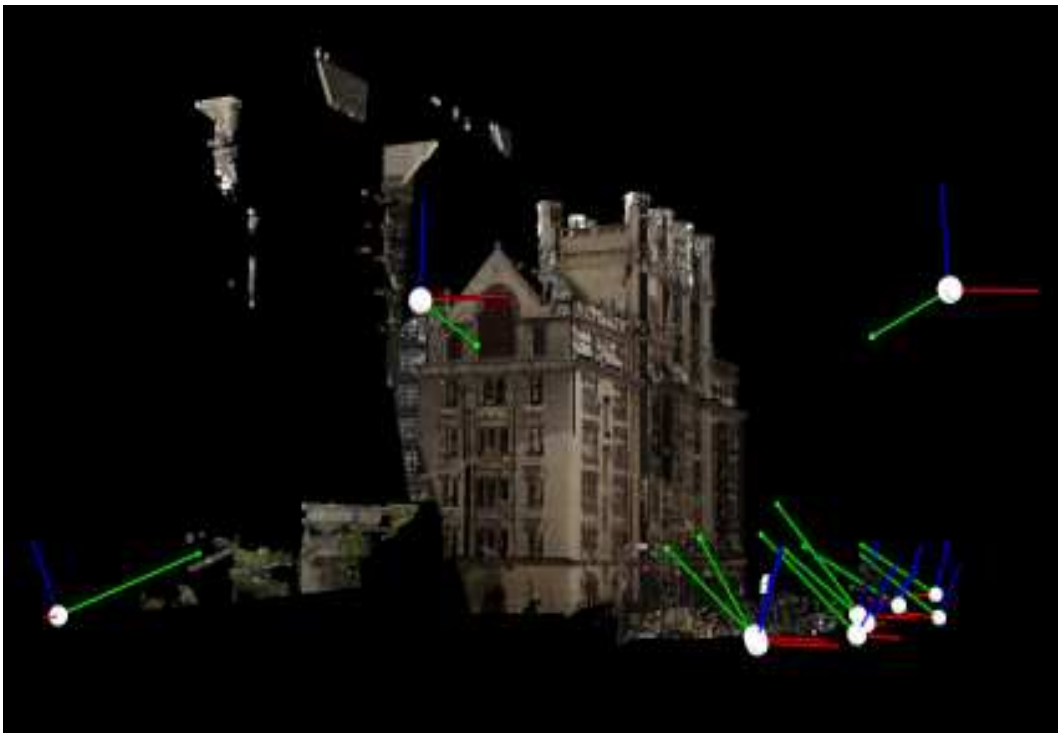

Figure 3.7: Camera configurations (11 cameras) with respect to texture-mapped 3D model of building 1 (Thomas Hunter building at Hunter College of CUNY).

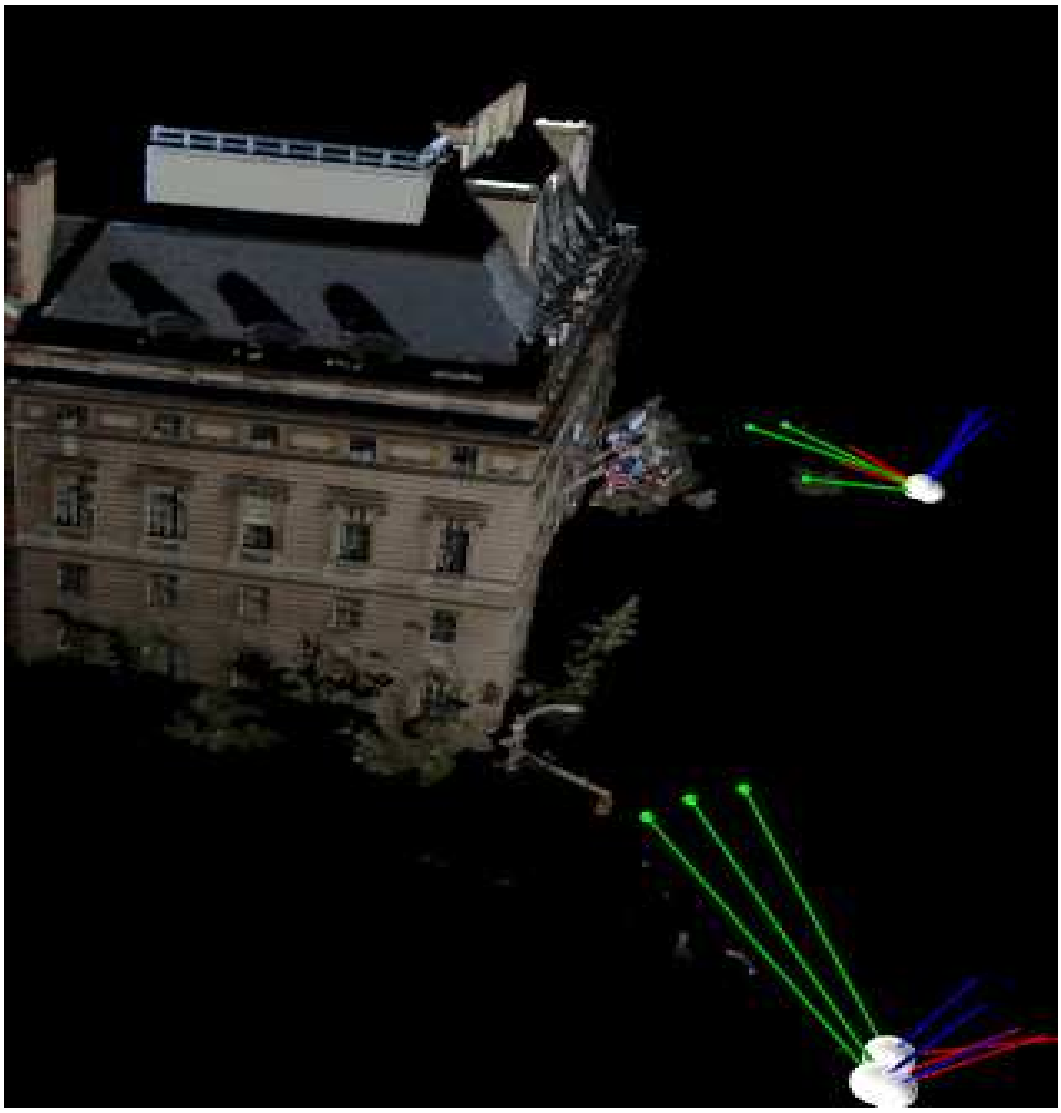

Figure 3.8: Camera configurations (6 cameras) with respect to texture-mapped 3D model of building 2 (corner of Park Avenue and 69th street, NYC).

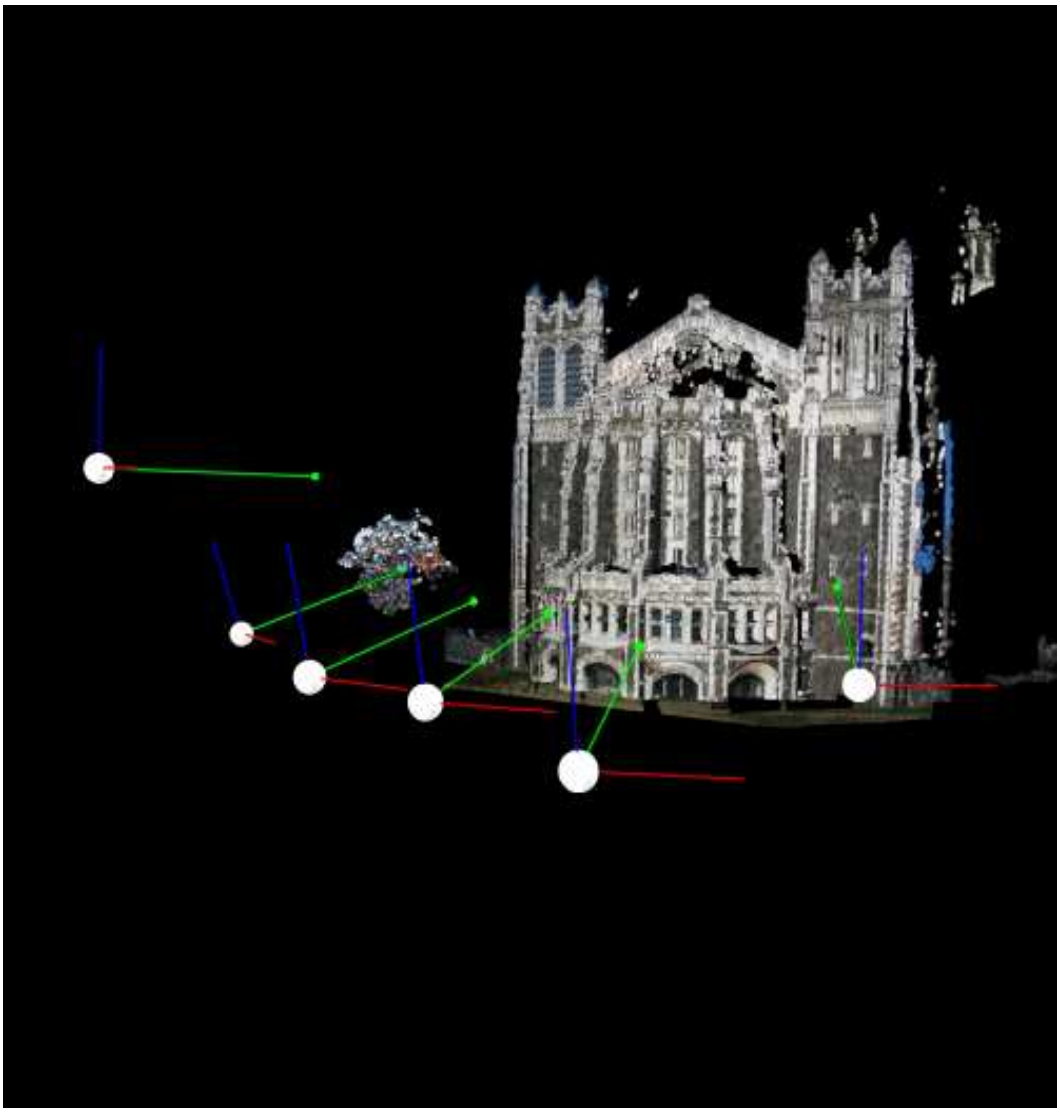

Figure 3.9: Camera configurations (6 cameras) with respect to texture-mapped 3D model of building 3 (Shepard Hall at CCNY, NYC).

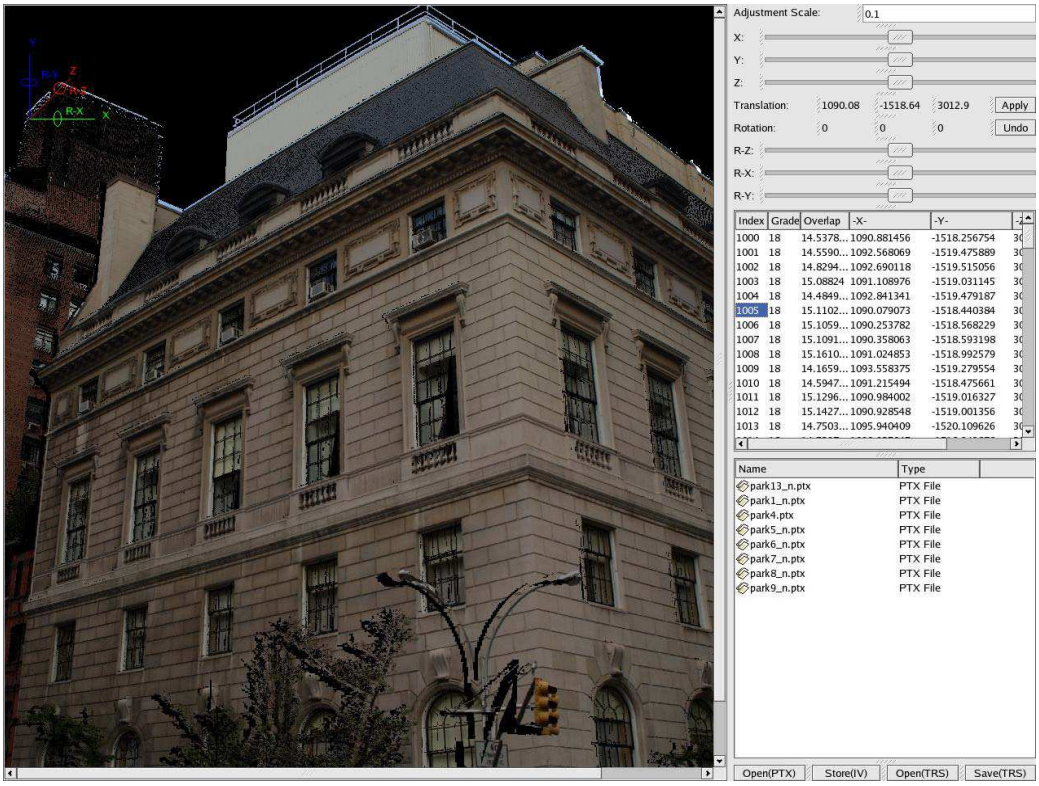

Figure 3.10: User interface for the translation selection. The final list of translations (obtained from the 6 steps in Sec. 3.4) are shown on the right panel. Those translations are sorted based on their grades. The user can click on an individual translation and the generated texture mapping result will be shown in the left window. By visually inspecting the generated texture mapping result, the user can easily select the best translation which produces the optimal texture mapping result.

Figure 3.11: Details of texture-maps for building 1 and 2.

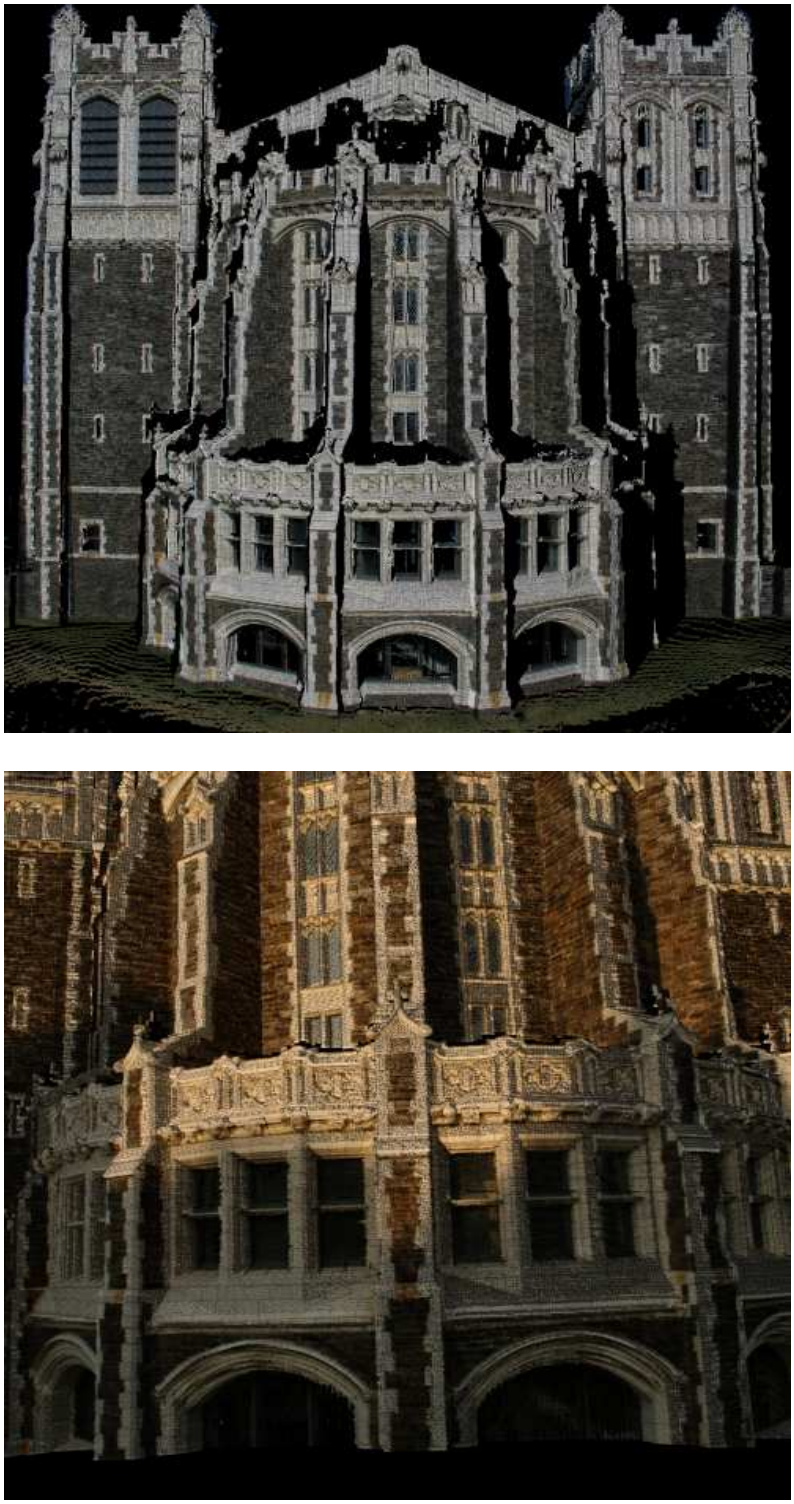

Figure 3.12: Details of texture-maps for building 3. Note the 2D images used for texture-mapping are taken at different time (top in the morning and bottom in the dawn) under quite different lighting conditions.

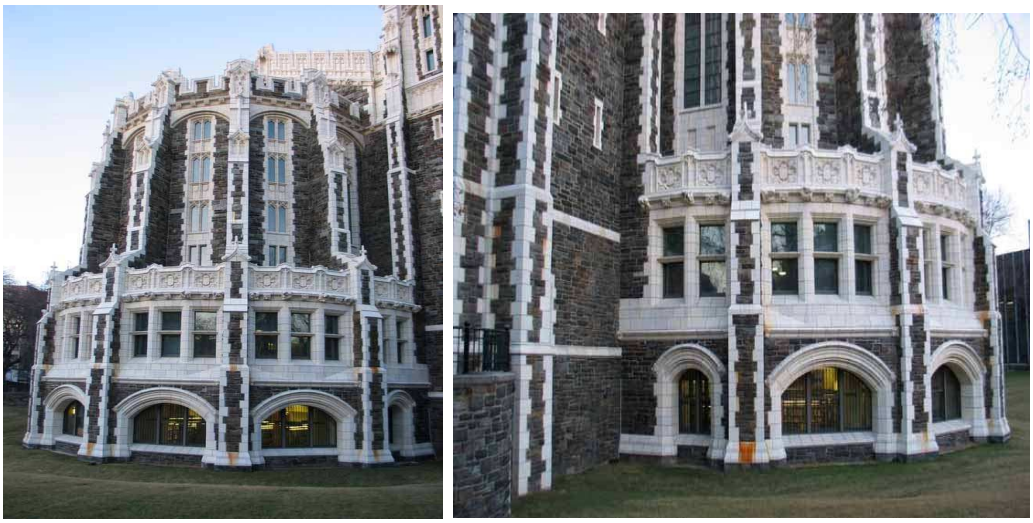

Figure 3.13: The algorithm of Chapter 3 cannot register these images with the 3D model of this building. This is due to the inability to extract two accurate orthogonal vanishing points. Note, that many not well-supported and inaccurate vanishing points can be extracted in the horizontal direction. However none of them is accurate enough due to the small number of lines that support them.

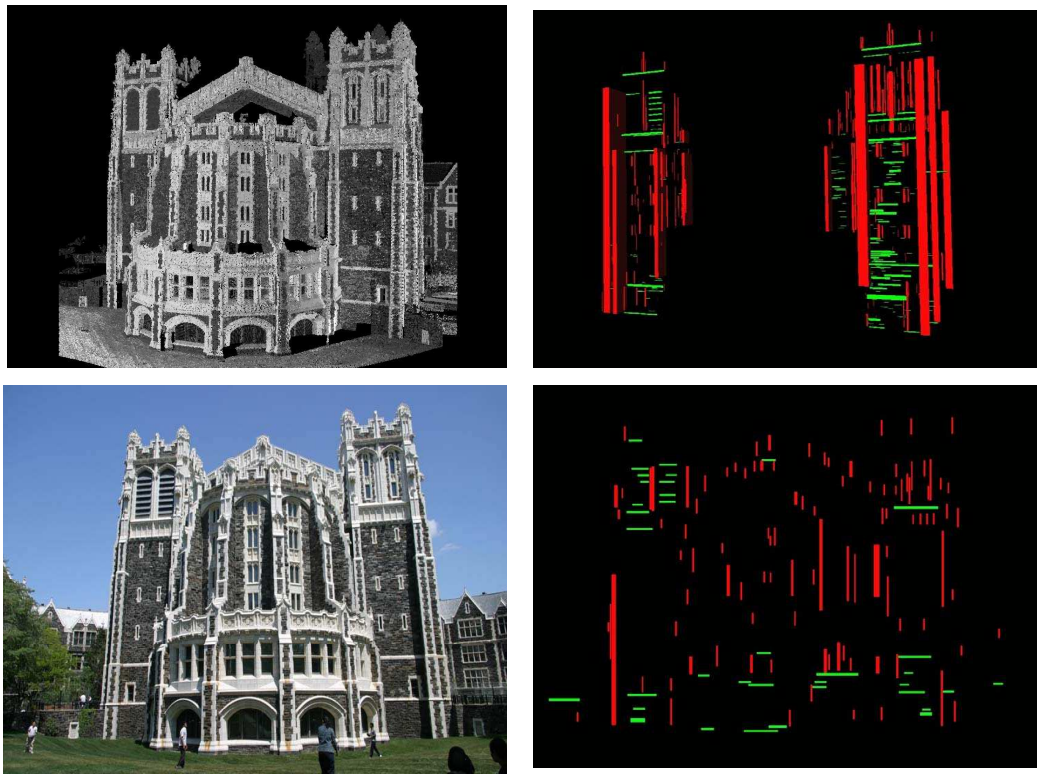

Figure 3.14: A successful image-to-range registration for Building 3. **top left**: 3D range data of Building 3. **top right**: extracted 3D rectangular parallelepipeds. **bottom left**: the 2D image of building 3, two towers are captured in the image. **bottom right**: extracted 2D rectangles. The abundance of linear structures in these two towers allows the extraction of sufficient 3D and 2D features. With adequate features, the system can register the 2D image with the 3D range data.

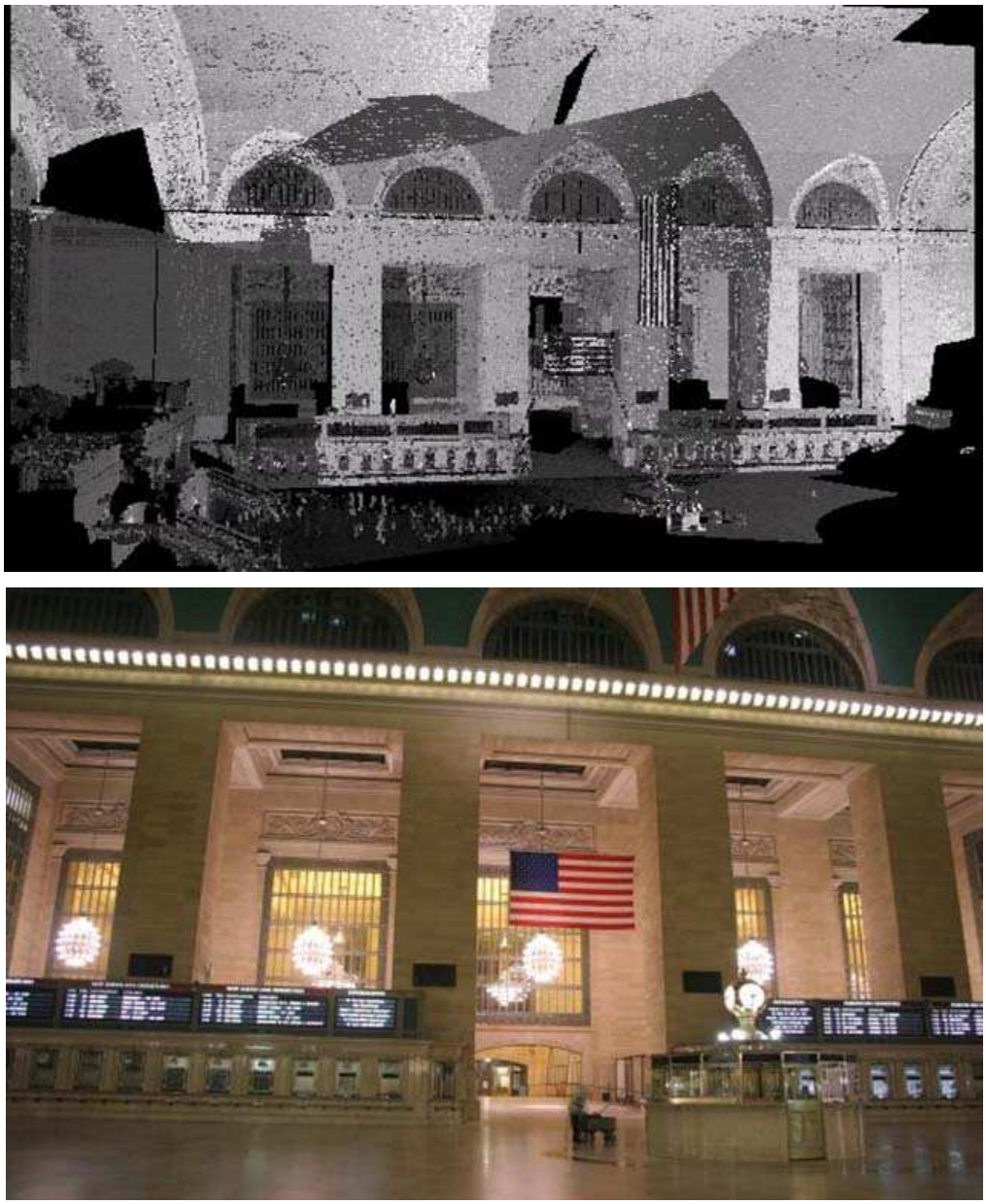

Figure 3.15: **Top**: 3D range model of Grand Central Station; **Bottom**: 2D image that captures the same part of the scene. The algorithm of Chapter 3 will fail in this case because the scene contains multiple layers of planar facades, such as cylinders, back walls, ticket-selling stand. Details are discussed in Sec. 3.6. The algorithm of Chapter 5 is able to address these limitations.
.

Chapter 4

Multiview Geometry for Texture Mapping 2D Images Onto 3D Range Data

Our image-to-range registration method described in Chapter 3 can register a single 2D image with the 3D range data at a time. In this chapter, a system that integrates multiview geometry with our automated image-to-range registration techniques for texture mapping a set of 2D images onto 3D range data is presented. This approach exploits all possible relationships between 2D images and 3D range data by combining image-to-range registration and structure-from-motion. Valuable information drawn from the overlapping part among the set of 2D images are utilized for the registration. Compared to

previous work [Stamos and Allen, 2001, Katsushi Ikeuchi *et al.*, 2003, Troccoli and Allen, 2004, Liu and Stamos, 2005], this approach provides a solution of increased robustness, efficiency and generality [Liu *et al.*, 2006].

4.1 Problem Formulation

Given a set of 2D images and 3D range data, our goal is to automatically register the whole set of 2D images with the 3D range data. The solution described in this chapter merges the benefits of multiview geometry with automated image-to-range registration to produce photorealistic models with minimal human interaction. The 3D range scans and the 2D photographs are respectively used to generate a pair of 3D models of the scene. The first model consists of a dense 3D point cloud, produced by using a range-to-range registration method that matches 3D lines in the range images to bring them into a common reference frame [Stamos and Leordeanu, 2003, Chen and Stamos, 2005]. The second model consists of a sparse 3D point cloud, produced by applying a multiview geometry (structure-from-motion) algorithm directly on a sequence of 2D photographs to simultaneously recover the camera motion and the 3D positions of image features. We developed a novel algorithm to automatically find the point correspondences between the sparse and dense models, and recover the similarity transformation (rotation/scale/translation) that best aligns these two models. This alignment is

necessary to enable the photographs to be optimally texture mapped onto the dense model. No a-priori knowledge about the camera poses relative to the 3D sensor's coordinate system is needed, other than the fact that at least one image frame should be registered with the 3D range data.

The framework of our system is shown in Fig. 4.1. An independent sequence of 2D images is gathered, taken from various viewpoints along a wide base-line. As the left branch of the framework indicates, a *subset* of the 2D images are automatically registered with the dense 3D range point cloud by the image-to-range registration method (Sec. 4.2). On the other hand, a sparse 3D point cloud is reconstructed from the whole collection of 2D images by using a structure-from-motion (SFM) algorithm (Sec. 4.3). As a result from this step, all the 2D images are registered with this reconstructed 3D point cloud in a local coordinate system. Finally, the *complete* set of 2D images is automatically aligned with the dense 3D range point cloud (Sec. 4.4) by combining the image-to-range registration results with the structure-from-motion results.

As described before (Chapter 2), a system [Zhao *et al.*, 2005] whose goals are very similar to ours is further discussed here. In that work, continuous video is aligned onto a 3D point cloud obtained from a 3D sensor. First, an SFM/stereo algorithm produces a 3D point cloud from the video sequence. This point cloud is then registered to the 3D point cloud acquired from the range scanner by applying the ICP algorithm [Besl and McKay, 1992]. One

limitation of this approach has to do with the shortcomings of the ICP algorithm. In particular, the 3D point clouds must be manually brought close to each to yield a good initial estimate that is required for the ICP algorithm to work. The ICP may fail in scenes with few discontinuities, such as those replete with planar or cylindrical structures. Also, in order for the ICP algorithm to work, a very dense model from the video sequence must be generated. This means that the method of [Zhao *et al.*, 2005] is restricted to video sequences, which limits the resolution of the 2D imagery. Finally, that method does not automatically compute the difference in scale between the range model and the recovered SFM/stereo model.

Our contributions can be summarized as follows:

- Like [Zhao *et al.*, 2005], we compute a model from a collection of images via SFM. Our method for aligning the range and SFM models, described in Sec. 4.4, does not rely on ICP and thus does not suffer from its limitations.

- We are able to automatically compute the scale difference between the range and SFM models.

- Like the method described in the previous chapter (Chapter 3), we perform image-to-range registration for a few (at least one) images of our collection. This feature-based method provides excellent results in the presence of a sufficient number of linear features. Therefore, the images

that contain enough linear features are registered using that method. The utilization of the SFM model allows us to align the remaining images with a method that involves robust point (and not line) correspondences.

- We generate an optimal texture mapping result by using contributions of all 2D images.

4.2 Image-to-Range Registrations

First, the 3D range model M_{range} is acquired in the same way as described in Chapter. 3. A set of range scans $R_m(m = 1, \ldots, M)$ that adequately covers the 3D scene is obtained by our laser range scanner. Through the automated feature-based range-to-range registration method of [Stamos and Leordeanu, 2003, Chen and Stamos, 2005], all the range scans are registered with respect to one selected pivot scan. The set of registered 3D points from the M scans is called M_{range} (Fig. 4.2(a) (top)).

The automated image-to-range registration method described in Chapter 3 is used for the automated calibration and registration of a single 2D image I_n with the 3D range data M_{range}. With this method, a few 2D images can be independently registered with the model M_{range}. The algorithm will fail to produce satisfactory results in parts of the scene where there is a lack of 2D and 3D features for matching. Also, since each 2D image is independently reg-

istered with the 3D model, valuable information that can be extracted from relationships between the 2D images (SFM) is not utilized. In order to solve the aforementioned problems, an SFM module (Sec. 4.3) and final alignment module (Sec. 4.4) has been added into the system. These two modules increase the robustness of the reconstructed model, and improve the accuracy of the final texture mapping results. Therefore, the image-to-range registration algorithm is used in order to register a few 2D images (five shown in Fig. 4.2(a) (top)) that produce results of high quality. The final registration of the 2D image sequence with the range data M_{range} is performed after SFM is utilized (Sec. 4.3).

4.3 Multiview pose estimation and 3D structure reconstruction

This part of work is a contribution from Prof. George Wolberg's group [Liu *et al.*, 2006]. The input to the system is a sequence $\mathbf{I} = \{I_n | n = 1, \ldots, N\}$ of high resolution still images that capture the 3D scene. This is necessary to produce photorealistic scene representations. Therefore we have to attack the problem of finding correspondences in a sequence of wide-baseline high resolution images, a problem that is much harder than feature tracking from a video sequence. Fortunately, there are several recent approaches that at-

tack the wide-baseline matching problem [Schaffalitzky and Zisserman, 2001, Tuytelaars and Gool, 2004, Lowe, 2004]. For the purposes of our system, Wolberg and Zokai have adopted the scale-invariant feature transform (SIFT) method [Lowe, 2004] for pairwise feature extraction and matching. In general, structure-from-motion (SFM) from a set images has been rigorously studied [Faugeras *et al.*, 2001, Hartley and Zisserman, 2004, Ma *et al.*, 2003, Beardsley *et al.*, 1997, Pollefeys *et al.*, 2004]. Here we use the method developed by Wolberg and Zokai [Liu *et al.*, 2006] for the structure-from-motion computation. Fig. 4.3 shows the recovered camera poses and the reconstructed 3D point cloud of Shepherd Hall (CCNY).

4.4 Alignment of 2D Image Sequences Onto 3D-Range Point Clouds

The dense 3D range model, which is captured by range scanner and generated by range-to-range registration [Stamos and Leordeanu, 2003, Chen and Stamos, 2005], is called M_{range}. On the other hand, the sequence of 2D images $\mathbf{I} = \{I_n | n = 1, \ldots, N\}$ produces a sparser 3D model of the scene (Sec. 4.3) called M_{sfm}. Both of these models are represented as clouds of 3D points. The distance between any two points in M_{range} corresponds to the actual distance of the points in 3D space, whereas the distance of any two points in M_{sfm} is

the actual distance multiplied by an unknown scale factor s. In order to align the two models, a similarity transformation that includes the scale factor s, a rotation R and a translation T needs to be computed. In this section, a novel algorithm that automatically computes this transformation is presented. The transformation allows for the optimal texture mapping of all images onto the dense M_{range} model, and thus provides photorealistic results of high quality.

Every point X from M_{sfm} can be projected onto a 2D image $I_n \in \mathbf{I}$ by the following transformation:

$$\mathbf{x} = \mathcal{K}_n[R_n \,|\, T_n\,]\,\mathbf{X} \tag{4.1}$$

where $\mathbf{x} = (x, y, 1)$ is a pixel on image I_n, $\mathbf{X} = (X, Y, Z, 1)$ is a point of M_{sfm}, $\mathcal{K}_n$ is the projection matrix, R_n is the rotation transformation and T_n is the translation vector. These matrices and points $\mathbf{X}$ are computed by the SFM method (Sec. 4.3).

Some of the 2D images $\mathbf{I}' \subset \mathbf{I}$ are also automatically registered with the 3D range model M_{range} (Sec. 4.2). Thus, each point of M_{range} can be projected onto each 2D image $I_n \in \mathbf{I}'$ by the following transformation:

$$\mathbf{y} = \mathcal{K}_n[{R'}_n \,|\, {T'}_n\,]\,\mathbf{Y} \tag{4.2}$$

where $\mathbf{y} = (x, y, 1)$ is a pixel in image I_n, $\mathbf{Y} = (X, Y, Z, 1)$ is a point of model M_{range}, $\mathcal{K}_n$ is the projection matrix of I_n, ${R'}_n$ is the rotation and ${T'}_n$ is the translation. These transformations are computed by the image-to-range registration method (Sec. 4.2).

The key idea is to use the images in $I_n \in \mathbf{I}'$ as references in order to find the corresponding points between M_{range} and M_{sfm}. The similarity transformation between M_{range} and M_{sfm} is then computed based on these correspondences. In summary, the algorithm works as follows:

1. Each point of M_{sfm} is projected onto $I_n \in \mathbf{I}'$ using Eq. (4.1). Each pixel $p_{(i,j)}$ of I_n is associated with the closest projected point $\mathbf{X} \in M_{sfm}$ in an $L \times L$ neighborhood on the image. Each point of M_{range} is also projected onto I_n using Eq. (4.2). Similarly, each pixel $p_{(i,j)}$ is associated with the projected point $\mathbf{Y} \in M_{range}$ in an $L \times L$ neighborhood (Fig. 4.4). Z-buffering is used to handle occlusions.

2. If a pixel $p_{(i,j)}$ of image I_n is associated with a pair of 3D points $(\mathbf{X}, \mathbf{Y})$, one from M_{sfm} and the other from M_{range}, then these two 3D points are considered as candidate matches. Thus, for each 2D-image in $\mathbf{I}'$ a set of matches is computed, producing a collection of candidate matches named $\mathbf{L}$. These 3D-3D correspondences between points of M_{range} and points of M_{sfm} could be potentially used for the computation of the similarity transformation between the two models. The set $\mathbf{L}$ contains many outliers, due to the very simple closest-point algorithm utilized. However, $\mathbf{L}$ can be further refined (Sec. 4.4.1) into a set of robust 3D point correspondences $\mathcal{C} \subset \mathbf{L}$.

3. Finally, the transformation between M_{range} and M_{sfm} is computed by

minimizing a weighted error function E (Sec. 4.4.1) based on the final robust set of correspondences $\mathcal{C}$.

4.4.1 Correspondence Refinement and Optimization

The set of candidate matches $\mathbf{L}$ computed in the second step of the previous algorithm contains outliers due to errors introduced from the various modules of the system (SFM, image-to-range registration, range sensing). It is thus important to filter out as many outliers as possible through verification procedures. A natural verification procedure involves the difference in scale between the two models. Consider two pairs of plausible matched 3D-points $(\mathbf{X_1}, \mathbf{Y_1})$ and $(\mathbf{X_2}, \mathbf{Y_2})$ ($\mathbf{X_i}$ denotes points from the M_{sfm} model, while $\mathbf{Y_j}$ points from the the M_{range} model). If these were indeed correct correspondences, then the scale factor between between the two models would be $s = \|\mathbf{X_1} - \mathbf{X_2}\|/\|\mathbf{Y_1} - \mathbf{Y_2}\|$. Since the computed scale factor should be the same no matter which correct matching pair is used, then a robust set of correspondences from $\mathbf{L}$ should contain only these pairs that produce the same scale factor s. The constant scale factor among correctly picked pairs is thus an invariant feature that we exploit. We now explain how we achieve this robust set of correspondences.

For each image $I_n \in \mathbf{I}'$, let us call the camera's center of projection as $\mathbf{C_n^{sfm}}$

in the local coordinate system of M_{sfm} and $\mathbf{C_n^{rng}}$ in the coordinate system of M_{range}. These two centers have been computed from two independent processes: SFM (Sec. 4.3) and image-to-range registration (Sec. 4.2). Then for any candidate match, $(\mathbf{X}, \mathbf{Y}) \in \mathbf{L}$, a candidate scale factor $s_1(\mathbf{X}, \mathbf{Y})$ can be computed as:

$$s_1(\mathbf{X}, \mathbf{Y}) = \frac{\|\mathbf{X} - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y} - \mathbf{C_n^{rng}}\|}$$

If we keep the match $(\mathbf{X}, \mathbf{Y})$ fixed and we consider every other match $(\mathbf{X}', \mathbf{Y}') \in \mathbf{L}$, $L-1$ candidate scale factors $s_2(\mathbf{X}', \mathbf{Y}')$ and $L-1$ candidate scale factors $s_3(\mathbf{X}', \mathbf{Y}')$ (L is the number of matches in $\mathbf{L}$) are computed as:

$$s_2(\mathbf{X}', \mathbf{Y}') = \frac{\|\mathbf{X}' - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y}' - \mathbf{C_n^{rng}}\|}, s_3(\mathbf{X}', \mathbf{Y}') = \frac{\|\mathbf{X} - \mathbf{X}'\|}{\|\mathbf{Y} - \mathbf{Y}'\|}$$

That means that if we keep the match $(\mathbf{X}, \mathbf{Y})$ fixed, and consider all other matches $(\mathbf{X}', \mathbf{Y}')$ we can compute a triple of candidate scale factors: $s_1(\mathbf{X}, \mathbf{Y})$, $s_2(\mathbf{X}', \mathbf{Y}')$, and $s_3(\mathbf{X}', \mathbf{Y}')$. We then consider the two pairs of matches $(\mathbf{X}, \mathbf{Y})$ and $(\mathbf{X}', \mathbf{Y}')$ as *compatible* if the scale factors in the above triple are close with respect to each other. By fixing $(\mathbf{X}, \mathbf{Y})$, all matches that are compatible with it are found. The confidence in the match $(\mathbf{X}, \mathbf{Y})$ is the number of compatible matches it has. By going through all matches in $\mathbf{L}$, their confidence is computed via the above procedure. Out of these matches the one with the highest confidence is selected as the most prominent: $(\mathbf{X_p}, \mathbf{Y_p})$. Let us call $\mathbf{L_n}$ the set that contains $(\mathbf{X_p}, \mathbf{Y_p})$ and all other matches that are compatible with it. Note that this set is based on the centers of projection of image I_n as

computed by SFM and image-to-range registration. Let us also call s_n the scale factor that corresponds to the set $\mathbf{L_n}$. This scale factor can be computed by averaging the triples of scale factors of the elements in $\mathbf{L_n}$. Finally a different set $\mathbf{L_n}$ and scale factor s_n is computed for every image $I_n \in \mathbf{I'}$.

From the previous discussion it is clear that each $\mathbf{L_n}$ is a set of matches that is based on the center of projection of each image I_n independently. A set of matches that will provide a globally optimal solution should consider all images of $\mathbf{I'}$ simultaneously. Out of the scale factors computed from each set $\mathbf{L_n}$, the one that corresponds to the largest number of matches is the one more robustly extracted by the above procedure. That computed scale factor, s_{opt}, is used as the final filtration for the production of the robust set of matches $\mathcal{C}$ out of $\mathbf{L}$. In particular, for each candidate match $(\mathbf{X}, \mathbf{Y}) \in \mathbf{L}$, a set of scale factors are computed as

$$s'_n = \frac{\|\mathbf{X} - \mathbf{C_n^{sfm}}\|}{\|\mathbf{Y} - \mathbf{C_n^{rng}}\|}$$

where $n = 1, 2, ..., K$, and K is the number of images in $\mathbf{I'}$. The standard deviation of those scale factors with respect to s_{opt} is computed and if it is smaller than a user-defined threshold, $(\mathbf{X}, \mathbf{Y})$ is considered as a *robust* match and is added to the final list of correspondences $\mathcal{C}$. The robustness of the match stems from the fact that it verifies the robustly extracted scale factor s_{opt} with respect to most (or all) images $I_n \in \mathbf{I'}$. The pairs of center of projections $(\mathbf{C_n^{sfm}}, \mathbf{C_n^{rng}})$ of images in $\mathbf{I'}$ are also added to $\mathcal{C}$.

The list $\mathcal{C}$ contains robust 3D point correspondences that are used for the accurate computation of the similarity transformation (scale factor s, rotation R, and translation T) between the models M_{range} and M_{sfm}. The following weighted error function is minimized with respect to sR and T:

$$E = \sum_{(\mathbf{X},\mathbf{Y})\in\mathcal{C}} w\|sR \cdot \mathbf{Y} + T - \mathbf{X}\|^2$$

where the weight $w = 1$ for all $(\mathbf{X}, \mathbf{Y}) \in \mathcal{C}$ that are not the centers of projection of the cameras, and $w > 1$ (user-defined) when $(\mathbf{X}, \mathbf{Y}) = (\mathbf{C_n^{sfm}}, \mathbf{C_n^{rng}})$. By associating higher weights to the centers we exploit the fact that we are confident in the original pose produced by SFM and image-to-range registration. The unknown sR and T are estimated by computing the least square solution from this error function. Note that s can be easily extracted from sR since the determinant of R is 1.

In summary, by utilizing the invariance of the scale factor between corresponding points in M_{range} and M_{sfm}, a set of robust 3D point correspondences $\mathcal{C}$ is computed. These 3D point correspondences are then used for an optimal calculation of the similarity transformation between the two point clouds. This provides a very accurate texture mapping result of the high resolution images onto the dense range model M_{range}.

4.5 Results

We tested our algorithms using range scans and 2D images acquired from a large-scale urban structure (Shepard Hall/CCNY) and from an interior scene (Great Hall/CCNY). 22 range scans of the exterior of Shepard Hall were automatically registered (Fig. 4.2) to produce a dense model M_{range}. In one experiment, ten images where gathered under the same lighting conditions(Fig. 4.8). All ten of them were independently registered (image-to-range registration Sec. 4.2) with the model M_{range}. The registration was optimized with the incorporation of the SFM model (Sec. 4.3) and the final optimization method (Sec. 4.4). In a second experiment, 22 images of Shepard Hall that covered a wider area were acquired. Although the automated image-to-range registration method was applied to all the images, only five of them were manually selected for the final transformation (Sec. 4.4) on the basis of visual accuracy (Fig. 4.6). For some of the 22 images the automated image-to-range method could not be applied due to lack of sufficient vanishing points and linear features (Chapter 3). However, all 22 images where optimally registered using our novel registration method (Sec. 4.4) after the SFM computation (Sec. 4.3). Fig. 4.2 (bottom) shows the alignment of the range and SFM models achieved through the use of the 2D images. In Fig. 4.7 the accuracy of the texture mapping method is visible. Fig. 4.9 displays the texture mapping result of an interior 3D scene. It should be noted that the image-to-range registration

(Sec. 4.2) fails to produce satisfactory results on all 7 images. This is because of the low quality 2D/3D features extracted from both 3D range data and 2D images. The low quality features are due to the fact that the dominate part of the scene is a big spherical structure and the linear features are very limited. We managed to manually register one image with the 3D range data. Combining with the SFM results (Sec. 4.4), all 7 images are registered with the range data.

We have visually verified that the SFM integration improves the texture-mapping results. In Fig. 4.5, texture mapping result after independent image-to-range registration (Sec. 4.2) and optimized texture mapping result after the integration of SFM (Sec. 4.4) are compared. Notice that the highlighted mis-registration region on the left (blue pixels due to the sky) vanishes after the integration of SFM (on the right).

Fig. 4.10 shows the easy-to-use interface for the integration. The left window shows the projection of M_{range} on a 2D image. The right window shows the projection of M_{sfm} on the same 2D image.

Table 4.1 provides some quantitative results of our experiments. Notice the density of the range models versus the sparsity of the SFM models. Also notice the number of robust matches in $\mathcal{C}$ (Sec. 4.4) with respect to the possible number of matches (i.e., number of points in SFM). The final row Table 4.1 displays the elapsed time for the final optimization on a Dell PC running Linux

	Shepard Hall		Great Hall
Number of points (M_{range})	12,483,568		13,234,532
Number of points (M_{sfm})	2,034	45,392	1,655
2D-images used	10	22	7
image-to-range registrations (Sec. 4.2)	10	5	3
No. of matches in $\mathcal{C}$ (Sec. 4.4)	258	1632	156
Final optimization (Sec. 4.4)	8.65 s	19.20 s	3.18 s

Table 4.1: Quantitative results.

on an Intel Xeon-2GHz, 2GB-RAM machine.

4.6 Discussion

We have presented a system that integrates multiview geometry and automated 3D registration techniques for texture mapping high resolution 2D images onto dense 3D range data. The benefits of multiview geometry (SFM) and automated image-to-range registration are merged for the production of photorealistic models with minimal human interaction. Our approach provides a solution of increased robustness, efficiency and generality with respect to previous methods. It should be noted the set of images taken under various lighting conditions may raise problem for the structure-from-motion (SFM), since the image-to-image registration utilizing SIFT features implemented in SFM is based on the color constancy assumption. In addition, this algorithm bears the same limitations of image-to-range registration method since at least one image is required to be registered with the 3D range data. If none of the 2D images in the set can provide at least two major vanishing points, then

no image can be registered with the 3D range data by the image-to-range registration. This will cause the integration to fail. Additional discussion on image-to-range registration can be found in Chapters 3 and 5.

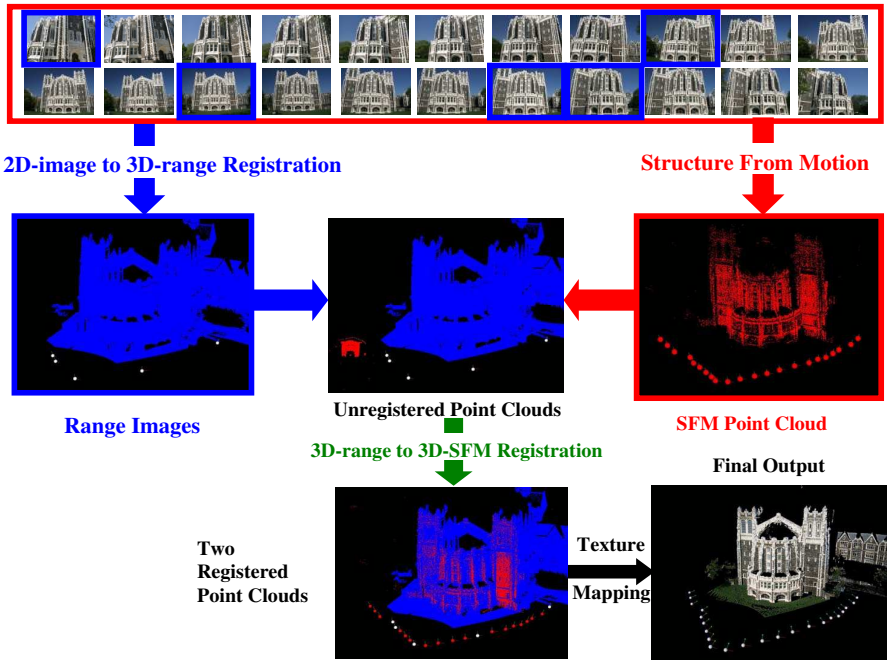

Figure 4.1: On the **left** branch (highlighted in blue): five color images out of 22 are automatically registered with the dense 3D point cloud M_{range} acquired from the range scanner (Sec. 4.2). The recovered camera positions are shown in white dots. On the **right** branch (highlighted in red): the set of 22 color images are registered together and a sparse 3D point cloud M_{sfm} is reconstructed from them by using a structure-from-motion (SFM) algorithm (Sec. 4.3). The camera positions (red dots) are shown with respect to the reconstructed 3D point cloud. Then, the sparse 3D point cloud M_{sfm} is registered with the dense 3D point cloud M_{range} (Sec. 4.4). This step brings all the 2D images into alignment with M_{range} . As a result, the whole collection of color images are texture mapped onto the 3D range model M_{range}.

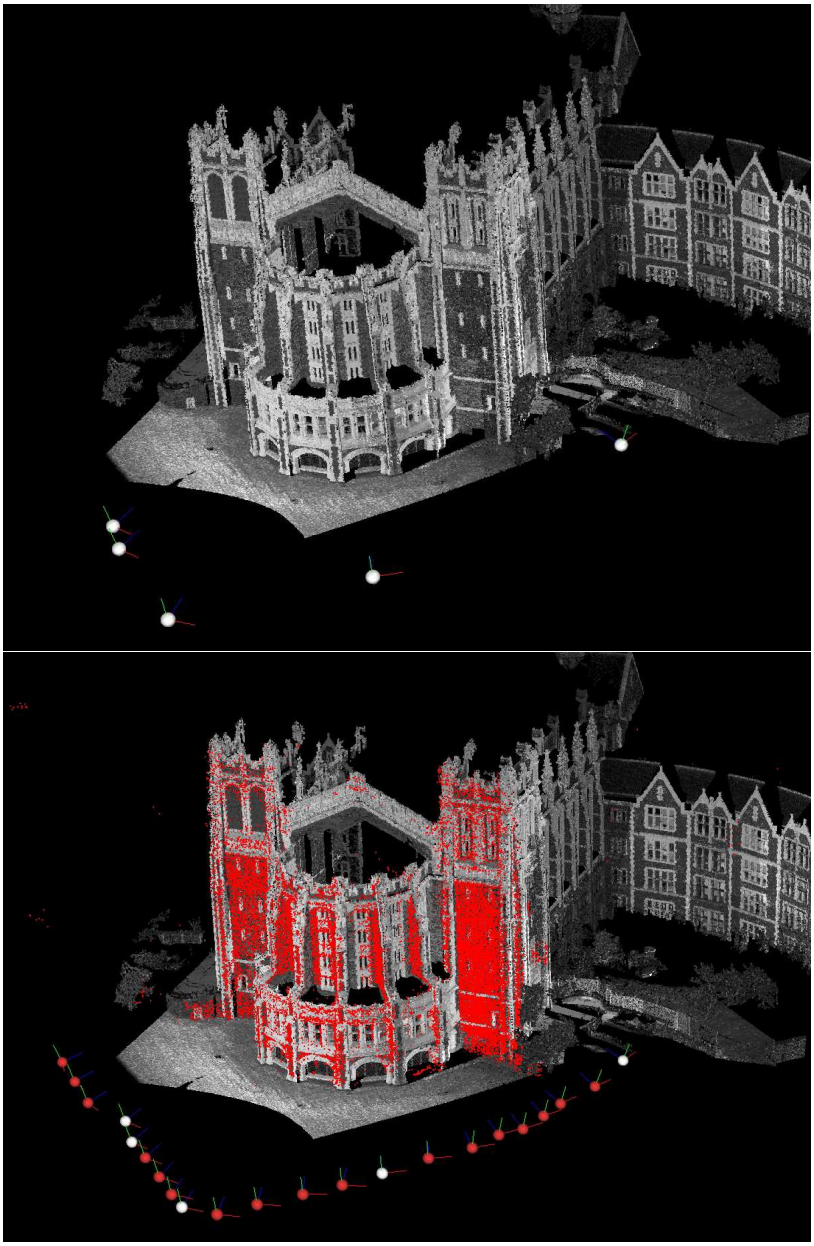

Figure 4.2: **(top)**: 3D range model M_{range} with five recovered 2D camera locations (white dots) produced by a image-to-range registration algorithm (Sec. 4.2). **(bottom)**: The 3D range model M_{range} overlaid with the 3D model M_{sfm} produced by SFM (Sec. 4.3) after the alignment method of Sec. 4.4. The points of M_{sfm} are shown in red, and the sequence of 2D images that produced M_{sfm} are shown as red dots in the figure. Their positions have been accurately recovered with respect to both models M_{range} and M_{sfm} (Sec. 4.4).

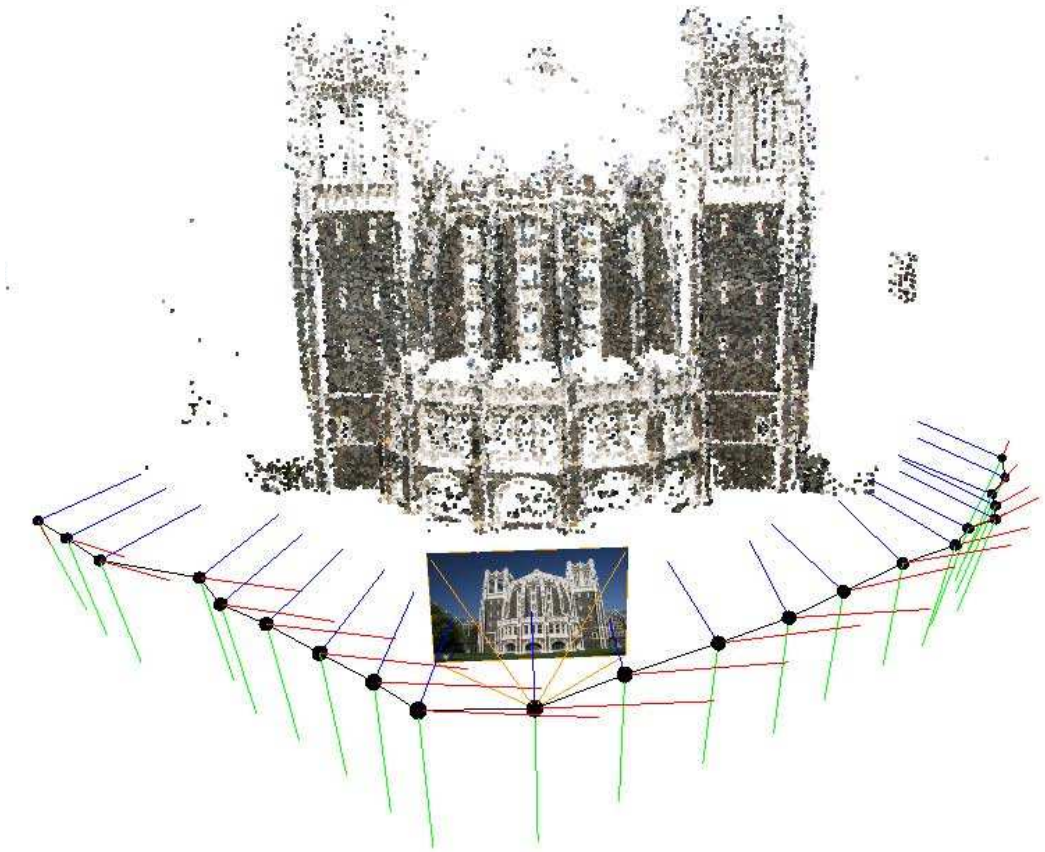

Figure 4.3: The path of camera motion (22 images used) and reconstruction of Shepherd Hall (CCNY) using SFM(see Sec. 4.3). This algorithm was designed and implemented by Wolberg and Zokai [Liu *et al.*, 2006].

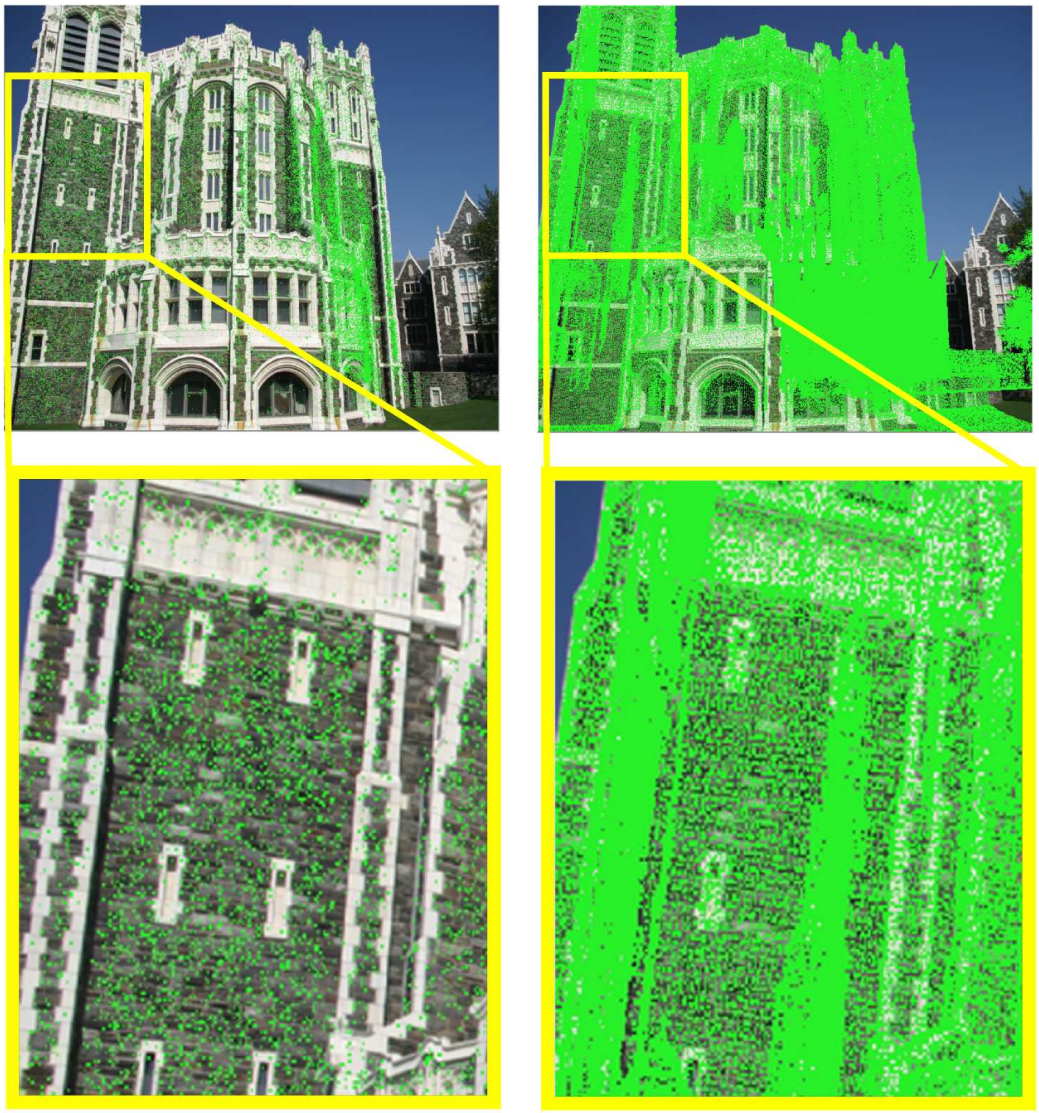

Figure 4.4: **(Left)**: The points of model M_{sfm} projected onto one 2D image I_n (Sec. 4.3). The projected points are shown in green. **(Right)**: The points of model M_{range} projected onto the same 2D image I_n (projected points shown in green) after the automatic image-to-range registration (Sec. 4.2). Note that the density of 3D range points is much higher than the density of the SFM points, due to the different nature of the two reconstruction processes. Finding corresponding points between M_{range} and M_{sfm} is possible on the 2D image space of I_n. This yields the transformation between the two models (Sec. 4.4).

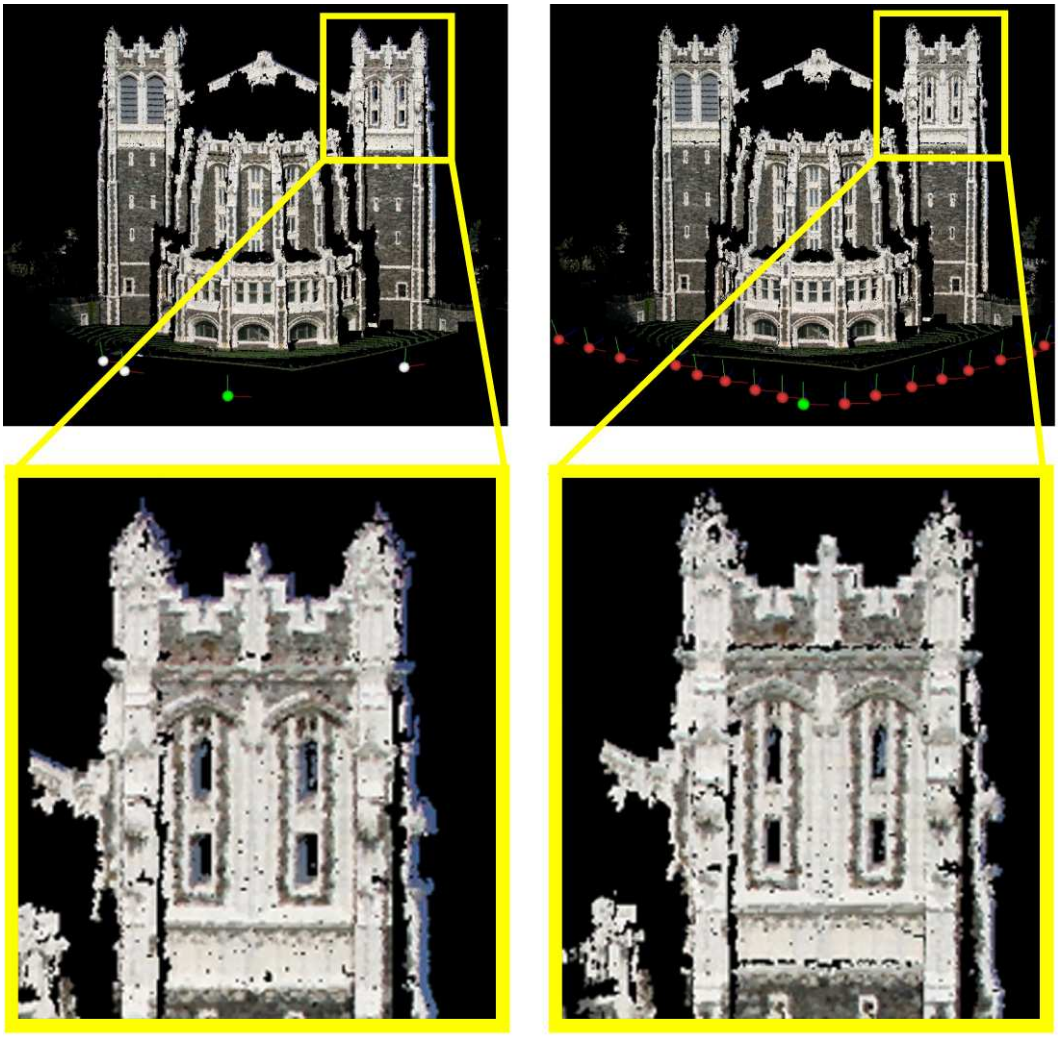

Figure 4.5: **(Left)**: Texture mapping result after independent image-to-range registration (Sec. 4.2) of one image with the model M_{range}. The location of the image is shown as a green dot. Notice that after magnifying the upper-right corner of the image, a mis-registration is evident; blue pixels due to the sky are mapped on the model. **(Right)**: Texture mapping result after all images have been considered (Sec. 4.3) and the optimized alignment of Sec. 4.4 performed. Notice that the mis-registration has vanished.

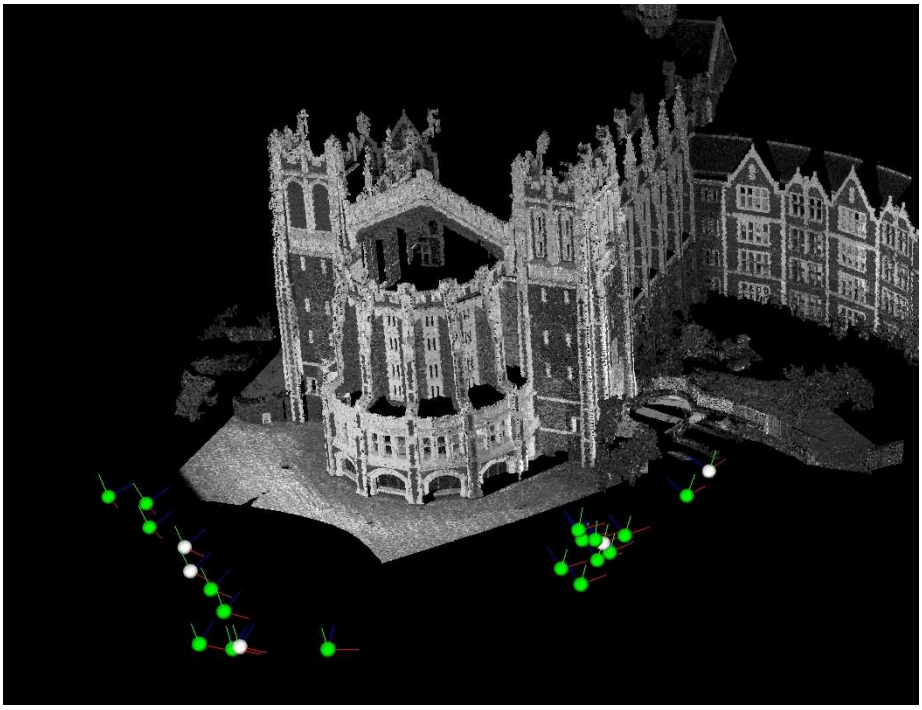

Figure 4.6: We applied our image-to-range registration method (Sec. 4.2) for the registration of all 22 images with the 3D range model. Five of them are successfully registered (white dots). For all other images, the image-to-range registration system fails to produce satisfactory results (green dots). For the reasons of this failure please refer to Sec. 3.6 of Chapter 3. The integration of multiview geometry allows us to successfully register all 22 images with the 3D model (see Figs. 4.2 and 4.7).

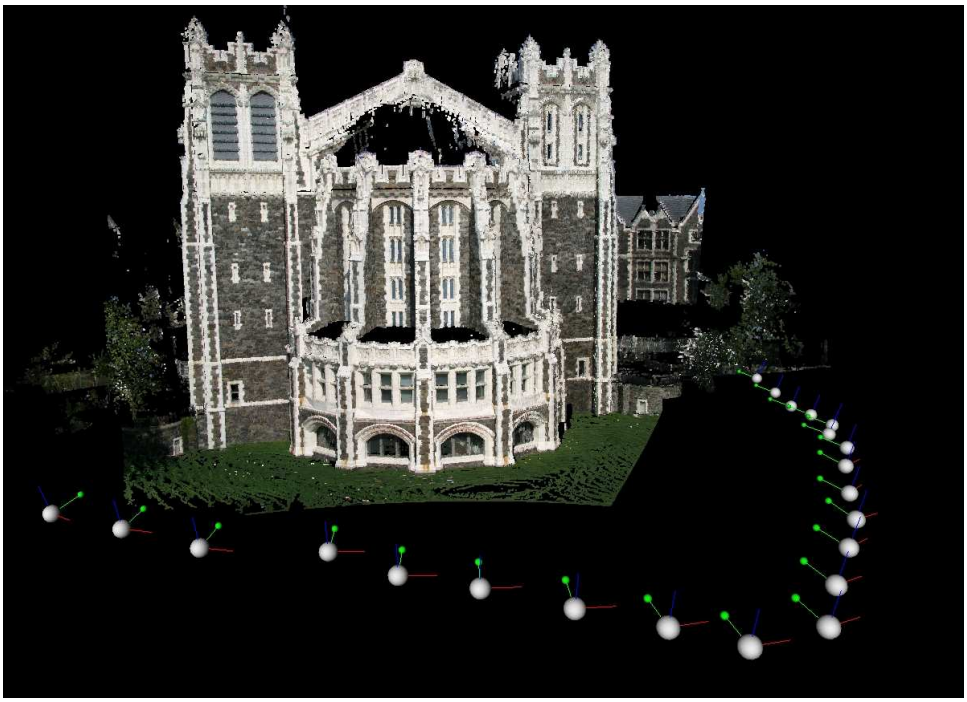

Figure 4.7: Shepard Hall with 22 automatically texture mapped images. Five images are registered with the range data by image-to-range registration (Sec. 4.2). All others are registered by the integration of SFM (Sec. 4.4).

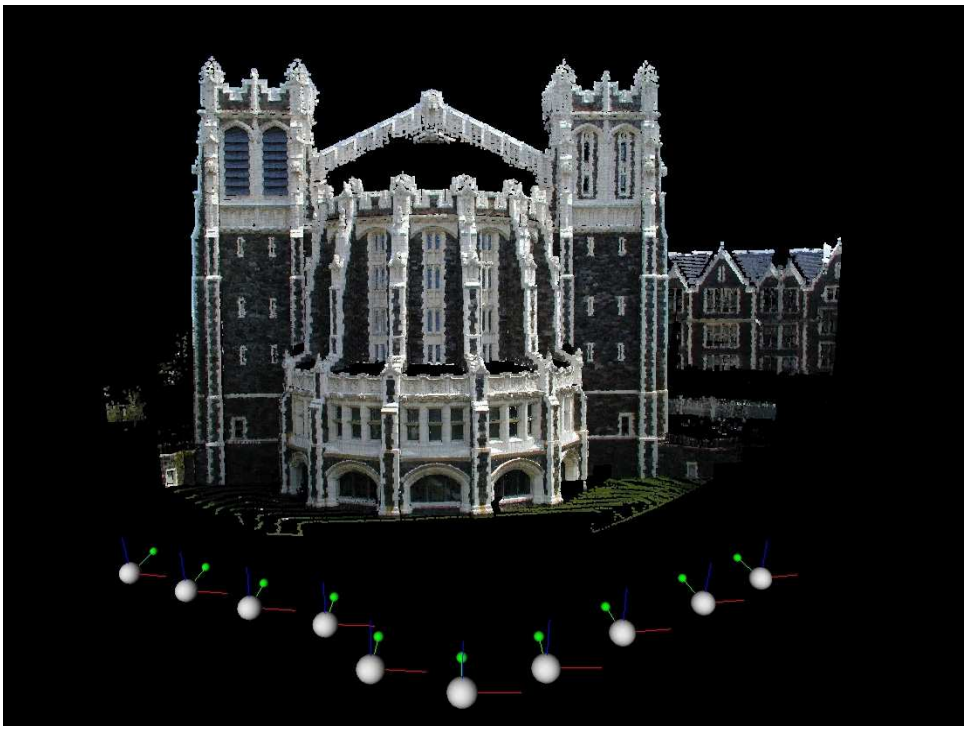

Figure 4.8: Shepard Hall with ten automatically texture mapped images. All ten images are registered with the range data by image-to-range registration (Sec. 4.2) and then the SFM is integrated (Sec. 4.4).

Figure 4.9: An interior scene (Great Hall) with seven texture mapped images. Note that we use only one image-to-range registration (achieved manually). Then, by utilizing multiview geometry the whole set of 2D images is automatically registered with the 3D range data.

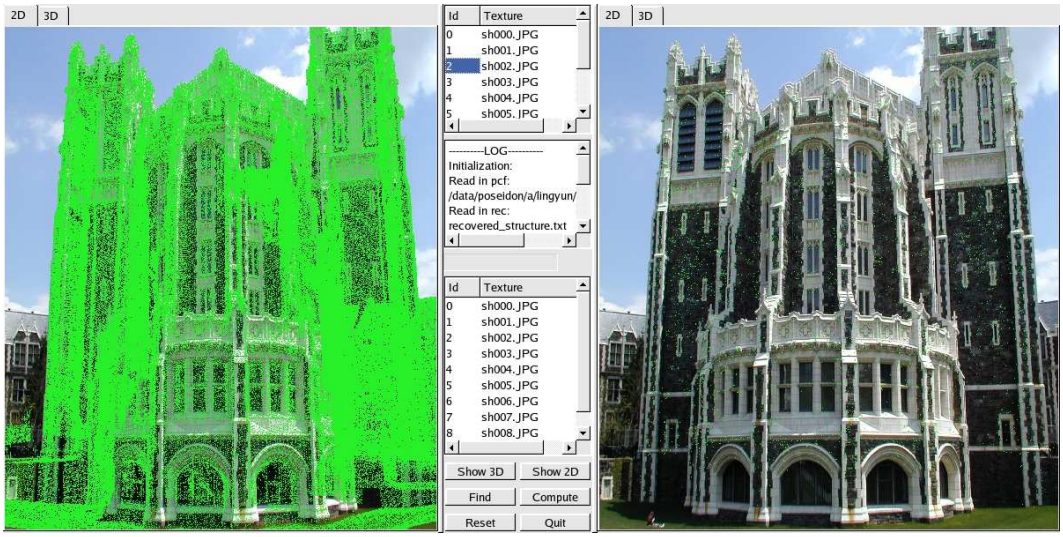

Figure 4.10: Easy-to-use interface for the SFM integration. The left window shows the projection of M_{range} on a 2D image (note the density of 3D range points). The right window shows the projection of M_{sfm} on the same 2D image (very few points are projected due to the sparse SFM model (Fig. 4.4)). The middle panel provides the major user controls (such as to find point correspondences ("find" button), to compute the transformation ("compute" button), etc.) and shows the log file during the process. The input to the system are 3D range point cloud (M_{range}), the whole set of 2D images, image-to-range registration results and structure-from-motion results (M_{sfm}). The output is the similarity transformation (scale factor, rotation and translation) that can bring M_{sfm} and M_{range} into alignment.

Chapter 5

Image-to-Range Registration: a new semi-automatic line-based system

In this chapter we present a new system that can automatically register 2D images with 3D range data assisted by a user-friendly user interface. New strategies for feature extraction, feature matching and camera pose estimation are developed in this system. The improvements of the new system with respect to our previous system (Chapter 3) can be summarized as follows:

- The new system is able to register 2D images to 3D models at interactive rates (approximately 10 secs per image on a 2GHz Pentium machine). This is due to new algorithms and improved user interface.

- Earlier systems ([Liu and Stamos, 2005, Stamos and Allen, 2001]) require the extraction of major facades, rectangles, or other higher-order structures from the 2D and 3D datasets. Our method, on the other hand, utilizes 3D and 2D linear features for matching without significant grouping. This increases the generality of our algorithm since we make fewer assumptions about the 3D scene. Scenes with various layers of planar facades, or without clear major facades can thus be handled.

- Our method utilizes vanishing points and major 3D directions, but it does not require them to be orthogonal as most earlier methods assume.

Our method consists of the following major steps: feature extraction (Sec. 5.1), internal calibration & rotation computation (Sec. 5.2), and camera position computation (Sec. 5.3). Our results are presented in Sec. 5.5.

5.1 Feature Extraction

In this section we describe our algorithms for extracting features from 3D-range and 2D-image data. These features are utilized for internal camera calibration and camera pose computation. The fact that our system uses linear features which do not require the significant grouping, makes our algorithms generally applicable to most exterior and interior urban scenes (see Sec. 5.5). Each linear feature is also associated with a radius r. In other words, a 3D

feature can be considered as a cylinder and a 2D feature as an oriented rectangle (Fig. 5.1). The value of the radius is initially defined by the user, and is then adapted based on the density of 3D and 2D lines (see following sections).

5.1.1 3D Feature Extraction

The 3D line extraction step is based on the segmentation method of [Stamos and Allen, 2002], whereas the major directions clustering is the same as in Chapter 3. The result of this process is a set of line clusters $\mathcal{L}^{3D}$. Each line in a cluster has similar orientation as every other line in the same cluster. The set of line clusters are then sorted based on the number of lines in each cluster. We do not assume knowledge of vertical or horizontal directions for the line clusters as in previous methods (Chapter 3). Each 3D line is thus associated with a cluster id, e.g. for the 3D lines in cluster $\mathcal{L}_i^{3D}$, their cluster id is i. In the next step, 3D features are extracted. First, an initial user-defined radius (e.g. 0.1m) is assigned to each 3D line. Then, a line merging step generates the final 3D features. This reduces the number of features, and thus increases the efficiency of the matching stage (Sec. 5.3). In this step, each pair of 3D lines (l_a, l_b) with the same cluster id are merged into a new line l_c (Fig. 5.1) if and only if: a) the distance between them are smaller than the sum of their radii, and b) their projections on l_c overlap. The merging procedure is continued until there are no two remaining 3D lines that can be merged. The final result

is a set of 3D lines, each of which is associated with a cluster id and radius.

5.1.2 2D Feature Extraction

The extraction of 2D features and vanishing points is based on well-known algorithms (e.g [Stamos and Allen, 2001, Mit City Scanning, 1999]). We can thus extract from each image a set lines that generate vanishing points $\mathbf{V}_1, \mathbf{V}_2, \cdots, \mathbf{V}_n$. Each vanishing point defines a cluster of 2D lines. The set of vanishing points are sorted based on the number of lines in the clusters [1]. Each 2D line is then associated with a cluster id (i.e. 2D lines of the cluster defined by $\mathbf{V}_i$ have id i). Lines that are close to each other are merged to generate the 2D features used for matching. The approach is similar to the 3D feature extraction as described above. Initially, a user defined radius is associated with each 2D line. In the merging step, if two lines, say l_a and l_b, have same cluster id, similar orientations and overlap with each other, then they are merged into a new 2D line l_c (Fig. 5.1). The merging stage continues until no two remaining 2D lines can be merged. The final result is a set of 2D lines, each of which is associated with a cluster id and radius.

[1] Note here that both 3D line clusters and 2D line clusters are sorted based on the number of lines they contain. Assuming that larger 3D clusters match with larger 2D clusters, this sort can provide a valuable hint for matching between 3D directions with 2D vanishing points.

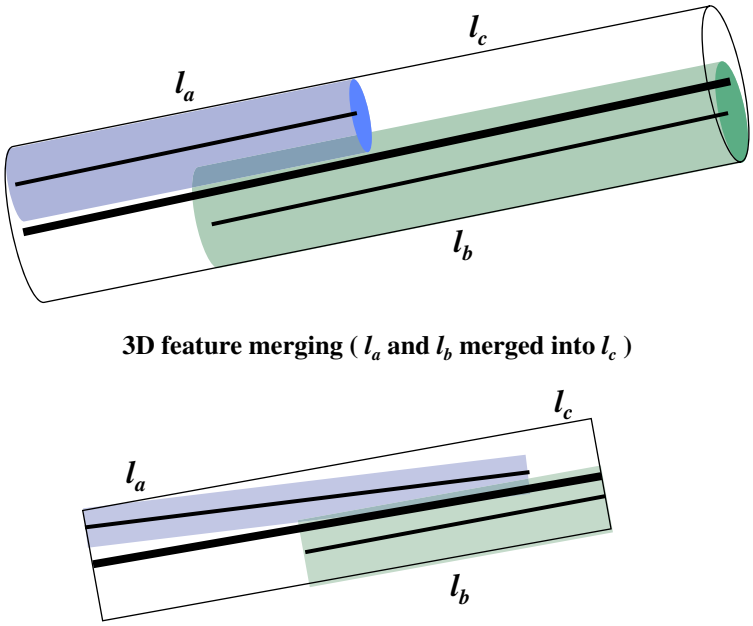

Figure 5.1: Example of new type of 3D and 2D features and their merging steps.

5.2 Internal Camera Calibration and Rotation Computation

The internal camera calibration parameters of each 2D camera (effective focal length and principal point[2]) can be computed by the utilization of three orthogonal vanishing points (closed form solution). An iterative solution can also estimate the effective focal length and principal point from two orthogonal vanishing points (Chapter 3). By matching two orthogonal vanishing points with two orthogonal 3D directions (see Sec. 5.1) the rotation $\mathbf{R}$ between the

[2]Note that we assume that radial distortions have already been corrected.

2D camera and 3D model can be computed.

Here we present an additional method for the calculation of the effective focal length f and of the rotation $\mathbf{R}$. We are using two vanishing points and two major 3D directions. We, however, do not assume that these directions are orthogonal to each other. Orthogonality is prominent in urban scenes, but is not always present. Our method starts with an initial estimate f_{init} of the effective focal length, and of the principal point $\mathbf{P}_{init}$. f_{init} is included in the Exif meta-data, information that is now provided by most digital cameras. $\mathbf{P}_{init}$ is estimated by the center of the image. Based on these estimates, we can have an initial center of projection $\mathbf{C}_{init}$, which is the origin of the camera coordinate system (Fig. 5.2).

Let us consider a vanishing point $\mathbf{V}_i$ extracted by the 2D images (see Sec. 5.1). The 3D coordinates of $\mathbf{V}_i$ in the camera coordinate system are $[(V_i)_x - (P_{init})_x, (V_i)_y - (P_{init})_y, f_{init}]^T$. Thus, the normalized vector $\mathbf{D}_i^{2D} = u(\mathbf{C}_{init}\mathbf{V}_i)$[3] represents the 3D direction that generates the vanishing point $\mathbf{V}_i$. This direction is expressed in the camera coordinate system. Our goal is to match each vanishing point with its corresponding 3D direction extracted from the 3D range model (see Sec. 5.1). This correspondence leads to the calculation of the focal length and the rotation $\mathbf{R}$. Let us represent each 3D line cluster in $\mathcal{L}^{3D}$ (Sec. 5.1) by its 3D direction $\mathbf{D}_j^{3D}, j = 1 \ldots n$ (where n is the number of extracted 3D clusters).

[3]We use the notation $u(\mathbf{v})$ for describing the unit vector derived from $\mathbf{v}$.

The next step is to find the matching pairs of directions $< \mathbf{D}_i^{2D}, \mathbf{D}_j^{3D} >$. Consider for the moment that we know the correspondence between vanishing points (expressed in the camera coordinate system) and 3D directions (expressed in the world coordinate system). It is known that with the principal point fixed at the center of image, two pairs ($< \mathbf{D}_a^{2D}, \mathbf{D}_a^{3D} >$, $< \mathbf{D}_b^{2D}, \mathbf{D}_b^{3D} >$) of matching vanishing point/3D directions are enough for the computation of the focal length f. The focal length f (which is $|\mathbf{CP}|$ in Fig. 5.2) can be computed via the following equations (triangles $\mathbf{CV_aP}, \mathbf{CV_bP}$ and $\mathbf{CV_aV_b}$):

$$|\mathbf{CV}_a|^2 = |\mathbf{PV}_a|^2 + f^2$$

$$|\mathbf{CV}_b|^2 = |\mathbf{PV}_b|^2 + f^2$$

$$|\mathbf{V}_a\mathbf{V}_b|^2 = |\mathbf{CV}_a|^2 + |\mathbf{CV}_b|^2 - 2 \cdot |\mathbf{CV}_a| \cdot |\mathbf{CV}_b| \cdot \cos\alpha$$

where α is the angle between $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$. (Note that the vanishing points $\mathbf{V_a}$ and $\mathbf{V_b}$ have been computed by using the initial estimates f_{init} and $\mathbf{P}_{init}$. The above computation leads to the calculation of a focal length that conforms to the 3D directions $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$.) From the above equations, we can get a quartic equation:

$$a \cdot f^4 + b \cdot f^2 + c = 0$$

where $a = \sin^2\alpha$, $b = \sin^2\alpha(|\mathbf{PV}_a|^2 + |\mathbf{PV}_b|^2) - |\mathbf{V}_a\mathbf{V}_b|^2$, $c = (\frac{|\mathbf{V}_a\mathbf{V}_b|^2 - |\mathbf{PV}_a|^2 - |\mathbf{PV}_b|^2}{2})^2 - \cos^2\alpha|\mathbf{PV}_a|^2|\mathbf{PV}_b|^2$. Solving this equation, one obtains the refined focal length: $f = \sqrt{\frac{\sqrt{b^2-4ac}-b}{2a}}$. Since $\mathbf{D}_a^{3D} \neq \mathbf{D}_b^{3D}$, $\sin\alpha$ will never be equal to 0. Finally,

the rotation $\mathbf{R}$ is computed based on these two pairs of matching directions [Faugeras, 1996].

Based on the above analysis, the task of our system is to find two matching pairs of vanishing points/3D directions. Intuitively, pairs ($< \mathbf{D}_a^{2D}, \mathbf{D}_a^{3D} >$, $< \mathbf{D}_b^{2D}, \mathbf{D}_b^{3D} >$) for which the angle between $\mathbf{D}_a^{2D}$ and $\mathbf{D}_b^{2D}$ is not similar to the angle between $\mathbf{D}_a^{3D}$ and $\mathbf{D}_b^{3D}$ can be rejected. As a result, we have a list of matching candidates, each of which contains two pairs of matching vanishing points and 3D directions, a refined focal length and a rotation. For each one of these candidates we can apply the algorithm described in the next section for calculating the camera position, and finally keep the result that provides the maximal alignment between the 2D image and 3D model.

In the worst case scenario though all pairs of directions have similar angles (this scenario is easily realizable in urban scenes where most angles between major directions is 90 degrees). In this case there are $\binom{n}{2}\binom{m}{2}$ candidate matching pairs of directions (where n is the number of 3D and m the number of vanishing points). Even though this is not a large search space (n and m are small in most urban scenes), testing all hypotheses involves the computation of the translation (see next section). This is computationally inefficient for the purposes of an interactive system, where a response time of up to 10 seconds per image is appropriate. For these reasons we let the user to implicitly provide the correct pair of matching directions, by rotating the 3D model to

an orientation that produces a rendering that is similar (but not exactly the same) to the real 2D image (see figures of our results). As shown in Figs. 5.11(b) and 5.12(b), the rotated 3D view (left) is similar (but not exactly the same) to the 2D image (right). This user-assisted rotation can approximately align the corresponding 2D and 3D directions.

The aforementioned user interaction not only increases the computational efficiency of the whole system, but also makes the registration problem tractable. In general, without constraining the possible locations of 2D cameras with respect to the 3D model, the image-to-range registration problem becomes intractable. This is due to the existence of a possible large set of solutions. For example, a photograph of one of the columns of the 3D structure of Fig. 5.12 can be matched with any of the symmetric 3D columns of the real scene. By selecting a synthetic view that is similar, but not exactly the same as the 2D image, the user can provide an approximate field of view to help the matching algorithm. In particular, only 3D features that are viewable in the synthetic 3D view are used for matching 2D image features. Comparing to the previous system (Chapter 3), in which the user needs to explicitly provide the match between vanishing points/3D directions and the user also needs to match facades between the 2D image and 3D model, the current approach is more natural and leads to faster interaction time. The selection of a synthetic view is very easy and intuitive for the user and can be done in little time.

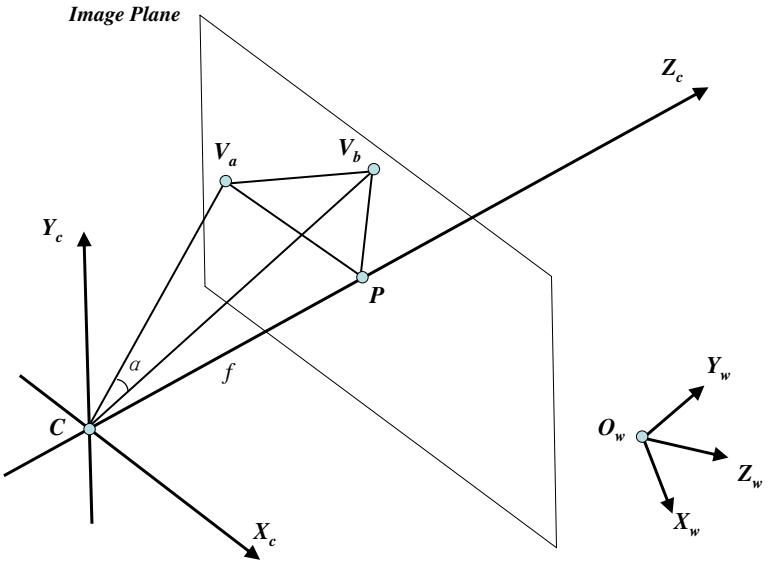

Figure 5.2: Rotation and focal length computation based on two vanishing points and their corresponding 3D directions (not shown in this image).

The final result of this module is a list of matching candidates, each of which contains two pairs of matching vanishing points/3D directions, a refined focal length and a rotation. The user can cycle through them, and a camera position is computed for each matching candidate (see next section). Then, each candidate is quantitatively evaluated (see next section).

5.3 Camera Position Computation

From the previous section, a list of matching candidates, named $\mathcal{M}$, is obtained. Each element in $\mathcal{M}$ contains a matching pair of two vanishing points and two 3D directions, a refined focal length and a rotation. In this section, a 2D camera position will be computed for each candidate in $\mathcal{M}$. Our method of finding the camera position follows a hypothesis-and-test scheme by matching the extracted 3D and 2D features based on the similar framework as in Chapter 3. A number of major differences with the aforementioned method make our algorithm more general and more robust. In particular, our algorithm does not require the extraction of planar facades, and does not require the significant grouping of low-level features into higher-order structures. Scenes that do not pertain clear major facades (such as the example of Figs. 5.12(a) and (b), where various layers of planar facades exist) can now be successfully handled. Also since all low-level features are used without significant grouping, more robust results are achieved. Due to the fact that we are utilizing the low-level linear features we have developed a new algorithm for the computation of camera position (Step 2 of the following algorithm).

We now present a detailed description of our algorithm. First, a candidate from $\mathcal{M}_i$ is selected, i.e. the matching pair of vanishing points and 3D directions are $< \mathbf{V}_a, \mathbf{V}_b >$ and $< \mathbf{D}_a^{3D}, \mathbf{D}_b^{3D} >$; the refined focal length is f_i and the rotation is $\mathbf{R}_i$. The camera position (translation) is then computed in the

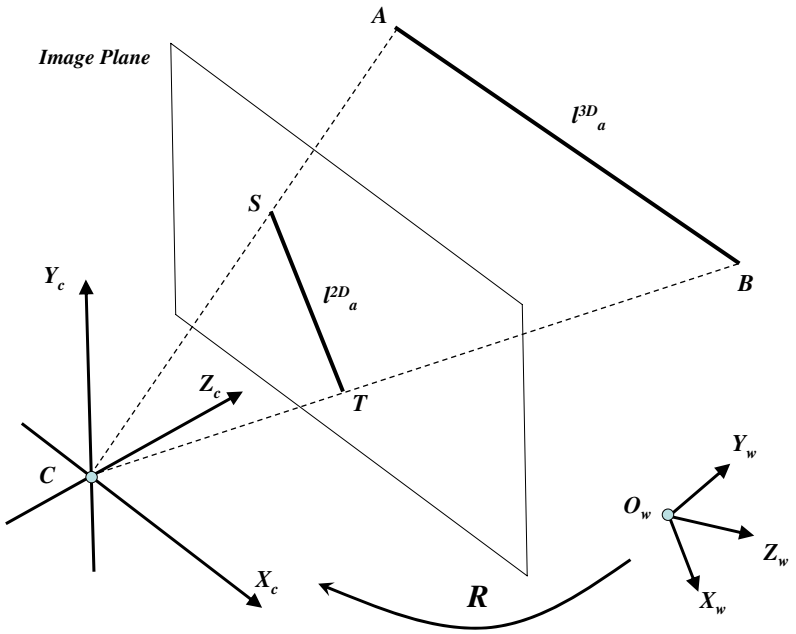

Figure 5.3: Camera position computation based on a match between 3D feature **AB** with image feature **ST**.

following six steps (Fig. 5.4):

Step 1 A hypothetical match between two pairs of 3D and 2D lines is selected (the algorithm will go over all possible such selections). Let us call these pairs $< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$ (l_a^{3D} and l_b^{3D} are 3D lines extracted from the 3D model, and l_a^{2D} and l_b^{2D} 2D lines extracted from the 2D image).

Step 2 In this step we compute the camera position in world coordinate system (translation) based on the match of l_a^{3D} with l_a^{2D}. As shown in Fig. 5.3, **A** and **B** are the endpoints of l_a^{3D} and **S** and **T** are the endpoints of l_a^{2D}. **C** is the center of projection. If l_a^{3D} matches exactly with l_a^{2D}, then in the camera

coordinate system, **C, S** and **A** should be collinear. The same applies for **C, T** and **B**. We thus consider **C** as the intersection point of the following two lines: a) one that passes through **A** having the orientation of line **CS** and b) one that passes through **B** having the orientation of line **CT**. To compute the world coordinates of **C**, we need to know the orientations of **CS** and **CT** in the world coordinate system. We know, however, the orientations of **CS** and **CT** in the camera coordinate coordinate system, say $\mathbf{n}_a$ and $\mathbf{n}_b$. We have also computed the rotation R which brings the camera and world coordinate systems into alignment (see previous section). We can thus compute the orientations of **CS** and **CT** in the world coordinate system as: $R \cdot \mathbf{n}_a$ and $R \cdot \mathbf{n}_b$. Then, the camera position is obtained by finding the intersection of two 3D lines: a) one of which passes through **A** with the orientation of $R \cdot \mathbf{n}_a$ and b) one which passes through **B** with the orientation of $R \cdot \mathbf{n}_b$ [4]. Finally, this computed center of projection is used to project l_b^{3D} onto the image plane. If the projection of l_b^{3D} overlaps with l_b^{2D} (within a threshold of 80%), then the camera position computed using (l_a^{3D}, l_a^{2D}) is verified by the pair (l_b^{3D}, l_b^{2D}). We therefore move to the next step. Otherwise, we return to step 1 (i.e. the match is discarded) to pick another set of hypothetical matching lines. It should be noted that to compute an accurate camera position, the end points of the matching lines should exactly match with each other. Due to the uncertainty of the automatically extracted 3D/2D features, this is not always the case. To compensate this inaccuracy,

[4] **A** and **B** are both expressed in the world coordinate system

in the following steps, the second pair of 3D/2D features are used to compute a camera position as well. The weighted average of the two computed camera positions is used as the camera position for these two pairs matching features (see discussion in Sec. 5.6).

Step 3 Step 2 is repeated assuming as hypothesis the match between l_b^{3D} and l_b^{2D}. The newly computed center of projection is used to compute the overlap between l_a^{2D} and the projection of l_a^{3D}. If this overlap is less than a threshold (i.e. the computed $\mathbf{C}$ is not verified by (l_a^{3D}, l_a^{2D}), we return to step 1 (i.e. the match is discarded). Otherwise, we proceed to the next step.

Step 4 Step 2 has thus computed a camera position $\mathbf{C_1}$ by the hypothesis (l_a^{3D}, l_a^{2D}) [verified by (l_b^{3D}, l_b^{2D})], while step 3 has computed a camera position $\mathbf{C_2}$ by the hypothesis (l_b^{3D}, l_b^{2D}) [verified by (l_a^{3D}, l_a^{2D})]. In this step, the weighted average (based on the amount of overlap) of these two camera positions is computed and saved in a list $\mathcal{T}$.

Step 5 Steps 1 to 4 are repeated for all possible pairs of pairs of 3D and 2D lines ($< l_a^{3D}, l_a^{2D} >, < l_b^{3D}, l_b^{2D} >$). All verified camera positions (see Step 4) are stored in a list $\mathcal{T}$. Then, for each position in $\mathcal{T}$, all 3D lines are projected onto the image plane. For each of the projected 3D lines, a possible matching 2D line is found by searching around its projection. This region is bounded by the radius of the 3D and 2D lines. The number of found matches grades this camera position. If the grade of a camera position is less than a threshold, it

is removed from the list $\mathcal{T}$.

Step 6 The remaining camera positions in $\mathcal{T}$ are optimized by two steps. First, for each camera position $\mathbf{C}_i$ a refined position is found. This is achieved by searching around a small neighborhood of $\mathbf{C}_i$ in order to maximize the overlap between the matching 3D and 2D lines. Then this refined position is further optimized by an iterative algorithm. In each iteration, the current camera position is used to generate a list of matching 2D and 3D lines from the whole 2D and 3D feature space. A new camera position is found in a small neighborhood around the current camera position. This new position produces the minimal error distance between the 2D lines and the projections of their matching 3D lines. For each pair of corresponding 2D and 3D lines, their error distance is the average of the two distances measured from the two end points of the projected 3D line to the matching 2D line. The algorithm converges at the point when the error distance remains constant. The camera position computed after the two optimization steps is the final result.

The camera position in $\mathcal{T}$ with the maximum grade is picked as the best one for the matching candidate $\mathcal{M}_i$. This is normally correct, but the list is still kept as well in case that the one with the maximum grade is not the best. Then, the user can select other positions in the list. This maximum grade is also used as the grade for $\mathcal{M}_i$. For each matching candidate in $\mathcal{M}$, a list of camera positions is computed by these 6 steps and a grade is assigned. Then,

the list $\mathcal{M}$ is sorted based on the grade and the one with the maximum grade is selected as the best one but the user also can select other results in $\mathcal{M}$.

5.4 Evaluation

To evaluate our registration results, we manually mark the line correspondences from 3D range data and 2D images as ground truth. The module of manual selection is implemented in our system. To mark a 3D line, the user only need to click on the 3D viewer to select two points from the range point cloud as the endpoints of a line. The 2D lines are similarly marked by clicking on the 2D viewer. The 2D and 3D lines should be picked such that the matching between them is obvious and intuitive. Usually silhouettes of architectural structures such as windows, cylinder, etc. are good candidates. Examples of marked 3D and 2D lines are shown in Figs. 5.5, 5.6 and 5.7. After the automatic registration steps, the marked 3D lines are projected onto the image plane based on this registration result (bottom right images in Figs. 5.5, 5.6 and bottom image in Fig. 5.7). The average distance (in pixels) between the marked 2D lines with the projections of their corresponding marked 3D lines is considered as an error for the evaluation. For each pair of corresponding 2D and 3D lines, the line-to-line distance is the average of the two distances measured from the two end points of the projected 3D line to the matching 2D line.

Table. 5.1 shows the successful registration results. The last column E contains the evaluation results. Intuitively, smaller errors correspond to better registration results. Taken the possible errors introduced during the manual selection process into account, an average error around 1 to 2 pixels is an acceptable range for a good registration. This is also verified visually by our high-quality texture-mapping results. Examples of texture-mapping details are shown in Fig. 5.8.

Table. 5.2 shows the results of failed registrations. The evaluation errors are far from the acceptable range. Examples of texture mappings from the failed registration results are shown in Fig. 5.9. These failed cases are caused by the low quality of 3D/2D features. The effects of 3D/2D feature quality on the registration are discussed in detail in Sec. 5.6.

5.5 Results

We are presenting results from real experiments in three urban settings that we name 1, 2, and 3. Buildings 1 (Thomas Hunter building, Hunter College NYC) and 2 (East 8th and Lafayette Street NYC) are the exteriors of regular urban structures. Building 3 is the interior of Grand Central Station, a scene of architectural complexity and beauty. Figs. 5.10, 5.11, and 5.12 provide individual registration results, as described in the technical sections

5.1, 5.2 and 5.3. For the cases shown in Fig. 5.13 and 5.14, the user only needs to orient the 3D range model in a position that simulates the 2D color image. As you can see from these figures, this simulation need not be exact. This step of user interaction is necessary to assist the matching between vanishing points and 3D directions (Sec. 5.2). Table 5.1 presents quantitative results for successful automated registrations and Table 5.2 for the failed registrations (see figure caption for more details). For building 1, we tested 16 images, 13 of them were registered successfully with errors in the acceptable range, whereas 3 others were not acceptable. Out of 9 images tried for building 2, 7 were registered successfully, whereas the rest 2 were not acceptable. Finally, out of 9 total images tried for building 3, 4 were registered successfully.

In all cases, the first step (Sec. 5.2) never fails, since all these scenes contain at least two vanishing points. The second step, however (Sec. 5.3) depends on the quality of the 2D/3D linear feature extractions. In cases that we cannot extract features of high quality (due to low contrast in 2D images), this method will not be able to perform correctly (for further discussion refer to next section). On the other hand, failure of few individual image-to-range registrations can be dealt with integration of multiview-geometry as described in Chapter 4.

5.6 Discussion

In summary, this new image-to-range registration system requires minimal user interaction and can register 2D images with 3D range data at interactive rates. The user interaction can not only increase the computational efficiency of the whole system, but also make the registration problem tractable. In addition, the whole space of possible matches between 3D and 2D linear features is explored efficiently in this system. This is a large improvement comparing to the previous method (Chapter 3), which requires the extraction of major facades, rectangles, or other higher-order structures from the 2D and 3D datasets. By searching the whole space, the possibility of convergence in our system is increased. As we have discussed before, the previous system requires the extraction of 2D/3D faces to calculate the rotation, match the features and compute the translation. This requirement can lead to system failure when dealing with scenes containing various layers of planar facades, or without clear major facades. The new system, on the other hand, computes the rotation by matching 3D directions with 2D vanishing points and computes the camera position by matching 3D/2D linear features without requiring 2D/3D face extractions. Therefore, the current system can work well with scenes containing multiple layers of planar facades, or without clear major facades. This increases the generality of our algorithm, since we make less assumptions about the 3D scene.

However, it should be noted that this system still requires the successful extraction of at least two 2D vanishing points and 3D major directions, although it does not require them to be orthogonal as the previous method assumes. At this stage, the existence of strong parallelism in the scene is still necessary for extracting at least two 3D major directions and two 2D vanishing points for registration. Therefore, a system failure will occur in the scenes containing dominant curvature structures. One important avenue of exploration for future work is the ability to handle scenes of general configuration not containing any major vanishing points. This would allow the exploration of registration algorithms in non-urban scenes.

It should also be noted that the accuracy of the registration result depends on the quality of the extracted 2D/3D features. This is because in the camera position computation step, correct matching 3D/2D linear features (i.e. their endpoints should be exact matches) are required. Since the 3D lines are mainly the borders and intersection lines of segmented planar regions from the range data [Stamos and Allen, 2002], if the scenes do not contain sufficient planar structures, the system may not extract good quality 3D features. In addition, the 2D image quality affects the 2D line/feature extraction. A good 2D image should be sharp and of high contrast such that the Canny edge detector can find good edges for 2D line/feature extraction. If the images are blurred or of low contrast, the Canny edge detector may not find good quality edges for

2D line/feature extraction. For example, in the case of building 3, all the images were taken in the interior of Grand Central Station at around 3:00 am. Due to the poor lighting condition, long exposure time was used. This long exposure leads to blurred images even due to very small camera motion. Our system was able to successfully register very few images with 3D range data for building 3 (Table. 5.1) due to the low quality 2D features extracted from the blurred images.

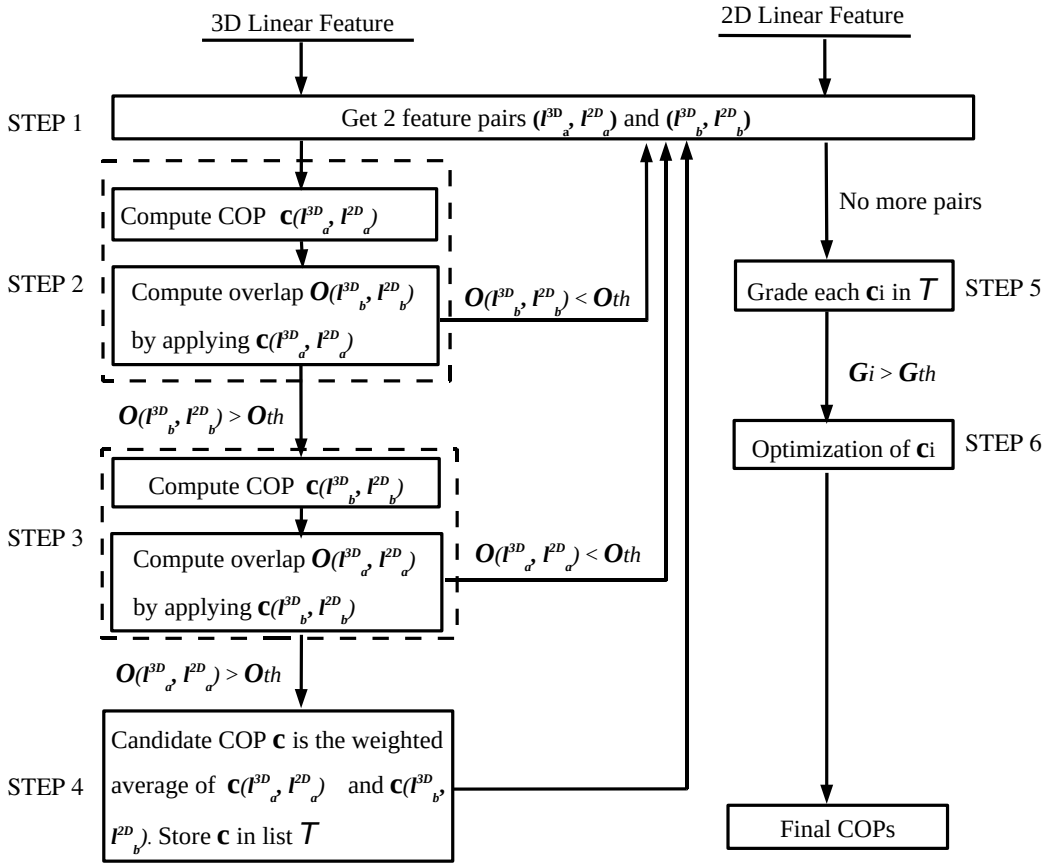

Figure 5.4: Camera position (translation) computation flowchart. Through step 1 all possible pairs of matched 3D and 2D lines ($< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$) are selected (l_a^{3D} and l_b^{3D} are 3D lines extracted from the 3D model, and l_a^{2D} and l_b^{2D} 2D lines extracted from the 2D image). Step 2 computes a camera position based on $< l_a^{3D}, l_a^{2D} >$. The pair $< l_b^{3D}, l_b^{2D} >$ is used for the verification of this position. If the overlap between l_b^{2D} and the projection of l_b^{3D} on the image is smaller than O_{th} (20%) (i.e. the position is not verified) a new pair is selected (step 1). Otherwise a similar computation is carried out for the pair $< l_b^{3D}, l_b^{2D} >$ (step 3). If steps 2 and 3 produce two verifiable camera positions, a weighted average is computed (step 4). This average represents the position that is generated by the hypothesis ($< l_a^{3D}, l_a^{2D} >$ and $< l_b^{3D}, l_b^{2D} >$). All verified camera positions are stored in a list $\mathcal{T}$. After all pairs have been explored, each position in $\mathcal{T}$ is graded by projecting all 3D lines on the 2D image space (step 5). Positions with high grade (greater than G_{th} number of matches) survive to the final optimization step 6.

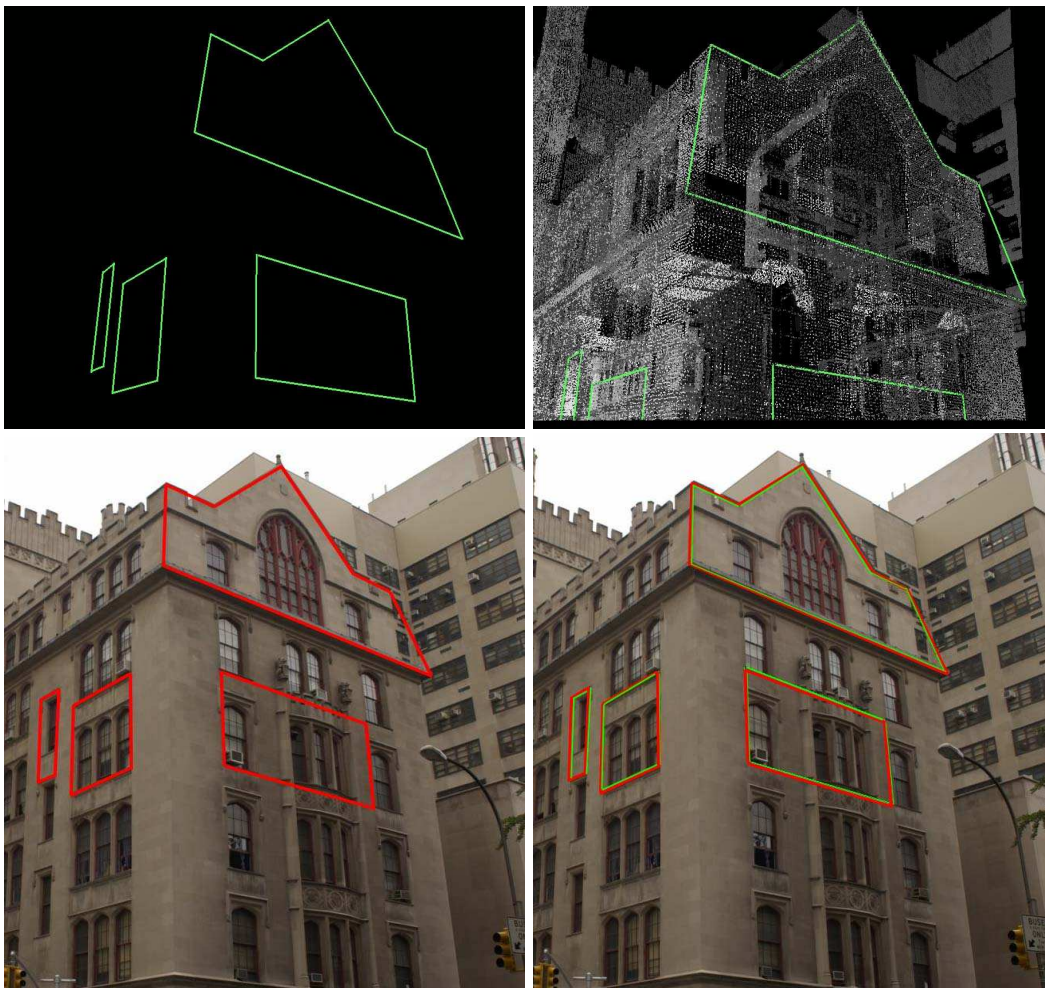

Figure 5.5: The evaluation of the registration result on building 1. For this case, 19 pairs of 3D and 2D lines are selected and the average error is 1.6833 in pixels (see Sec. 5.4 and Table 5.1 for description of error measurements). **Top Left**: The manually marked 3D lines; **Top Right**: Closed view of marked 3D lines with 3D range point cloud; **Bottom Left**: The manually marked 2D lines; **Bottom Right**: The marked 3D lines are projected onto the image plane (green) based on the registration result and overlaid with their corresponding 2D lines (red).

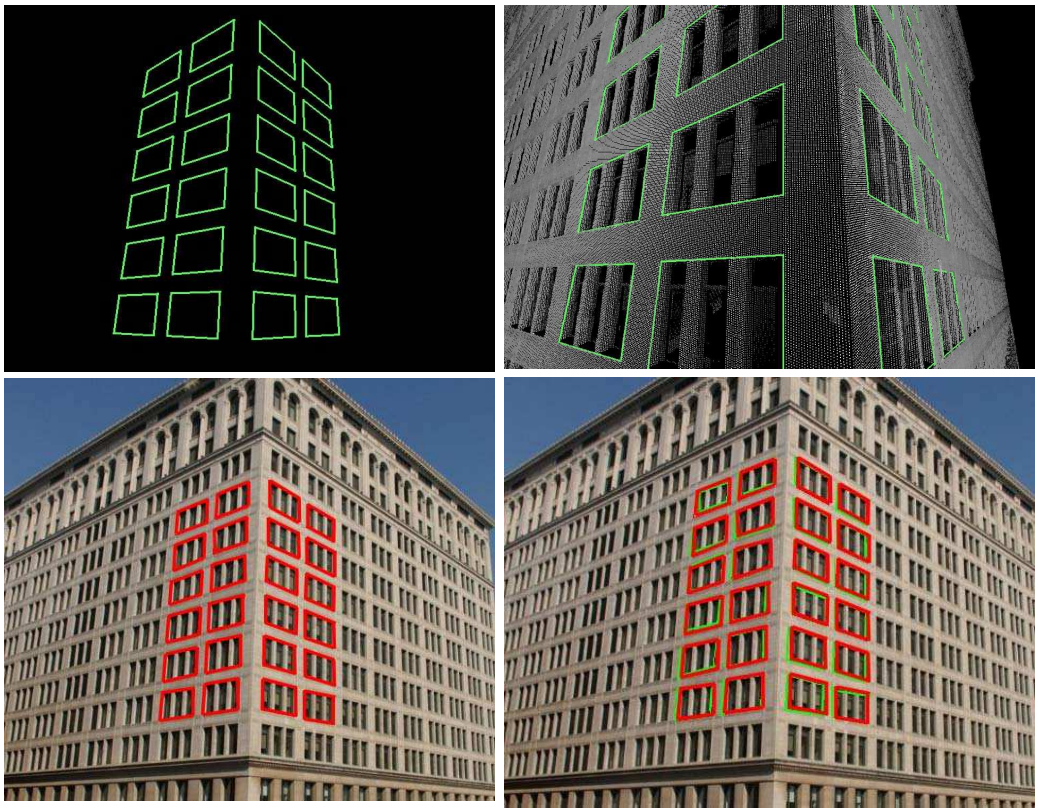

Figure 5.6: The evaluation of the registration result on building 2. For this case, 96 pairs of 3D and 2D lines are selected and the average error is 1.4923 in pixels (See Sec. 5.4 and Table 5.1 for description of error measurements). **Top Left**: The manually marked 3D lines; **Top Right**: Closed view of marked 3D lines with 3D range point cloud; **Bottom Left**: The manually marked 2D lines; **Bottom Right**: The marked 3D lines are projected onto the image plane (green) based on the registration result and overlaid with their corresponding 2D lines (red).

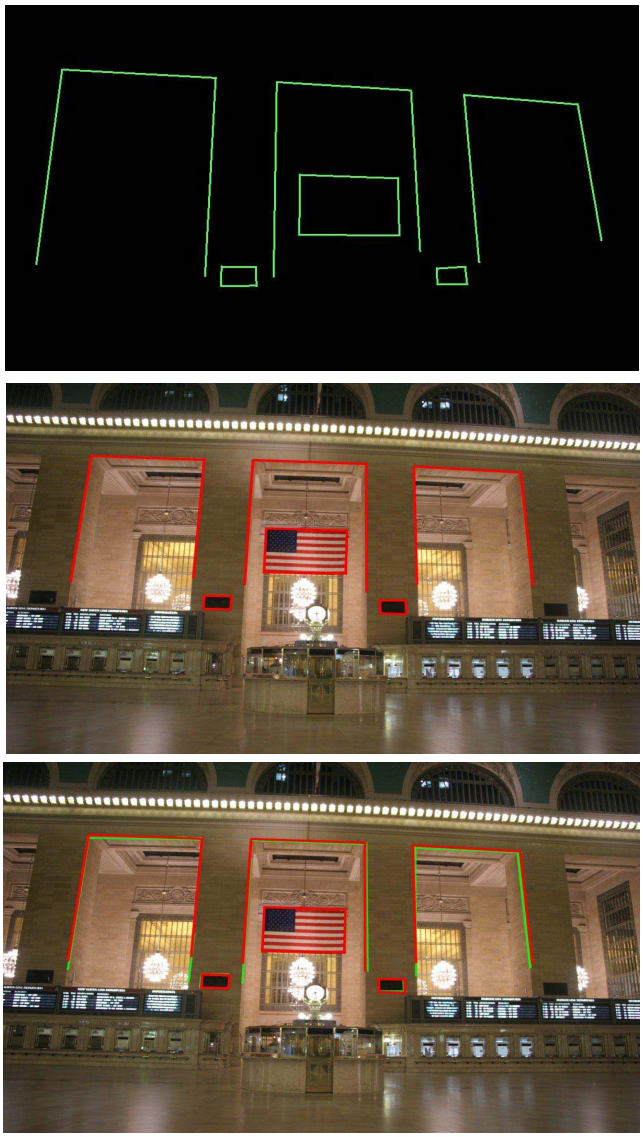

Figure 5.7: The evaluation of the registration result based on manually selected line correspondences. For this case, 21 pairs of 3D and 2D lines are selected and the average error is 1.7217 in pixels (See Sec. 5.4 and Table 5.1 for description of error measurements). **Top**: The manually marked 3D lines; **Middle**: The manually marked 2D lines; **Bottom**: The marked 3D lines are projected onto the image plane (green) based on the registration result and overlaid with their corresponding 2D lines (red).

Figure 5.8: **Top**: Texture mapping details for Building 1 (left) and Building 2 (right). The texture mapped urban structures (windows, street signs, AC units, etc.) show the high accuracy of our registration system. **Bottom**: Texture mapping details for Building 3. The flag and lights show the accuracy of our registration system.

Figure 5.9: The texture mapping results from wrong registration. **Top**: Building 1, the white sky is mapped onto the building. The evaluation error is 24.2109 pixels in this case. **Bottom**: Building 3, the texture mapping is very messy. For example, the flag is mapped everywhere, on cylinder, back wall, etc. The evaluation error is 43.5432 pixels in this case. The reason for these errors is discussed in Sec. 5.6. Note that these images were captured under poor lighting conditions.

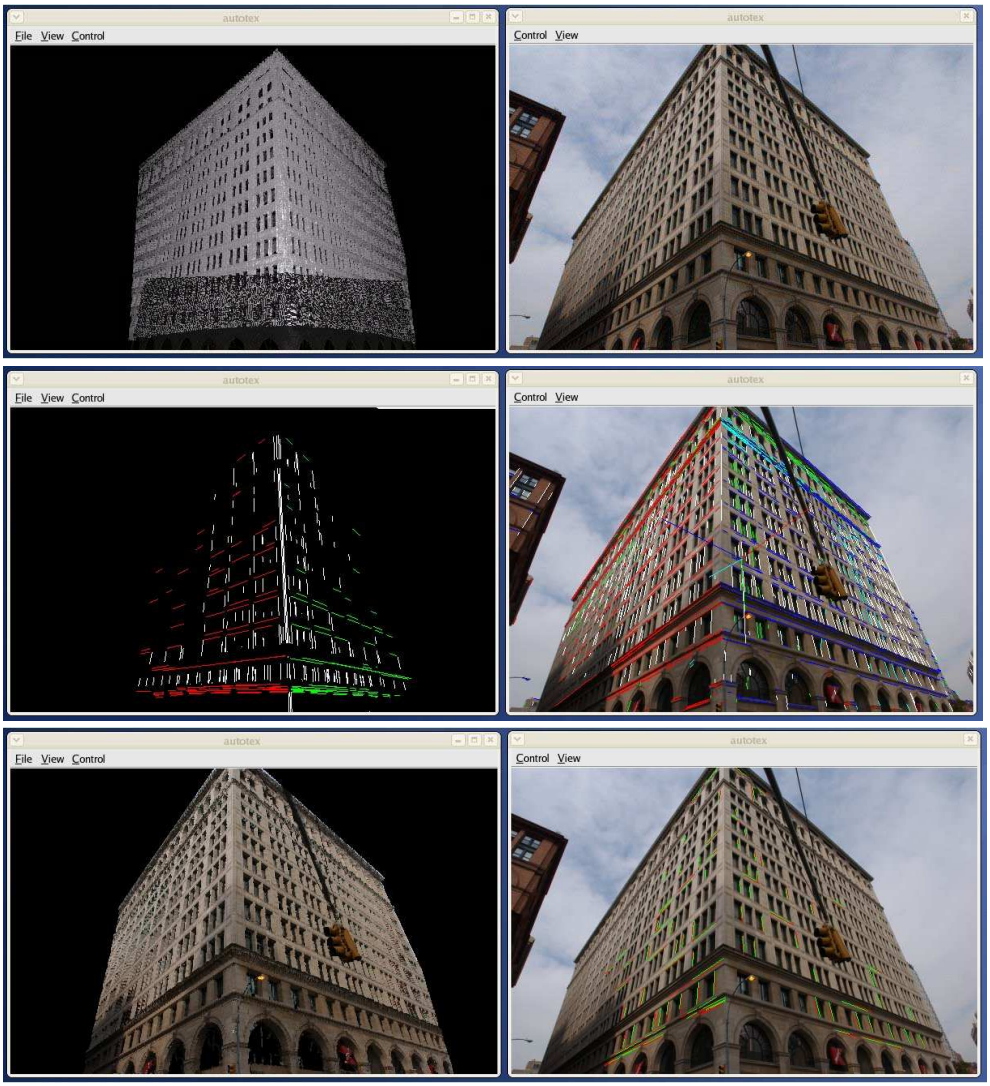

Figure 5.10: Registration result of Building 2. **Top row**: Initial state (before registration). The 3D range model (left column) and 2D image (right column) are loaded and displayed in the interface. **Middle row**: The state of the system after the feature extraction. The 3D viewer (left column) shows the clustered 3D lines while the 2D viewer (right column) shows the clustered 2D lines that are drawn on the original 2D image. Different clusters are represented by different colors for clarity. **Bottom row**: The final registration. The 2D image is automatically registered with the 3D range data. The 3D viewer (left) shows the texture mapped 3D range data. The 2D viewer (right) shows the matching 2D and 3D line features (2D lines are displayed as red, while projected 3D lines are highlighted in green). Note that objects that are not part of the 3D model cannot be texture-mapped (corner of other building shown in the 2D image).

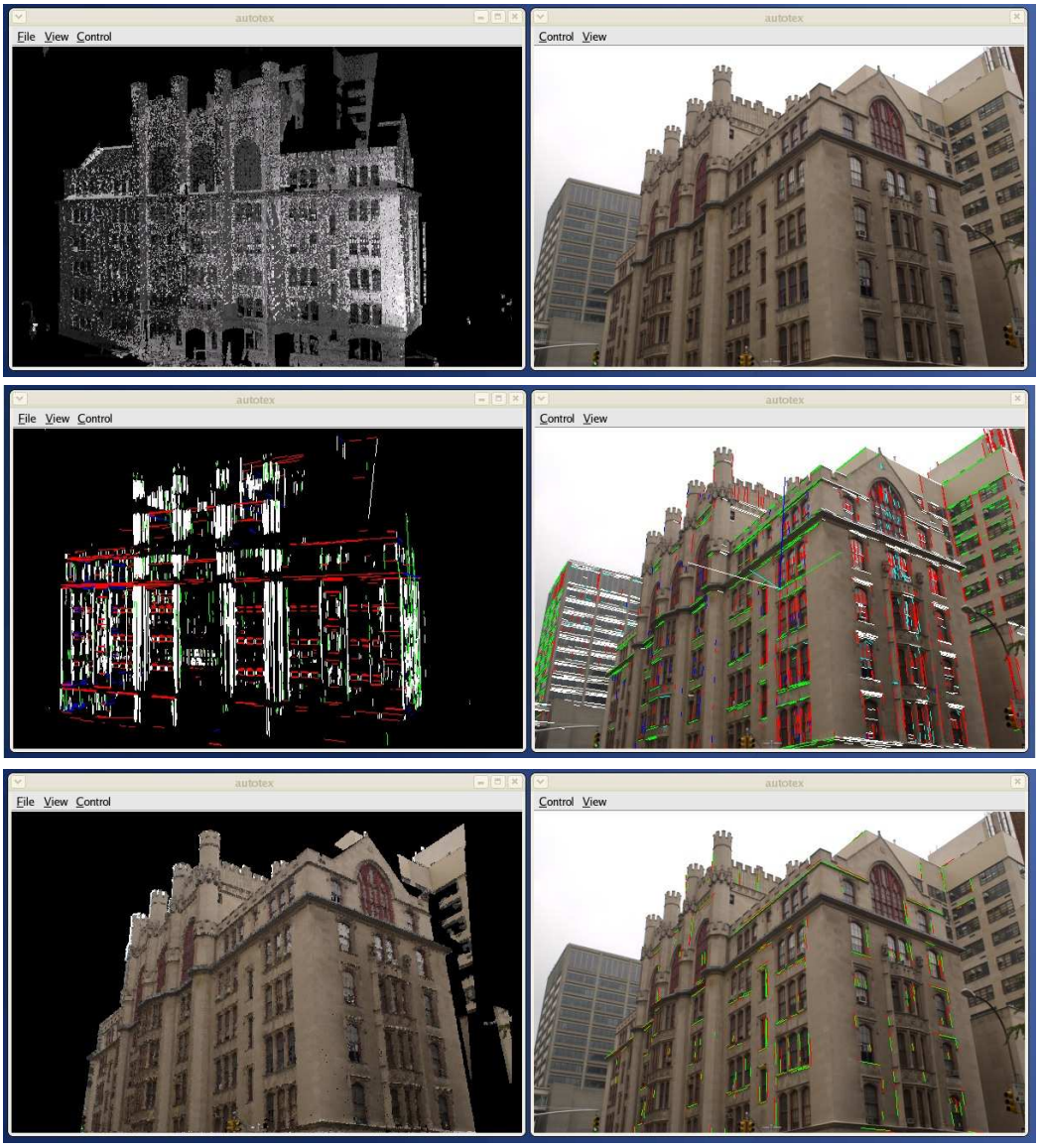

Figure 5.11: Registration result of Building 1. For description see caption of Fig. 5.10

.

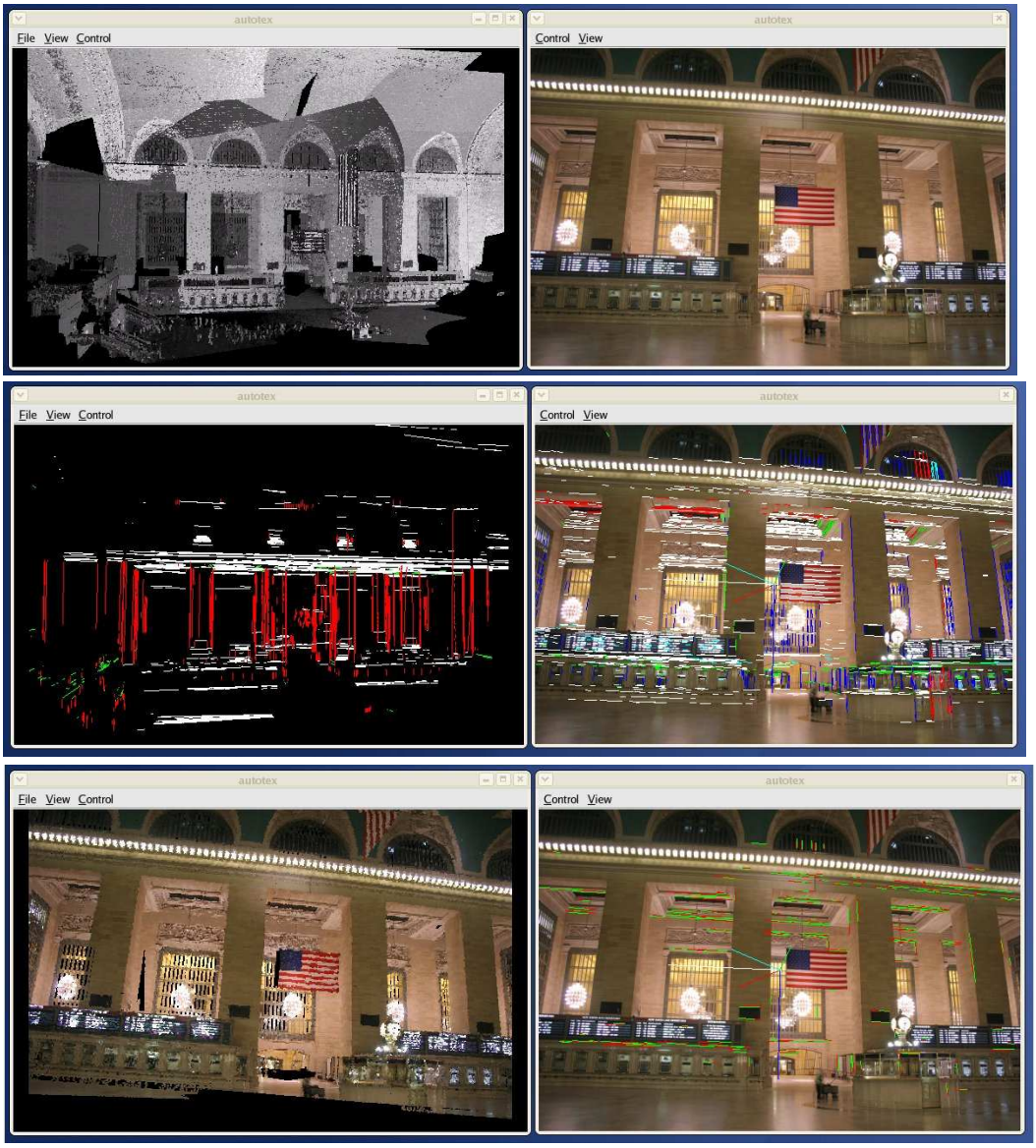

Figure 5.12: Registration result of Building 3. For description see caption of Fig. 5.10

.

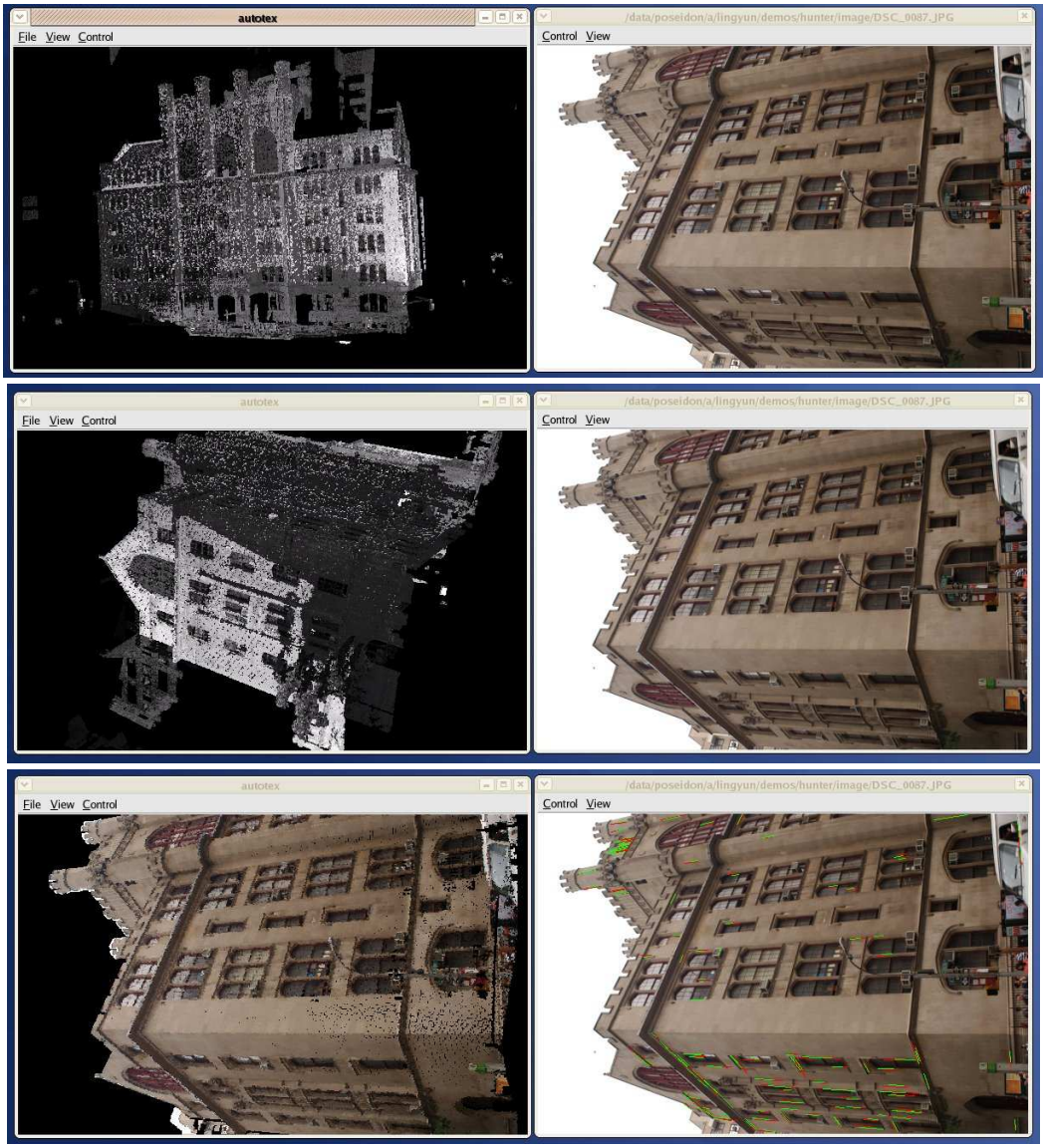

Figure 5.13: User interaction for the registration on building 1. **(Top row):** The 2D image is in a very different orientation with respect to the acquired 3D range model . **(Middle row):** The user rotates the 3D model so that it is orientated similarly (note that it does not have to be exactly matched) to the 2D image. **(Bottom row):** The right image shows the 2D image along with the matched 2D and projected 3D features (see caption of Fig. 5.10). The left image shows the texture-mapped 3D range model after successful registration.

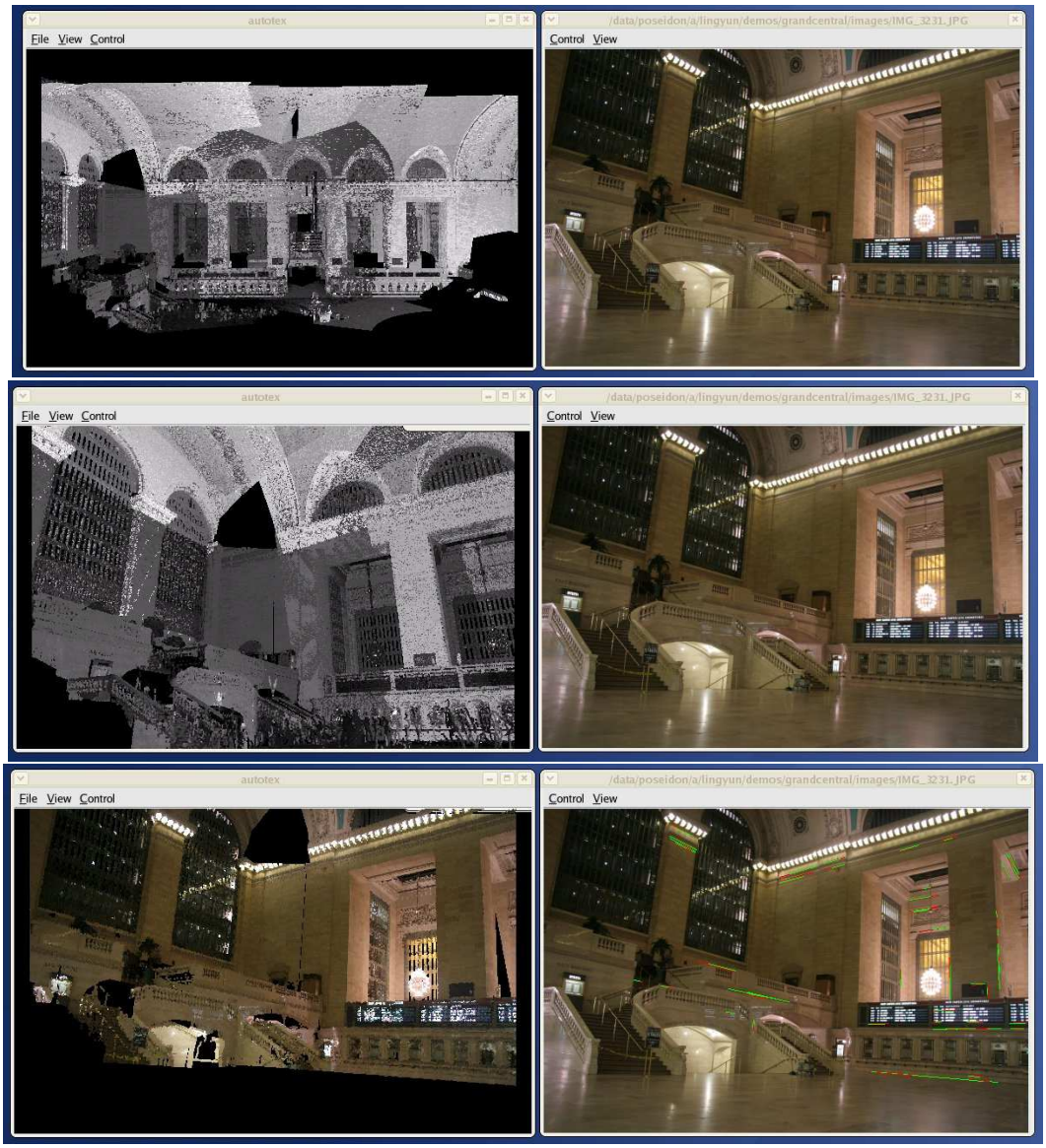

Figure 5.14: User interaction for the registration on building 3. **(Top row):** The 2D image is viewing a small part of the 3D model. **(Middle row):** The user rotates the 3D model so that it is orientated similarly (note that it does not have to be exactly matched) to the 2D image. **(Bottom row):** The right image shows the 2D image along with the matched 2D and projected 3D features (see caption of Fig. 5.10). The left image shows the texture-mapped 3D range model after successful registration. Note that surfaces that are not part of the 3D model cannot be texture-mapped and appear as black holes. For example the floor is missing from our range model.

F3D	F2D	Fi	Fr	M	E
672	412	3065.83	3072.42	119	1.9872
583	345	3065.83	3075.34	103	2.0121
409	390	3065.83	3071.90	112	2.1029
392	230	3065.83	3069.45	93	1.8752
321	312	3065.83	3073.23	187	1.6523
456	387	3065.83	3072.12	134	1.3892
402	390	3065.83	3071.29	94	1.8973
390	219	3065.83	3069.22	87	1.9653
592	539	3065.83	3071.90	212	1.2393
390	416	3065.83	3061.39	145	1.4203
271	392	3065.83	3073.38	123	1.9153
430	456	3065.83	3076.19	209	1.0872
390	549	3065.83	3063.56	115	1.6847
438	789	1185.03	1165.65	114	1.4328
421	654	1185.03	1175.89	83	1.5832
389	520	1185.03	1172.90	88	1.2348
402	432	1185.03	1179.34	101	1.5932
389	598	1185.03	1172.90	91	1.6932
435	621	1185.03	1169.39	156	1.5120
419	535	1185.03	1178.17	182	1.7684
543	245	2805.81	2833.45	63	1.9574
390	190	2805.81	2839.93	50	2.2383
493	231	2805.81	2812.24	63	2.4892
301	189	2805.81	2829.39	58	1.9432

Table 5.1: Building 1 (13 images) - Building 2 (7 images) - Building 3 (4 images). Each row presents results from successful registration (with acceptable evaluation errors) of a different 2D image with the 3D range model. The upper part of the table presents results of the registration of 13 images with a 3D range model of building 1. The middle part shows results from registering 7 images with a 3D range model of building 2. Finally, the lower part describes results from the registration of 4 images with the 3D range model of building 3. The registration (matching phase) of each image requires on average 5 to 10 seconds (2GHz Xeon Intel processor, 2GB of RAM). The first two columns show the numbers of 3D and 2D features used for matching. "Fi" is the initial focal length extracted from the Exif meta-data of the image, while "Fr" is the refined focal length. "M" is the number of matched features of the best transformation. Finally, "E" is the evaluation error which is average line-to-line distance (in pixels) based on the manually selected line correspondences.

F3D	F2D	Fi	Fr	M	E
392	230	3065.83	3069.45	93	24.2109
398	341	3065.83	3072.32	121	13.9034
386	276	3065.83	3068.81	91	9.3920
390	219	3065.83	3069.22	87	8.2134
389	598	1185.03	1172.90	91	7.5009
421	654	1185.03	1175.89	83	22.4323
389	520	1185.03	1172.90	88	43.5432
543	245	2805.81	2833.45	63	33.6743
390	190	2805.81	2839.93	50	28.6434
301	189	2805.81	2829.39	58	19.8653

Table 5.2: Building 1 (3 images) - Building 2 (2 images) - Building 3 (5 images). Each row presents results from a failed registration (with not acceptable evaluation errors) of a different 2D image with the 3D range model. For building 1 (top part), 3 images fail out of total 16 images. For building 2 (middle part), 2 images fail out of total 9 images. For building 3, 5 images fail out of total 9 images. The upper part of the table presents results of the registration of 8 images with a 3D range model of building 1. The reason that cause the failure is the low quality of extracted 3D/2D features. Detailed discussions are in Sec. 5.6. The first two columns show the numbers of 3D and 2D features used for matching. "Fi" is the initial focal length extracted from the Exif meta-data of the image, while "Fr" is the refined focal length. "M" is the number of matched features of the best transformation. Finally, "E" is the evaluation error which is average line-to-line distance (in pixels) based on the manually selected line correspondences.

Chapter 6

Conclusions and Future Work

6.1 Contributions

With the advent and commercialization of accurate range finders, the photorealistic 3D modeling by combining range sensing technology with 2D photography has become an important research area. In this thesis, we have attacked the key problem for this combination, the registration of 2D images with 3D range data, in a unique and efficient manner. With an emphasis on the large-scale urban scenes, a set of algorithms were developed and implemented into a comprehensive registration system, which served as an important tool for photorealistic modeling. Our major contributions are summarized as follows:

Automated Image-to-Range Registration We presented a feature-based

method to automatically register 2D images with 3D range data (Chapters 3 and 5). As we mentioned before, the registration of 2D color image with 3D range data involves two very different domains. In the 2D domain, each pixel on the image is a result of the complex interactions between various lighting sources and the object material. In the domain of 3D range data, each 3D point represents a geometrical location in space. In our method, both domains are abstracted into higher-order linear features. These features are then used for the efficient search for the correspondences and hence solve for the transformation that can bring them into alignment. Images captured by a high-resolution 2D camera, which moves and adjusts freely, are mapped on range data providing photorealistic renderings of high quality.

It should be noted that picking the right kind of features to match can be a challenge. While using higher-order feature can decrease the search space and thus increase the efficiency, it can also narrow the domain of applicability of the algorithm. On the other hand, lower-level features can be applied to more general cases but may greatly increase the search space. An ideal feature selection should consider the trade-off between the size of search space and generality properly. From our experience, switching from the heavily grouped higher-order structures (such as rectangular parallelepiped and rectangles) to slightly grouped linear features

increases the generality of the system without significantly sacrificing efficiency.

Integration of Multiview Geometry for Registration The creation of a photorealistic 3D model of a large-scale urban scene normally requires a collection of 2D images that sufficiently cover the whole scene. We presented an approach that combines the automated image-to-range registration with structure-from-motion. With at least one color image registered with the 3D range data, the remaining set of the 2D images can come into alignment by exploring multiview geometry. This is achieved by aligning the 3D range data with a model derived by structure from motion based on point correspondences. In comparison to registering one color image at a time, our method merges the benefits of multiview geometry and automated image-to-range registration. It compensates the feature-based image-to-range registration and produces the optimal texture mapped 3D model with increased robustness, efficiency and generality.

6.2 Limitations and Future Work

As we discussed before, our image-to-range registration system requires the extraction of at least two major vanishing points and two major 3D directions

from the 2D and 3D data. In addition, the system is based on matching linear features, so it requires a large number of linear structures in the scene. These requirements limit our system to those scenes with strong parallelism constraint. Although this constraint is available for the majority of urban structures, a large amount of man-made architectural structures do not contain such constraint, for example, the Statue of Liberty, the Sydney Opera House, etc. One possible avenue of future exploration is handling scenes of general configurations, including those without any vanishing points or even linear structures. One possible solution is to use the corner points as features for registration. Corner points are more general than lines and exist in most urban scenes (including structures such as the Statue of Liberty or the Sydney Opera House). With sufficiently extracted corner point correspondences, a robust point-based pose estimation algorithm can be used to register 2D images with 3D range data. Of course the major challenge is how to automatically compute the correct correspondences.

For more general cases, the ideal solution for registration is to develop a generic feature descriptor that can enable the comparison between 2D and 3D features. This is one of the most fundamental problems in pattern recognition and still an open research area. As we have mentioned before, there is no guarantee of true match between features extracted from two quite different domains. A possible alternative approach would be to match 2D features from

color images with 2D features that can be derived from 3D range data. The underlying logic to this idea is that for most laser scanners, the 3D range data is associated with a reflectance image, which contains the returned laser intensity. This intensity depends on the factors of view distance, light incident angle and object surface material, which are similar to those that affect the pixel values of the color images. Therefore, it is likely that the features extracted from reflectance image could be comparable to the features from 2D color image. In the domain of 2D-to-2D image registration, the scale invariant feature (SIFT) [Lowe, 2004] has been proven to work well and widely used. It would be interesting to see whether SIFT could also work for the registration of reflectance images with 2D color images.

In recent years, with the emerging of several virtual reality systems, e.g. Google Earth and Microsoft Virtual Earth, the photorealistic modeling of large-scale urban scenes receives more and more attention. Tools for manually generating simplified 3D models are widely available (e.g. Google's Sketch-Up) and user can easily download 3D models from the Internet. However, generating the photorealistic 3D model by texture mapping still requires quite a lot manual work and certain training. It will be interesting to look into this area with our already established system in hand. An interesting future work would be to apply our feature-based registration method and the integration of multiview geometry for the registration of color images with those simplified

3D models. This would introduce new challenges such as how to adjust the manually generated 3D model into the correct scale for the texture mapping and how to combine multiple images to generate seamless texture mappings. We believe this could lead to a very interesting research area to explore.

Bibliography

[Alter and Grimson, 1997] T. D. Alter and W. E. L. Grimson. Verifying model-based alignments in the presence of uncertainty. In *CVPR '97: Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 344, Washington, DC, USA, 1997. IEEE Computer Society.

[Beardsley *et al.*, 1997] P. A. Beardsley, A. P. Zisserman, and D. W. Murray. Sequential updating of projective and affine structure from motion. *International Journal of Computer Vision*, 23(3):235–259, 1997.

[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.

[Caprile and Torre, 1990] B. Caprile and V. Torre. Using vanishing points for camera calibration. *Int. J. Comput. Vision*, 4(2):127–140, 1990.

[Cass, 1997] Todd A. Cass. Polynomial-time geometric matching for object recognition. *Int. J. Comput. Vision*, 21(1-2):37–61, 1997.

[Chen and Stamos, 2005] Chao Chen and Ioannis Stamos. Semi-automatic range to range registration: A feature-based method. *Proceedings of the Fifth International Conference on 3-D Digital Imaging and Modeling*, pages 254–261, 2005.

[Christy and Horaud, 1999] Stéphane Christy and Radu Horaud. Iterative pose computation from line correspondences. *Comput. Vis. Image Underst.*, 73(1):137–144, 1999.

[Curless and Levoy, 1996] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, New York, NY, USA, 1996. ACM Press.

[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 *SIGGRAPH '96: Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 11–20, New York, NY, USA, 1996. ACM Press.

[Dementhon and Davis, 1995] Daniel F. Dementhon and Larry S. Davis. Model-based object pose in 25 lines of code. *Int. J. Comput. Vision*, 15(1-2):123–141, 1995.

[Dhome *et al.*, 1989] M. Dhome, M. Richetin, and J.-T. Lapreste. Determination of the attitude of 3D objects from a single perspective view. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(12):1265–1278, 1989.

[Faugeras *et al.*, 2001] O. Faugeras, Q. T. Luong, and T. Papadopoulos. *The Geometry of Multiple Images.* MIT Press, 2001.

[Faugeras, 1996] Olivier Faugeras. *Three–Dimensional Computer Vision.* The MIT Press, 1996.

[Fischler and Bolles, 1981] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, 1981.

[Geosystems, 2000] Leica Geosystems, 2000. http://hds.leica-geosystems.com.

[Haralick *et al.*, 1989] Robert M. Haralick, Hyonam Joo, Chung nan Lee, Xinhua Zhuang, Vinay G. Vaidya, and Man Bae Kim. Pose estimation from

corresponding point data. *IEEE Transactions on System, Man and Cybernetics*, 19(6):1426–1446, 1989.

[Hartley and Zisserman, 2004] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision.* Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[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 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, page 1106, Washington, DC, USA, 1997. IEEE Computer Society.

[Horaud *et al.*, 1989] Radu Horaud, Bernard Conio, Oliver Leboulleux, and Bernard Lacolle. An analytic solution for the perspective 4-point problem. *Comput. Vision Graph. Image Process.*, 47(1):33–44, 1989.

[Huttenlocher and Ullman, 1990] Daniel P. Huttenlocher and Shimon Ullman. Recognizing solid objects by alignment with an image. *Int. J. Comput. Vision*, 5(2):195–212, 1990.

[Katsushi Ikeuchi *et al.*, 2003] Katsushi Ikeuchi, Atsushi Nakazawa, Kazuhide Hasegawa, and Takeshi Ohishi. The great buddha project: Modeling cultural heritage for vr systems through observation. *International Symposium on Mixed and Augmented Reality*, 00:7, 2003.

[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.

[Kurazume *et al.*, 2002] Ryo Kurazume, Ko Nishino, Zhengyou Zhang, and Katsushi Ikeuchi. Simultaneous 2D images and 3D geometry model registration for texutre mapping utilizing reflectance attribute. In *ACCV '02: the 5th Asian Conference on Computer Vision*, 2002.

[Lensch *et al.*, 2001] Hendrik P.A. Lensch, Wolfgang Heidrich, and Hans-Peter Seidel. A silhouette-based algorithm for texture registration and stitching. *Graph. 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 *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[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 (CVPR'05) - Volume 2*, pages 137–143, Washington, DC, USA, 2005. IEEE Computer Society.

[Liu *et al.*, 1990] Y. Liu, T. S. Huang, and O. D. Faugeras. Determination of camera location from 2-D to 3-D line and point correspondences. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(1):28–37, 1990.

[Liu *et al.*, 2006] Lingyun Liu, Ioannis Stamos, 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*, pages 2293–2300, Washington, DC, USA, 2006. IEEE Computer Society.

[Lowe, 2004] David Lowe. Distinctive image features from scale-invariant keypoints. *Intl. Journal of Computer Vision*, 60(2), 2004.

[Ma *et al.*, 2003] Yi Ma, S. Soatto, J. Kosecka, and S. Sastry. *An Invitation to 3-D Vision: From Images to Geometric Models.* Springer-Verlag, 2003.

[Mit City Scanning, 1999] MIT City Scanning, 1999. http://city.lcs.mit.edu.

[Oberkampf *et al.*, 1996] Denis Oberkampf, Daniel F. DeMenthon, and Larry S. Davis. Iterative pose estimation using coplanar feature points. *Comput. Vis. Image Underst.*, 63(3):495–511, 1996.

[Pollefeys *et al.*, 2004] M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, and R. Koch. Visual modeling with a hand-held camera. *International Journal of Computer Vision*, 59(3):207–232, 2004.

[Pulli *et al.*, 1998] Kari Pulli, Habib Abi-Rached, Tom Duchamp, Linda G. Shapiro, and Werner Stuetzle. Acquisition and visualization of colored 3D objects. In *ICPR '98: Proceedings of the 14th International Conference on Pattern Recognition-Volume 1*, page 11, Washington, DC, USA, 1998. IEEE Computer Society.

[Quan and Lan, 1999] Long Quan and Zhongdan Lan. Linear N-point camera pose determination. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21(8):774–780, 1999.

[Reed and Allen, 1999] M.K. Reed and P.K. Allen. 3-D modeling from range imagery: An incremental method with a planning component. *Image and Vision Computing*, 17:99–111, 1999.

[Rocchini *et al.*, 1999] Claudio Rocchini, Paolo Cignomi, Claudio Montani, and Roberto Scopigno. Multiple textures stitching and blending on 3D objects. In *Eurographics Rendering Workshop 1999*, 1999.

[Schaffalitzky and Zisserman, 2001] F. Schaffalitzky and A. Zisserman. Viewpoint invariant texture matching and wide baseline stereo. *Proc. ICCV*, pages 636–643, July 2001.

[Sequeira and Goncalves, 2002] Vitor Sequeira and Joao G.M. Goncalves. 3D reality modelling: Photo-realistic 3D models of real world scenes. *3dpvt*, 00:776, 2002.

[Stamos and Allen, 2001] Ioannis Stamos and Peter K. Allen. Automatic registration of 2-D with 3-D imagery in urban environments. In *ICCV*, pages 731–737, 2001.

[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 and Leordeanu, 2003] Ioannis Stamos and Marius Leordeanu. Automated feature-based range registration of urban scenes of large scale. *cvpr*, 02:555, 2003.

[Troccoli and Allen, 2004] Alejandro J. Troccoli and Peter K. Allen. A shadow based method for image to model registration. In *CVPRW '04: Proceedings of the 2004 Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'04) Volume 11*, page 169, Washington, DC, USA, 2004. IEEE Computer Society.

[Tsai, 1987] Roger 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(4):323–344, 1987.

[Tuytelaars and Gool, 2004] Tinne Tuytelaars and Luc J. Van Gool. Matching widely separated views based on affine invariant regions. *International Journal of Computer Vision*, 59(1):61–85, 2004.

[VITG, 2000] VITG. Visual Information Technology Group, Canada, 2000. http://iit-iti.nrc-cnrc.gc.ca/about-sujet/vit-tiv_e.html.

[William M. Wells, 1997] III William M. Wells. Statistical approaches to feature-based object recognition. *Int. J. Comput. Vision*, 21(1-2):63–98, 1997.

[Yu, 2000] Y. Yu. Modeling and editing real scenes with image-based techniques, 2000.

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

[Zhao and Shibasaki, 2001] Huijing Zhao and Ryosuke Shibasaki. Reconstructing textured CAD model of urban environment using vehicle-borne laser range scanners and line cameras. In *ICVS '01: Proceedings of the Second International Workshop on Computer Vision Systems*, pages 284–297, London, UK, 2001. Springer-Verlag.

[Zhao *et al.*, 2005] Wenyi Zhao, David Nister, and Steve Hsu. Alignment of continuous video onto 3D point clouds. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1305–1318, 2005.