SIMULTANEOUS REGISTRATION OF 2D IMAGES ONTO 3D MODELS FOR TEXTURE MAPPING
テクスチャマッピングのための ３次元モデルと複数枚画像の同時位置合わせ

by

Ryo Ohkubo
大久保 亮

A Master Thesis
修士論文

Submitted to
the Graduate School of Information Science and Technology
the University of Tokyo
on February 4-5, 2003
in Partial Fulfillment of the Requirements
for the Degree of Master of Information Science and Technology
in Computer Science

Thesis Supervisor: Katsushi Ikeuchi　池内 克史
Title: Professor of Computer Science

ABSTRACT

Recently, creation of realistic 3D contents through sensing the real world has become fundamental for many applications. To enhance 3D geometric models obtained through laser range scanners with their textures reconstructed from several photographic 2D images taken from various view points, it is necessary to determine the camera position and orientation relative to the 3D models for each of the images.

In this thesis, a registration method is proposed, which automatically and simultaneously aligns multiple 2D images onto a 3D model. For each iteration process, correspondences between 2D edge pixels and 3D edge points are automatically searched and updated. Besides these 2D-3D edge correspondences, 2D-2D edge correspondences on 3D surface model are also considered simultaneously for global optimization among all the images. Errors are minimized by using conjugate gradient search, utilizing M-estimator for robustness. From texture mapped objects, the usefulness of the proposed simultaneous registration method is shown. Also, it is applied to the creation of digital cultural assets.

論文要旨

近年、実世界を測定することによって自動的に 3 次元コンテンツを生成する技術の重要性が高まってきている。より現実感の高いモデルを作成するためには、レーザレンジファインダ計測によって得られた 3 次元幾何形状モデルに、写真から得られる物体表面のテクスチャを貼り付けることが有効であるが、そのためには写真を撮影した位置・向きなどを対象物に対して正確に推定する必要がある。

本論文では、3 次元幾何モデルと複数枚のテクスチャ画像間での位置合わせを自動的に行う手法を提案する。画像上の 2 次元エッジピクセルとモデル上の 3 次元エッジ点との対応関係を随時更新しながら、反復解法で求める。またそれら 2D-3D 間での対応関係に加え、3 次元表面上での 2D-2D 間のエッジ対応関係も同時に考慮に入れることで、複数方向からの画像全体の間で整合のとれた最適化を行うことができる。誤差最小化計算には共役勾配法を用い、またロバストに動作するように M-推定法を使用する。テクスチャマッピングへ適用例から、本同時位置合わせ手法の有効性を示す。また、文化財のデジタルアーカイブへの応用例も示す。

Acknowledgements

I would like to express my sincere gratitude to professor Katsushi Ikeuchi for his valuable advice. I would also like to thank the following: Ryo Kurazume for his previous work of the texture mapping, Ko Nishino and Hiroki Unten for their helpful suggestions, and Robby T. Tan for reviewing this thesis. And finally, I am grateful to all the members of Ikeuchi Laboratory.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

In recent years, widespread demand for 3D contents have been greatly increased in many areas: computer graphics, entertainment, E-commerce, preservation of cultural assets, ITS(Intelligent Transportation System), etc. However, most of them are created manually by human experts using 3D modeling systems and this input process is normally very time-consuming. To simplify the process, some research have been investigated to aid designers through novel human interfaces, like SKETCH[34] and Teddy[11].

On the other hand, in many situations, to obtain 3D models by observation of real world objects is much more convenient and reasonable. One obvious example is the faithful modeling of cultural heritages. In this case, it is essential to create realistic 3D models by measuring those objects through sensors. Recently, such measuring-techniques and algorithms for processing acquired data have been rapidly developed by many researchers.

The term of "3D model" can be classified into three detailed categories: geometric model, physical model and environmental model. Therefore, to acquire the complete 3D model through observation, several processes are necessary and vast numbers of studies have been made in wide fields.

A geometric model represents the shape of objects. It is usually composed of the vertices and meshes structure (or sometimes by voxels). To build these data, several steps are necessary. First, several range images are measured by laser range scanners from various viewpoints and directions. For each pixel of a range image, the distance

to the object at respective direction is stored. Therefore one range image contains shape information from one direction. Next, registration calculation is applied, which aligns multiple range images from various viewpoints to obtain the whole shape [2, 4, 24, 21]. Finally, they are merged to form a unique consensus surface of the object [17, 25].

A physical model represents colors and reflectance properties of surfaces and is an essential factor for rendering. There are numbers of research for decades and various reflection models have been studied [12]. However, the pursuit of the exact analysis is significantly difficult because the radiance observed in the scene is caused by complex interactions among surface intrinsic colors, surface reflection functions, viewing position, illumination conditions, inter-reflection, etc. Recently, several novel methods have been proposed to model realistic appearances of real objects utilizing 3D geometric models [29, 22].

An environmental model includes illumination distributions and interactions between surrounding objects (like shadows and inter-reflection), and plays an important role to achieve the mixed reality. Although it is quite difficult to formulate such a model, several approaches have been investigated lately. Global illumination is measured using a fisheye lens or a mirror ball, so that virtual objects are seamlessly synthesized onto an image of a real scene with correct shadings [27, 5, 6]. High dynamic range radiance maps which are supposed to be necessary in illumination measurements, are recovered from multiple photographs [19, 7]. Imari et al. [28] have directly estimated the illumination distribution of a real scene from a radiance distribution inside shadows cast by an object in the scene.

Recently, much interest has been focused on a physical model since nowadays the geometric models can be obtained accurately, and the need for realistic rendering of these geometric objects has increased.

Although there are various algorithms to recover detailed physical properties, the texture mapping method is a good compromise between the complexity and the quality of appearance. It does not require a large number of photographs which are usually necessary to obtain more complex physical models, and makes the measurement process easier and more practical for the wide range of applications. Indeed, the restrictions at the measurement time can become a big bottle neck in practice. Another advantage of the texture mapping method is that it can be processed entirely by normal 3D graphics hardwares.

However, for the texture mapping and the other methods which acquire the photo-

metric attributes of 3D geometric models using 2D photographic images, it is necessary to know camera positions and directions relative to the 3D geometric models. In other words, camera parameters are required to map the coordinates between the 3D world and the 2D images.

1.2 Obtaining Camera Parameters

In most researches concerning the physical models, camera parameters have been usually assumed to be known. They can be estimated using camera calibration, which is a well-studied problem [10, 32, 36, 9]. In addition, there exist some 3D scanners which can obtain both a range image and a photometric image at the same time, which means the precise camera parameters relative to the 3D geometry are always known for each measurement. However, there are several drawbacks in the use of camera calibration and these 3D scanners, and we cannot always assume camera parameters to be known. First, although camera calibration methods are practical for the experiments taken place in the laboratories, it is inconvenient and often quite difficult to use them in the outdoor environments, especially in large-scale environment. Second, in the case of 3D-2D integrated sensors, 2D capturing systems attached to such sensors are often inferior than normal digital cameras. The image captured by them has worse quality and lower resolution, and they cannot allow sufficient configurations of capturing system like shutter speed, aperture, etc. Mounting a separate high-quality digital camera on a 3D scanner and fixing their relationship completely can solve this problem. Relative camera parameters against the 3D scanner can be calculated by camera calibration beforehand and such a system can emulate 3D-2D integrated scanners. Indeed, it can become a practical solution in many cases. Even so, there are several situations where it is favorable or necessary to take photographs separately from 3D geometry. Generally, the required sampling density for 2D photometric images is often different from 3D geometry, so extra capture of 3D data may burden the capacity and the processing time. Furthermore, the measuring situation in practice often causes various constraints and may make it impossible to use such large-scale devices: e.g., the measurements of the unfavorably located objects like cultural heritages, the measurements under controlled lighting conditions, etc.

Under the condition of uncalibrated cameras, 2D-3D registration is necessary to estimate camera parameters. There have been numerous 2D-3D registration researches and

their aims are not necessarily restricted to the texture mapping, e.g., for object recognition, robot navigation, medical image processing, and etc.

2D-3D registration algorithms require some kinds of information about correspondences between 2D features and 3D features. The simplest correspondence information is specified by a set of point pairs between the 2D image and the 3D geometric model. From these correspondences the camera parameters for the 2D image can be directly calculated using standard camera calibration algorithms [10, 32, 36]. However, the problem is to find these points and pairs. Without using markers, it is difficult and not robust to detect these points and pairs automatically through image processing techniques. Therefore, specifying a set of corresponding pairs manually, i.e., the pixels on the 2D image and the corresponding points in the 3D geometries, is a commonly used approach [23, 1, 20]. Since the accuracy of the obtaind camera parameters heavily depends on the accuracy of point-pairs specified by the user, Neugebauer et al. [20] have refined the registration results by considering the outline of the object and the intensities of images. Instead of using a set of point pairs, a set of corresponding lines is also used to derive the camera parameters of each image [31]. They extract planar regions from the range image and a 3D line is obtained by the intersection of these regions. It is manually matched to a 2D line extracted from the image.

Debevec et al. [8] have used simple predefined models like a box and a wedge, to recover both camera parameters and 3D geometries from only photographic images. By manually specifying locations of parametric primitives for each photograph, both primitive parameters and camera parameters can be obtained at the same time.

Although the methods which need 2D-3D correspondence information specified manually by the user are robust and practical in some cases, they require tedious labor and they would fail when the number of input photographs increases. For this reason, automatic algorithms to create these correspondence information are investigated. Instead of directly searching corresponding features like points and lines, which are not usually robust and practical process, the methods which use more structured features such as contours and edges and take the error minimization framework are proposed.

Lavallée et al. [15] have proposed a registration method which use the outline of a 3D object from volumetric medical data. The pose of a 3D smooth surface is estimated by minimizing the distance between a 3D object surface and the projection of camera-contours in 2D X-ray projections.

In the field of robot vision, the pose information of the object is estimated by the

edgel correspondences [33]. The edgel is the element of the edge and their correspondences are automatically searched and updated through iterative calculations. Based on this algorithm, the registration methods for texture mapping have been studied [14, 13] and this thesis also utilizes this technique.

Lensch et al. [16] have used the silhouette information. The silhouette of a 3D object generated by the 3D geometric model and the silhouette extracted from the 2D image are compared and their distance is minimized by using downhill simplex method. It can utilize the acceleration of graphics hardwares for a calculation speed and register the image without user intervention through multi-resolutional approach. However, extracting the exact silhouette from 2D image is very difficult in real outdoor environments.

A few works mentioned above [16, 20, 13] have also considered the global registration problem. Besides registering each 2D image respectively, they also consider a multi-view global optimization. This is because even if one 2D image is thoroughly registered to the 3D object in the error metric of that viewpoint, it does not necessarily mean it is globally optimal. Due to various errors such as the inaccuracy of 3D geometry, the resolution of pixels, lens distortions, etc., it is impossible to make exactly correct registration. Therefore, the errors always exist and they need to be distributed globally. Otherwise, when textures are mapped using multiple photographs, undesirable artifacts may be caused around the boundary where textures from different views intersect.

In [20] and [16], the points on the 3D surface which are visible from multiple images are used for the optimization of 2D-2D registration. For such points, the former method calculates a 3D euclidean distance to the nearest edge on each visible image and minimizes these differences. The latter uses the difference of colors projected from each visible image. For the color component, hue and saturation channels are used to reduce the influence of specular.

Kurazume et al. [13] have used the technique of epipolar geometry, instead of minimizing the differences of image attributes on the 3D surface points. It extracts the point correspondences between adjacent images using KLT method [30] and calculates the relative camera transformation and the epipolar lines of corresponding point pairs [35]. For the global registration, the sum of distance between the point and its corresponding epipolar line on each image is considered. However, finding corresponding points between two images is a very difficult task, so directly depending on these results makes the registration process not robust. Further, when epipolar lines are almost parallel, which is

often the case in adjacent photographs taken closely, this method does not work along the direction of these lines.

1.3 Thesis Overview

In this thesis, a novel registration method is proposed, which automatically and simultaneously aligns multiple 2D images onto a 3D model. Throughout iterative calculations, the correspondence information between 2D edge pixels and 3D edge points are automatically searched and updated. Therefore, there is no need to specify corresponding points or lines manually. In addition, the global optimization among all the images are also executed by the simultaneous registration of 2D-2D edge correspondences on 3D surfaces. Outliers are eliminated using M-estimates and the errors are minimized by conjugate gradient search. Registration results are shown with the texture mapped objects and the usefulness of the proposed simultaneous registration method is shown. In addition, the application for the creation of digital cultural assets is also presented.

The remainder of this thesis is organized as follows. In Chapter 2, mathematical notations and camera parameters which are used in this thesis, are explained. In Chapter 3, a registration algorithm concerning the single 2D image and the 3D geometric model is presented. In Chapter 4, multiple 2D images are simultaneously registered to the 3D model through the global optimization. In Chapter 5, experiments are shown and results are examined. Finally, in Chapter 6, the summary and future work are mentioned.

Chapter 2

Preliminaries

2.1 Mathematical Notation

- Vectors are expressed in boldface type: $\mathbf{x}$ is a vector, x is a scalar.
- Unit vectors have the hat symbol: $\hat{\mathbf{x}}$ is a unit vector.
- Matrices are expressed by the capitalized and boldface character: $\mathbf{M}$ is a matrix, and especially, $\mathbf{I}$ is the identity matrix and $\mathbf{R}$ is a rotation matrix.
- $\mathbf{x}$ will be used to denote the 3D coordinate of the 3D geometric models.
- $\mathbf{y}$ will be used to denote the 2D coordinate of the 2D photometric images.
- $\mathbf{U}$ will be used to denote the 2D coordinate which is projected from the 3D world (note, this is the only vector that will be capitalized).
- Vectors should be assumed to be three dimensional except for the above.

2.2 Camera Parameter

The 3D geometric objects are located in the world coordinate system and the camera is also located in the same world, viewing the objects. Seen from the camera, the coordinates of objects are expressed in the camera coordinate system. They are illustrated in Figure 2.1 The camera is located at C and this point is named “focal point”. Z_c represents the viewing direction.

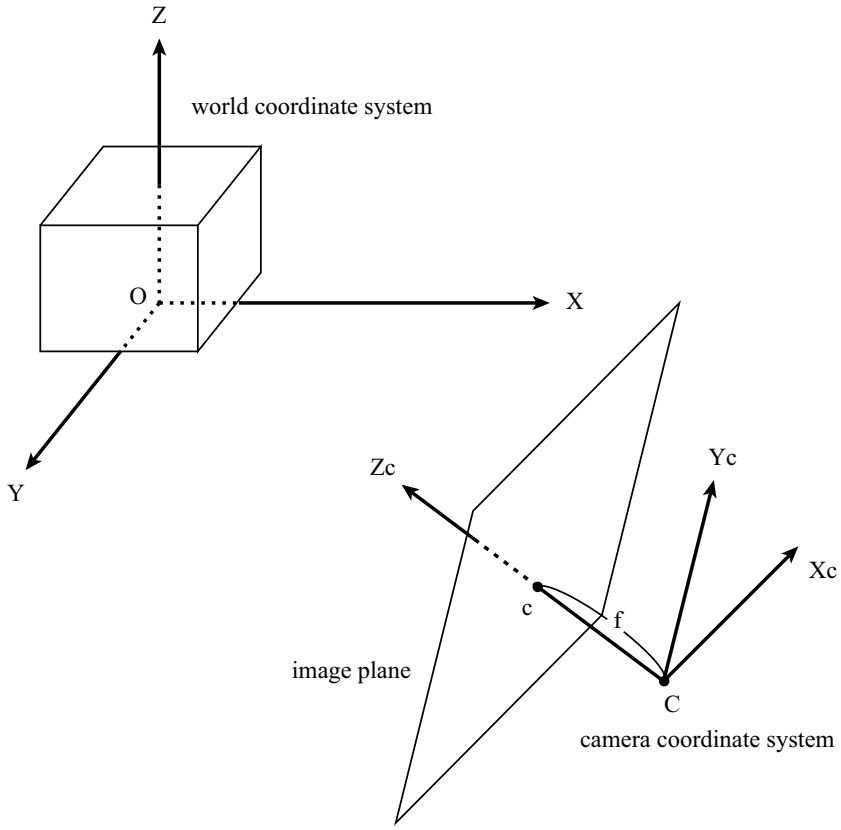


Figure 2.1: The world coordinate system and the camera coordinate system

The transformation between world and camera coordinates can be described with the set of rotation and translation, $\langle \mathbf{R}, \mathbf{t} \rangle$. Since they represent the camera position and orientation, they are called "camera extrinsic parameters". Let a 3D point in the world coordinate be $\mathbf{x}_w = (x_w, y_w, z_w)$. Then, the coordinate of the point in the camera coordinate system, $\mathbf{x}_c = (x_c, y_c, z_c)$, is expressed as follows:

$$\begin{pmatrix} \mathbf{x}_c \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_w \\ 1 \end{pmatrix} \tag{2.1}$$

The photographic image can be obtained by projecting the camera-centered view onto the image plane (in Figure 2.1). The point *c* at which the viewing direction and the image plane intersect, is named the "principal point", and the distance between that point *c* and

the optical point C is called the "focal length". Let the projected 2D point on the image plane be $\mathbf{U}$, then the projection equation can be written as follows:

$$\mathbf{u} = \mathbf{P} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_w \\ 1 \end{pmatrix} \tag{2.2}$$

$$= \mathbf{P} \begin{pmatrix} \mathbf{x}_c \\ 1 \end{pmatrix} \tag{2.3}$$

$$\text{where } \mathbf{u} = \begin{pmatrix} u \\ v \\ w \end{pmatrix}, \ \mathbf{U} = \begin{pmatrix} \frac{u}{w} \\ \frac{v}{w} \end{pmatrix} \tag{2.4}$$

$\mathbf{P}$ is a 3×4 projection matrix and contains various parameters. They are called "camera intrinsic parameters", and details are shown below.

$$\mathbf{P} = \begin{pmatrix} k_u & -k_u \cot\theta & u_0 \\ 0 & k_v / \sin\theta & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.5}$$

They consist of the focal length, principal point, aspect ratio, and skew.

As we have seen, there are two kinds of camera parameters: the intrinsic parameters and the extrinsic parameters. However, estimating both the extrinsic parameters and the intrinsic parameters simultaneously makes the registration process unstable and not robust. Therefore, only the extrinsic parameters, i.e., the camera rotation and translation $\langle \mathbf{R}, \mathbf{t} \rangle$ are optimized in this thesis. To robustly refine the focal length along with the registration process remains one of the future work.

Aside from the 2D-3D registration process, the camera intrinsic parameters need to be estimated using the camera calibration method. Among intrinsic parameters, important components are the focal length f, and the principal point (u_0, v_0). Although the precise parameters are only acquired through camera calibration, we can obtain their approximate estimates in a easy way. First, the skew and the aspect ratio can be ignored in the modern digital cameras. And often, the principal point are also presumed to be $(0, 0)$. Further, the approximate focal length can be obtained by EXIF (Exchangeable Image File) and DCF (Design rule for Camera File system) data which are recorded in JPEG/TIFF files captured by digital cameras.

In practice, besides the camera intrinsic and extrinsic parameters, lens distortions also affect the obtained photographic image. They primarily consists of the radial distortions and the tangential distortions, which are especially outstanding when the wide-angle lenses or the small handy cameras are used. Lens distortions can be estimated by various camera calibration methods and they should be removed before any image processing.

2.3 Quaternion Representation

In the following Chapters, the set of camera parameters to be estimated is expressed as the vector $\mathbf{p}$. It consists of the camera extrinsic parameters, that is, the camera position and the camera orientation. In general, it is convenient to represent them as the set of the camera rotation matrix and the camera translation vector, $\langle \mathbf{R}, \mathbf{t} \rangle$.

However, representing a rotation as the matrix form, $\mathbf{R}$, causes a great difficulty in the computation of the optimal rotation. While a rotation in 3D space has only three degrees of freedom, a rotation matrix has nine degrees. This restricts the values of $\mathbf{R}$ in a non-linear way as follows:

$$\mathbf{R}\mathbf{R}^T = I \tag{2.6}$$

$$|\mathbf{R}| = 1 \tag{2.7}$$

$\mathbf{R}$ must always satisfy these constraints to represent a rotation and this makes difficult to take advantage of the linear matrix form of rotation.

The generally accepted alternative for the representation of rotation is the use of quaternion. A quaternion is a 4-vector, consisting of a 3-vector $(u, v, w)^T$ and a scalar s, that is, $\mathbf{q} = (u, v, w, s)^T$ and it can represent an arbitrary rotation in the 3D space. It has several useful characteristics.

- The constraint of rotation is easily maintained by standard vector normalization.
- The inverse rotation is obtained by simply negating first 3 components of the quaternion vector.
- It can avoid the gimbal lock problem. Roughly speaking, the continuous change of the elements always lead to the smooth change of rotation, and vice versa.
- The intermediate rotation between two quaternions can be calculated linearly.

- With the quaternion representation, the rotation between two sets of corresponding 3D points can be solved in closed form.

In addition, another important advantage of the quaternion representation is utilized in Section 3.7

Thus, the following vector is used to express the camera parameters:

$$\mathbf{p} \quad = \quad (\mathbf{q}^T \ \mathbf{t}^T)^T \tag{2.8}$$

where **p** is a 7-vector, **q** is a quaternion representing a camera rotation, and **t** is a 3-vector representing a camera translation. If necessary, the form of rotation matrix is also used and the rotation matrix corresponding to **q** is denoted by **R**(**q**).

Chapter 3

2D-3D Registration Algorithm

In this chapter, the registration method which optimizes the camera position and orientation of a 2D texture image with respect to the 3D geometric models, is described. It is accomplished through the iterative algorithms. In each stage, corresponding 2D-3D point pairs are automatically searched and the estimated camera parameters are updated. In addition, the robust estimation framework is used to eliminate the unfavorable effects of outliers.

3.1 Outline of 2D-3D Registration

Nowadays, we can capture the precise 3D geometric models through sensing the real world objects. In addition, 2D photographic images of those objects can be easily obtained with digital cameras. The 2D-3D registration shown in this chapter is the problem to estimate the camera positions and orientations from which the photograph is taken, and to make the correspondence between 3D geometries and 2D photometric attributes (colors, etc.). The camera parameters consist of the camera rotation and translation and are written as $\mathbf{p} = (\mathbf{q}^T,\ \mathbf{t}^T)^T$.

To align the 2D image with the 3D model, the edge features of the 2D image and the edge features of the 3D model are considered. The outline of the registration algorithm is shown in Figure 3.1. The “edgel” refers to the edge element (cf. pixel as the picture element). First, 2D edgels and 3D edgels are extracted from the 2D image and the 3D geometric model. Next, their correspondences are automatically searched and then, camera parameters are adjusted to minimize their distances. After that, the new 3D edgels

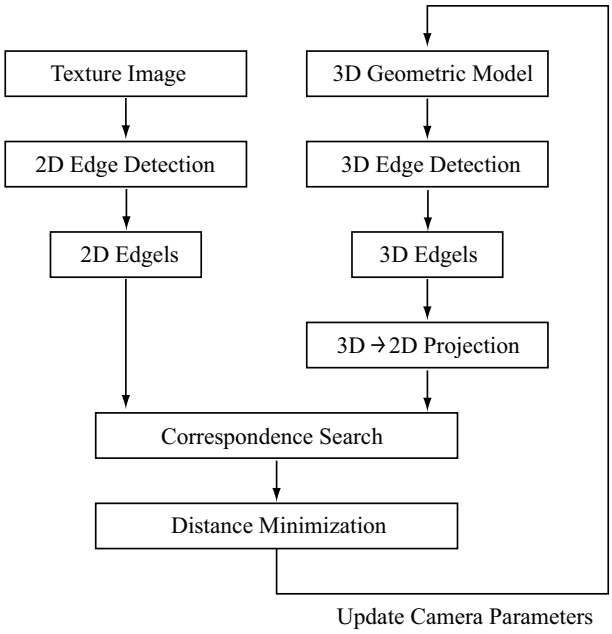


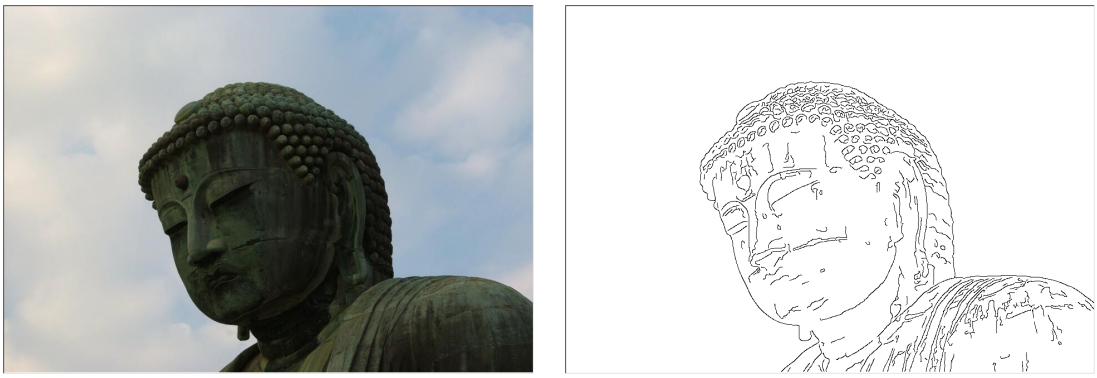
Figure 3.1: Outline of the 2D-3D registration algorithm

are detected using the newly estimated camera parameters and the above processes are repeated iteratively.

Note that in this 2D-3D registration algorithm the 3D geometric model is not necessarily restricted to the one object. We can assume many objects as long as they provide the 3D edgels, so this method is applicable to the outdoor environment, too. However, in the presence of multiple objects, especially when they are located at the different distances, the small change of camera parameters is likely to cause the large separate movements of objects. Therefore, the algorithm is supposed to be not so robust compared to the environment of the indoor experiments, and the importance of initial position specification grows.

3.2 2D Edgels

The detection of 2D edgels from the 2D photographic image is the very important stage in the registration algorithm. To achieve the stable and robust registration, well-structured edges are crucial. If the edges are too scattered and dense, the mismatch rate of 2D-3D correspondences will expand. Simple edge detection methods such as the Sobel operator is likely to cause such noisy edges. In out experiment, we use the Canny edge detector [3]. The example of texture image and the result of 2D edge detection are shown in Figure 3.2. Each edge pixel drawn as the black pixels in Figure 3.2(b) constructs the 2D edgel. Note that 2D edgels do not change throughout the whole registration process and they are detected only once.



(a) (b)

Figure 3.2: Result of the 2D edge detection: (a) original 2D photographic image, (b) edge image of (a). Each black pixel on (b) constructs the 2D edgel.

3.3 3D Edgels

Since the appearance of the 3D geometric models changes as the estimated camera viewpoint changes, the 3D edgels have to be detected at every iteration. The desirable characteristics of the 3D edgels are

- They should have the similar edge structures as the 2D edgels (similarity).

- They should contain sufficient details to register the image precisely, while they should not have too much minor junks (density).
- They should be robustly detected from various kinds of 3D models, e.g., the models might be noisy (robustness).

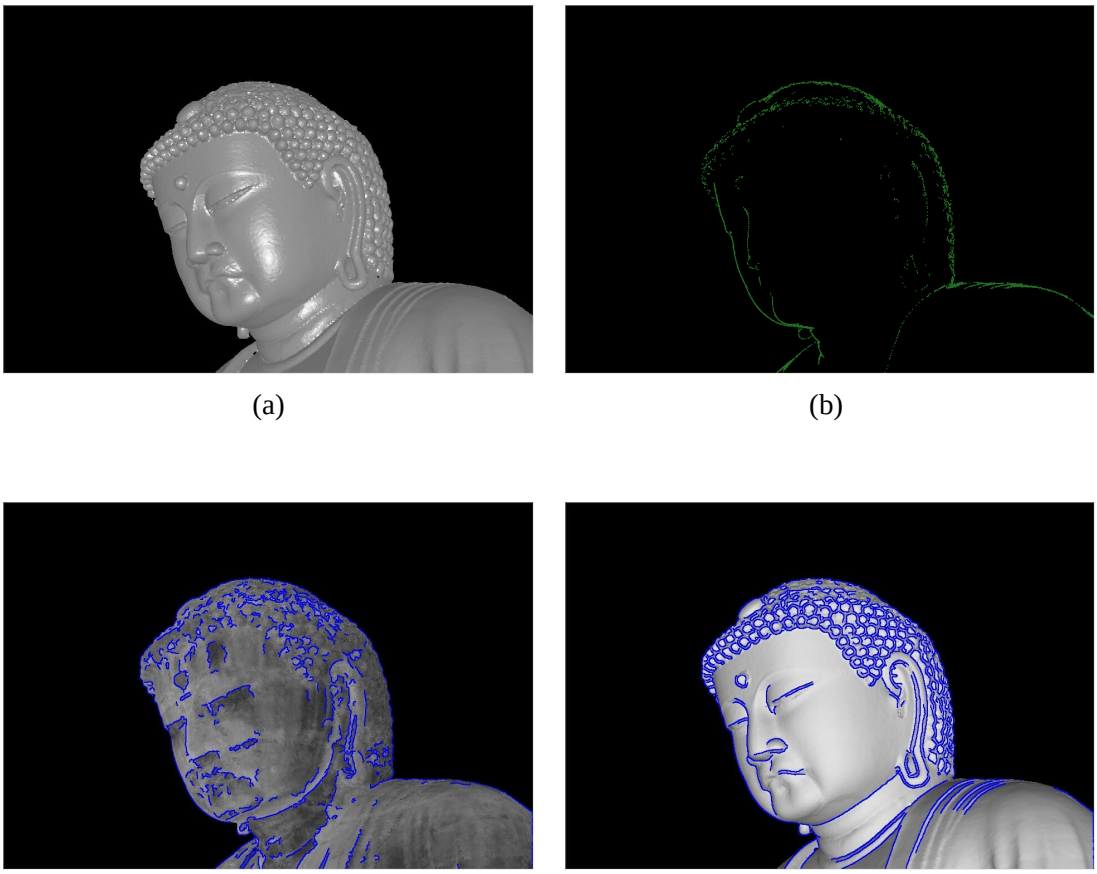
Considering these conditions, the three types of 3D edgels are proposed in this thesis: occluding edgels, reflectance edgels, and rendered edgels (in Figure 3.3).

1. Occluding Edgels:
   Occluding edgels are detected around the surfaces whose normal are almost perpendicular to the viewing direction (in Figure 3.3(b)). They are supposed to cause the distance gap and can be seen as the edge. To reduce the effects of noise, surface normals are calculated by the PCA (Principal Component Analysis) method around the neighboring vertices. Although the occluding edgels can be detected robustly, they are likely not to have much information to align details.

2. Reflectance Edgels:
   Usually, in the process of measuring 3D geometric objects by using the laser range finder, the data concerning the reflectance ratio of the laser are also obtained. Since the reflectance values have already corresponded to the 3D geometries, it is reasonable to use these values for the registration. The change of reflectance ratio results from the difference of the surface material, which also causes the change of surface colors. Therefore, the edges of the reflectance values are supposed to have the similar structures as the edges of 2D photometric image and used as the 3D edgels (in Figure 3.3(c)).

3. Rendered Edgels:
   Although occluding edgels do not have much information, reflectance edgels have sufficient details and their combination works well. However, we cannot assume the 3D geometric data always contain the reflectance values. For example, if we use different kinds of laser range finders at the same time, the consistent reflectance values cannot be obtained. Such situation easily occurs in practice because we would need various kinds of sensors such as the accurate one for neighborhood measurement and the wide-angle one which covers the wide range of distance.

   Therefore, an alternative method to detect detailed 3D edgels becomes necessary. One possibility is to detect edgels using geometric features such as the curvature



(a) (b)

(c) (d)

Figure 3.3: Example of the three types of 3D edgels: (a) original 3D geometric model, (b) detected occluding edgels, (c) reflectance values of the laser range sensor and the edgels they form, (d) edgels obtained by the rendering result.

of the surface. However, these methods are very sensitive to noise and many undesirable junk edgels would be detected. To overcome this problem, the rendered edgels are proposed here. Instead of detecting the features directly from the 3D geometric model, edges are detected from the rendering result. In the rendering process, the 3D surface is assumed to be Lambertian (no specular highlights) and the smooth shading is executed, so that the unnecessary edges should disappear. After the rendering result is obtained, the edgels are detected by the Canny edge filter. As a result, the edge structures are supposed to be similar to the one which results from the 2D edge detection, and also they have enough density of edgels.

These three types of 3D edgels are used properly and in combination.

3.4 2D-3D Correspondence

After detecting both 2D edgels and 3D edgels, their correspondences are searched. In advance, the visibility of 3D edgels has to be checked, because only part of the 3D edgels can be observed from the camera viewpoint of the 2D image. This visibility checking stage utilizes the z-buffer resulting from the rendering process. Each 3D edgel is transformed to the camera-centered coordinate and if it is located within some threshold range from the z-value, it is marked as visible.

Then, each visible 3D edgel is projected to the 2D image coordinate using the currently estimated camera parameters. The nearest 2D edgel is searched according to the 2D Euclidean distance and the pairs of 2D-3D edgels are established.

3.5 Error Metric of Corresponding 2D-3D Pairs

Given a set of N corresponding points $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$, where $i = 0, ..., N-1$ and $\mathbf{x}_i$ is a 3D edgel and $\mathbf{y}_i$ is a 2D edgel, the registration problem is to compute the camera parameters $\mathbf{p}$, i.e., the camera rotation and translation $\langle \mathbf{R}, \mathbf{t} \rangle$, which aligns the projections of 3D edgels $\mathbf{x}_i$ with 2D edgels $\mathbf{y}_i$. The projection of $\mathbf{x}_i$ is written as

$$\mathbf{u}_i = \mathbf{P} \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0^T} & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_i \\ 1 \end{pmatrix} \tag{3.1}$$

$$\mathbf{u}_i = \begin{pmatrix} u_i \\ v_i \\ w_i \end{pmatrix}, \ \mathbf{U}_i = \begin{pmatrix} \frac{u_i}{w_i} \\ \frac{v_i}{w_i} \end{pmatrix} \tag{3.2}$$

where $\mathbf{P}$ is a 3×4 projection matrix, and $\mathbf{U}_i$ is the coordinate of the projected point on the 2D image.

To facilitate further analysis, several assumptions are made in the following equations: the focal length is unity, the principal point lies exactly at $(0, 0)$ on the image, the aspect ratio is unity and the skew is zero. These assumptions can be done without loss of generality. Thus, the projection equation 3.1 is simplified to

$$\mathbf{u}_i = \mathbf{R}\,\mathbf{x}_i + \mathbf{t} \tag{3.3}$$

$$\mathbf{u}_i = \begin{pmatrix} u_i \\ v_i \\ w_i \end{pmatrix}, \ \mathbf{U}_i = \begin{pmatrix} \frac{u_i}{w_i} \\ \frac{v_i}{w_i} \end{pmatrix} \tag{3.4}$$

One way of defining the error metric of corresponding 2D-3D point pairs is the squared distance on the 2D image.

$$z_i = \|\mathbf{U}_i - \mathbf{y}_i\|^2 \tag{3.5}$$

However, it does not take the distance to the 3D point $\mathbf{x}_i$ into account, and it only accounts for the direction of the 3D point. Consequently, it would favor parts of the 3D edgels that are closer to the camera.

Instead of using a 2D error metric, a similar 3D error metric can be considered. It can be expressed as the distance between a 3D edge point and a line connecting the focal point to a 2D edge point. Figure 3.4 shows an example of such a point and a line. Let $\hat{\mathbf{v}}_i$ be the unit vector of that line, i.e., the viewing direction to a 2D edge point $\mathbf{y}_i$ from the focal point. Now we can determine the closest point on that line to the 3D edge point $\mathbf{u}_i$ ($\mathbf{x}_i$ is transformed into $\mathbf{u}$ for the camera-centered coordinates).

$$\mathbf{y}'_i = (\mathbf{u}_i \cdot \hat{\mathbf{v}}_i)\,\hat{\mathbf{v}}_i \tag{3.6}$$

Subsequently, the error z_i is expressed as follows.

$$z_i = \|\mathbf{u}_i - \mathbf{y}'_i\|^2 \tag{3.7}$$

This error computation is now in 3D rather than 2D.

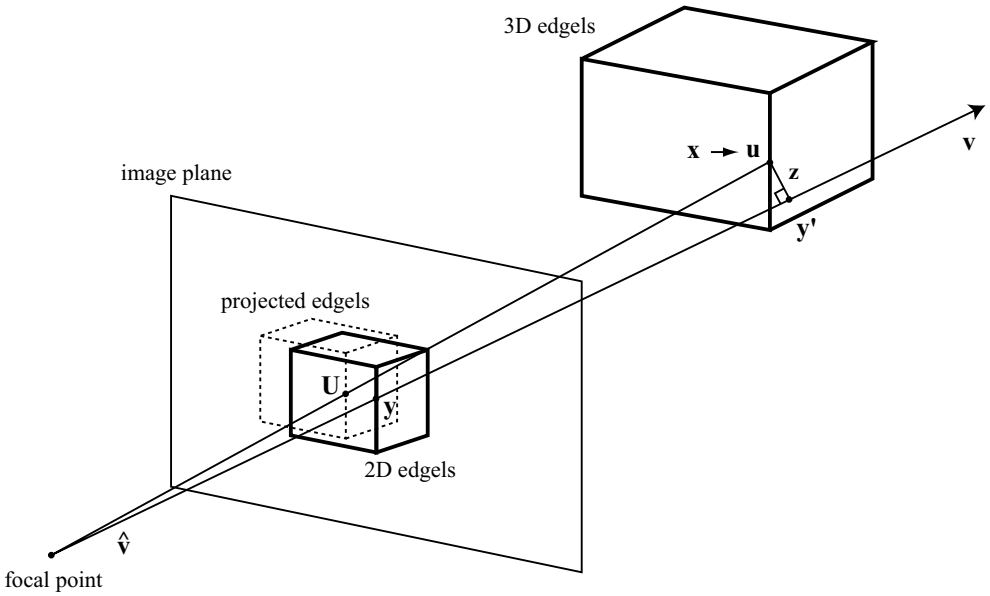


Figure 3.4: The error metric of corresponding 2D-3D edgels in 3D space. The 3D euclidean distance between the 3D edgel **u** and the line stretching to the 2D edgel **y** is used for the error metric.

3.6 Robust Estimation

In the registration process, the camera position and orientation are updated toward the direction which reduces the sum of corresponding 2D-3D errors.

$$E(\mathbf{p}) = \sum_i z_i(\mathbf{p}) \tag{3.8}$$

$$= \sum_i \|\mathbf{u}_i - \mathbf{y}'_i\|^2 \tag{3.9}$$

where $\mathbf{p}$ is the camera extrinsic parameters that we want to estimate, and $E(\mathbf{p})$ is the evaluation function. This form of the equation 3.9 represents the least squares estimation of parameters $\mathbf{p}$.

However, it is not practical to use the above formulation directly since the least squares method is very sensitive to the outliers and the estimated parameters tend to

be strongly biased by them. This is because the least squares method is the maximum-likelihood estimator which assumes that the errors are distributed according to the normal distribution function. In the problems of computer vision, there exist much more outliers which can have fractionally large departures than expected in the normal distribution. Furthermore, much worse situation is expected in this case. Since the corresponding point pairs are automatically searched, a large part of them is supposed to be incorrectly matched, especially in the initial stage of registration.

Outlier thresholding is the simplest and commonly used technique to remove outliers. It regards the data values outside some range as outliers and simply eliminates those data points. The range is often determined by estimating the standard deviation σ of the errors in data and the value $k\sigma$ is used for thresholding, where k is typically greater than or equal to 3. Although it is computationally easy and cheap, there are significant problems. One problem is that the hard threshold is used to eliminate the outliers. This means, regardless of where the threshold is chosen, some of the valid data are rejected as the outliers and some of the outliers are classified as valid. In addition, the hard threshold makes the objective function discontinuous and causes the difficulties for the numerical optimization. The other problem is that in our case the initial correspondences are supposed to be highly incorrect. Therefore, both valid data and incorrect data may exist in the same range and distinguishing them may be meaningless.

To deal with outliers, various sorts of robust statistical estimators have been studied. The two representative classes of robust estimation are the least-median-of-squares (LMedS) method and the M-estimation.

The former class, LMedS method estimates the parameters by solving the following non-linear minimization problem.

$$E(\mathbf{p}) = \underset{i}{\text{med}}\ z_i(\mathbf{p}) \tag{3.10}$$

$$\mathbf{p} = \arg\min_{\mathbf{p}} E(\mathbf{p}) \tag{3.11}$$

The concept of LMedS is to select the median value of the errors for each observation and use that value as the error value at the current parameters. The logic behind this is that the median is almost guaranteed not to be an outlier as long as half of the data is valid. Essentially, this requires an exhaustive search of possible values $\mathbf{p}$, by testing least-squares estimates of $\mathbf{p}$ for all possible combinations of matches between 2D-3D

edgels. Although this median based technique can be very robust, its computation cost is extremely high.

The M-estimation is another representative method for robust estimation and it is used in this thesis. The "M" refers to maximum-likelihood estimation and the arbitrary error model can be used. Assuming that each error z_i is independently random and it is observed according to the probability density **P**,

$$P = \prod_i e^{\rho(z_i)} \tag{3.12}$$

the maximum-likelihood parameters can be obtained by minimizing the following objective function,

$$E = \sum_i \rho(z_i) \tag{3.13}$$

where $\rho(z) = -\log P(z)$.

For example, assuming that the errors z_i follow the normal distribution, they are written as follows.

$$P \propto \prod_i e^{-z_i^2}, \quad \rho(z) = z_i^2 \tag{3.14}$$

$$E = \sum_i z_i^2 \tag{3.15}$$

Notice that this is equivalent to the least-squares formulation.

Using the framework of M-estimation, our evaluation function (Equation 3.8) can be modified to

$$E(\mathbf{p}) = \sum_i \rho(z_i(\mathbf{p})) \tag{3.16}$$

By taking the derivative of E with respect to **p** and setting it to 0, the parameters **p** that minimize E can be obtained.

$$\frac{\partial E}{\partial \mathbf{p}} = \sum_i \frac{\partial \rho}{\partial z_i} \cdot \frac{\partial z_i}{\partial \mathbf{p}} = 0 \tag{3.17}$$

By substituting

$$w(z) = \frac{1}{z}\frac{\partial \rho}{\partial z} \tag{3.18}$$

we get

$$\frac{\partial E}{\partial \mathbf{p}} = \sum_i w(z_i)\, z_i \frac{\partial z_i}{\partial \mathbf{p}} = 0 \tag{3.19}$$

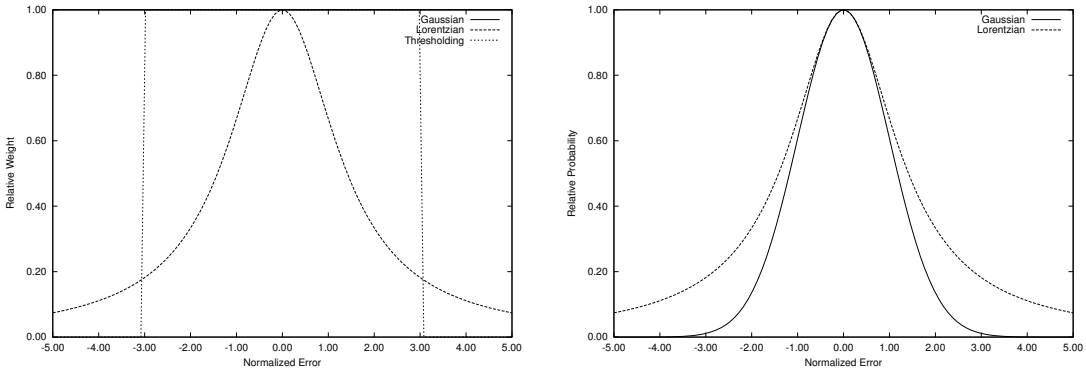
Function Name	$\rho(z)$	$w(z)$
Gaussian	$\rho(z) = z^2$	$w(z) = 1$
Lorentzian	$\rho(z) = \log\left(1 + \frac{1}{2}z^2\right)$	$w(z) = \frac{1}{1+\frac{1}{2}z^2}$
Thresholding	$\rho(z) = \begin{cases} z & \lvert z\rvert \le \theta \\ 0 & \lvert z\rvert > \theta \end{cases}$	$w(z) = \begin{cases} 1 & \lvert z\rvert \le \theta \\ 0 & \lvert z\rvert > \theta \end{cases}$

Table 3.1: Comparison of weight functions. θ in the row "Thresholding" is the threshold value.

If we temporarily forget that w is a function of z, this can be interpreted as weighted-least squares minimization, which has the form $\rho(z) = w\,z^2$. In other words, the term $w(z)$ represents the weight of contribution of errors of magnitude z with respect to a weighted-least squares estimate.

There are many possible choices of $\rho(z)$ to reduce the sensitivity to outliers on the estimation. The famous functions are: Lorentz's, Tukey's, Andrew's, Huber's and the sigmoid function. Among them, the Lorentzian function is used in the current implementation.

In the weighted-least squares sense, the behavior of M-estimation function can be intuitively understood by analyzing the weight function $w(z)$. The Figure 3.5(a) shows the graph of weight functions. While the normal distribution (Gaussian function) has the constant weight value for all ranges of data, the Lorentzian function discounts observations with large errors, which makes this function more robust against outliers. For comparison, the simple thresholding method is also drawn, with the threshold value 3σ. Figure 3.5(b) compares the error probability distribution functions. Both the Gaussian and the Lorentzian function look similar around the center, however, the Gaussian function hardly allow large errors, in particular the errors larger than 3σ. For this reason, the least-squares estimate which assumes the Gaussian distribution does not work correctly in the presence of such outliers.



(a) (b)

Figure 3.5: Plots of weight and probability distribution functions. (a) shows the weight functions. While the Lorentzian function discounts observations with large errors, the Gaussian function always weighs constantly. The thresholding method is also drawn for comparison, with the threshold value 3σ. (b) compares the error probability distribution. They look similar around the center, however, the Gaussian function hardly allow the errors which are larger than 3σ.

3.7 Iterative Refinement of Camera Parameters

Now, we review the registration problem in detail. Given a set of N corresponding point pairs $\langle \mathbf{x}_i, \mathbf{y}_i \rangle$, where $i = 0, ..., N-1$ and $\mathbf{x}_i$ is a 3D edgel point and $\mathbf{y}_i$ is a 2D edgel point, the objective function to be minimized can be written as follows:

$$E(\mathbf{p}) = \frac{1}{N} \sum_{i}^{N} \rho(z_i(\mathbf{p})) \tag{3.20}$$

$$\text{where} \quad z_i(\mathbf{p}) = \|\mathbf{u}_i - \mathbf{y}'_i\|^2 \tag{3.21}$$

$\rho(z)$ is the M-estimate function, the Lorentzian function in this case, and the parameters $\mathbf{p}$ is a 7-vector which denotes the camera rotation and translation $(\mathbf{q}^T \ \mathbf{t}^T)^T$. Both $\mathbf{u}_i$ and $\mathbf{y}'_i$ are the function of $\mathbf{p}$ and they are shown in Equation 3.3 and 3.6. The normalization factor $1/N$ is introduced to take the average distance of corresponding point pairs, since the number of them change through the iterative process by the automatic generation and visibility check of 3D edgels.

The difficulty in minimizing $E(\mathbf{p})$ is that the 2D edgel $\mathbf{y}_i$ corresponding to the 3D edgel $\mathbf{x}_i$ is also the function of $\mathbf{p}$, that is, the movement of $\mathbf{p}$ may cause the change of their correspondences. Although ignoring this fact can lead to inefficiency and possibility of incorrect results, it seems impossible to take these effects into account in the above mathematical formulation.

To overcome this problem, iterative minimization processes are used. In each iterative process, the current correspondences are searched using the current camera parameters. Within each minimization calculation, they are regarded as fixed and the better camera parameters are estimated under such constraints. It starts with a crude set of correspondences and gradually converge to the correct correspondences and at the same time finds the true camera parameters. An improvement in $E(\mathbf{p})$ should correspond to an improvement in $\mathbf{p}$, and that leads to an improvement in the correspondences as well.

Each minimization calculation is accomplished by the conjugate gradient search. Other non-linear optimization methods, such as the Levenberg-Marquardt method, can also be used.

To use these non-linear optimization methods, the gradient of the objective function E with respect to the camera parameters $\mathbf{p}$ must be computed:

$$\frac{\partial E}{\partial \mathbf{p}} = \frac{1}{N}\sum_{i}^{N} w(z_i) z_i \frac{\partial z_i}{\partial \mathbf{p}} \tag{3.22}$$

In particular,

$$\frac{\partial z_i}{\partial \mathbf{p}} = \frac{\partial \mathbf{u}_i}{\partial \mathbf{p}} \frac{\partial z_i}{\partial \mathbf{u}_i} \tag{3.23}$$

The former component, $\frac{\partial \mathbf{u}_i}{\partial \mathbf{p}}$, is the Jacobi matrix of the camera coordinates with respect to the camera parameters. The latter component, $\frac{\partial z_i}{\partial \mathbf{u}_i}$, tells us how we must move $\mathbf{u}_i$, the camera-centered coordinates of $\mathbf{x}_i$, to reduce z_i.

First, the former component, $\frac{\partial \mathbf{u}_i}{\partial \mathbf{p}}$, is inspected in detail.

$$\mathbf{u}_i(\mathbf{p}) = \mathbf{R}(\mathbf{q})\mathbf{x}_i + \mathbf{t} \tag{3.24}$$

The difficult point is the differentiation of $\mathbf{R}(\mathbf{q})\mathbf{x}$ with respect to the rotation quaternion $\mathbf{q}$. To simplify the computation, we pre-rotate the model points so that the current quaternion

is $\mathbf{q_I} = (0, 0, 0, 1)^T$, i.e., the unit quaternion. It has the property $\mathbf{R}(\mathbf{q_I}) = \mathbf{I}$ and considering the gradient around it depends on the fact that this becomes the very simple form:

$$\left.\frac{\partial \mathbf{Rx}}{\partial \mathbf{q}}\right|_{\mathbf{q_I}} \mathbf{x} = 2\mathbf{C}(\mathbf{x})^T \tag{3.25}$$

where $\mathbf{C}(\mathbf{x})$ is the 3×3 skew-symmetric matrix of the vector $\mathbf{x}$. The skew-symmetric matrix is defined as follows.

$$\mathbf{x} \times \mathbf{a} = \mathbf{C}(\mathbf{x})\,\mathbf{a} = \begin{pmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{pmatrix} \mathbf{a} \tag{3.26}$$

where $\mathbf{x} = (x, y, z)^T$. In other words, the cross product of the vector $\mathbf{x}$ is equivalent to the multiplication of its skew-symmetric matrix $\mathbf{C}(\mathbf{x})$. Notice that, by the skew-symmetry of $\mathbf{C}(\mathbf{x})$, $\mathbf{C}^T = -\mathbf{C}$. Therefore,

$$\frac{\partial \mathbf{u}_i}{\partial \mathbf{p}}\mathbf{a} = \begin{bmatrix} \mathbf{a} \\ 2\mathbf{C}(\mathbf{x_i})^T\mathbf{a} \end{bmatrix} \tag{3.27}$$

can be obtained.

Next, the differentiation of z_i by $\mathbf{u}_i$ is derived. From Equation 3.21,

$$\frac{\partial z_i}{\partial \mathbf{u}_i} = \left(\mathbf{I} - \frac{\partial \mathbf{y}_i'}{\partial \mathbf{u}_i}\right)\left\{2(\mathbf{u}_i - \mathbf{y}_i')\right\} \tag{3.28}$$

where

$$\frac{\partial \mathbf{y}_i'}{\partial \mathbf{u}_i} = \frac{\partial (\mathbf{u}_i \cdot \hat{\mathbf{v}})\hat{\mathbf{v}}}{\partial \mathbf{u}_i} \tag{3.29}$$

$$= \frac{\partial (\hat{\mathbf{v}}^T \mathbf{u}_i)\hat{\mathbf{v}}}{\partial \mathbf{u}_i} \tag{3.30}$$

$$= \hat{\mathbf{v}}\hat{\mathbf{v}}^T \tag{3.31}$$

Since $\mathbf{y}_i'$ is on the line stretched from the 3D point $\mathbf{u}_i$ and that line is perpendicular to the viewing direction $\hat{\mathbf{v}}$,

$$(\mathbf{u}_i - \mathbf{y}_i') \cdot \hat{\mathbf{v}} = 0 \tag{3.32}$$

Consequently, we get

$$\frac{\partial z_i}{\partial \mathbf{u}_i} = \mathbf{I}\left\{2(\mathbf{u}_i - \mathbf{y}_i')\right\} \tag{3.33}$$

$$= 2(\mathbf{u}_i - \mathbf{y}_i') \tag{3.34}$$

Finally, from Equation 3.23, 3.27 and 3.34, the derivative of z_i with respect to the camera parameters $\mathbf{p}$ can be obtained.

$$\frac{\partial z_i}{\partial \mathbf{p}} = \frac{\partial \mathbf{u}_i}{\partial \mathbf{p}} \frac{\partial z_i}{\partial \mathbf{u}_i} \tag{3.35}$$

$$= \left\{ \frac{\partial (\mathbf{R}(\mathbf{q})\mathbf{x}_i + \mathbf{t})}{\partial \mathbf{p}} \right\} \left\{ 2(\mathbf{u}_i - \mathbf{y}'_i) \right\} \tag{3.36}$$

$$= \begin{bmatrix} 2(\mathbf{u}_i - \mathbf{y}'_i) \\ 4\mathbf{C}(\mathbf{x}_i)^T (\mathbf{u}_i - \mathbf{y}'_i) \end{bmatrix} \tag{3.37}$$

$$= \begin{bmatrix} 2(\mathbf{u}_i - \mathbf{y}'_i) \\ -4\mathbf{x}_i \times (\mathbf{u}_i - \mathbf{y}'_i) \end{bmatrix} \tag{3.38}$$

Now, we can compute the gradient of E and the minimization calculation can be executed by the conjugate gradient search.

Chapter 4

Simultaneous Registration Algorithm

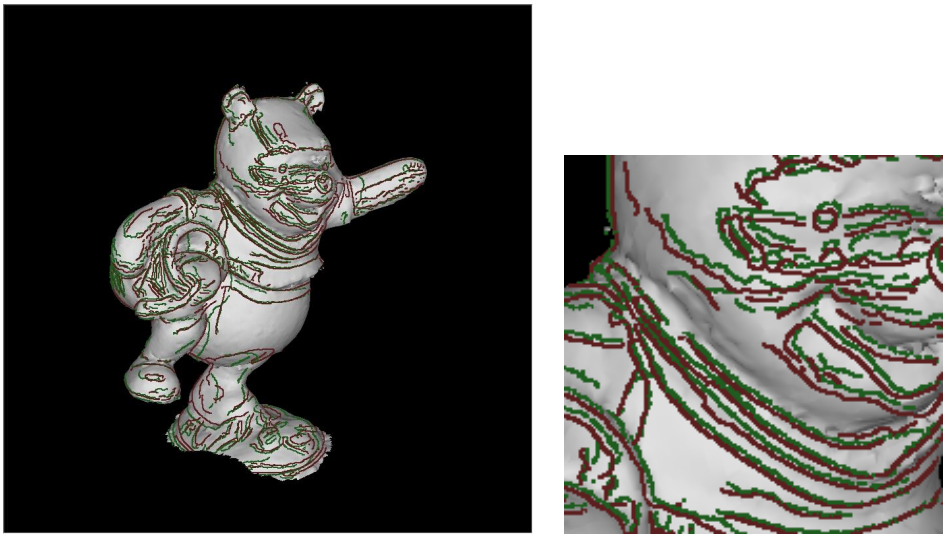
In the previous chapter, the registration method which aligns one 2D image and estimates the single viewpoint against the 3D geometric models, is described. In this chapter, multiple 2D images are taken into account and multiple viewpoints are simultaneously estimated.

4.1 Illustration of Simultaneous Registration

Since one photographic image taken from one viewing point is a partial view of the model, multiple images must be measured to cover the entire 3D geometric models. To obtain the whole texture-mapped model, the apparent approach is to sequentially align each 2D image with the 3D geometric model using the 2D-3D registration technique mentioned in the previous Chapter.

However, it may cause undesirable artifacts around the boundary where texture images from different views intersect, since there would be a gap between two adjacent texture images. Figure 4.1 shows the example. After registering two images separately, aligned 2D edgels are projected onto the 3D surface. We can observe lots of gaps between the edge projected from one texture image and the one projected from another texture image. These gaps lead to the discontinuity at the boundary switching from one texture image to another and result in the visual artifacts.

These gaps result from the fact that even if each 2D image is thoroughly registered to



(a) (b)

Figure 4.1: Example of the gap between two adjacent texture images. Adjacent 2D image edges which are already aligned by the single-viewpoint 2D-3D registration, are projected onto the 3D surface. (b) is a zoomed view of (a).

the 3D object in the error metric of respective viewpoint, it does not necessarily mean it is globally optimal. Due to various errors such as the inaccuracy of 3D geometry, the resolution of pixels, irremovable lens distortions, etc., it is impossible to seek exactly correct registration. Accordingly, we have to assume errors always exist and they need to be distributed globally. Otherwise, if they are minimized only in terms of the single-viewpoint registration, each adjacent image is aligned toward the different kind of objective function and it results in the gaps between adjacent images. Therefore, the multi-view global optimization is necessary which registers multiple images simultaneously.

The simultaneous registration method has the other good point, too. In Section 3.3, the topic of density and similarity of edge features was mentioned. The occluding edgels have less features than other kinds of edgels and the reflectance and rendered edgels might have different edge structures compared to the photometric edges. In the global registration, the gaps seen in Figure 4.1 are optimized, i.e., the 2D edgels from adjacent images are taken into account. These features have the highly similar structures in the

photographs taken from neighboring viewpoints, and also, they contain sufficient details. Consequently, the global registration process utilizing these features is supposed to lead to more accurate and detailed registration results.

4.2 Interactive Error Term

To minimize gaps shown in the previous section, the interactive term is introduced into the objective function $E(\mathbf{p})$:

$$E^{(t)}(\mathbf{p}^{(t)}) = E^{(t)}_{\text{single}}(\mathbf{p}^{(t)}) + E^{(t)}_{\text{interactive}}(\mathbf{p}^{(t)}) \tag{4.1}$$

Since there are multiple 2D photographic images and multiple camera parameters to estimate in the simultaneous registration problem, the upper script (t) is used to denote that the value is related to t-th 2D image. The above formula represents that the evaluation function with respect to t-th image, $E^{(t)}(\mathbf{p}^{(t)})$, comprises two parts, i.e., the term regarding the single-viewpoint registration and the term considering the interaction among neighboring images.

The former term is the same as the one shown in Equation 3.20 and can be expressed as follows.

$$E^{(t)}_{\text{single}}(\mathbf{p}^{(t)}) = \frac{1}{N^{(t)}} \sum_i \rho(z_i^{(t)}(\mathbf{p}^{(t)})) \tag{4.2}$$

It is slightly rewritten to distinguish multiple viewpoints, that is, the script (t) are added. $z_i^{(t)}(\mathbf{p}^{(t)})$ is the distance between i-th visible 3D edgel in the 3D geometric model and the corresponding 2D edgel on the t-th image at the camera parameter $\mathbf{p}^{(t)}$. The normalization factor $1/N^{(t)}$ is presented to obtain the average distance of corresponding points, since the number of them change through the iterative process by the automatic detection and visibility check of 3D edgels.

In addition to the term concerning the single-viewpoint registration, the interactive term which aligns the edgels among neighboring images is introduced in the simultaneous registration. It minimizes the distances of newly generated 3D edgels on the 3D surface. These processes are explained below and illustrated in Figure 4.2.

1. After each image is registered to the 3D geometric model, its 2D edgels are projected onto the surface of the 3D geometric model.

2. Subsequently, they form the new sets of 3D edgels.

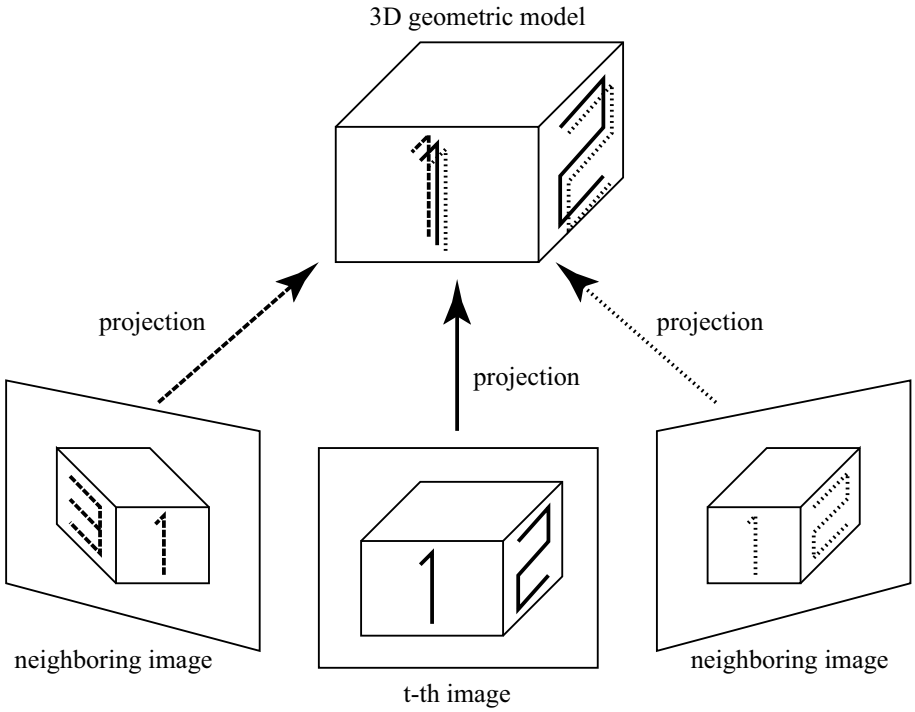


Figure 4.2: Projecting 2D edgels from neighboring images onto the 3D surface. These projected 2D edgels compose the new 3D edgels and they are aligned on the 3D surface.

* We now consider the objective function concerning the *t*-th image.

3. The sets generated from neighboring images are chosen.

4. Among them, the edgels which are not visible from *t*-th viewpoint are removed.

5. Visible neighboring edgels are registered with the edgels projected from *t*-th image, that is, *t*-th viewpoint is modified to make them agree on the 3D surface.

Note that at the projection stage, only the edgels projected onto the smooth gradual surface are used, i.e., edgels projected onto the discontinuous surface or onto the steep slope are eliminated.

The error metric of corresponding edgel pairs on the 3D surface is illustrated in Figure 4.3; **u** is the novel 3D edgel projected from the neighboring image, **y** is its correspond-

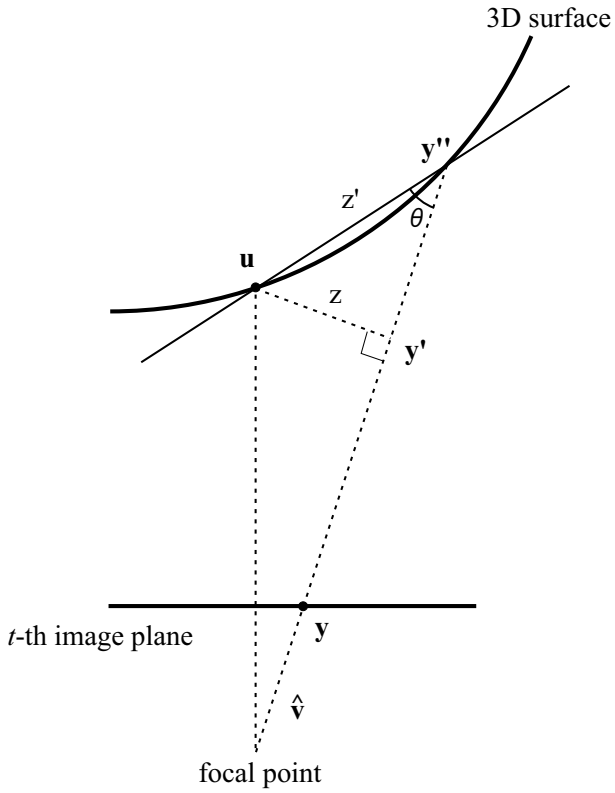


Figure 4.3: Error metric of corresponding edgel pairs on the 3D surface

ing 2D edgel on the t-th image, and $\hat{\mathbf{v}}$ is the viewing direction. These formulations are constructed to imitate the normal 2D-3D registration. In the 2D-3D error metric, the distance z between the 3D edgel $\mathbf{u}$ and the line connecting from the focal point to the 2D edgel $\mathbf{y}$ is considered and minimized. On the other hand, in the simultaneous registration, the distance between projected edgels along the 3D surface is minimized. Let $\mathbf{y}''$ be the projected point of the 2D edgel on the t-th image. Here, we can assume that the distance of corresponding edgel pairs along the 3D surface is approximated by the Euclidean distance between $\mathbf{u}$ and $\mathbf{y}''$. This is because the projected edgels exist on the smooth surface and their neighborhood can be approximated by the tangential plane. Consequently, the error metric for the interactive term can be written as follows:

$$z_i' = \frac{z_i}{\sin\theta_i} \tag{4.3}$$

where θ is the angle between the viewing direction and the tangential plane.

Assuming that θ is fixed in one iteration step, z'_i and its gradient $\partial z'_i / \partial \mathbf{p}$ are almost the same as the 2D-3D registration case. Therefore, this mutual registration algorithm can take advantage of the similar framework to the single-viewpoint registration between 3D edgels and 2D edgels.

Now, the formula of the interactive term is shown below.

$$E^{(t)}_{\text{interactive}}(\mathbf{p}^{(t)}) = \frac{1}{N'^{(t)}} \sum_{s \in U(t)} \sum_{i} \rho(z'^{(t,s)}_i(\mathbf{p}^{(t)})) \tag{4.4}$$

$U(t)$ denotes the set of neighboring images of t-th image. Among them, s-th image is chosen and $z'^{(t,s)}_i(\mathbf{p}^{(t)})$ is the distance of i-th edgel pair which comprises the edgels projected from t-th image and the edgels projected from s-th image.

Thus, the objective function regarding the t-th camera parameter $\mathbf{p}^{(t)}$ is constructed to meet both the 3D edgels from 3D geometric model and the edgels projected from neighboring images.

4.3 Iterative and Simultaneous Refinement

In the global registration problem, we have to estimate N sets of camera parameters $\mathbf{p}^{(i)}$ $(0 \leq i \leq N-1)$. First of all, separate single-viewpoint registrations need to be accomplished to approximately align all images. After that, simultaneous refinement process starts and it is also achieved by the iterative calculations.

The outline of the simultaneous refinement algorithm is described in Figure 4.4. Since it takes advantage of the similar framework as the single-viewpoint registration, there are not so many differences. However, some points are described in detail below.

1. For each iterative step, 2D texture edgels are projected onto the 3D surface using their current camera parameters and they form the new temporal 3D edgels. At this projection stage, uncertain edgels which are projected to a steep surface or around occluding boundaries should be removed.

2. The t-th objective function consists of the single-viewpoint term and the interactive term. The latter considers the differences of projected edgels on the 3D surface. It minimizes the distance between the edgels projected from t-th image and the edgels projected from neighboring images on the 3D surface.

3. By minimizing the t-th objective function, the update of t-th camera parameters is estimated. Minimization is executed by the conjugate gradient search.

4. The estimated update of t-th camera parameters is "not" applied at this point. Instead, it is recorded in the update list.

5. After all objective functions are minimized and all camera updates are estimated, they are finally applied to the sets of camera parameters.

6. The above loops are repeated until the objective functions converge.

Note that the camera parameters are not transformed immediately. Considering that the changes of camera parameters caused by each step will not so large, this latency of propagation will not cause a problem. The strategy of updating every camera parameters at once is not taken because the order of processing texture images matters in that case, and further, it requires duplicated calculations of projecting 2D edgels.

```
// separate single-viewpoint registration stage
foreach t in AllTextureImages {
    do {
        Model3DEdgel[t] = GetVisibleEdgel(GeometricModel, Camera[t]);
        PointPairs = [];
        foreach i in PointsOf(Model3DEdgel[t])
            PointPairs += CorrespondenceSearch(i, Texture2DEdgel[t]);
        UpdateList[t] = EstimateCameraUpdate(PointPairs);
        TransformSingle(Camera[t], UpdateList[t]);
    } until converge
}

// simultaneous registration stage
do {
    foreach t in AllTextureImages {
        Model3DEdgel[t] = GetVisibleEdgel(GeometricModel, Camera[t]);
        Projected3DEdgel[t] = Project2DEdgel(GeometricModel, Camera[t],
                                             Texture2DEdgel[t]);
    }

    foreach t in AllTextureImages {
        PointPairs = [];
        PointPairs2 = [];

        // single-viewpoint term
        foreach i in PointsOf(Model3DEdgel[t])
            PointPairs += CorrespondenceSearch(i, Texture2DEdgel[t]);

        // interactive term
        foreach s in NeighboringImages
            foreach i in PointsOf(Projected3DEdgel[s])
                PointPairs2 += CorrespondenceSearch(i, Texture2DEdgel[t]);

        UpdateList[t] = EstimateCameraUpdate(PointPairs, PointPairs2);
    }

    // update all camera parameters at this point
    TransformAll(Camera, UpdateList);
} until converge
```

Figure 4.4: Outline of simultaneous registration algorithm

Chapter 5

Experiments and Results

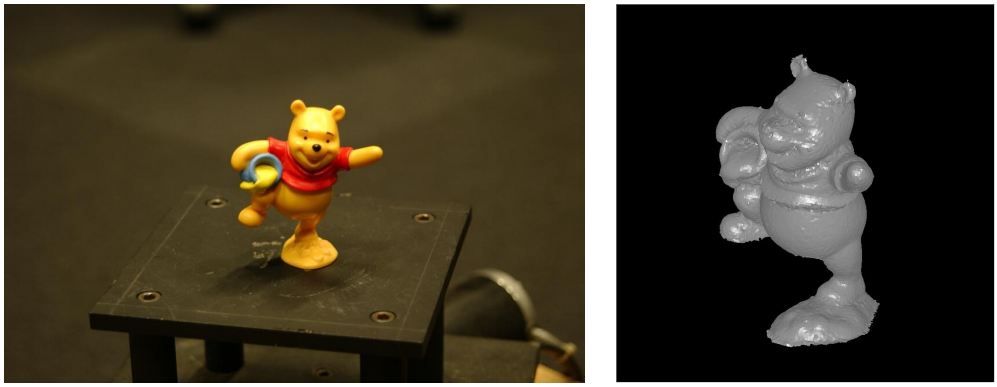
5.1 Implementation Details

- Rough and detailed registration:
  In the experiment, the single-viewpoint registration was divided into two separate stages: the rough registration stage and the detailed registration stage. At first, only the occluding edgels are used so that the rough position can be easily aligned without the interference of small edge structures. After that, the reflectance edgels (if available) or the rendered edgels are also considered to align the detailed structures.

  Each 2D image is registered separately as described above. Finally, the all images are simultaneously registered by the global optimization.

- σ of the Lorentzian function:
  In the M-estimation framework, the argument of the Lorentzian function must be the normalized value with respect to the proper standard deviation σ. Otherwise, reduction of outliers might be too weak or too strong. Therefore, σ is always updated by analyzing the distribution of corresponding 2D-3D errors. Every time the 2D-3D correspondences are updated, their lower quartile error is chosen as σ. Since lower quartile is the 1/4th smallest value, proper σ will be chosen as long as the quarter of the correspondences are correct.

- Selection of the neighboring images:
  In the simultaneous registration, 2D edgels are projected onto the 3D surface and they are registered among the neighboring images. In the experiment, only two



(a) (b)

Figure 5.1: Plastic bear object: (a) photographic image, (b) 3D geometric model.

adjacent images, i.e., the left and the right neighbors, are used as the neighborhood since the texture images are captured on the circular position surrounding the target object. However, in practice, the neighborhood within some range should be automatically chosen.

- Selection of the texture image:
  For each mesh, the texture image which minimizes the inner product of the mesh normal and the viewing direction, is chosen. To avoid too much fragmentation, the mesh normal is averaged around the neighborhood.

5.2 Results

Proposed registration method is applied to a plastic bear object (In Figure 5.1). Range images are measured with a Minolta VIVID 900, and the 3D geometric model has been constructed using these alignment and merging methods [21, 25]. The obtained geometry has 31300 vertices and 62277 meshes. 2D photographic images are taken with a NIKON D1x digital camera which yields an image of 3008x1960 resolution. Lens distortions are eliminated using the camera calibration method [36], and at the same time, the camera focal length is also obtained. Other camera intrinsic parameters are assumed to be idealized value, i.e., the principal point is $(0, 0)$, the aspect ratio is unity and the skew is zero. Registration calculations are carried out on the PC which has the AMD Athlon processor of 1400MHz and 512MB memory.

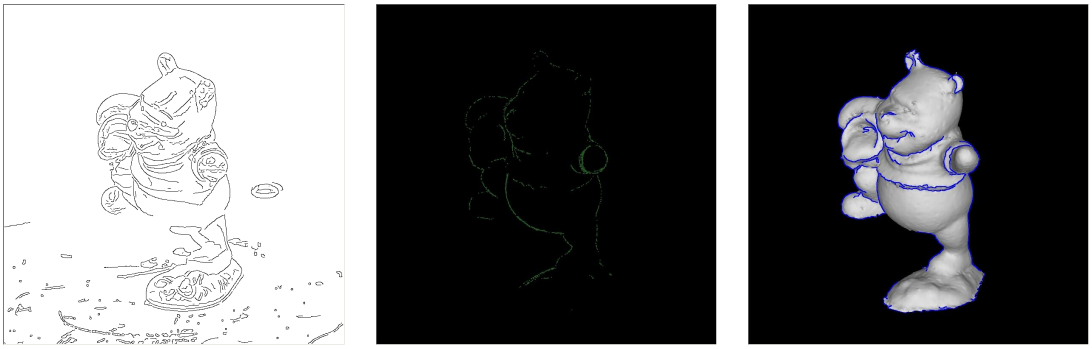
To begin with, the single-viewpoint 2D-3D registration method is examined. Figure 5.2 shows detected 2D and 3D edgels. In this experiment, the occluding edgels and the rendered edgels are used as the 3D edgels. The process of the iterative calculation is illustrated in Figure 5.3. The camera extrinsic parameters have been refined to align corresponding 2D edgels and 3D edgels and the proper camera viewpoint is estimated. This registration took approximately 30 seconds. More than half of them is consumed in the process of the 2D edge detection using Canny method which is executed several times to obtain 3D rendered edgels. Other time consuming processes are: the rendering process of 3D geometries using OpenGL which is necessary to obtain the z-buffer for visibility checking, and the calculation of the objective function which is evaluated many times in the conjugate gradient search.

Thus, 11 photographs taken from different viewpoints can be separately registered to the 3D geometric model. However, the set of images which are registered separately is not necessarily consistent around the boundary where images from different views intersect. Since there always remain some registration errors due to the inaccuracy of 3D geometries, irremovable lens distortions, incorrect camera intrinsic parameters, etc., the perfectly correct registration cannot be achieved, and such errors must be distributed globally. Therefore, the simultaneous registration is applied after the separate single-viewpoint registrations, and the effects are examined. 2D edgels of two adjacent texture images are projected onto the 3D surface and their gaps before and after the simultaneous refinement are compared in Figure 5.4. Here, we can observe that these gaps undoubtedly shrink, thanks to the simultaneous registration. For this simultaneous refinement, 20 iterations were necessary and it took roughly 10 minutes.

In the simultaneous registration, the objective function consists of two parts, i.e., the single-viewpoint error terms relating to the separate 2D-3D registration and the interactive error terms concerning the global errors.

$$E(\mathbf{p}) = E_{\text{single}}(\mathbf{p}) + E_{\text{interactive}}(\mathbf{p}) \tag{5.1}$$

The behavior of these two kinds of components is examined in Figure 5.5. This graph contains two experiments: one is the separate single-viewpoint registration of 40 iterations, and the other is also the single-viewpoint registration for first 20 iterations but the simultaneous registration follows for successive 20 iterations. Although the interactive error terms do not exist in the single-viewpoint registration, they are temporarily evaluated



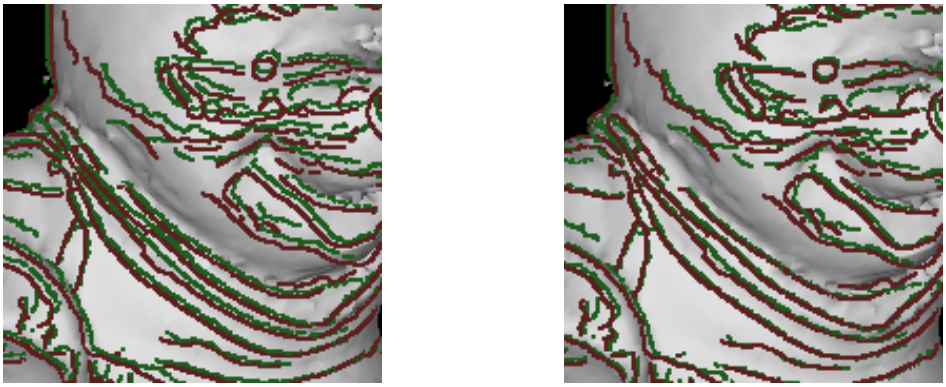
(a) (b) (c)

Figure 5.2: Detected 2D and 3D edgels: (a) 2D edgels, (b) 3D occluding edgels, and (c) 3D rendered edgels.

(a) (b) (c)

Figure 5.3: 2D-3D registration: (a) initial position, (b) after 8 iterations, and (c) after registration calculation. The 3D geometry and 3D edgels are overlaid on the photographic image; red pixels are the 2D edgels, green pixels are the occluding edgels, and blue pixels are the rendered edgels.

(a) (b)



(c) (d)

Figure 5.4: Comparison of the alignment gap: 2D edgels of the two adjacent images are projected onto the 3D surface. (a) Aligned using the separate single-viewpoint registrations. (b) Aligned using the simultaneous registration. (c), (d) Zoomed views of (a) and (b), respectively.

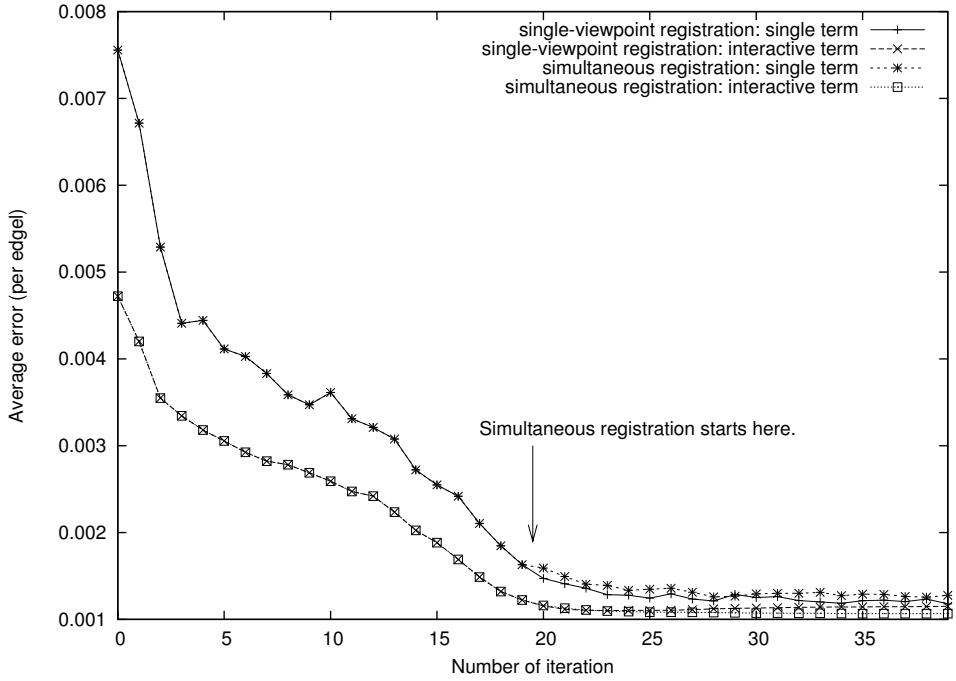
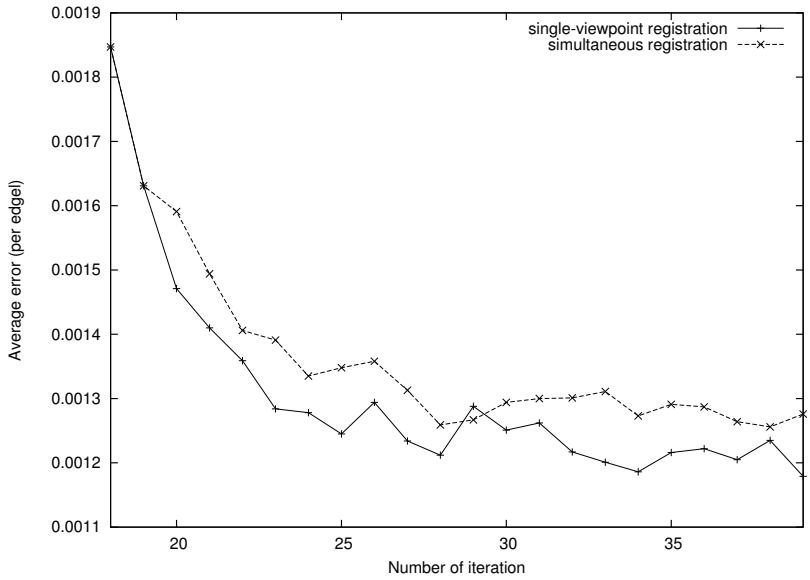


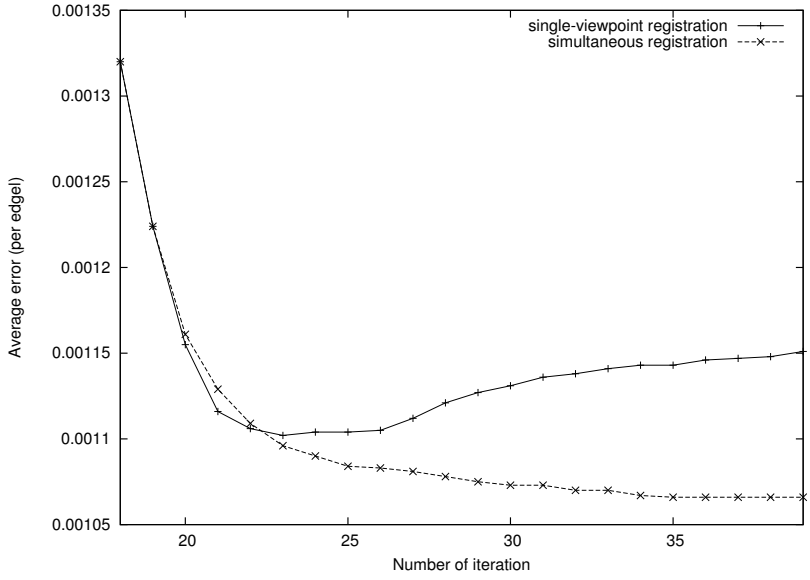
Figure 5.5: Plotting two kinds of error terms: the single error terms relating to the single-viewpoint errors, and the interactive error terms concerning the global errors.

at each iteration to observe the global errors. While the first half of the plots are exactly the same, we can observe the interesting difference after the simultaneous registration starts in one experiment. It is plainly seen in the zoomed views around the simultaneous registration (in Figure 5.6). Although the single-viewpoint registration reduces the single error terms slightly better than the simultaneous registration, the interactive errors do not necessarily decrease. Indeed, further single-viewpoint registration tries to reduce the single error terms too much at the expense of the global errors.

The quality of the texture-mapped object is also compared in Figure 5.7. Since the registration errors are absorbed globally, visual artifacts are reduced in simultaneously registered results. However, when examined carefully, there still remain some defects and we can consider two major reasons: registration errors and color inconsistency. The former means that although the simultaneous registration distributes errors globally, there



(a)



(b)

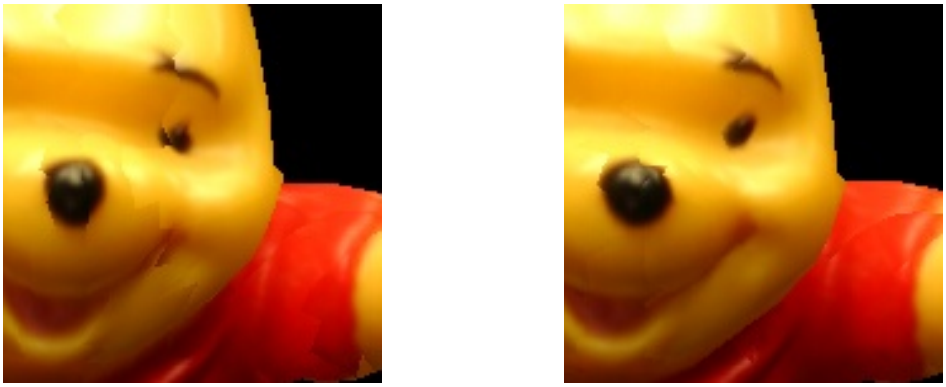
Figure 5.6: Behavior of the single error terms (a) and the interactive error terms (b). These are the zoomed views of Figure 5.5.

should remain excessive errors. The latter is the more serious problem. Even if the images are perfectly aligned, there might exist the color gaps between adjacent images. This is because the observed color in the photograph changes due to various factors: illumination conditions, viewing positions, specular highlights, etc. Note that, to avoid such problems, many researches concerning the texture mapping adopt the blending strategy of textures from neighboring images.

Recently, our laboratory has been conducting the project of creating digital cultural assets through observation, and the precise 3D geometric models of such objects have been constructed using accurate laser scanners [18, 21, 25, 26]. Thus, the proposed registration method is applied to one of them, the Great Buddha of Kamakura (in Figure 5.8(a)) and its texture-mapped model is created. The Great Buddha of Kamakura is a 13m tall statue sitting in an open air. It was scanned using a Cyrax 2400 sensor and the fine geometric model has been reconstructed, which has approximately 0.7 million vertices and 1.3 million meshes (in Figure 5.8(b)). Since registering 2D images to such high resolutional data requires massive computational time, the simplified model was used, which has approximately 100 thousand vertices and 200 thousand meshes.

18 photographs are taken with D1x digital camera and they are registered to the geometric model. Results of the textured model are shown in Figure 5.9. Although the registration process is almost the same as the previous bear example, reconstructed Great Buddha has several visual artifacts. This is because there exist excess difficulties in this case due to the outdoor environment and the size of the object. First, the illumination condition should easily change in the outdoor environment. Although all measurements of photographs are carried out within only a few minutes, the observed colors are slightly changed. This would be caused by the imperceptible movement of the sun and the clouds. Second, a 17mm wide lens was necessary to capture the unoccluded whole image of the Great Buddha. The wide lens leads to larger lens distortions particularly in the periphery of the image, and indeed, the camera calibration could not remove part of the distortions around the leg of the Great Buddha (in Figure 5.8(a)). As a result, the simultaneous registration did not perform well especially in the lower half of the Great Buddha and this leads to the alignment gaps around that region.

(a) (b)



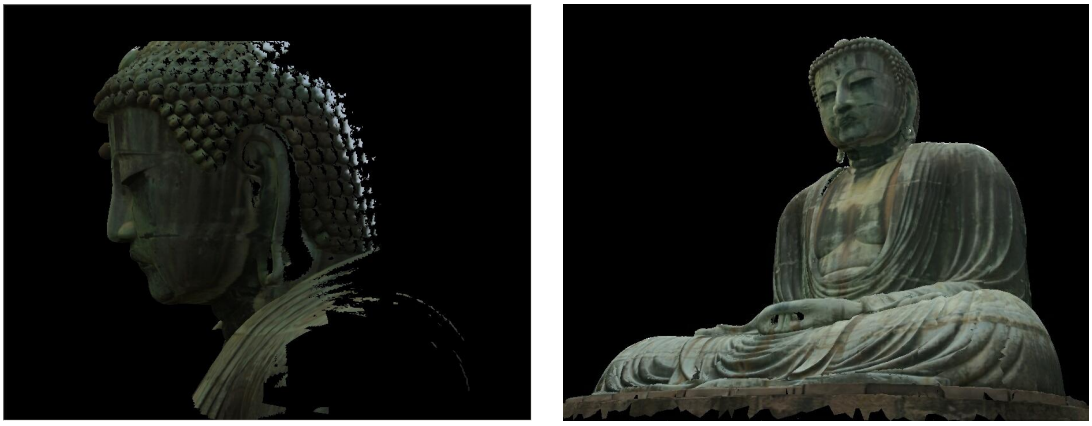
(c) (d)

(e) (f)

Figure 5.7: Comparison of the texture-mapped model: (a) Separate single-viewpoint registrations. (b) Simultaneous registration. (c), (d), (e), (f) Zoomed views of the top images.

(a) (b)

Figure 5.8: The Great Buddha of Kamakura: (a) a photograph taken with the 17mm wide lens, (b) the high resolutional geometric model.



(a) (b)

Figure 5.9: Textured Great Buddha. (a) one image is mapped, (b) 18 images are mapped.

Chapter 6

Conclusions

6.1 Summary

In this thesis, a novel registration method is introduced and described, which automatically and simultaneously aligns multiple 2D images onto 3D geometric models. Usually, corresponding features between the 2D image and the 3D model have to be specified to estimate the camera position and orientation. However, in the proposed method, the correspondence information between 2D edge pixels and 3D edge points are automatically searched and updated throughout the iterative calculations. Considering the robustness and the density of edge features, three types of 3D edge features are proposed and used in combination. Further, the global optimization among all the 2D images are also achieved by the simultaneous registration which considers the 2D-2D edge correspondences on 3D surfaces. To make the algorithm robust against the outliers, the framework of M-estimates is employed. Registration results are examined with the texture mapped objects and the meaningful importance of the simultaneous registration is presented. Also, this method is applied to the creation of digital cultural assets and the issues concerning the measurement in large-scale outdoor environments are revealed.

6.2 Future Work

To achieve the accurate texture mapping, lens distortions must be removed. Therefore, the practical camera calibration is needed, which can be easily performed at the measurement time even in the large-scale outdoor environments.

To improve the quality of texture-mapped objects, the intrinsic color of the object surface must be estimated. Since the observed texture image contains various factors at the measurement time: illumination conditions, shadows, specular highlights, etc., the colors of the corresponding points from different viewpoints are not consistent. Therefore, in order to reconstruct the precise 3D models, such factors must be canceled out and the intrinsic color of the surface needs to be estimated.

References

[1] P. K. Allen, I. Stamos, A. Troccoli, B. Smith, M. Leordeanu, and Y. C. Hsu. 3D modeling of historic sites using range and image data. *submitted to the International Conference of Robotics and Automation*, 2003.

[2] P. Besl and N. McKay. A method for registration of 3-D shapes. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 14, No. 2, pp. 239–256, February 1992.

[3] J. Canny. A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 679–698, 1986.

[4] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. *Image and Vision Computing*, Vol. 10, No. 3, pp. 145–155, 1992.

[5] P. E. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. *Computer Graphics*, pp. 189–198, 1998. Proc. of SIGGRAPH '98 (July 1998, Orlando, Florida).

[6] P. E. Debevec. A tutorial on image-based lighting. *IEEE Computer Graphics and Applications*, pp. 26–34, 2002.

[7] P. E. Debevec and Jitendra Malik. Recovering high dynamic range radiance maps from photographs. *Computer Graphics*, pp. 369–378, 1997. Proc. of SIGGRAPH '97 (August 1997, Los Angeles, California).

[8] P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *Computer Graph-*

ics, pp. 11–20, 1996. In Proc. of SIGGRAPH ‘96 (August 1996, New Orleans, Louisiana).

[9] K. Deguchi and T. Okatani. Calibration of multi-view cameras for 3D scene understanding (in japanese). *CVIM*, January 2002.

[10] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT Press, 1993.

[11] T. Igarashi, S. Matsuoka, and H. Tanaka. Teddy: A sketching interface for 3D freeform design. *Computer Graphics*, pp. 409–416, 1999. In Proc. of SIGGRAPH ‘99 (Los Angels, 1999).

[12] B. Klaus and P. Horn. *Robot Vision*. The MIT Press, McGraw-Hill Book Company, 1986.

[13] R. Kurazume, K. Nishino, Z. Zhang, and K. Ikeuchi. Simultaneous 2D images and 3D geometric model registration for texture mapping utilizing reflectance attribute. In *Proc. 5th Asian Conf. on Computer Vision*, January 2002.

[14] R. Kurazume, M. D. Wheeler, and K. Ikeuchi. Mapping textures on 3D geometric model using reflectance image. *Data Fusion Workshop in IEEE Int. Conf. on Robotics and Automation*, May 2001.

[15] S. Lavallée and R. Szeliski. Recovering the position and orientation of free-form objects from image contours using 3D distance maps. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 17, No. 4, pp. 378–390, April 1995.

[16] H. P. A. Lensch, W. Heidrich, and H. Seidel. Automated texture registration and stitching for real world models. In *Proc. Pacific Graphics ’00*, pp. 317–326, October 2000.

[17] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, Vol. 21, No. 4, pp. 163–169, July 1987. In Proc. SIGGRAPH ‘87.

[18] D. Miyazaki, T. Ooishi, T. Nishikawa, R. Sagawa, K. Nishino, T. Tomomatsu, Y. Takase, and K. Ikeuchi. The great buddha project: Modelling cultural heritage through observation. In *Proc. 6th Inter. Conf. on Virtual Systems and MultiMedia (VSMM 2000)*, pp. 138–145, 2000.

[19] S. K. Nayar and T. Mitsunaga. High dynamic range imaging: Spatially varying pixel exposures. In *Proc. IEEE Inter. Conf. on Computer Vision and Pattern Recognition*, Hilton Head Island, South Carolina, June 2000.

[20] P. J. Neugebauer and K. Klein. Texturing 3D models of real world objects from multiple unregistered photograpihic views. In *Proc. Eurographics '99*, pp. 245–256, Milan, September 1999.

[21] K. Nishino and K. Ikeuchi. Robust simultaneous registration of multiple range images. In *Proc. 5th Asian Conf. on Computer Vision*, pp. 454–461, Melbourne, Australia, January 2002.

[22] K. Nishino, Y. Sato, and K. Ikeuchi. Eigen-texture method: Appearance compression and synthesis based on a 3D model. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 23, No. 11, pp. 1257–1265, 2001.

[23] C. Rocchini, P. Cignoni, and C. Montani. Multiple textures stitching and blending on 3D objects. In *10th Eurographics Workshop on Rendering*, pp. 127–138, June 1999.

[24] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. *3DIM*, 2001.

[25] R. Sagawa, K. Nishino, and K. Ikeuchi. Robust and adaptive integration of multiple range images with photometric attributes. In *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pp. 172–179, December 2001.

[26] R. Sagawa, T. Oishi, A. Nakazawa, R. Kurazume, and K. Ikeuchi. Iterative refinement of range images with anisotropic error distribution. In *Proc. of 2002 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 79–85, October 2002.

[27] I. Sato, Y. Sato, and K. Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Trans. Visualization and Computer Graphics*, Vol. 5, No. 1, pp. 1–12, 1999.

[28] I. Sato, Y. Sato, and K. Ikeuchi. Illumination distribution from brightness in shadows: adaptive estimation of illumination distribution with unknown reflectance properties in shadow regions. In *Proc. IEEE Inter. Conf. Computer Vision*, pp. 875–882, September 1999.

[29] Y. Sato, M. D. Wheeler, and K. Ikeuchi. Object shape and reflectance modeling from observation. *Computer Graphics*, pp. 379–387, August 1997. In Proc. SIGGRAPH '97.

[30] J. Shi and C. Tomasi. Good features to track. In *Proc. IEEE Inter. Conf. on Computer Vision and Pattern Recognition*, pp. 593–600, 6 1994.

[31] I. Stamos and P. K. Allen. 3-D model construction using range and image data. In *Proc. IEEE Inter. Conf. on Computer Vision and Pattern Recognition*, Vol. 1, pp. 531–536, South Carolina, June 2000.

[32] R. 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*, Vol. 3, No. 4, pp. 323–344, August 1987.

[33] M. D. Wheeler. *Automatic Modeling and Localization for Object Recognition*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, October 1996.

[34] R. C. Zeleznik, K. P. Herndon, and J. F. Hughes. Sketch: An interface for sketching 3D scenes. *Computer Graphics*, pp. 163–170, 1996. In Proc. of SIGGRAPH '96 (New Orleans,1996).

[35] Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 1997.

[36] Z. Zhang. A flexible new technique for camera calibration. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 11, pp. 1330–1334, 2000.