# Acquisition and Rectification of Shape Data Obtained by a Moving Range Sensor

By

Atsuhiko Banno

A Doctoral Dissertation

Submitted to the Graduate School of
The University of Tokyo

東京大学
The University of Tokyo

for the Degree of
Doctor of Information Science and Technology

on December 16, 2005

Committee:
Takeshi NAEMURA (Chair)
Masao SAKAUCHI
Masaru KITSUREGAWA
Yoichi SATO
Shunsuke KAMIJO

Supervisor:
Katsushi IKEUCHI

# Acknowledgments

First of all, I would like to express my sincere gratitude to professor Katsushi Ikeuchi for taking me on as his student. He provided his valuable advice and an excellent studying environment with many opportunities to develop new ideas, work on practical scenes and discuss with interesting researchers.

I wish to express my deepest gratitude to great senior researchers in Ikeuchi Laboratory. Some of them gave me significant advice to my research, some of them gave me exciting stimulations and some of them left the FLRS. Without them, I could not have achieved my thesis. And I am grateful to all the members of the laboratory.

My special thanks are due to Mr. Markus Mettenleiter and Mr. Frantz Härtl of Zoller+Fröhlich GmbH for technical supports on the FLRS.

I would like to thank Dr. Joan Knapp for proofreading my manuscripts. She kindly improved them and gave me a lot of appropriate suggestions.

I also wish to thank my former bosses, Mr. Kazuo Mogami and Dr. Takehiko Takatori of National Research Institute of Police Science.

Finally, I wish to express my gratitude to all people who have have supported my research activities.

# ABSTRACT

”Modeling from Reality” techniques are making great progress because of the availability of accurate geometric data from three dimensional digitizers. These techniques contribute to numerous applications in wide areas such as academic investigation, industrial management, and entertainment. Among them, one of the most important and comprehensive applications is modeling cultural heritage objects. For a large object, scanning from the air is one of the most efficient methods of obtaining 3D data. Nevertheless, in the case of large cultural heritage objects, there are some difficulties in scanning with respect to safety and efficiency. To remedy these problems, we have been developing a novel 3D measurement system, the Floating Laser Range Sensor (FLRS), in which a range sensor is suspended beneath a balloon. The obtained data, however, have some distortion due to movement during the scanning process. We propose two novel methods to rectify the shape data obtained by a moving range sensor in this thesis. One method rectifies the distorted range data by using image sequences and another one rectifies the data without images. Both methods are applicable not only to our FLRS, but also to a general moving range sensor.

In Chapter 2, we explain our FLRS system. While we use a commercial product as a scanning unit, we have designed whole system and the mirror configurations of the FLRS. Thus, the FLRS is our original system. The system overview and components are introduced in this chapter. In addition, we explain the algorithm of 3D reconstruction by using mirrors from a fixed-point measurement range data.

In Chapter 3, we explain a full perspective factorization, which is utilized as the initial solution for the camera motion. We use a weak perspective factorization iteratively for the perspective projection camera model. Interest point detectors are essential for the factorization. We explain two detectors, Harris operator and SIFT key. Finally, we estimate the performance of our full perspective factorization.

In Chapter 4, we describe our proposed algorithm for refinement of the parame-

ters. Our method applies the three constraints for optimization, which are tracking, smoothness and range data constraint. Applying these constraints and optimizing the cost function, we can estimate more precise parameters. For the optimization method, we apply a conjugated gradient method and the golden section search.

In Chapter 5, some topics on calibration for our FLRS system are described. In fact, the video camera is assumed to be calibrated with range sensor in the previous chapters. With respect to the FLRS, fixing it on the ground, we can easily acquire shape model and its image simultaneously. In the case of the calibration with 3D reference objects, many methods suffer from noise in accuracy. Using the RANSAC (Random Sampling Consensus) technique, we propose a robust calibration method in the first half of this chapter. Furthermore, in the second half of the chapter, we show that our algorithm is also applicable for the uncalibrated system.

In Chapter 6, we describe another method for shape rectification that needs no image sequences. Instead of using images, this method requires range data obtained by another range sensor fixed on the ground. Incomplete range data of the fixed sensor are sufficient to rectify FLRS range data. There are many cases such a situation in real measurements. Originally, the FLRS has been proposed in order to complement the fixed sensors. Based on overlapping shape between two data sets, we rectify FLRS range data. In this method, it is also assumed that the sensor moves smoothly. We can easily build a graphic user interface (GUI) onto this method and therefore produce practical software.

In Chapter 7, we evaluate our algorithms with known models. Constructing a virtual FLRS on a PC by using CG model, we estimate the accuracy of our methods.

In Chapter 8, we show several experimental results conducted in the Bayon Temple in Cambodia. To evaluate our methods, the rectified shapes are compared with other data sets obtained by a range sensor on the ground. Now, we are conducting the Digital Bayon Project, in which our algorithms are actually applied for range data processing and the results show the effectiveness of our methods.

Finally, we present our conclusions and summarize our possible future works in Chapter 9.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Background

Nowadays, many researches on real object modeling are making great progress because of the availability of accurate geometric data from three dimensional digitizers. The techniques of real object modeling contribute toward numerous applications in wide areas such as academic investigation, industrial management, and entertainment.

Among them, one of the most important and comprehensive applications is modeling cultural heritage objects. Modeling these heritage objects has great significance in many aspects. Modeling them leads to digital archives of the object shapes. Utilizing these data enables us to restore the original shapes of the heritage objects, even if the objects have been destroyed due to natural weathering, fire, disasters and wars. In addition, we can provide images of these objects through the Internet to people in their homes or in their offices. Thus, the techniques of real object modeling are available for many applications.

We have been conducting some projects to model large scale cultural heritage objects such as great Buddhas, historical buildings and suburban landscapes [MNS$^+$00] [INHO03]. Basically, to scan these large objects, a laser range finder is usually used with a tripod positioned on stable locations. In the case of scanning a large scale object, however, it often occurs that some part of the object is not visible from the laser range finder on the ground. In spite of such a difficulty, we have scanned large objects from scaffolds temporally constructed nearby the object. However, this scaffold method requires costly, tedious construction time. In addition, it may be impossible to scan some parts of the object due to the limitation of available space for scaffold-building.

We are now conducting a project [IHN+04] to model the Bayon Temple [VZG01] in Cambodia; the temple's scale is about 150 × 150 square meters with over 40 meter height. Scanning such a huge scale object from several scaffolds is unrealistic. To overcome this problem, several methods have been proposed. For example, aerial 3D measurements can be obtained by using a laser range sensor installed on a helicopter platform[TDH03]. High frequency vibration of the platform, however, should be considered to ensure that we obtain highly accurate results. To avoid irrevocable destruction, the use of heavy equipment such as a crane should be eschewed when scanning a cultural heritage object.

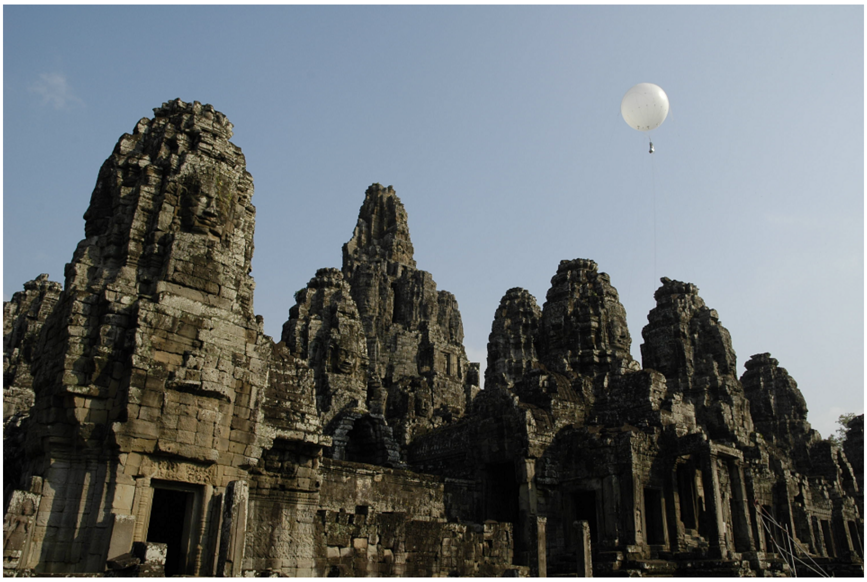


Figure 1.1: The FLRS and the Bayon Temple

Based upon the above considerations, we proposed a novel 3D measurement system, a Floating (or Flying) Laser Range Sensor (FLRS)[HMK+04a] [HMK+04b] [HHO+04] [HHO+05]. This system digitizes large scale objects from the air while suspended from the underside of a balloon platform (Fig.1.1). Our balloon platform is certainly free from high frequency vibration such as that of a helicopter engine. The obtained range data are, however, distorted because the laser range sensor itself is moving during the scanning processes (Fig.1.2).

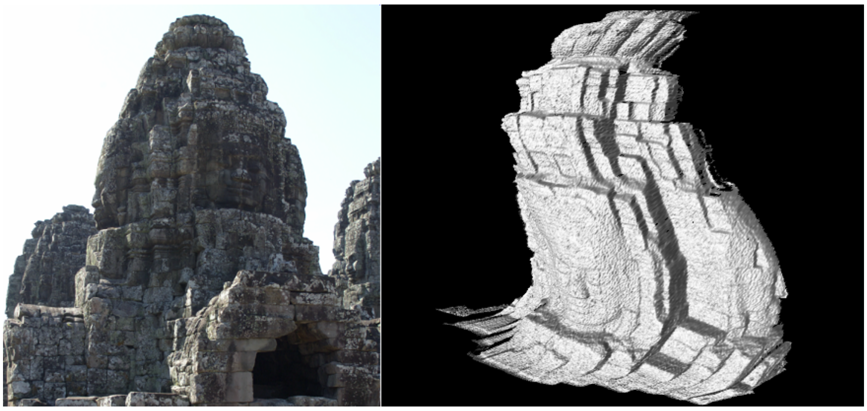


Figure 1.2: An sample snap shot and the distorted range data obtained by the FLRS.

## 1.2 Our Contributions

In this thesis, we propose two methods to rectify 3D range data obtained by a moving laser range sensor. Not only is this method limited to the case of our FLRS, but it is also applicable to a general moving range sensor.

In fact, several attempts have been made to rectify the deformed FLRS data. The following three strategies have been considered to solve this problem:

- Window matching-based method [HHO$^+$04] [HHO$^+$05]
- 3D registration-based method [HMK$^+$04a] [HMK$^+$04b] [MHNI05]
- Structure from motion-based method

In the first strategy, under the assumption that translation of the balloon is very small and within a plane parallel to the image plane without any rotation, the shape is recovered by using a video sequence image. Then supposing that the changes in sequential images are very small, the balloon motion is estimated by a local window matching technique. This method is very fast, but it restricts the balloon to a simple and small motion.

In the second strategy, the balloon motion is parametrized motion beforehand (e.g. the velocity vector for a linear uniform motion or a constant angular velocity). Then, an extended ICP algorithm is applied to align the deformed model obtained by the FLRS with the correct model obtained by a range sensor located on the ground. This method does not require image sequences, but it assume the simple motions.

In this thesis, we adopt two strategies for the rectification. Firstly, we adopt the third strategy among the methods listed above, and propose a method with image sequences and destorted range data by FLRS. Next, we adopt the second strategy.

In the first method based on ”Structure from Motion”, We use distorted range data obtained by a moving range sensor and image sequences obtained by a video camera mounted on the FLRS. The motion of the FLRS is roughly estimated only by the obtained images. And then the more refined parameters are estimated based on an optimization imposing some constraints, which include information derived from the distorted range data itself. Finally, using the refined camera motion parameters, the distorted range data are rectified.

In the second method based on ”3D registration”, we adopt a method similar with [HMK$^+$04a] [HMK$^+$04b] [MHNI05], but supposing smooth and more generalized balloon motion.

These methods are not limited to the case of our FLRS but also applicable to a general moving range sensor that has smooth motion. In this thesis, we do not utilize physical sensor such as gyros, INS and GPS for estimation of self position and pose. We try to solve our problems only by range sensors and video cameras through the techniques of ”Computer Vision [Fau93] [TV98] [FL01] [FP02] [HZ04] ”.

## 1.3 Outline of the Thesis

In this dissertation, we have been wrestled with the FLRS throughly. Then we propose two novel methods to rectify the shape data obtained by a moving range sensor.

One method rectifies the distorted range data by using image sequences and another one rectifies data without images.

In the method with images, the initial motion parameters are estimated by using a full perspective factorization. Then they are refined through an optimization with some constraints. In fact, this method is based on the technique of *”Structure from Motion”*. This technique is applicable both calibrated cameras and uncalibrated cameras.

In the method without images, the original distorted shape is rectified based on the correct shape obtained by another range sensor fixed on the ground.

This thesis is organized as follows.

We explain our FLRS system in Chapter 2. While we use a commercial product

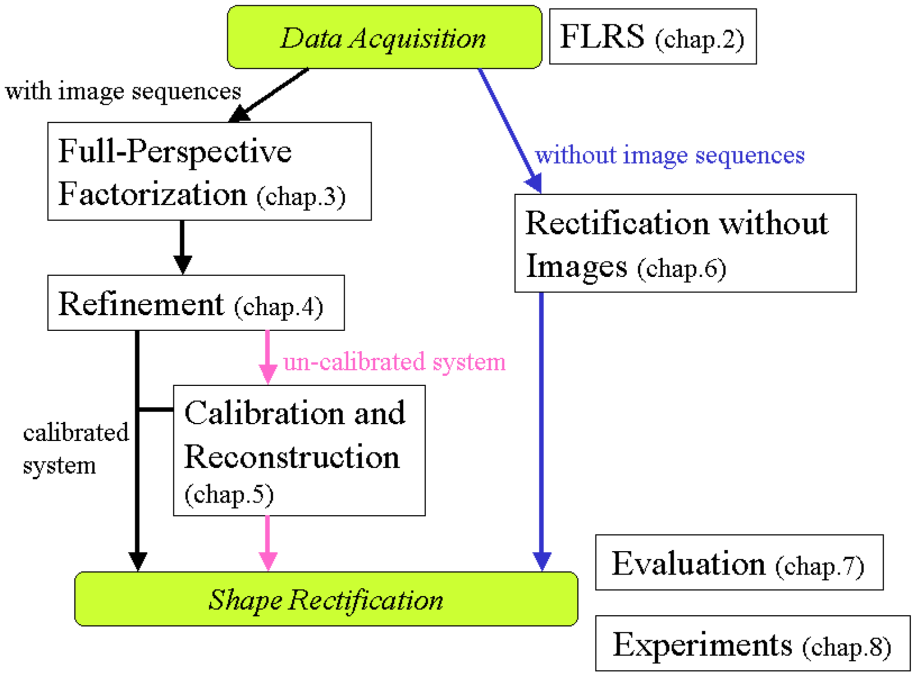


Figure 1.3: The context of this thesis.

as a scanning unit, we have designed whole system and the mirror configurations of the FLRS. Thus the FLRS is our original system. The system overview and components are introduced in this chapter. And we explain the algorithm of 3D reconstruction by using mirrors from a fixed-point measurement range data.

In Chapter 3, we explain a full perspective factorization, which is utilized as the initial value for the camera motion. We use a weak perspective factorization iteratively for the perspective projection camera model. Interest point detectors are essential for the factorization. Two detectors, Harris operator and SIFT key, are explained in this chapter. Furthermore, in the last part, we estimate the performance of our full perspective factorization.

In Chapter 4, we describe our proposed algorithm for refinement of the parameters. Our method applies three constraints for the optimization, which are tacking, smoothness and range data constraint. Implying these constraints and optimizing the cost function, we can estimate more precise parameters. For the optimization method, we apply a conjugated gradient method and the golden section search.

In the above method, the video camera is assumed to be calibrated with the range sensor. The method for the calibration is described in the first half of Chapter 5. By using 3D reference model, the video camera is calibrated. With respect to the FLRS, when we fix the FLRS on the ground and obtain range data and image sequence, we can easily acquire shape model and its image simultaneously. In the case of the calibration with 3D reference model, many methods suffer from noise in accuracy. Combining RANSAC (Random Sampling Consensus) technique, we propose a robust calibration method with 3D model in this chapter. Furthermore, we show that our algorithm is also applicable for the uncalibrated system in the second half of Chapter 5.

In Chapter 6, we describe another method for shape rectification which need not any image sequences. Instead of using images, this method requires range data obtained by another range sensor fixed on the ground. Incomplete range data of the fixed sensor are sufficient to rectify FLRS range data. There are many cases such a situation in real measurements. Originally, the FLRS has been proposed in order to complement fixed sensors. Based on overlapped shape between two data sets, we rectify FLRS range data. In this method, it is also assumed that the sensor moves smoothly. We can easily build a graphic user interface (GUI) onto this method and a practical software.

In Chapter 7, we evaluate our algorithms with known models. Constructing a virtual FLRS in PC by using CG model, we estimate the accuracy of our method.

In Chapter 8, we show several experimental results conducted in the Bayon Temple in Cambodia. To evaluate our methods, the recovered shapes are compared with other data sets obtained by a range sensor on the ground. Now, we are conducting the Digital Bayon Project, in which our algorithms are actually applied for range data processing.

Finally, we present our conclusions and summarize our possible future works in Chapter 9.

# Chapter 2

# FLRS

## 2.1 System Overview

FLRS(Floating Laser Range Sensor) has been developed to measure large objects from the air by using a balloon without constructing any scaffolds (Fig. 2.1). There are several demands for the system because of dangling the entire system under the balloon.

In the beginning of the development, the following points were required:

- The entire system should be light and compact in order to float in the air.
- The structure of the platform should be firm.
- The range sensor can measure quickly to minimize the influence of the balloon motion.

Several considerations suggested that we determined the configuration of the system and scanning time as 1.0 second.

With respect to measurement principle, passive stereopsis method could capture images without the influence of balloon motion. However it would be forecast to cause the fatal inadequate accuracy in the eye of the cultural heritage preservation and repair.

On the other hand, there are many active stereopsis methods with laser range sensors which can measure within 1 second. There are, however, a few problems in this measurement principle.

- unsuitable for large scale objects because they need wide baselines.

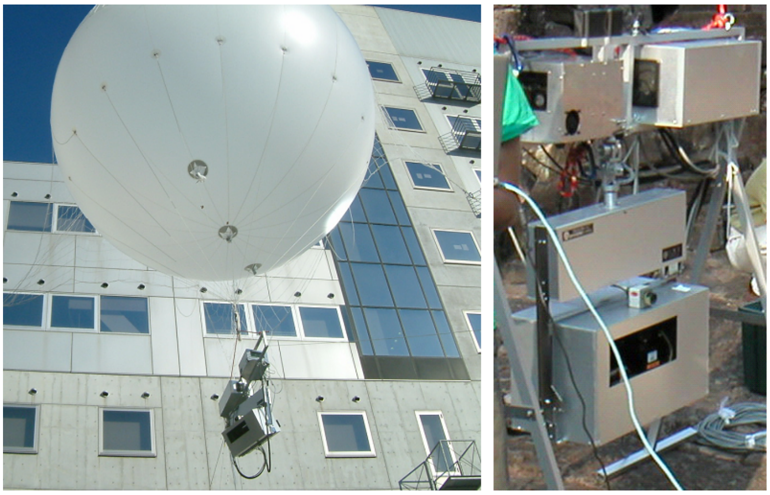


Figure 2.1: The FLRS (25m sensor)

- dangerous because they require strong laser beams for long range measurement.
- not adequate to measurement in daytime.

Generally, laser radar method is suitable for outdoor measurement for large objects. Therefore, we had adopt a range sensor of ”time-of-flight” in principle. Moreover we were able to utilize two kinds of mirrors to shorten the measurement processing time. Then we have designed and developed a novel measurement system based on the laser radar method.

Some details of the system will explained in the next section.

## 2.2 The Components of the FLRS

We have two types of FLRSs. Each FLRS is composed of a scanner unit, a controller and a personal computer (PC). These three units are suspended beneath a balloon.

### 2.2.1 The Scanner Unit

The scanner unit includes a laser range finder, especially designed to be suspended from a balloon. Figure 2.2 shows the interior of the scanner unit. It consists of a spot laser radar unit and two mirrors. We chose the LARA25200 and LARA53500 supplied by Zoller+Fröhlich GmbH[Z+F] as laser radar units because of their high sampling rate. Each laser radar unit is mounted each FLRS scanner unit. Two systems equipped with Lara25200 and LARA53500 are respectively referred to as ”25m sensor” and ”50m sensor”.

The specifications of two units are shown in Table 2.1.

Table 2.1: The specifications of the 25m (LARA25200) and 50m (LARA53500) Sensors

| | 25m Sensor | 50m Sensor |
|---|---|---|
| Ambiguity interval | 25.2 m | 53.5 m |
| Minimum range | 1.0 m | 1.0 m |
| Resolution 16bit range | 1.0 mm | 1.0 mm |
| Data acquisition rate | ≤ 625,000 pix/sec | ≤ 500,000 pix/sec |
| Linearity error | ≤ 3 mm | ≤ 5mm |
| Range noise at 10m | ≥ 1.0 mm | ≥ 1.5mm |
| Range noise at 25m | ≥ 1.8 mm | ≥ 2.7mm |
| Laser output power | 23 mW | 32mW |
| Laser wavelength | 780nm | 780nm |

Both sensors have the similar mirror configurations. There are two mirrors inside each unit to give a direction to the laser beam. One is a polygon mirror with 4 reflection surfaces, which determines the azimuth of the beam. In normal use, the polygon mirror, which rotates rapidly(2400rpm), controls the horizontal direction of the laser beam. Another is a plane mirror (swing mirror) which determines the elevation of the beam. The plane mirror swings slowly to controls the vertical direction of the laser beam.

The lase beam emitted from the LARA is hit on a surface of the polygon mirror at first. Then the polygon mirror reflects the laser beam into the plane mirror. The plane mirror also reflects the beam into the outside of the unit(lower of Fig.2.2).

The combination of two mirror demonstrate the following specifications.

### 2.2.2 The Controller and the PC

The controller is composed of a signal processing unit, an interface unit, a mirror controller and a power supply unit. The signal processing unit receives the signals from the PC and performs actual control of rotation angles of the mirrors and the

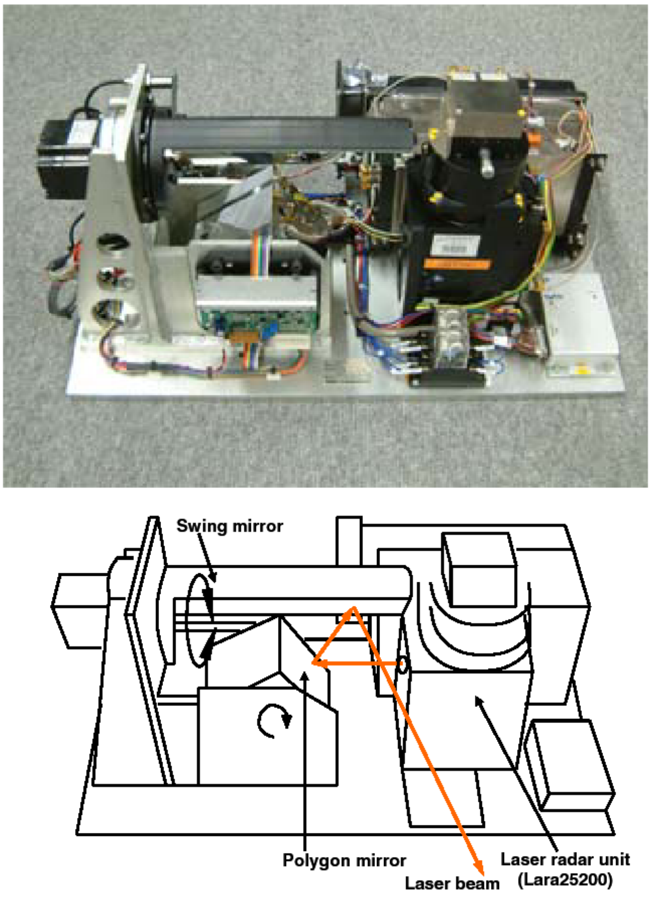


Figure 2.2: The interior of scanner unit (25m sensor)

Table 2.2: The specifications of the 25m sensor and 50m sensor

| | 25m Sensor | 50m Sensor |
|---|---|---|
| Angle Resolution | | |
| Horizontal | 0.05 deg | 0.05 deg |
| Vertical | 0.02 deg | 0.02 deg |
| Horizontal field | ≤ 90 deg | ≤ 90 deg |
| Vertical field | ≤ 30 deg | ≤ 30 deg |
| Scanning period/range image | ≤ 15 sec | ≤ 1 sec |

laser radar unit. The range data obtained by the laser radar and the angle data obtained by the mirror encoders are subsequently combined in the interface board.

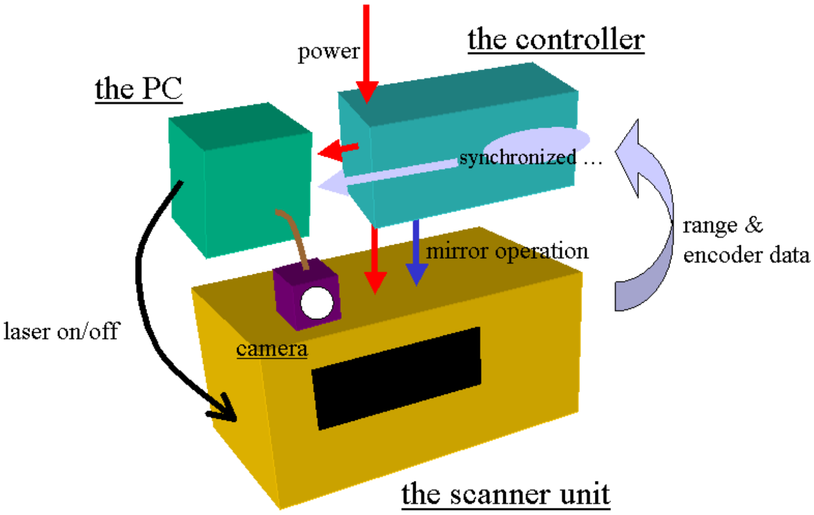


Figure 2.3: The diagram of the signals in the FLRS system

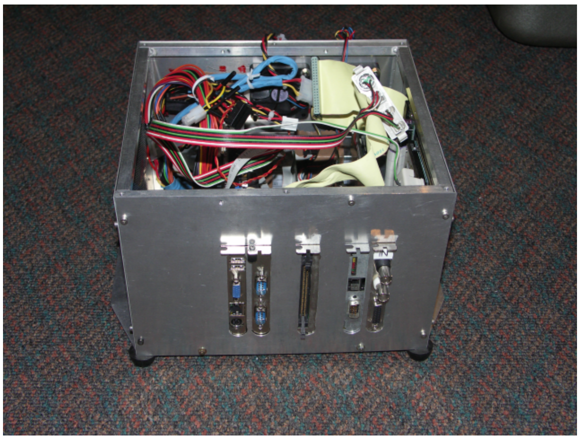


Figure 2.4: The PC of the FLRS system

The PC includes a CPU board, a DIO(Digital Input/Output) board, an image capture board and a LARA-PCI board. The DIO board outputs the signal of the laser on/off. The commands for the mirror operations are send through a LAN cable. Then synchronized range and encoder data (*.zfs) are transmitted to the PC via the LARA-PCI board. The zfs data consist of range data, reflectance and two kind of encoder data sets (the polygon and the plane mirror). These data sets are stored in the PC and converted into 3D shape data (*.pts). The PC of the FLRS is mounted on the balloon platform. Therefore the PC is actually operated remotely via another mobile PC on the ground through a LAN cable.

### 2.2.3 The Monitoring Camera

In order to monitor the object whose shape the FLRS scans, a camera is mounted on the platform(Fig.2.5). Because bulky data are transmitted into the PC in the scanning process, it is necessary to avoid CPU load with respect to the image capture. Therefore we adopt a capture board with SDRAM (Interface Corporation [Int]), which enable to stock image data temporally without any CPU load.

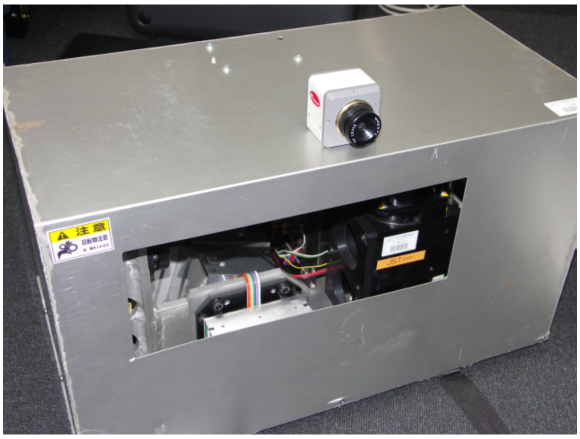


Figure 2.5: The monitoring camera mounted on the FLRS

The acquirable frame number is determined by the capacity of the SDRAM (64MByte). In a short period scanning (1 second) the capture board stocks an image sequence of VGA size (640x480) while in a long period scanning (over 3 seconds) it stocks images of 320x240 size.

By using this board, we can obtain whole images during a scanning process.

The ordinary use of the FLRS, the camera is calibrated before measurements. Calibration is to estimate the camera position and pose in the sensor-oriented coordinate system. Before floating the balloon in the air, we adjust the video camera roughly in order to capture the area where the range sensor scans (Fig.2.7). Then fixing the whole system on the ground, we measure several still scenes. By using these measurement data sets, the camera is calibrated through the method men-

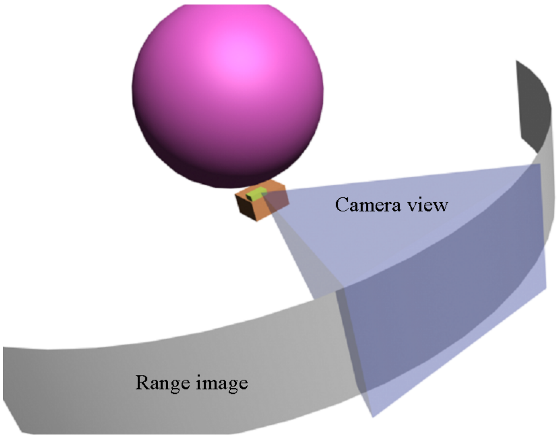


Figure 2.6: A range image and a camera view

tioned in Section 5.1.

### 2.2.4 The Balloon and the Platform

We use a ready-made device ”Photo Balloon AS-21”(Asahi Co., Ltd.), which is modified for the FLRS. The balloon is filled with helium gas. Floating in the air, the balloon is controlled by several hands on the ground with four peaces of rope. The balloon is made of particular flexible chloroethene, which avoid rapid expansion of a hole in an emergency.

Table 2.3: The specifications of the balloon

| | |
|---:|:---:|
| Diameter | 5.0 m |
| Weight | about 12 kg |
| Capacity | about 65.45 $m^3$ |
| Maximum buoyancy | about 60 kgf |

The platform is equipped with pan and tilt mechanism, which can point the sensor at from the horizontal direction to the directly below, scope of 180 degree from side to side. We can operate the pan and tilt mechanism via the cable for the video monitor.

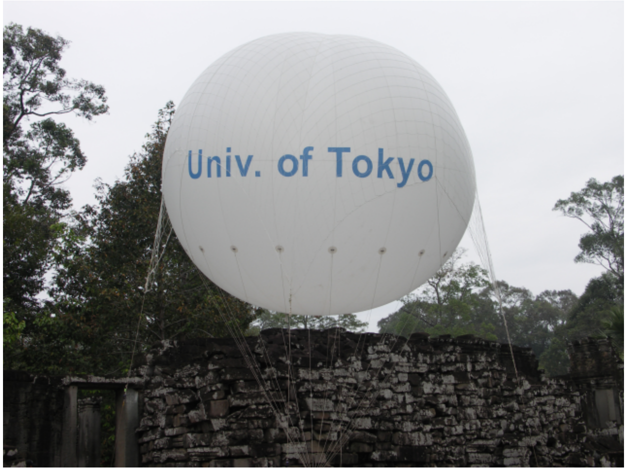


Figure 2.7: The balloon for the FLRS

### 2.2.5 The Operation

During the scanning process, the laser beam is directed horizontally by the rotating polygon mirror and vertically by the swinging plane mirror. The scanning with respect to the horizontal line of a range image is a fast one-way scanning. On the other hand, the vertical motion is a slow reciprocating one. To make up a range image with the raster scan order, we take plenty of time for a scanning process. By a single scan we actually utilize a portion of the whole data, which are acquired in a one side of a reciprocating motion. For example, it takes 1 second for a single scanning period. In this case, the FLRS actually acquires range data for 2 seconds. During the 2 seconds, a thorough one side motion must be contained, that is the top-to-bottom or bottom-to-top motion. It is the timing of a scan start that determines whether the top-to-bottom or bottom-to-top order. By using the half data, range data are constructed through the method mentioned in the next section. In the case of a 1 second scanning with a 2400rpm of the polygon mirror rotation, incidentally, the acquired range image includes 160 horizontal scan lines because

$$\frac{2400\ [rpm]}{60\ [sec]} \times 1\ [sec] \times 4\ [face/rev] = 160$$

In the case of a 5 seconds scanning, by the same token, the FLRS operates for 10 seconds in practice.

## 2.3 Data Reconstruction

As mentioned above, the stored data in the PC consist of range data, reflectance values and encoders' values of two rotors. By using these data, 3D coordinate values of measured points are reconstructed. In this section, we explain the reconstruction method.

First, let's determine the axes of the coordinate system. Taking account of the mirror configuration of the FLRS, we set $x$ as the direction where the laser beam is emitted from the laser radar unit. Then it corresponds to $z$ direction that the axis of rotation of the polygon mirror, while the rotary shaft of the plane mirror is parallel to the $x$-axis (Fig.2.8).

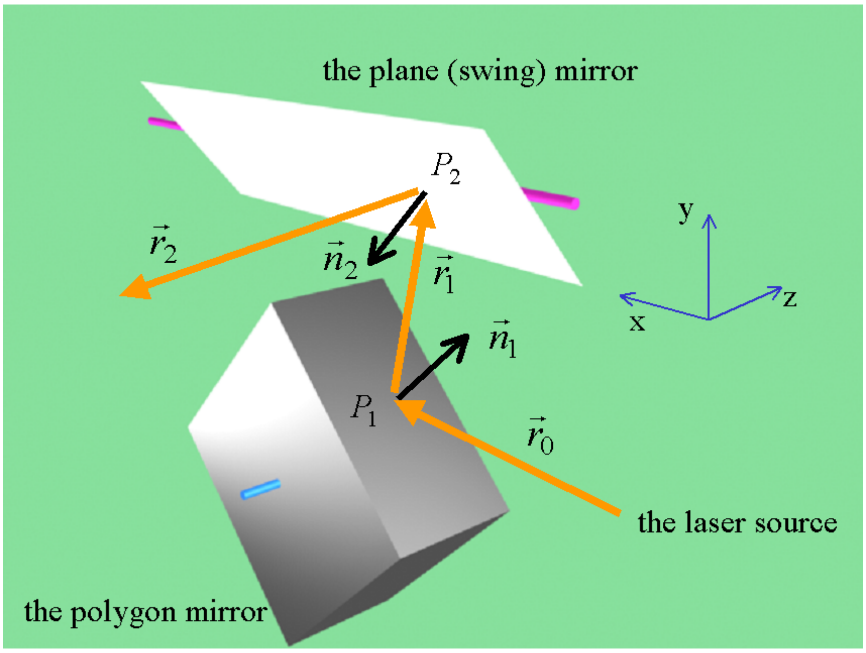


Figure 2.8: The mirror configuration of the FLRS

Then, we set the unit vector $\vec{r_0}$ $(= (1, 0, 0))$ of the laser beam from the laser radar unit. Supposing the normal vector of a polygon mirror surface as $\vec{n_1}$, the direction of the reflected beam toward the plane mirror can be described as $\vec{r_1}$

$$\vec{r_1} = \vec{r_0} - 2\,(\vec{r_0} \cdot \vec{n_1})\,\vec{n_1} \qquad (2.1)$$

Here, let us consider the cross section of the 3D configuration by the plane $z = 0$ (Fig.2.9). Setting the origin of the coordinate system at the center of the polygon

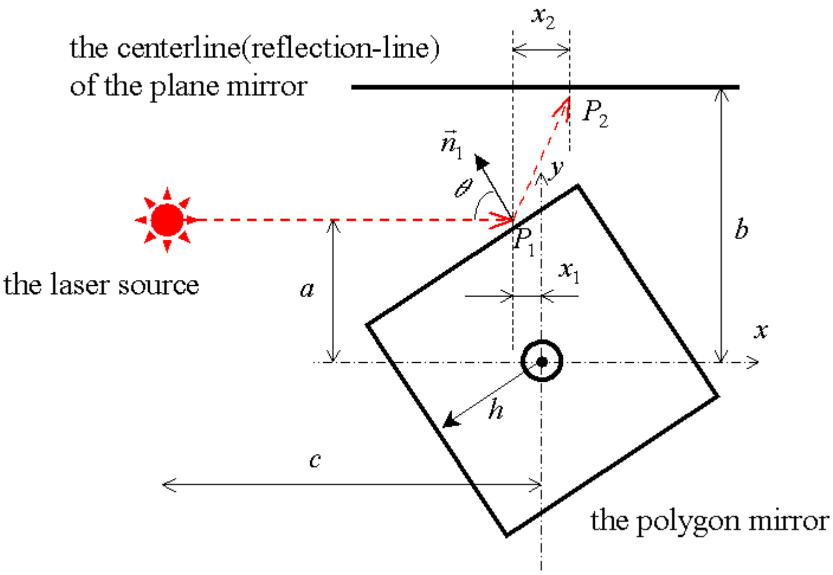


Figure 2.9: The 2-dimensional mirror configuration. (projected on the plane $z = 0$)

mirror and given the position of the laser light source at $(-c, a, 0)$, the reflection point $P_1$ on the polygon mirror surface is obtained by the following system.

$$\begin{cases} n_{1x}x \;+\; n_{1y}y - h = 0 \\ y - a = 0 \end{cases} \tag{2.2}$$

where, $\vec{n_1} = (n_{1x}, n_{1y}, 0)$ is the unit normal vector of the polygon mirror surface. Then we can obtain the reflection point $P_1 = (x, a, 0) = \left(\frac{h - n_{1y}a}{n_{1x}}, a, 0\right)$.

As in Fig.2.9, point $P_1$ is always has the minus value of x. We therefore define $x_1(\geq 0)$ the offset along the $x$-axis between the origin and the cross point as in Fig.2.9.

$$x_1 = -\frac{h - n_{1y}a}{n_{1x}} = \frac{h - a\sin\theta}{cos\theta} \tag{2.3}$$

Here, $\theta$ $\left(0 \leq \theta \leq \frac{\pi}{2}\right)$ indicates the angle between the normal vector of the polygon mirror surface and -$x$ direction as in Fig.2.9.

Next, let us consider the reflection on the plane mirror. From $P_1$ to the reflection point on the plane mirror $P_2$, the laser beam travels distance $l = P_1P_2$. Setting the offset between the origin and the centerline of the plane mirror as $b$, we can estimate the distance $l$ by using $\sin(\pi - 2\theta) = \dfrac{b-a}{l}$.

$$l = \frac{b-a}{\sin 2\theta} \tag{2.4}$$

Similarly, the $x$ element of $\vec{P_1P_2}$ is calculated as

$$x_2 = -\frac{b-a}{\tan 2\theta} \tag{2.5}$$

Note that $x_2$ takes a minus value in the case of $\theta < \dfrac{\pi}{4}$ and takes a positive value in the case of $\theta > \dfrac{\pi}{4}$.

Therefore, a laser beam hits the plane mirror at point $P_2$.

$$P_2 = (-x_1 + x_2, b, 0) = \left(-\frac{h - a\sin\theta}{\cos\theta} - \frac{b-a}{\tan 2\theta},\ b,\ 0\right); \tag{2.6}$$

Until arriving at point $P_2$, a laser beam flies $l + (c - x_1)$.

A laser beam is emitted outside of the scanner unit from point $P_2$ along direction $\vec{r_2}$.

$$\vec{r_2} = \vec{r_1} - 2\ (\vec{r_1} \cdot \vec{n_2})\ \vec{n_2} \tag{2.7}$$

where, $\vec{n_2} = (0, n_{2y}, n_{2z})$ is the unit normal vector of the plane mirror surface.

Therefore, when the laser finder outputs the range as $L$ for a point in space, the point is located at

$$(L - l - (c - x_1))\,\vec{r_2}$$

in the coordinate system with the origin $P_2$, which moves according to the mirror configuration.

Translating the origin to the center of polygon mirror,

$$(x, y, z) = P_2 + (L - l - c + x_1)\vec{r_2}$$
$$\begin{cases} x = \left(L - \dfrac{b-a}{\sin 2\theta} - c + \dfrac{h - a\sin\theta}{cos\theta}\right) r_{2x} - \dfrac{h - a\sin\theta}{cos\theta} - \dfrac{b-a}{\tan 2\theta} \\ y = \left(L - \dfrac{b-a}{\sin 2\theta} - c + \dfrac{h - a\sin\theta}{cos\theta}\right) r_{2y} + b \\ z = \left(L - \dfrac{b-a}{\sin 2\theta} - c + \dfrac{h - a\sin\theta}{cos\theta}\right) r_{2z} \end{cases} \tag{2.8}$$

Here, $\theta$ and $\vec{n_2}$ are estimated based on the encoders of the motors which rotate the mirrors. Then, we can reconstruct the 3D data from the array of 1D range data and encoded values.

# Chapter 3

# Full Perspective Factorization

In this chapter, we explain a full perspective factorization, which is utilized as the initial value for the camera motion. We use a weak perspective factorization iteratively for the perspective projection camera model. Interest point detectors are essential for the factorization. Two kind of the detectors, Harris operator and SIFT key, are explained in this chapter. Furthermore, in the last part, we estimate the performance of our full perspective factorization.

First, we briefly refer to some projection models which are utilized in computer vision. Then we explain weak-perspective factorization, which is subsequently extended to the perspective factorization as in [HK99]. In the next section, the weak-perspective factorization is extended to full perspective factorization. The solution by the full perspective factorization is utilized as the initial value for the optimizing problem described in the next chapter. Finally, some demonstrations of the full perspective factorization are shown.

## 3.1 Projection Model

The perspective projection model(Fig.3.1) can faithfully represent ordinary cameras. This model is corresponds to a pinhole camera.

$$\begin{cases} u = f\dfrac{x}{z} \\ v = f\dfrac{y}{z} \end{cases} \tag{3.1}$$

The mathematical description is, however, non-linear and that makes it difficult to treat the model. Therefore, some linear projection models have been formulated,

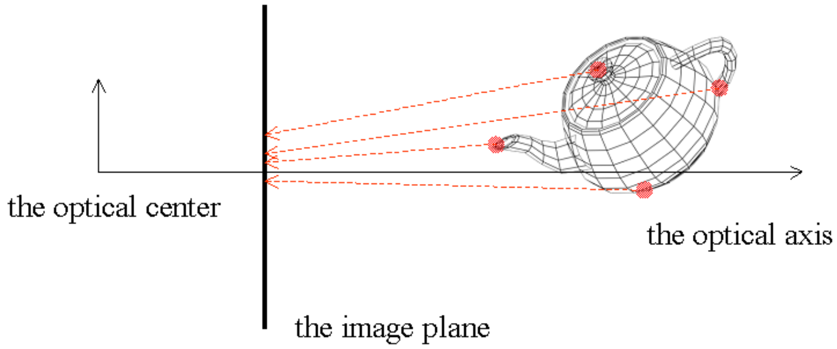


Figure 3.1: The perspective projection model (Pinhole camera model)

which are well-approximated to the non-linear projection model under certain condition.

In the rest of this section, we briefly explain three common approximation models.

### 3.1.1 Orthographic Model

The orthographic projection model(Fig.3.2) projects 3D points onto the image plane along the optical axis. This model is generally utilized in the field of technical designs such as drafts of buildings and machine designs. In this model, the coordinate values with respect to $x$ and $y$ in the 3D world are projected onto the image coordinates directly while the depth, $z$, is ignored.

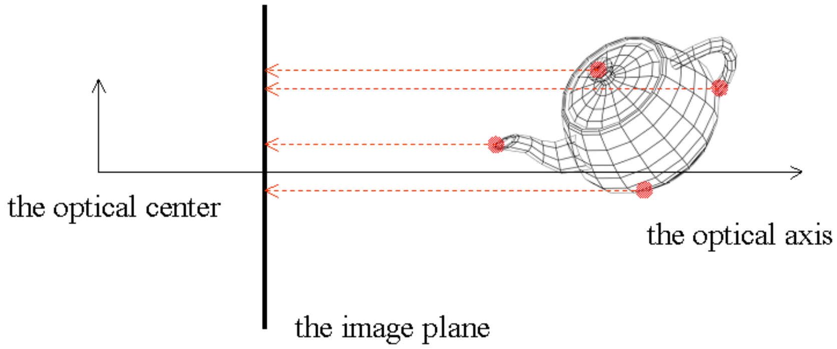


Figure 3.2: The orthographic projection model

Therefore, the orthographic projection is represented by the next equations:

$$\begin{cases} u = x \\ v = y \end{cases} \tag{3.2}$$

In this representation, the equation is simple linear and easy to handle while the perspective model is non-linear. However the assumption of the orthographic model is too simple to be applied to real cameras. There are few cases applicable to actual data for this model. The original factorization[TK92] was developed under the assumption of this simple projection model.

### 3.1.2 Weak-Perspective Model

This model is considered as an intermediate one between the perspective and orthographic projection. The weak-perspective model(Fig.3.3), which is also called scaled orthographic model, is approximated more accurately than orthographic model since the weak-perspective model has the scaling effect (closer objects appear bigger then further objects). But it is not so accurate as para-perspective model.

Consider a reference plane ($z = z_0$), which is located at the center of the object and parallel to the image plane. All points are, firstly, projected onto the reference plane along the optical axis. Then these projected points on the reference plane are projected again onto the image plane with a simple scale factor $f$ .

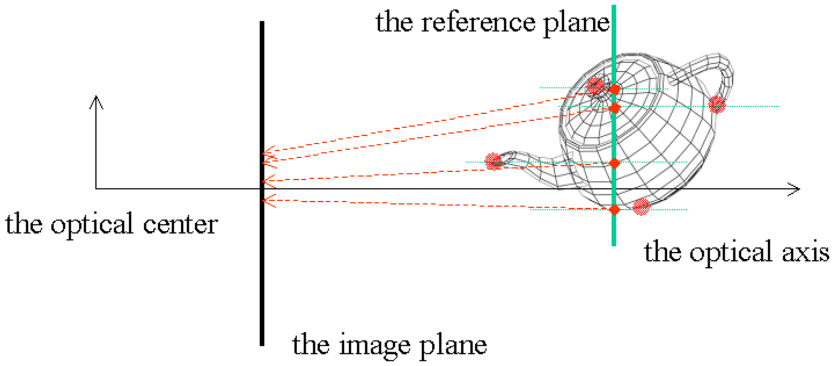


Figure 3.3: The weak-perspective projection model

The weak-perspective projection model supposes the depths of all point have the same value $z_0$. Therefore, this model is represented in linear manner with the focal length $f$ and the constant depth $z_0$:

$$\begin{cases} u = f\dfrac{x}{z_0} \\ v = f\dfrac{y}{z_0} \end{cases} \tag{3.3}$$

If the scene depth is enough small relative to the distance between the camera and the center of the object, the depths of all points can be taken to be constant. Therefore, this approximate model is valid only when the depth range of the object is considerably smaller than the distance to the object.

### 3.1.3 Para-Perspective Model

Para-perspective projection model(Fig.3.4) will be the closest approximate model among the linear models. It has the scaling effect and the position effect (objects in the periphery of the image are viewed from a different angle than those near the center of projection [Alo90]).

In this model, similarly consider the reference plane ($z = z_0$) located at the center of the object and parallel to the image plane. Next, object points are projected onto the plane along the direction of the line between the optical center and the object's center of mass. Then, the points projected on the reference plane are projected again onto the image plane with the scale factor $f$, which is equivalent to a simple scaling effect by the ration of the focal length $f$ and the distance to the reference plane. Therefore, the difference between the weak- and para-perspective model is the projection method onto the reference image.

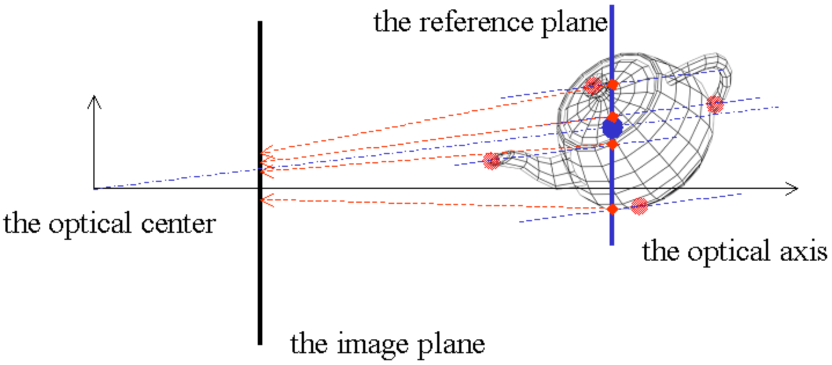


Figure 3.4: The para-perspective projection model

Given the object's center of mass $(x_0, y_0, z_0)$, this model is represented in linear manner with the focal length $f$ and the constant $z_0$:

$$\begin{cases} u = f\,\dfrac{x - x_0\dfrac{z}{z_0} + x_0}{z_0} \\[2ex] v = f\,\dfrac{y - y_0\dfrac{z}{z_0} + y_0}{z_0} \end{cases} \tag{3.4}$$

The above equations are certainly linear because of constant $z_0$.

### 3.1.4 Generalized Approximate Model

In the eyes of mathematics, above three approximate models are interpreted as follow:

Let us consider a point around the object's center of mass in space. The 3D coordinate value of the point $(x, y, z)$ is represented with the center $(x_0, y_0, z_0)$ as follows:

$$(x, y, z) = (x_0 + \delta x, y_0 + \delta y, z_0 + \delta z)$$

The coordinate value $u$ under the perspective projection model is estimated as $f\dfrac{x}{z}$ (Eq.5.1). Therefore, supposing $\delta z \ll z_0$.

$$\begin{aligned} u &= f\frac{x}{z} = f\,\frac{x}{z_0 + \delta z} = f\,\frac{x}{z_0}\frac{1}{1 + \dfrac{\delta z}{z_0}} \\ &\simeq f\frac{x}{z_0}\left(1 - \frac{\delta z}{z_0}\right) \end{aligned} \tag{3.5}$$

In the case of $f \rightarrow 1$, $z_0 \rightarrow 1$ and $\delta z \rightarrow 0$, this equation is close to the orthographic model.

In the case of $\delta z \rightarrow 0$, it is close to the weak-perspective model.

Then, from Eq.3.5,

$$u = f\,\frac{x_0}{z_0}\left(1 + \frac{\delta x}{x_0}\right)\left(1 - \frac{\delta z}{z_0}\right) \simeq f\,\frac{x_0}{z_0}\left(1 + \frac{\delta x}{x_0} - \frac{\delta z}{z_0}\right)$$

Here, the term with respect to $\delta x \delta z$ is ignored.

$$u = f\,\frac{1}{z_0}\left(x_0 + \delta x - x_0\frac{z - z_0}{z_0}\right) = f\,\frac{x - x_0\dfrac{z}{z_0} + x_0}{z_0} \tag{3.6}$$

Consequently, we can obtain the formulation for the para-perspective projection model.

As seen above transformations, these linear models are built upon the assumptions of $\delta x \ll x_0$, $\delta y \ll y_0$ and $\delta z \ll z_0$.

## 3.2 Factorization

### 3.2.1 Previous Works

Estimations of the shape of an object or of camera motion by using images are called ”Shape from Motion” or ”Structure from Motion”, and are main research fields in computer vision.

The factorization method proposed in [TK92] is one of the most effective algorithms for simultaneously recovering the shape of an object and the motion of the camera from an image sequence. By using the singular value decomposition(SVD), the shape and motion are estimated from the trajectories of interest points. Originally, this method was limited to the orthographic model. Then the factorization was extended to several perspective approximations and applications [CK95] [MK97] [CH96] [PK97] [HK99] [GW04]. Among them, in [PK97] a factorization method on the weak-perspective (or scaled orthographic projection) model was proposed, in which the scaling effect of an object is accounted for as it moves toward and away from the camera. At the same time, they applied the factorization method under the para-perspective projection model, which is a better approximation of the perspective model than that of the weak-perspective model. In the para-perspective model, the scaling effect as well as the different angles from which an object is viewed are accounted for as the object moves in a direction parallel to the image plane. In [PK97], they also presented perspective refinement by using the solution under the para-perspective factorization as the initial value. In [HK99] a factorization method with a perspective camera model was proposed. Using the weak-perspective projection model, they iteratively estimated the shape and the camera motion under the perspective model.

### 3.2.2 Weak-Perspective Factorization

Given a sequence of F images, in which we have tracked P interest points over all frames, each interest point p corresponds to a single point $\vec{S_p}$ on the object. In image coordinates, the trajectories of each interest point are denoted as $\{(u_{fp}, v_{fp})| f =$

$1, ..., F, p = 1, ..., P \;\; 2F \geq P\}$.

Using the horizontal coordinates $u_{fp}$, we can define an $F \times P$ matrix $U$. Each column of the matrix contains the horizontal coordinates of a single point in the frame order, while each row contains the horizontal coordinates for a single frame. Similarly, we can define an $F \times P$ matrix $V$ from the vertical coordinates $v_{fp}$. With respect to the coordinate values of $u_{fp}$ and $v_{fp}$, we set the origin of the coordinate system as the principal point.

The combined matrix of $2F \times P$ becomes the measurement matrix as follows,

$$W = \left( \frac{U}{V} \right) \tag{3.7}$$

Each frame f is taken at camera position $\vec{T_f}$ in the world coordinates. The camera pose is described by the orthonormal unit vectors $\vec{i_f}$, $\vec{j_f}$ and $\vec{k_f}$. The vectors $\vec{i_f}$ and $\vec{j_f}$ correspond to the $x$ and $y$ axes of the camera coordinates, while the vector $\vec{k_f}$ corresponds to the $z$ axis along the direction perpendicular to the image plane (Fig.3.5).

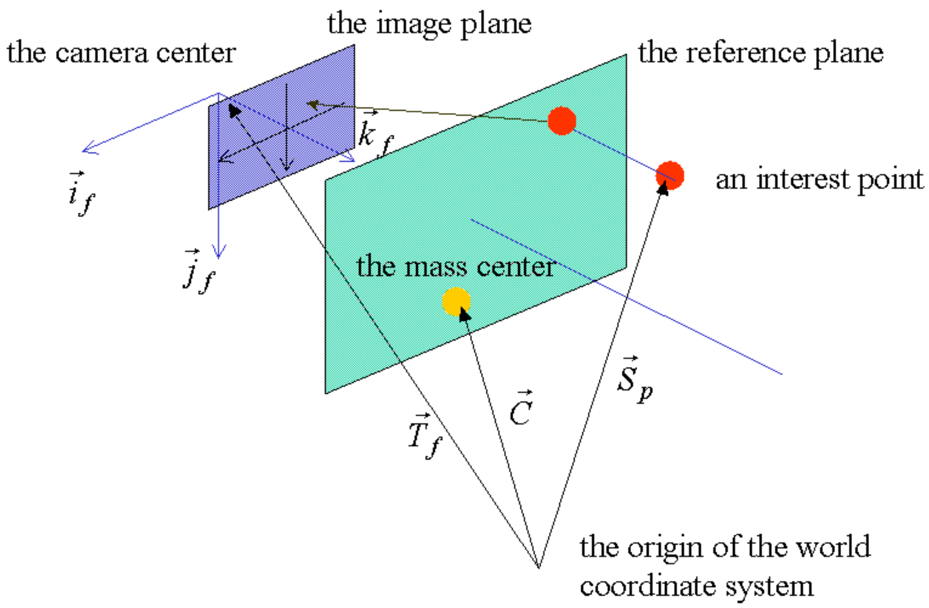


Figure 3.5: the coordinate system: $\vec{T_f}$ denotes the position of the camera at time of frame f. The camera pose is determined by three unit basis vectors.

Under the weak-perspective camera model, we can derive the following equation from (Eq. 3.3).

$$\begin{cases} u = f\dfrac{x}{z_0} = f\dfrac{\vec{i_f}^{\,t} \cdot (\vec{S_p} - \vec{T_f})}{z_f} \\ v = f\dfrac{y}{z_0} = f\dfrac{\vec{j_f}^{\,t} \cdot (\vec{S_p} - \vec{T_f})}{z_f} \end{cases} \tag{3.8}$$

Here, a single point in the world coordinates $\vec{S_p}$ is projected onto the image plane f at $(u_{fp}, v_{fp})$ of a camera at $\vec{T_f}$ in the world coordinate system.

We denote the distance between the camera center and the reference plane (the mass center of the object) as $z_f$. Then we obtain the following,

$$z_f = \vec{k_f}^{\,t} \cdot (\vec{C} - \vec{T_f}) \tag{3.9}$$

The vector $\vec{C}$ is the center of mass of all interest points. Without loss of generality, the origin of the world coordinates can be placed at the centroid, that is $\vec{C} = \sum \vec{S_p} = \mathbf{0}$. Then this means that

$$z_f = -\vec{k_f}^{\,t} \cdot \vec{T_f} \tag{3.10}$$

to simplify the expansion of the following formulations.

They are summarized as follows:

$$\begin{aligned} u_{fp} &= \vec{m_f}^{\,t} \cdot \vec{S_p} + \mathbf{x_f} && (3.11)\\ v_{fp} &= \vec{n_f}^{\,t} \cdot \vec{S_p} + \mathbf{y_f} && (3.12)\\ \vec{m_f} &= \frac{f}{z_f}\,\vec{i_f} && (3.13)\\ \vec{n_f} &= \frac{f}{z_f}\,\vec{j_f} && (3.14)\\ \mathbf{x_f} &= -\frac{f}{z_f}\,\vec{i_f}^{\,t} \cdot \vec{T_f} && (3.15)\\ \mathbf{y_f} &= -\frac{f}{z_f}\,\vec{j_f}^{\,t} \cdot \vec{T_f} && (3.16) \end{aligned}$$

and these equations are expressed in a matrix form:

$$\begin{pmatrix} u_{11} & \dots & u_{1P} \\ u_{21} & \dots & u_{2P} \\ \vdots & \vdots & \vdots \\ u_{F1} & \dots & u_{FP} \\ v_{11} & \dots & v_{1P} \\ \vdots & \vdots & \vdots \\ v_{F1} & \dots & v_{FP} \end{pmatrix} = \begin{pmatrix} \vec{m_1}^{\,t} \\ \vec{m_2}^{\,t} \\ \vdots \\ \vec{m_F}^{\,t} \\ \vec{n_1}^{\,t} \\ \vdots \\ \vec{n_F}^{\,t} \end{pmatrix} (\vec{s_1} \ \dots \ \vec{s_P})$$

$$+\begin{pmatrix} \mathbf{x_1} \\ \mathbf{x_2} \\ \vdots \\ \mathbf{x_F} \\ \mathbf{y_1} \\ \vdots \\ \mathbf{y_F} \end{pmatrix} (1 \ \ldots \ 1) \tag{3.17}$$

Using the setting that the center of all interest points is the origin, from Eq.(3.11),

$$\sum_{p=1}^{P} u_{fp} = \sum_{p=1}^{P} \vec{m_f}^{\,t} \cdot \vec{s_p} + \sum_{p=1}^{P} \mathbf{x_f} = P\mathbf{x_f} \tag{3.18}$$

similarly from Eq.(3.12),

$$\sum_{p=1}^{P} v_{fp} = P\mathbf{y_f} \tag{3.19}$$

Therefore, $\mathbf{x_f}$ and $\mathbf{y_f}$ are easily calculated with all interest points.

$$\begin{cases} \mathbf{x_f} = \dfrac{1}{P} \displaystyle\sum_{p=1}^{P} u_{fp} \\ \mathbf{y_f} = \dfrac{1}{P} \displaystyle\sum_{p=1}^{P} v_{fp} \end{cases} \tag{3.20}$$

We obtain the registered measurement matrix $\tilde{W}$, after translation $\tilde{W} = W - (\mathbf{x_1}\ \mathbf{x_2}\ \ldots\ \mathbf{x_F}\ \mathbf{y_1}\ \ldots\ \mathbf{y_F})^t(1\ 1\ldots 1)$ as a product of two matrixes $M$ and $S$.

$$\tilde{W} = M \cdot S \tag{3.21}$$

where $M$ is a $2F \times 3$ matrix and $S$ is a $3 \times P$ matrix.

In [TK92], they stated the following;

**Theorem (Rank Theorem)** *Without noise, the registered measurement matrix $\tilde{W}$ is at most of rank 3.*

This theorem means that the registered measurement matrix $\tilde{W}$ of $2F \times P$ are highly redundant. The matrix $\tilde{W}$ is originally the product of the $2F \times 3$ matrix $M$ and the $3 \times P$ matrix $S$. Therefore, it follows that the matrix $\tilde{W}$ has at most rank three.

With respect to the decomposition of the matrix $\tilde{W}$, we utilized the Singular Value Decomposition(SVD) [GL96]. By the SVD, supposing that $2F \geq P$, the matrix $\tilde{W}$ is decomposed as follows.

$$\tilde{W} \ = O_1 \, \Sigma \, O_2 \tag{3.22}$$

where $O_1$ is a $2F \times P$ matrix, $\Sigma$ is a diagonal $P \times P$ matrix and $O_2$ is a $P \times P$ matrix. In addition, the matrices $O_1$ and $O_2$ fulfill that $O_1^t \ O_1 = O_2^t \ O_2 = E$, where $E$ is the $P \times P$ unit matrix. The matrix $\Sigma$ has only diagonal elements (other elements are 0), which are the singular values $\sigma_1 \geq \ldots \geq \sigma_P \geq 0$ sorted in non-decreasing order.

The above Rank Theorem also says that the matrix $\Sigma$ has at most three singular values of non-zero. It is, therefore, only necessary to consider the first three columns of $O_1$, the most upper left $3 \times 3$ submatrix of $\Sigma$ and the first 3 rows of $O_2$.

In real case, the observed values, that mean the coordinate values of the interest points $u_{fp}$ and $v_{fp}$, include noises. Consequently more than three diagonal elements of matrix $\Sigma$ are non-zero. In the case of noisy measurements, the following theorem is provided.

**Theorem (Rank Theorem for Noisy Measurements)** *All the shape and rotation information in $\tilde{W}$ is contained in its three greatest singular values.*

Therefore, we can deal with the registered measurement matrix $\tilde{W}$ with noise in the same manner. The singular values out of the first three corresponds to the noises.

As mentioned above, we have only to deal with the first three columns of $O_1$, the most upper left $3 \times 3$ submatrix of $\Sigma$ and the first three rows of $O_2$. Then, we suppose the following partitions of $O_1$, $\Sigma$ and $O_2$.

$$\begin{aligned} O_1 &= (O_1' \ O_1'') \\ \Sigma &= \begin{pmatrix} \Sigma' & \mathbf{0} \\ \mathbf{0} & \Sigma\text{''} \end{pmatrix} \\ O_2 &= \begin{pmatrix} O_2' \\ O_2'' \end{pmatrix} \end{aligned} \tag{3.23}$$

where the $O_1'$ is a $2F{\times}3$ matrix, $\Sigma'$ is a diagonal $3{\times}3$ matrix and $O_2'$ is a $3{\times}P$matrix.

Without noise, the following equations are perfectly satisfied.

$$\tilde{W} = M \cdot S \ = O_1 \, \Sigma \, O_2 = O_1' \, \Sigma' \, O_2' \tag{3.24}$$

$$O_1'' \, \Sigma'' \, O_2'' = \mathbf{0} \tag{3.25}$$

In noisy cases, the term of $O_1'' \ \Sigma'' \ O_2''$ corresponds to noise. As a consequence, we can regard

$$\tilde{W} \simeq \hat{W} = O_1 \ \Sigma \ O_2 = O_1'$$

and consider $\hat{W}$ form here on.

Let us return the decomposition of the registered measurement matrix into the rotation matrix $M$ and the shape matrix $S$ (Eq.3.21). In the meanwhile, we define

$$\hat{M} = O_1' \ \sqrt{\Sigma'} \tag{3.26}$$

$$\hat{S} = \sqrt{\Sigma'} \ O_2' \tag{3.27}$$

we obtain the next equation.

$$\hat{W} = \hat{M} \ \hat{S} \tag{3.28}$$

Here, $\hat{M}$ is a $2F \times 3$ matrix and $hatS$ is a $3 \times P$ matrix, which posses the same configurations of Eq.3.21.

The above decomposition, however, is not unique because any invertible $3 \times 3$ matrix A makes a valid decomposition of $\hat{W}$ as

$$(\hat{M}A)(A^{-1}\hat{S}) = \hat{M}(AA^{-1})\hat{S} = \hat{M}\hat{S} = \hat{W} \tag{3.29}$$

To get rid of the ambiguity, using the fact that the matrix $M$ represents the axes of the camera coordinates(Eq.3.13 and 3.14), the following constraints should be satisfied.

$$|\vec{m_f}| = |\vec{n_f}| \tag{3.30}$$

$$\vec{m_f}^{\,t} \cdot \vec{n_f} = 0 \tag{3.31}$$

$$where, \hat{M}A = \left(\begin{array}{c} [\vec{m_f}^{\,t}] \\ \hline [\vec{n_f}^{\,t}] \end{array}\right) \tag{3.32}$$

These constraints give us the motion matrix $M$ and the shape matrix $S$.

Then, we need to estimate a $3 \times 3$ matrix $A$. From Eq.3.32, two $F \times 3$ matrices $\hat{M}'$ and $\hat{N}'$ are defined as follows.

$$\left(\begin{array}{c} [\vec{m_f}^{\,t}] \\ \hline [\vec{n_f}^{\,t}] \end{array}\right) = \hat{M}A = \left(\begin{array}{c} \hat{M}' \\ \hline \hat{N}' \end{array}\right) A \tag{3.33}$$

$$\hat{M}' = \begin{pmatrix} \vec{m'_1}^{\,t} \\ \vec{m'_2}^{\,t} \\ \vdots \\ \vec{m'_F}^{\,t} \end{pmatrix} \tag{3.34}$$

$$\hat{N}' = \begin{pmatrix} \vec{n'}_1^{\,t} \\ \vec{n'}_2^{\,t} \\ \vdots \\ \vec{n'}_F^{\,t} \end{pmatrix} \tag{3.35}$$

Considering Eq.3.32 and each vector $\vec{m'}_f$ ,

$$|\vec{m}_f|^2 = \vec{m}_f^{\,t} \cdot \vec{m}_f = (\vec{m'}_f^{\,t} A) \cdot (\vec{m'}_f^{\,t} A)^t = \vec{m'}_f^{\,t} A A^t \vec{m'}_f = \vec{m'}_f^{\,t} T \vec{m'}_f \tag{3.36}$$

$T = AA^t$ is a $3 \times 3$ symmetric matrix. Similarly on $\vec{n'}_f$ ,

$$|\vec{n}_f|^2 = \vec{n'}_f^{\,t} T \vec{n'}_f \tag{3.37}$$

In addition,

$$\vec{m}_f^{\,t} \cdot \vec{n}_f = (\vec{m'}_f^{\,t} A) \cdot (\vec{n'}_f^{\,t} A)^t = \vec{m'}_f^{\,t} T \vec{n'}_f \tag{3.38}$$

Here, estimating the matrix $A$ corresponds to estimating the matrix $T$. Then based on the constraints of Eq.3.30 and 3.31, the next cost function $G$ should be minimized to estimate the symmetric matrix $T$.

$$\begin{aligned} G &= \sum_{f=1}^{F} \Big( \ ( \ |\vec{m}_f|^2 - \ |\vec{n}_f|^2 \ )^2 \ + \ w \ (\vec{m}_f^{\,t} \cdot \vec{n}_f)^2 \ \Big) \\ &= \sum_{f=1}^{F} \Big( \ ( \ \vec{m'}_f^{\,t} T \vec{m'}_f \ - \ \vec{n'}_f^{\,t} T \vec{n'}_f \ )^2 \ + \ w \ (\vec{m'}_f^{\,t} T \vec{n'}_f)^2 \ \Big) \end{aligned} \tag{3.39}$$

$w$ : a weighted coefficient

In this thesis, $w$ is set at 1. We can easily minimize the cost function $G$ by a linear method to obtain the symmetric matrix $T$ (see Appendix A).

Once obtaining the matrix T, we can calculate the $3 \times 3$ matrix $A$ as follows. First, $T$ is decomposed as

$$T = U \Lambda V^t \tag{3.40}$$

where $U$ and $V$ are both $3 \times 3$ matrices and $\Lambda$ is a diagonal $3 \times P$ matrix, just like Eq.3.22. In this particular case of the symmetric matrix $T$, the matrix $U$ is identical with the matrix $V$. Consequently,

$$\begin{aligned} T = U \Lambda U^t &= U \sqrt{\Lambda} \sqrt{\Lambda} U^t = \Big( U \sqrt{\Lambda} \Big) \Big( U \sqrt{\Lambda} \Big)^t = AA^t \\ \therefore \ A &= U \sqrt{\Lambda} \end{aligned} \tag{3.41}$$

Then based on Eq.3.29 we obtain the approximate shape matrix $S$ as

$$S = (\vec{S}_1\ \vec{S}_2\ \ldots\ \vec{S}_P) = A^{-1}\hat{S} \tag{3.42}$$

For $\vec{m}_f$ and $\vec{n}_f$, from Eq.(3.32)

$$\vec{m}_f = A^t\vec{m'}_f \tag{3.43}$$

$$\vec{n}_f = A^t\vec{n'}_f \tag{3.44}$$

The distance between the camera center and the reference plane, $z_f$, is calculated by Eq.3.13 and 3.14;

$$|\vec{m}_f|^2 = \frac{f^2}{{z_f}^2} \quad and \quad |\vec{n}_f|^2 = \frac{f^2}{{z_f}^2}$$

$$\therefore z_f = f\sqrt{\frac{2}{|\vec{m}_f|^2 + |\vec{n}_f|^2}} \tag{3.45}$$

Then, the axises of the camera coordinate system $\vec{i}_f$ and $\vec{j}_f$ can be calculated. Another axis $\vec{k}_f$ is estimated as the cross product $\vec{i}_f \times \vec{j}_f$. However, it does not assure the orthogonality between $\vec{i}_f$ and $\vec{j}_f$. Then three axises are given in practice by the following post-treatment with the SVD.

$$\begin{pmatrix} \vec{i}_f & \vec{j}_f & \vec{k}_f \end{pmatrix} = U\Sigma V^t, \quad then \quad U\begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix}V^t \rightarrow \begin{pmatrix} \vec{i}_f & \vec{j}_f & \vec{k}_f \end{pmatrix} \tag{3.46}$$

When it comes to the camera position $\vec{T}_f$, we can obtain the next equation from Eq.3.10, 3.15 and 3.16.

$$\begin{pmatrix} \vec{m}_f{}^t \\ \vec{n}_f{}^t \\ \vec{k}_f{}^t \end{pmatrix}\vec{T}_f = \begin{pmatrix} \mathbf{x_f} \\ \mathbf{y_f} \\ z_f \end{pmatrix} \tag{3.47}$$

$\vec{T}_f$ is easily calculated as the linear solution for the above system.

Under the assumption of the weak perspective projection model, by using known values of $(u_{fp},\ v_{fp})$ and $f$, we can obtain unknown parameters of $\vec{i}_f$, $\vec{j}_f$, $\vec{k}_f$, $\vec{S}_p$ and $\vec{T}_f$.

Finally, there is one further problem that we can't ignore. It is an enantiomorph problem. As a matter of fact, the weak perspective factorization gives two kinds of solutions. If a certain shape, $\vec{S}_p$, is proper for the solution, the enentimorph is also proper for the solution. This means, there is another solution in Eq.3.29

$$\hat{W} = \hat{M}\hat{S} = (\hat{M}A)(A^{-1}\hat{S}) = \left(\hat{M}(-A)\right)\left((-A)^{-1}\hat{S}\right) \tag{3.48}$$

The shape $(-A)^{-1}\hat{S}$ is the enantimorph for the model $(A)^{-1}\hat{S}$. Thus, the weak perspective factorization leaves the shape ambiguity. The only way to determine which shape should be adopted as the correct one would be the choice by eye observation. However our algorithm can select the proper shape automatically by using deformed range data as mentioned in next section.

### 3.2.3 Extension to Full-Perspective Factorization

The above formulation is under the weak perspective projection model, which is a linear approximation of the perspective model. Next, using an iterative framework, we obtain approximate solutions under the non-linear, full perspective projection model.

Under the perspective projection model, the projective equations between the object point $\vec{S_p}$ in 3D world and the image coordinate $(u_{fp}, v_{fp})$ are written as

$$u_{fp} = f\frac{\vec{i_f}^t \cdot (\vec{S_p} - \vec{T_f})}{\vec{k_f}^t \cdot (\vec{S_p} - \vec{T_f})} \tag{3.49}$$

$$v_{fp} = f\frac{\vec{j_f}^t \cdot (\vec{S_p} - \vec{T_f})}{\vec{k_f}^t \cdot (\vec{S_p} - \vec{T_f})} \tag{3.50}$$

Replacing $z_f = -\vec{k_f}^t \cdot \vec{T_f}$, we obtain the following equations.

$$\begin{aligned}(\lambda_{fp} + 1)u_{fp} &= \frac{f}{z_f}\vec{i_f}^t \cdot (\vec{S_p} - \vec{T_f}) && (3.51)\\ (\lambda_{fp} + 1)v_{fp} &= \frac{f}{z_f}\vec{j_f}^t \cdot (\vec{S_p} - \vec{T_f}) && (3.52)\\ \lambda_{fp} &= \frac{\vec{k_f}^t \cdot \vec{S_p}}{z_f} && (3.53)\end{aligned}$$

Note that the right hand sides of Eq.3.51 and Eq.3.52 are the same form under the weak-perspective model (see Eq.3.8). This means, multiplying a image coordinate $(u_{fp}, v_{fp})$ by a real number $\lambda_{fp}$ maps the coordinate in the full perspective model space into the coordinate in the weak-perspective model space. Solving for the value of $\lambda_{fp}$ iteratively, we can obtain motion parameters and coordinates of interest points under the full perspective projection model in the framework of weak-perspective factorization.

The entire algorithm of the perspective factorization is as follows:

***Input:*** An image sequence of F frames tracking P interest points.

***Output:*** The 3D positions of P interest points $\vec{S_p}$. The camera position $\vec{T_f}$ and poses $\vec{i_f}$, $\vec{j_f}$, $\vec{k_f}$ at each frame f.

1. Given $\lambda_{fp} = 0$
2. Supposing the Equations 3.51 and 3.52, solve for $\vec{S_p}$, $\vec{T_f}$, $\vec{i_f}$, $\vec{j_f}$, $\vec{k_f}$ and $z_f$ through the weak perspective factorization.
3. Calculate $\lambda_{fp}$ by Equation 3.53.
4. Substitute $\lambda_{fp}$ into step 2 and repeat the above procedure.

***Until:*** $\lambda_{fp}$'s are close to ones at the previous iteration.

We must now return to the point which we postponed in the previous section, the enantiomorph problem. In fact, the ambiguity of enantiomorph is removed in our method. With respect to $\lambda$, if a point $\vec{S_p}$ is located on the reference plane, the value of $\lambda_{fp} = 0$ because the value of $z_f$ means the depth of the reference plane for the camera $f$, and the value of $\vec{k_f}^t \cdot \vec{S_p}$ means the depth of the point $\vec{S_p}$ for the camera $f$. The value of $\lambda_{fp}$ takes more than 0 for the point $\vec{S_p}$ located further away than the reference plane from the camera $f$. Similarly, $\lambda_{fp}$ for the point closer than the reference plane from the camera takes a negative value.

On the other hand, we measure the temporal relative position of each interest points (see Section 4.3). Supposing the frame $f_p$ in which the range sensor scans the interest point $p$, we can obtain $\vec{k_f}^t(\vec{S_p} - \vec{T_{f_p}})$, the depth of the interest point $p$ at frame $f_p$ as the observed value by a moving range sensor. But we can not obtain the value $\lambda_{f_p}$ exactly because we do not have any information about the depth of the reference plane at frame $f_p$.

Nevertheless, if the sensor does not move so widely along the optical axes direction, we can roughly estimate the $\lambda_{f_p}$.

Roughly speaking we can regard $z_f$ as a constant $z_c$ at all frames in the case of small sensor motion along to the optical axis. Then $\lambda_{fp}$ is roughly calculated as constant for each interest point $p$, $\lambda_{fp} \simeq \lambda_p \simeq \lambda_{f_p}$. Supposing that we observe the depth values of interest point $p$ as $D_p$ at frame $f_p$ from the distorted range data of FLRS. Therefore, we can estimate $\lambda_{f_p}$ from the range data based on the approximation of $z_c \simeq \frac{1}{P}\sum D_p$.

Let us consider the enantiomorph with two arrays, $\{\lambda_1^A + 1, \lambda_2^A + 1, \cdot, \lambda_P^A + 1\}$ and $\{\lambda_1^B + 1, \lambda_2^B + 1, \cdot, \lambda_P^B + 1\}$. When we also obtain the array of $\{\frac{D_1}{z_c}, \frac{D_2}{z_c}, \cdots, \frac{D_P}{z_c}\}$ from the range data, we can select the proper shape model $\{\lambda^A\}$ or $\{\lambda^B\}$ by comparing the correlations.

## 3.3 Tracking

As input stuff, we need P interest points at each frame whole a sequence, which are tracked identified points in the 3D world. There are several methods to derive interest points of images [Mor77] [SB97]. Among them, we adopt *Harris operator* [HS88] and *SIFT key* [Low99] [Low04] for derivation of interest points. Harris operator, a corner detector, is the most famous operator in the field of image processing. While SIFT key was originally proposed for the purpose of object recognition. This operator is robust for scale, rotation and affine transformation changes. Many operators with robustness for these changes are recently proposed. The main reasons why we adopt SIFT key are its stability of points derivation and usefulness of the key, which has 128 dimensional elements and can be used for the identification for each point.

### 3.3.1 Harris Operator

First, let us consider the spatial gradient of intensities, $(E_x, E_y) = (\frac{\partial X}{\partial x}, \frac{\partial X}{\partial y})$. Then we define a matrix $C$ at a point $p$ based on its neighborhood as follows,

$$C = \begin{pmatrix} \sum {E_x}^2 & \sum E_x E_y \\ \sum E_x E_y & \sum {E_y}^2 \end{pmatrix} \tag{3.54}$$

The key for feature detection is the eigenvalue of matrix $C$ and their geometric meanings. Matrix $C$ is a symmetric one and without any loss of generality it can be diagonalize by a rotation of the two coordinate axes.

$$C \mapsto \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$$

$\lambda_1$ and $\lambda_2$ are the eigenvalues of $C$ ($\lambda_1 > \lambda_2 > 0$).

If the region around point $p$ in the image is perfectly even and $E_x = E_y = 0$, matrix C has eigenvalues $\lambda_1 = \lambda_2 = 0$. If point $p$ is located on a line or an edge where is even along one direction and has a intensity gradient along the another, we obtain $\lambda_1 > 0,\ \lambda_2 = 0$. In fact, the larger the intensity gradient, the larger its corresponding eigenvalue. That is, the eigenvectors encode edge directions and the eigenvalues encode the strengths of the edges. Then, if point $p$ is located on a corner which has gradients along the both directions, we obtain $\lambda_1 > \lambda_2 > 0$. It means that point $p$ is located on a strong corner in the case of large $\lambda_2$. If the smaller eigenvalue is larger than a threshold, the point $p$ is to be an interest point.

Instead of actual calculating the smaller eigenvalue, the next value is evaluated.

$$r = \det C - \kappa(traceC)^2 \tag{3.55}$$

In the most studies with Harris operator, $\kappa = 0.04$ is used and we adopt it. Then the interest points are detected if the values $r$ at corresponding points are greater than the threshold.

With respect to the inter-frame connections of each interest point, we adopt a local window matching method.

Consequently, our tracking algorithm with Harris operator is as follows.

1. Given a image sequence of F frames.

2. Harris operator is applied to the all images and detects $P_{max}$ interest points at each frame ($P_{max} > P$).

3. Each interest point at frame $f$ is identified at the point as frame $(f+1)$. Point $p_i$ at frame $f$ and point $p_j$ at the next frame $f+1$ is considered as the same point if a similarity index is lower than a threshold. The similarity index is defined as follows based on the window matching around the point.

$$\sum_{neighbor} \left(I_f(p_i) - I_{f+1}(p_j)\right)^2 \tag{3.56}$$

   The traveling distance of each point is restricted inter neighboring frames because of small image changes. The next constraint is also implied.

$$||p_i - p_j|| \leq d_{threshold} \tag{3.57}$$

4. The interest points tracked from start to finish in the sequence are recorded and utilized in the factorization.

### 3.3.2 SIFT Operator

Recently, several detectors of interest points are proposed which are invariant with respect to scale, image resolution and wide view point changes [SM97] [MS01] [MS02] [MS04] [DSH04]. In addition there are many studies on the evaluations for these detectors [SMB98] [MS03]. Among them, we adopt SIFT key [Low99].

As mentioned above, SIFT key was proposed originally for object recognition. The features detected by SIFT key are invariant to image scaling ans rotation, and partially invariant to change in illumination and 3D camera viewpoint [Low04]. In our research, we utilize it for tracking.

For the detection of interest points from an image, SIFT key searches them in the 3D scale space. The scale space is a volumetric space of 2D images applied by various Gaussian smoothening. Given an image $I(x, y)$, the scale space $L(x, y, \sigma)$ is defined as the following convolution,

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \tag{3.58}$$

where $G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$ is a Gaussian.

For an efficient detection of interest points, a method is proposed [Low99], which searches the scale space peaks in the difference-of-Gaussian (DoF) function convoluted with the image.

$$\begin{aligned} D(x, y, \sigma) &= (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{3.59}$$

This means the difference of two nearby scales separated by a constant factor $k \ (k > 1)$.

The interesting point detection corresponds to the detection of all local maximums and minimums in $D(x, y, \sigma)$ as in Fig.3.6. We can not specify the number of interest points in SIFT key since the detector picks up all these peaks [1]. For the most efficient search, $k = \sqrt{2}$ is chosen.

For the accurate localizations of the interest points, they are estimated in subpixel level. At the accurate positions of peaks, the derivatives of $D(\mathbf{x}) = D(x, y, \sigma)$ take 0. By Taylor expansion,

$$D(\mathbf{x}) = D + \left(\frac{\partial D}{\partial \mathbf{x}}\right)^t x + \frac{1}{2}\mathbf{x}^t \frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x} \tag{3.60}$$

Therefore, the peak positions $\hat{\mathbf{x}}$ are calculated as

$$\hat{\mathbf{x}} = -\left(\frac{\partial^2 D}{\partial \mathbf{x}^2}\right)^{-1} \frac{\partial D}{\partial \mathbf{x}} \tag{3.61}$$

Here, $\sigma$ corresponds to the scale at the interest point, which makes this operator scale-invariant.

[1] It is possible to specify the total number of the interest points in the Harris operator according to the values of Eq.3.55, for example.

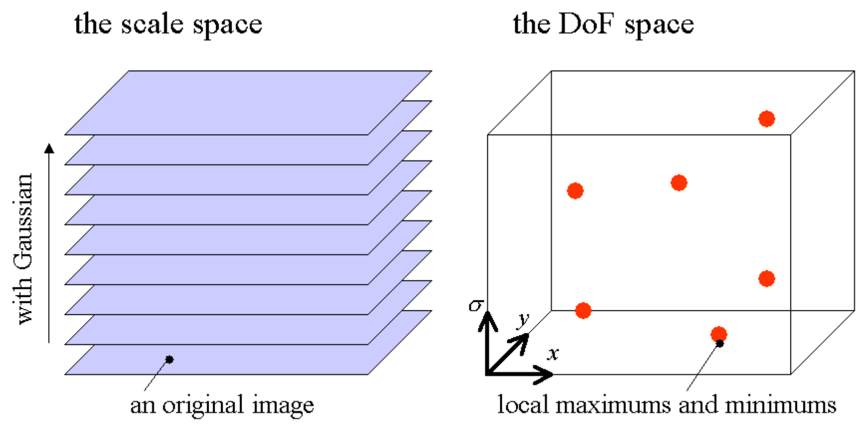


Figure 3.6: The scale scape and the different-of-Gaussian space.

Besides the localizations of interest points, SIFT key detect the orientations $\theta$ and the local image descriptors $\vec{k}$ around the points. The gradient magnitude $m$ and orientation $\theta$ at all pixels around each interest point is calculated by the local Gaussian smoothed image.

$$m = \sqrt{(L_{x+1,y} - L_{x-1,y})^2 + (L_{x,y+1} - L_{x,y-1})^2} \tag{3.62}$$

$$\theta = \tan^{-1} \frac{L_{x,y+1} - L_{x,y-1}}{L_{x+1,y} - L_{x-1,y}} \tag{3.63}$$

Based on the gradient orientations around an interest point, an orientation histogram is formed. The orientation histogram has 36 bins for the 360° range. Then the peak in the histogram corresponds to the dominant direction of the region, which makes a rotation-invariant detector.

For the local description, the region around each interest point is normalized in advance with respect to scale $\sigma$ and rotation $\theta$ . Then the gradient magnitude and orientation are compared at each interest point. In the left figure in Fig.3.7, the circle shows a Gaussian window. These samples are accumulated into orientation histograms summarizing the contents over large region with the length each arrow corresponding to the summation of the gradient magnitudes (the right figure in Fig.3.7). In practice, a $4 \times 4$ array of the sample regions with 8 orientation bins in each region is used. Therefore, each SIFT key $\vec{k}$ has $4 \times 4 \times 8 = 128$ dimension.

The advantage of SIFT key in the tracking process is that we can use the 128-dimensional vector $\vec{k}$ for the inter-frame identification. Consequently, our tracking algorithm with SIFT key is as follows.

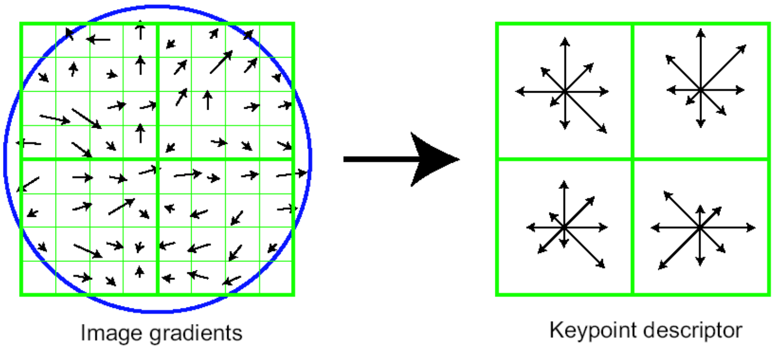


Figure 3.7: The local descriptor by SIFT key [Low04].

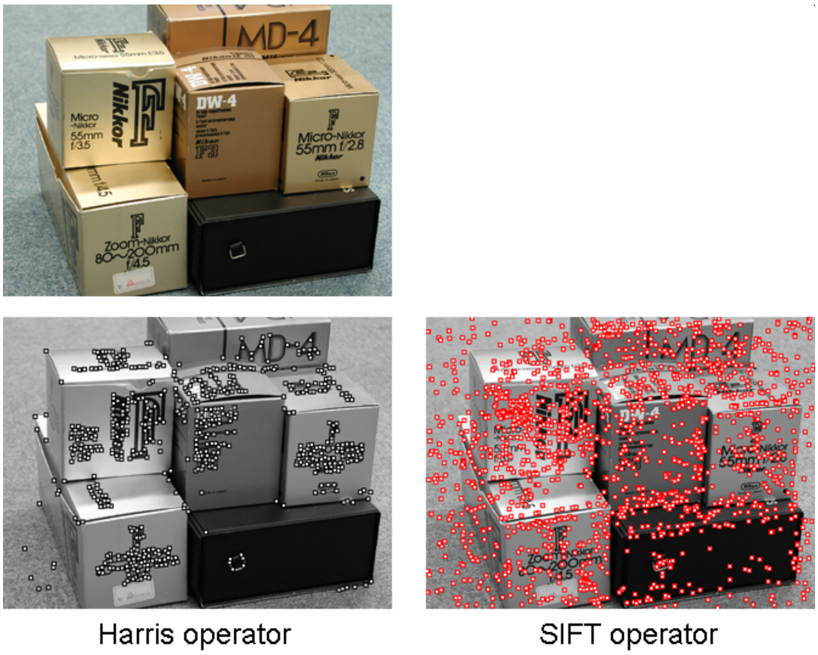


Figure 3.8: The results of the two detectors. In Harris detector, the total number of interest points is set at 500. And 1623 points are detected by SIFT operator.

1. Given a image sequence of F frames.
2. SIFT key is applied and detects interest points at each frame.
3. Each interest point at frame $f$ is identified at the point as frame $(f + 1)$. Just like in the Harris operator, Eq.3.56 and 3.57 are applied. In addition, the next

constraint is taken into account.

$$||\vec{k}_f(p_i) - \vec{k}_{f+1}(p_j)|| < threshold \tag{3.64}$$

4. The interest points tracked from start to finish in the sequence are recorded and utilized in the factorization.

Then, we show the results of the interest point detection by two operators. The top picture in Fig.3.8 is the original one and applied with the operators. The left bottom image shows the result by Harris operator. In this example, the strongest 500 interest points are detected according to the values of Eq.3.55. The right bottom image shows the result by SIFT key, which detects 1623 interest points. Many part of them are detected in large $\sigma$ space, where the locations of interest points are unstable. While SIFT key detects more points from an image than Harris operator, there is not a large number of the interest points which can be tracked in the whole sequence. Consequently, there are not large differences between the results by both operators.

## 3.4 Demonstration

In this section, we demonstrate our algorithm by using two kinds of sequences. As the first image sequence, we use an CG animation which means an ideal image sequences taken by an ideal camera. As the second example, we use a real image sequences taken by a digital camera in laboratory which shows that the method is applicable to real data.

### CG Sequence

We made an image sequence by 3ds max ® [Aut]. In this sequence, CG pictures of a textured box putted on the textured floor are taken by a virtual camera in a linear uniform motion. It consists of 72 frames, which is the same frame number as the data by the FLRS. Some examples of the sequence are shown in Fig.3.9.

Then, we extract the interest points by Harris operator, which are all observable on the entire frames from start to finish. Consequently, 136 interest points are extracted and an example picture is shown in Fig.3.10.

The history of the total residual errors defined as the next function is shown in

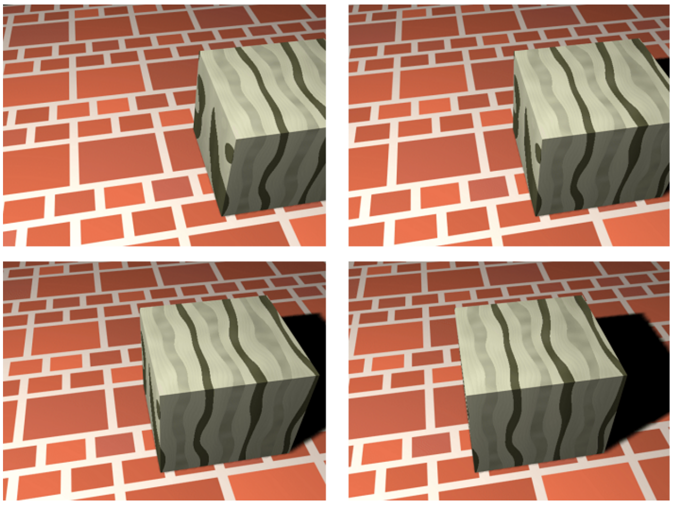


Figure 3.9: The image sequence of ”BOX”. (top left → top right → bottom left → bottom right)

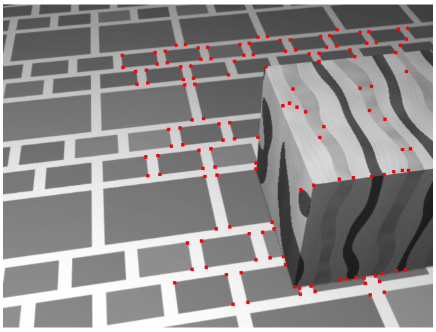


Figure 3.10: The interest points of the ”BOX” sequence.

Fig.3.11.

$$\sum_{f=1}^{F}\sum_{p=1}^{P}\left[\left((\lambda_{fp}+1)u_{fp}-\frac{f}{z_f}\vec{i}_f^{\,t}\cdot(\vec{S}_p-\vec{T}_f)\right)^2+\left((\lambda_{fp}+1)v_{fp}-\frac{f}{z_f}\vec{j}_f^{\,t}\cdot(\vec{S}_p-\vec{T}_f)\right)^2\right] \tag{3.65}$$

We can see that the total error is decreasing with iteration smoothly and converges within finite iterations adequately. The estimated shape by the full perspective factorization after 50 iterations is shown in Fig.3.12. While the floor is not completely flat nor the object is not a complete cube, it is considered that the result is practical for estimation of the shape. One can safely state that the full perspective factoriza-

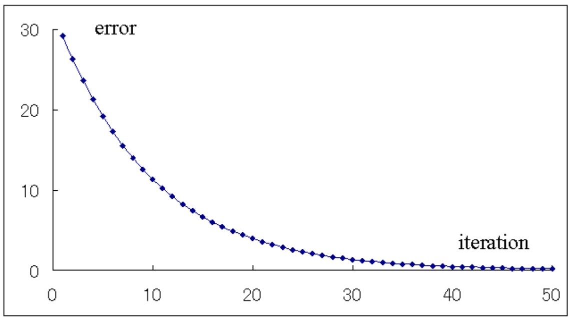


Figure 3.11: The histry of the error convergence in the ”BOX” sequence.

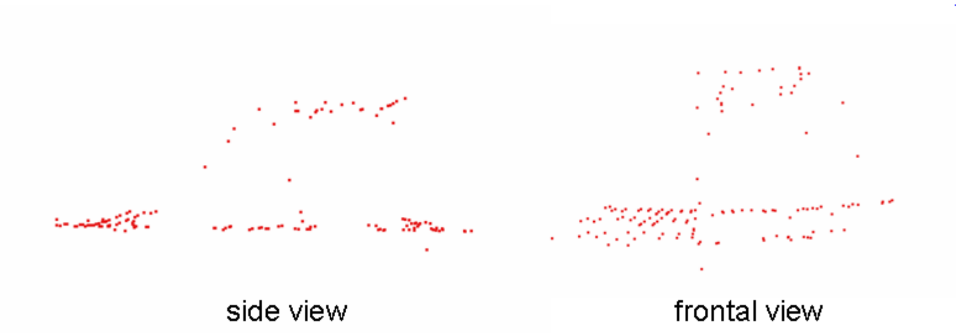


Figure 3.12: The estimated shape of the ”BOX” sequence by the full perspective factorization.

tion is effective for shape estimation only from images.

### Real Sequence

We apply the full perspective factorization to a real sequence. By using a commercial digital camera and a miniature house model, we verify practical effectiveness of our implementation.

Similarly, some examples of the sequence are shown in Fig.3.13. In this case, the motion of the hand-held camera consists of arbitrary translation and rotation.

The number of tracked interest point, 40, is rather small because the camera motion in this dataset is wide. An example picture of the interest points is shown in Fig.3.14.

The similar history of the error convergence is shown in Fig.3.15, and we can find that the error converges adequately.

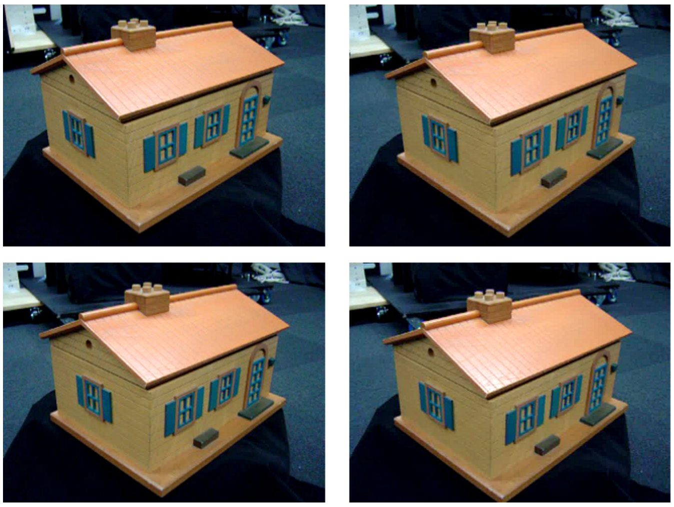


Figure 3.13: The real image sequence of ”Miniature House”.

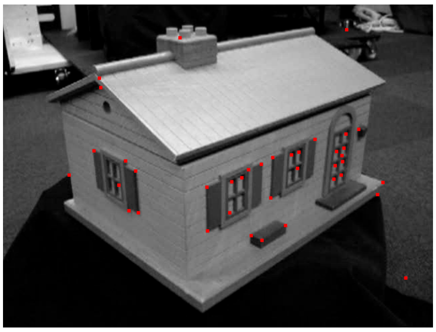


Figure 3.14: The interest points of the ”Miniature House” sequence.

The recovered shape after 100 iterations is shown in Fig.3.16.

Figure 3.16 shows the full perspective factorization comes in very useful also for practical operations.

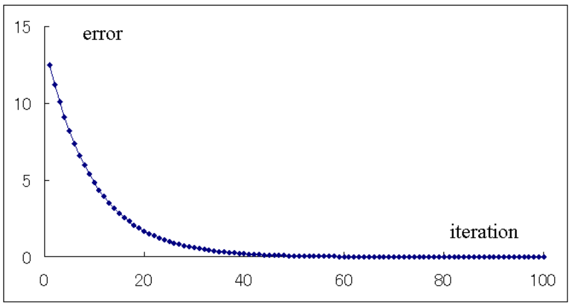


Figure 3.15: The histry of the error convergence in the ”Miniature House” sequence.

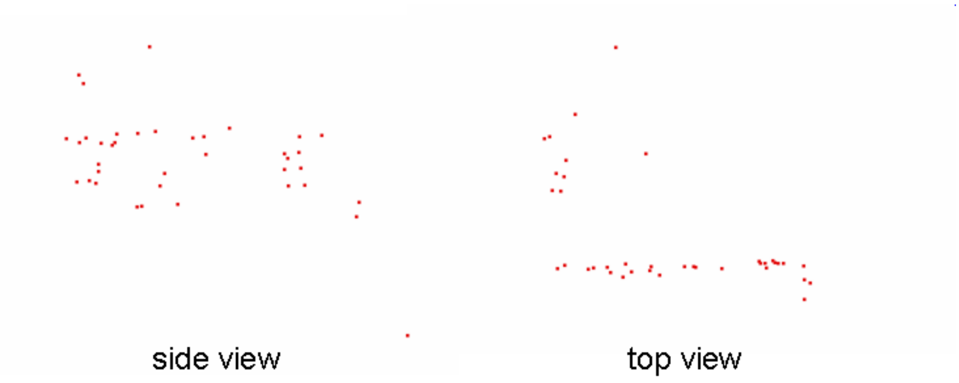


Figure 3.16: The estimated shape of the ”Miniature House” sequence by the full perspective factorization.

# Chapter 4

# Refinement

Without noise in the input, the factorization method leads to the excellent solution. As a result, the rectified 3D shape through the estimated camera parameters is valid. Real images, however, contain a bit of noise. Therefore, it is not sufficient to rectify range data obtained by the FLRS only through the factorization. For the sake of a more refined estimation of motion parameters, we impose three constraints: for tracking, movement, and range data. The refined camera motion can be found through the minimization of a global functional. To minimize the function, the solution by the full perspective factorization is utilized as the initial value to avoid local minimums.

## 4.1 Tracking Constraint

As the most fundamental constraint, any interest point $\vec{S_p}$ must be projected at the coordinates $(u_{fp}, v_{fp})$ on each image plane. This constraint is well known as Bundle Adjustment [Bro76]. When the structure, motion and shape have been roughly obtained in the meantime, this technique is utilized to refine them through the image sequence. In our case, the constraint conducts the following function:

$$\begin{aligned} F_A = \sum_{f=1}^{F}\sum_{p=1}^{P}\Bigg(&\Big(u_{fp} - f\frac{\vec{i_f}^{\,t}\cdot(\vec{S_p}-\vec{T_f})}{\vec{k_f}^{\,t}\cdot(\vec{S_p}-\vec{T_f})}\Big)^2 \\ &+\Big(v_{fp} - f\frac{\vec{j_f}^{\,t}\cdot(\vec{S_p}-\vec{T_f})}{\vec{k_f}^{\,t}\cdot(\vec{S_p}-\vec{T_f})}\Big)^2\Bigg) \end{aligned} \tag{4.1}$$

The minimization of $F_A$ leads to the correct tracking of fixed interest points by a moving camera. However, we can see that the presence of parameters we are trying to estimate in the denominator makes this equation a difficult one. We have to seek the optimal solution via some non-linear minimization techniques. Then, suppose that instead, we consider the following function:

$$F'_A = \sum_{f=1}^{F} \sum_{p=1}^{P} \Bigg( \Big( \vec{k_f}^t \cdot (\vec{S_p} - \vec{T_f}) u_{fp} - f \cdot \vec{i_f}^t \cdot (\vec{S_p} - \vec{T_f}) \Big)^2 + \Big( \vec{k_f}^t \cdot (\vec{S_p} - \vec{T_f}) v_{fp} - f \cdot \vec{j_f}^t \cdot (\vec{S_p} - \vec{T_f}) \Big)^2 \Bigg) \tag{4.2}$$

The term $\vec{k_f}^t \cdot (\vec{S_p} - \vec{T_f})$ means the depth, the distance between the optical center of camera $f$ and a plane, which is parallel to the image plane and include the point $\vec{S_p}$. The cost function $F_A$ is the summation of squared distances on the image plane while the cost function $F'_A$ is estimated on the plane of the point $\vec{S_p}$. It is true that we can only observe the image points on the image sequence, therefore the noise occurs on these images. However it is also true that the cost function $F_A$ does not assure that the reconstructed points are close to the correct ones in the real 3D world. In [BCS01], it has reported that these functions are likely to give good results.

Based on the above consideration, we choose to minimize the cost function $F'_A$ for the facility of the differential calculation.

## 4.2 Smoothness Constraint

One of the most significant reasons for adopting a balloon platform is to be free from the high frequency that occurs with a helicopter platform [HMK+04a]. A balloon platform is only under the influence of low frequency: the balloon of our FLRS is held with some wires swayed only by wind. This means that the movement of the balloon is expected to be smooth. Certainly, the movement of the balloon is free from rapid acceleration, rapid deceleration, or acute course changing. Taking this fact into account, we consider the following function:

$$F_B = \int \Big( w_1 \Big( \frac{\partial^2 \vec{T_f}}{\partial t^2} \Big)^2 + w_2 \Big( \frac{\partial^2 \mathbf{q}_f}{\partial t^2} \Big)^2 \Big) dt \tag{4.3}$$

Here, $\vec{T_f}$ denotes the position of the camera; $t$ is time; $w_1, w_2$ are weighted coefficients; and $\mathbf{q}_f$ is a unit quaternion (see Appendix B) that represents the camera pose. The bases $\vec{i_f}$, $\vec{j_f}$ and $\vec{k_f}$ are described by the quaternion immediately as follows:

$$\mathbf{q} = ((s, )u, v, w) \tag{4.4}$$

$$s^2 + u^2 + v^2 + v^2 = 1 \tag{4.5}$$

$$\vec{i_f} = \begin{pmatrix} s^2 + u^2 - v^2 - w^2 \\ 2(uv - sw) \\ 2(uw - sv) \end{pmatrix} \tag{4.6}$$

$$\vec{j_f} = \begin{pmatrix} 2(uv + sw) \\ s^2 - u^2 + v^2 - w^2 \\ 2(vw - su) \end{pmatrix} \tag{4.7}$$

$$\vec{k_f} = \begin{pmatrix} 2(uw - sv) \\ 2(vw + su) \\ s^2 - u^2 - v^2 + w^2 \end{pmatrix} \tag{4.8}$$

The first term of the above integrand represents smoothness with respect to the camera's translation while the second represents smoothness with respect to the camera's rotation. When the motion of the camera is smooth, the function $F_B$ becomes a small value.

For a quaternion, there are three independent variables which we have to estimate. The parameter $s$ is, for example, calculated by other 3 parameters as $\sqrt{1 - u^2 - v^2 - w^2}$. Therefore, we take account of only $u$, $v$ and $w$ with respect to $\mathbf{q}$.

We implement in practice the following discrete form:

$$F'_B = \sum_{f=1}^{F} \left( w_1 \left( \frac{\partial^2 \vec{T_f}}{\partial t^2} \right)^2 + w_2 \left( \frac{\partial^2 \mathbf{q}_f}{\partial t^2} \right)^2 \right) \tag{4.9}$$

As discrete approximation formulation for the 2nd-order partial derivatives with respect to time ($\Delta t = 1$), we use the next forms [Ban96].

$$\frac{\partial^2 F_t}{\partial t^2} = \begin{cases} 2F_t - 5F_{t+1} + 4F_{t+2} - F_{t+3} & (t = 0) \\ F_{t-1} - 2F_t + F_{t+1} & (0 < t < T - 1) \\ 2F_t - 5F_{t-1} + 4F_{t-2} - F_{t-3} & (t = T - 1) \end{cases} \tag{4.10}$$

## 4.3 Range Data Constraint

Taking a broad view of range data obtained by the FLRS, the data are distorted by the swing of the sensor. We can find, however, that these data contain instantaneous precise information locally; that information is utilized for refinement of the camera motion.

The FLRS re-radiates laser beams in raster scan order. This means that we can instantly obtain the time when each pixel in the range image is scanned because the camera and the range sensor are calibrated (Fig.4.1). If the video camera is synchronized with the range sensor, we can find the frame among the sequence when the pixel is scanned. With the video camera calibrated with the range sensor, we can also obtain the image coordinate of each interest point in the 3D world with respect to the instantaneous local coordinate.

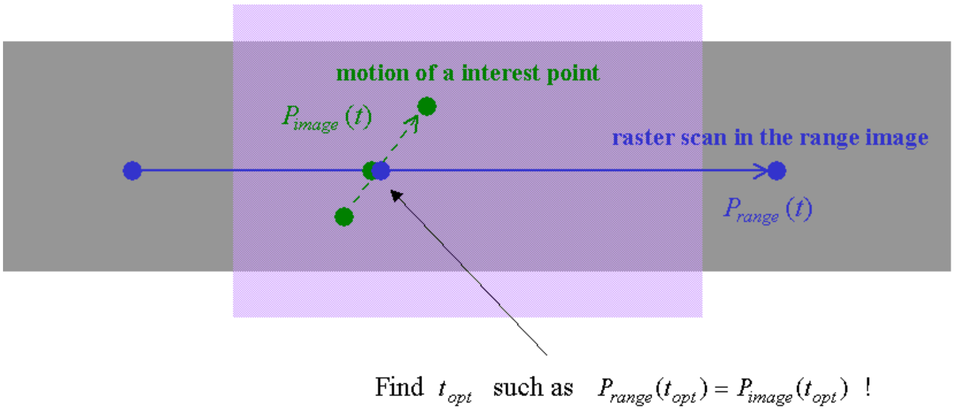

motion of a interest point
$P_{image}(t)$
raster scan in the range image
$P_{range}(t)$
Find $t_{opt}$ such as $P_{range}(t_{opt}) = P_{image}(t_{opt})$ !

Figure 4.1: Finding the time when a pixel in the picture is scanned by the range sensor.

Considering this constraint, we can compensate the camera motion.

At time $t$, suppose that the sensor position is $\vec{T}(t)$ and the 3 bases $\vec{i}_f$, $\vec{j}_f$, $\vec{k}_f$ are described as $\vec{i}(t)$, $\vec{j}(t)$, $\vec{k}(t)$. At this moment, suppose that the range sensor output $\vec{x}(t)$(in the local coordinate) as the measurement of the point $\vec{X}$, which is described in the world coordinate.

Based on Fig.4.2, the following equation is obtained.

$$\vec{X} = x\vec{i} + y\vec{j} + z\vec{k} + \vec{T} = \begin{pmatrix} \vec{i} & \vec{j} & \vec{k} \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \vec{T} = R\vec{x} + \vec{T} \tag{4.11}$$

Then, based on $R^t = R^{-1}$ (because $R^tR = (\vec{i}\,\vec{j}\,\vec{k})^t(\vec{i}\,\vec{j}\,\vec{k}) = E$), when the range sensor scans interest point $\vec{S}_p$, we can conduct the third constraint to be minimized as follows:

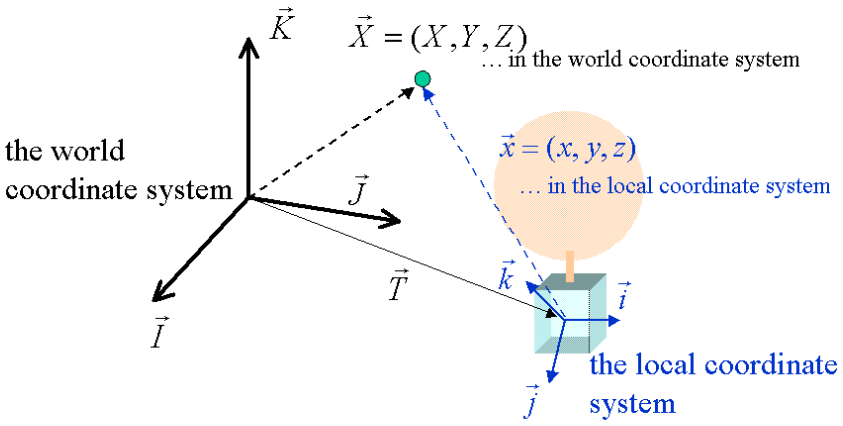

$\vec{K}$
$\vec{X} = (X, Y, Z)$
… in the world coordinate system
the world coordinate system
$\vec{J}$
$\vec{x} = (x, y, z)$
… in the local coordinate system
$\vec{T}$
$\vec{I}$
$\vec{k}$
$\vec{i}$
the local coordinate system
$\vec{j}$

Figure 4.2: The global position and its description in the local coordinates

$$F_C = \sum_{p=1}^{P} \left\| \mathbf{x}_{fp} - R^t(\vec{S_p} - \vec{T_{fp}}) \right\|^2 \tag{4.12}$$

Here, the index $fp$ denotes the frame number when the range sensor scans interest point $\vec{S_p}$. It is very significant to note that $\mathbf{x}_{fp}$ is the 3D coordinate values not described in the sensor-oriented coordinate system but in the camera-oriented one, which is rewritten based on the range data and camera-sensor calibration. In practice, we find sub-frame $fp$ by using a linear interpolating technique for the motion of interest points between frames. The main purpose of the above constraint is to adjust the absolute scale.

As $\mathbf{x}_{fp} = (x_{fp}, y_{fp}, z_{fp})$, the above function can be rewritten as the stronger constraint:

$$\begin{aligned} F'_C = \sum_{p=1}^{P} \Big( & (x_{fp} - \vec{i_{fp}}^{\,t} \cdot (\vec{S_p} - \vec{T_{fp}}))^2 \\ & + (y_{fp} - \vec{j_{fp}}^{\,t} \cdot (\vec{S_p} - \vec{T_{fp}}))^2 + (z_{fp} - \vec{k_{fp}}^{\,t} \cdot (\vec{S_p} - \vec{T_{fp}}))^2 \Big) \end{aligned} \tag{4.13}$$

## 4.4 The Global Cost Function

Based on the above considerations, it will be found that the next cost function should be minimized. Consequently, the weighted sum

$$F = w_A F'_A + w_B F'_B + w_C F'_C \tag{4.14}$$

leads to a global function. The coefficients $w_A$, $w_B$ and $w_C$ are determined experimentally and we are going to discuss them later.

To minimize this function, we employ Fletcher-Reeves method or Polak-Ribiere method [Pol71] [Jac77] [SR80], which are types of the conjugate gradient method (in the next section, we explain the conjugate gradient method briefly). Then, we use the golden section search to determine the magnitude of gradient directions. For optimization, Levenberg-Marquardt method [Mar63] is generally employed to minimize a functional value. Levenberg-Marquardt method is very effective to estimate function's parameters, especially to fit a certain function. However in our function, it is not a parameter fitting problem to minimize the value of $F'_B$. What we only have to do is to decrease $F'_B$ simply. Therefore we adopt the conjugate gradient method.

As mentioned in the previous parts, we input the solution by the perspective factorization as the initial value. Minimizing the function $F$ is basically quite difficult because this function has many local minimums. By employing the solution of the factorization as a fairly good approximation, we try to avoid them.

## 4.5 Optimization

There are many methods to minimize a multi-dimensional function value. Their strategies are, nevertheless, almost the same. First, an initial approximate solution is located in the space, which is expected to be close to the correct answer(global minimum). Then the approximate solution moves to search the global minimum iteratively. The method to decide the directions for the search differs according to the method.

The simplest method is a steepest descent method. To minimize a function $f(\vec{x})$ for example, it searches the next approximate solution along to the direction of $\nabla f(\vec{x})$. It is certainly effective to use the steepest descent method for minimization of a simple function. In the case of a complex function, it is not always effective to search along the differential direction.

Generally it is Newton method that can determine the search direction effectively. To determine the direction, however, the method needs inversion of huge Hessian matrices.

Then in this thesis, we apply a conjugate gradient method, which need not inverse a huge matrix.

### 4.5.1 Conjugate Gradient Method

As mentioned above, the steepest descent directions are not always the most suitable directions. In a conjugate gradient method, the conjugate direction direction for the previous search direction is applied. There are two familiar methods in this category, Fletcher-Reeves method and Polak-Ribiere method.

First, we define $A$ as a $n\times n$ positive definite symmetric matrix, $\vec{g_0}$ as an arbitrary vector and $\vec{h_0} = \vec{g_0}$. Then two types of gradient vectors are defined as follows:

$$\vec{g_{i+1}} = \vec{g_i} - \lambda_i A \cdot \vec{h_i} \qquad \vec{h_{i+1}} = \vec{g_{i+1}} + \gamma_i \vec{h_i} \tag{4.15}$$

Two vectors $\vec{g_i}$ and $\vec{h_i}$ satisfy $\vec{g_{i+1}}^t \cdot \vec{g_i}$ and $\vec{h_{i+1}}^t \cdot A \cdot \vec{h_i}$. That means

$$\lambda_i = \frac{\vec{g_i}^t \cdot \vec{g_i}}{\vec{g_i}^t \cdot A \cdot \vec{h_i}} \qquad \gamma_i = -\frac{\vec{g_{i+1}}^t \cdot A \cdot \vec{h_i}}{\vec{h_i}^t \cdot A \cdot \vec{h_i}} \tag{4.16}$$

Consequently, the next equations are introduced in the case of $i \neq j$.

$$\vec{g_i}^t \cdot \vec{g_j} = 0 \qquad \vec{h_i}^t \cdot A \cdot \vec{h_j} = 0 \tag{4.17}$$

Above equations mean that $\vec{g_i}$ is orthogonal to $\vec{g_j}$ and that $\vec{h_i}$ is conjugate to $\vec{h_j}$.

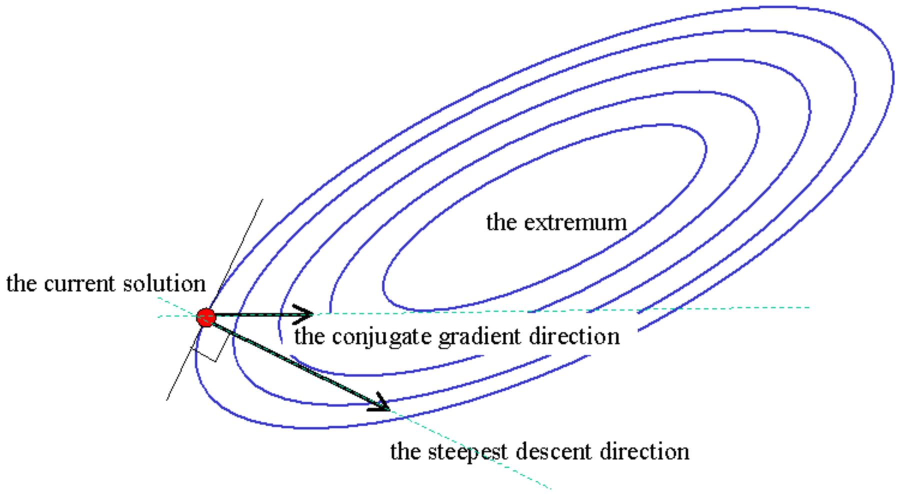

the extremum
the current solution
the conjugate gradient direction
the steepest descent direction

Figure 4.3: The search directions of the steepest descent and the conjugate gradient method.

From Eq.4.15 and 4.17, we obtain the followings.

$$\gamma_i = \frac{\vec{g_{i+1}}^t \cdot \vec{g_{i+1}}}{\vec{g_i}^t \cdot \vec{g_i}} = \frac{(\vec{g_{i+1}} - \vec{g_i})^t \cdot \vec{g_{i+1}}}{\vec{g_i}^t \cdot \vec{g_i}} \tag{4.18}$$

$$\lambda_i = \frac{\vec{g_i}^t \cdot \vec{h_i}}{\vec{h_i}^t \cdot A \cdot \vec{h_i}} \tag{4.19}$$

Here, $\vec{g_i}$ is interrupted as the steepest descent direction at $i - th$ step. Suppose that the equation $\vec{g_i} = -\nabla f(\mathbf{P_i})$ is satisfied at point $\mathbf{P_i}$. Then searching along $\vec{h_i}$, point $\mathbf{P_{i+1}}$ is to be found where the function $f(\mathbf{P_{i+1}})$ takes the minimal value. A theorem shows that vector $-\nabla f(\mathbf{P_{i+1}})$ coincides vector $\vec{g_{i+1}}$ of Eq.4.15. Moreover, by using Eq.4.15 and 4.18, we can find vector $\vec{h_i}$ without calculating matrix $A$ (matrix $A$ corresponds to the Hessian). The conjugate gradient method is summarized as follows.

***Input:*** A cost function $f(\vec{x})$ and the initial approximate solution $\vec{x_0}$.

***Output:*** The extremum $\hat{\mathbf{x}}$.

1. Given the initial approximate solution $\vec{x_0}$
2. Calculate the deviation of $f$ at current point $\vec{x_0}$.
3. Set $\vec{g_0} = \vec{h_0} = -\nabla f(\vec{x_0})$.
4. Search for the minimal point, $\vec{x_1}$ along $\vec{h_0}$.

   (replace 1 → $i$)
5. Calculate $\vec{g_i} = -\nabla f(\vec{x_i})$ $(i = 1, 2, \cdots)$. Then also calculate $\vec{h_i}$ by using Eq.4.15 and 4.18.
6. Search for the next minimal point, $\vec{x_{i+1}}$ along $\vec{h_i}$.
7. Return to the step 5.

***Until:*** $\vec{x_i}$ are close to the previous step.

The point is that the essential techniques in this method are the calculations of $\vec{h_i}$ and the minimization along the search line. For the line minimization we explain in the next. For your information, Fletcher-Reeves method adopts the first equation of Eq.4.18 as the definition of $\gamma$ and Polak-Ribiere method adopts the second one.

### 4.5.2 Golden Section Search

We adopt Golden section search method as the line minimization technique, which searches the minimal point along a line effectively and does not need any deviations. Golden section search is based on an enclosure method.

Here, the underlying problem is to find $\hat{x}$ which satisfies $\hat{x} = \arg\min_x f(x)$. The strategy of Golden section search is as follows. First, to enclose the minimal point, we suppose three points $a$, $b$ and $c$ ($a < b < c$). If $f(b)$ is smaller than both $f(a)$ and $f(c)$, the minimal point $\hat{x}$ does exist between $(a, c)$. Then, we narrow down the search range $(a, c)$ iteratively to find out the minimal point $\hat{x}$.

When $f(b)$ is smaller than both $f(a)$ and $f(c)$, we consider another point $x$ between $(a, b)$ or $(b, c)$. For example, let us consider the case of $a < x < b$. If $f(x) > f(b)$, the minimal point should exist between $(x, c)$. Then, $x$ is relabeled as $a$ in the next step and then the minimal point is to exist between $(a, c)$. If $f(x) < f(b)$, the point is between $(a, b)$. For the next step, point $x$ and $b$ are relabeled as $b$ and $c$ respectively. The search region becomes narrower in this way. Repeating this procedure, the search region is getting narrower and we can find out the minimal point, $\hat{x}$ numerically.

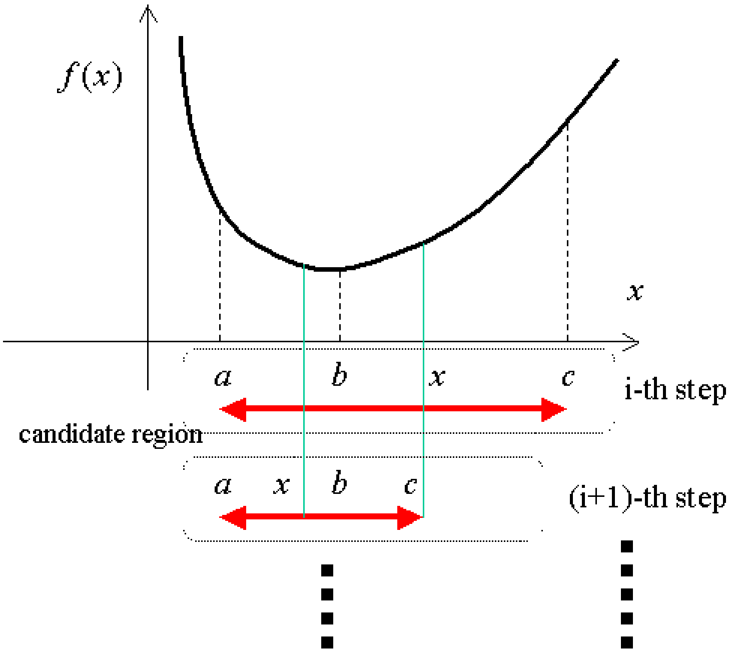


Figure 4.4: The Golden section search for line minimization.

Then, where shall we set the point $x$ for an effective search? In Golden section search method, the most effective position of $x$ is rigidly determined [PFTV88]. First, given the initial range of $(a, c)$ (Fig.4.4), the first position of $b$ is locates so as to

$$\overline{ab} : \overline{bc} = 0.38197 : 0.61803$$

Then, $x$ is to be set in the wider region $(b, c)$ so as to $\overline{bx} : \overline{xc} = 0.38197 : 0.61803$. Comparing the values of $f(x)$ and $f(b)$, the next step's search region is determined as mentioned above. The next point $x$ is located so that its fraction is 0.38197 into the larger of the two intervals $\overline{ab}$ and $\overline{bc}$. The ratio of $0.38197 : 0.61803$ is called the golden section ratio.

## 4.6 Shape Rectification

After the refinement, we possess the vector $\vec{T_f}$ and three bases $\vec{i_f}$, $\vec{j_f}$ and $\vec{k_f}$ at each frame. That means we know the position and pose of the camera at discrete time. To rectify the deformed shape data by using these extrinsic parameters quantized with respect to time, these parameters have to be interpolated. To be more precise, we have to interpolate three components with respect to translation $\vec{T_f} = (T_{xf}, T_{yf}, T_{zf})$, and three components with respect to rotation $\mathbf{q}_f = \left((s_f,)\ u_f, v_f, w_f\right)$. Each parameter's variation with respect to time is, therefore, approximated by a polynomials. In this study, we adopt 7-order polynomials.

A range sensor outputs the temporal coordinate values $\vec{x}(t) = (x(t), y(t), z(t))$ in the temporal sensor-oriented coordinate system. That means, suppose the range sensor with position $\vec{T}(t)$ and three bases $\vec{i}(t)$, $\vec{j}(t)$ and $\vec{k}(t)$ outputs $\vec{x_i}$ when a point $\vec{X} = (X, Y, Z)$ in the world coordinate system is scanned.

Therefore, the next equation should be satisfied.

$$\vec{X} = x(t) \cdot \vec{i}(t) + y(t) \cdot \vec{j}(t) + z(t) \cdot \vec{k}(t) + \vec{T}(t) \tag{4.20}$$

Consequently, defining the matrix $R(t) = \left(\vec{i}(t)\ \vec{j}(t)\ \vec{k}(t)\right)$ as the rotation matrix, we can rectify the deformed range data as;

$$\vec{X} = R(t)\vec{x}(t) + \vec{T}(t) \tag{4.21}$$

Combining the initial estimation for camera parameters by the full perspective factorization (Chapter 3) and the refinement method mentioned in this section, we can estimate the more accurate motion parameters. Then, for parameter estimation of a moving sensor, we utilize not only image sequences but also distorted range data.

In this method, we use a calibrated camera-sensor system as a precondition. Then a robust method for the calibration is described in the next section. Moreover, we show that this method is applicable for uncalibrated system too.

# Chapter 5

# Calibration and Reconstruction

The method described in the previous chapters is based on a calibrated system, in which the relative positions are known between the range sensor-oriented coordinate system and the camera-oriented one. In the first half of this chapter, we describe how to calibrate two coordinate systems. In the second half, we apply our method to an uncalibrated system, in which the configuration between the two systems is unknown. We use ”Shape from Motion” techniques to calibrate them *a posteriori*.

## 5.1 Calibration

Calibration is to obtain camera parameters. There are two kinds of camera parameters, intrinsic and extrinsic. The intrinsic parameters are proper to each camera and the extrinsic parameters are in reference to position and pose of a camera. In the FLRS system, we assume the intrinsic camera parameters are known in advance (weak calibrated camera). On the hand, the extrinsic camera parameters are unknown. Moreover, on the FLRS, the sensor-oriented coordinate system differs from the camera-oriented one. Therefore, we have to estimate the relative orientation between the range sensor and the monitoring camera on the FLRS. For the acquisition of the relative orientation, we use calibration techniques. Given a known 3D geometry model by the range sensor and some 2D images by the monitoring camera with known intrinsic parameters, we have to estimate the extrinsic parameters. That means, calibration corresponds to 2D-3D registration.

There are many techniques for camera calibration by using 3D reference objects [Tsa86], 2D reference planes [SM99] [Zha99], [Zha00] and 1D lines [Zha04]. Most of the techniques using 3D reference objects estimate the lens distortions simultaneously [Tsa87] [WCH92]. In [UT03], a method for simultaneous calibration of multi cameras is proposed. By using more simple object(circles [WZHW04], spheres [Agr03]), several methods are proposed to estimate only intrinsic parameters. On the other hand, many calibration methods without any reference objects, called ”Self calibration”, have been published recently [Hr96] [LF97] [PKG99] [PG99].

Calibration algorithms require some kinds of information about the correspondences between 2D features on images and 3D features in space (in most cases, the features mean points). In order to find 2D-3D correspondences, in some cases, calibration boxes or checker boards which has prominent markers are utilized for calibration. In some methods, the correspondences are specified manually by users. These methods work, but they are labor intensive. On the other hand, many researchers are tying 2D-3D registration automation. In [Ohk03], they aligned 2D images and a 3D model on the optimization framework, in which conventional edges in a 2D image were aligned to edges in the rendered image by using the 3D model.

There are many studies, textbooks and reviews on camera calibration [Hor86] [Fau93] [Dav97] [Pol02] [UOS05], because it is one of the most difficult and important problems. The main reason of the difficulty is that the accuracy of parameter estimation is very sensitive to noises. With severe errors and noises, incoherent parameters are estimated. To overcome this difficulty, we adopt a robust estimation of the intrinsic parameters that rejects the incoherent parameters.

Calibration, the process of estimating the intrinsic and extrinsic parameters of a camera, is divided into 2 steps.

1. Estimate the 3 × 4 projection matrix, which describes the direct mapping of a 3D point onto the 2D image.

2. Divide the projection matrix into the intrinsic and extrinsic matrices.

We will explain the process and our robust estimation of the intrinsic parameters.

As mentioned in 3.1, a 3D point at $(x, y, z)$ described in the camera coordinate system is projected on to a 2D point at $(u, v)$ according to Eq.5.1.

$$\begin{cases} u = f\dfrac{x}{z} \\ v = f\dfrac{y}{z} \end{cases} \tag{5.1}$$

Here, defining a real number $\kappa \equiv z$, we can describe Eq.5.1 in a matrix form.

$$\kappa\tilde{\mathbf{m}} = \kappa \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & & \\ & f & \\ & & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} \tag{5.2}$$

Suppose that the camera is located at $\vec{T}$ and has the bases $\vec{i}$, $\vec{j}$ and $\vec{k}$ in the world coordinate system. A 3D point $\vec{X}$ described in the world coordinate system is described as $\vec{x} = (x, y, z)$ in the camera coordinate system as follows:

$$\begin{cases} x = \vec{i}^t \cdot (\vec{X} - \vec{T}) \\ y = \vec{j}^t \cdot (\vec{X} - \vec{T}) \\ z = \vec{k}^t \cdot (\vec{X} - \vec{T}) \end{cases} \tag{5.3}$$

Here, $x$ is, for example, rewritten as follows:

$$x = \vec{i}^t \cdot (\vec{X} - \vec{T}) = \vec{i}^t \cdot \vec{X} - \vec{i}^t \cdot \vec{T} = \begin{pmatrix} \vec{i}^t, & -\vec{i}^t \cdot \vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} \tag{5.4}$$

Therefore, the system (Eq. 5.3) is described in a matrix form as,

$$\vec{x} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} \vec{i}^t & -\vec{i}^t \cdot \vec{T} \\ \vec{j}^t & -\vec{j}^t \cdot \vec{T} \\ \vec{k}^t & -\vec{k}^t \cdot \vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} = \begin{pmatrix} R^t & -R^t\vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} \tag{5.5}$$

Substituting Eq.5.5 into Eq.5.2,

$$\kappa\tilde{\mathbf{m}} = \kappa \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} f & & \\ & f & \\ & & 1 \end{pmatrix} \begin{pmatrix} R^t & -R^t\vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} \tag{5.6}$$

According to Eq.5.6, 3D point $(X, Y, Z)$ described in the world coordinate system is mapped onto point $(u, v)$ in the image.

The coordinate values $u$ and $v$ are not described in the general coordinate system utilized by many studies. In our method, the origin of the image coordinate system $(u, v) = (0, 0)$ is located on the center of the image. Moreover, we have assumed that the image on the image plane is the same picture that we can obtain like as a photo. Generally, the origin of an image coordinate system is located on the left top corner of the image, and we have to take into account a deformation associated with the mapping from the image plane to the actual photo. That means, an image $(u, v)$ projected by Eq.5.1 is fairly ideal without any distortions. Therefore we have to consider the mapping from the ideal image $(u, v)$ to the actual photo image $(u_a, v_a)$ with the left top origin.

The coordinate system of the the ideal image is centered at $c$, the intersection of the optical axis and the image plane. Point $c$ is mapped to the point at $(u_0, v_0)$ in the new coordinate system; point $c$ is called the *principal point*. Then we set one basis $\vec{i_a}$ of the actual image coordinate parallel to the basis $\vec{i}$ of the camera coordinate system. For the angle $\theta$ between $\vec{i_a}$ and another basis $\vec{j_a}$ (ideally, $\theta = \frac{\pi}{4}$), the mapping is obeyed in the next equation.

$$\begin{pmatrix} u_a \\ v_a \\ 1 \end{pmatrix} = \begin{pmatrix} k_u & -k_u \cot\theta & u_0 \\ 0 & \dfrac{k_v}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} \tag{5.7}$$

where, $k_u$ and $k_v$ are scale factors with respect to $\vec{i}$ and $\vec{j}$, respectively. For simplicity, we set $k_u = 1$.

Consequently, the relationship between 3D point $\vec{X}$ in the world coordinate system and the corresponding 2D point $(u_a, v_a)$ in the observed image is described as follows:

$$\begin{aligned} \kappa\tilde{\mathbf{m}}_a = \kappa \begin{pmatrix} u_a \\ v_a \\ 1 \end{pmatrix} &= \begin{pmatrix} 1 & -\cot\theta & u_0 \\ 0 & \dfrac{k_v}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} f & & \\ & f & \\ & & 1 \end{pmatrix} \begin{pmatrix} R^t & -R^t\vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} f & -f\cot\theta & u_0 \\ 0 & f\dfrac{k_v}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} R^t & -R^t\vec{T} \end{pmatrix} \begin{pmatrix} \vec{X} \\ 1 \end{pmatrix} \end{aligned} \tag{5.8}$$

The 3×4 matrix in the middle of Eq.5.8, $(R^t \quad -R^t\vec{T})$ consist of camera position $\vec{T}$ and pose $R$, and it is thus called the *extrinsic matrix*.

The first matrix in Eq.5.8 is rewritten:

$$A = \begin{pmatrix} f & -f\cot\theta & u_0 \\ 0 & f\dfrac{k_v}{\sin\theta} & v_0 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} f & s & u_0 \\ 0 & \alpha f & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{5.9}$$

where $s$ is the *skew* and $\alpha$ is the *aspect ratio*. The five parameters ( the focal length $f$, the principal point $(u_0, v_0)$, the skew $s$ and the aspect ratio $\alpha$ ) do not depend on the position and orientation of the camera in space and differ from camera to camera, or from lens to lens. They are, therefore, called the *intrinsic parameters* and matrix $A$ is called the *intrinsic matrix*. In fact, the skew $s$ and the aspect ratio $\alpha$ have roots in the manufacturing accuracy of image pickup devices. Generally, the skew can be ignored and the aspect ratio $\alpha$ is almost 1.0 in the modern digital camera. And often the principal point is also presumed on the center of the images.

In practice, besides the skew and the aspect ration, lens distortions also affect the observed image deformation. They primarily consist of radial distortion and tangential distortion, which are especially notable when the wide-angle lenses or small handy cameras are used. Lens distortion can be estimated by various methods [Bro66] [SN00] [STEY05]. We adopt the method in [Zha99] [Zha00] as described later.

Equation 5.8 is rewritten as follows:

$$\kappa\tilde{\mathbf{m}}_a = \kappa\begin{pmatrix} u_a \\ v_a \\ 1 \end{pmatrix} = A\left(R^t \quad -R^t\vec{T}\right)\tilde{\mathbf{W}} = \mathbf{P}\tilde{\mathbf{W}} = \mathbf{P}\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \mathbf{P}\tilde{W} \tag{5.10}$$

The $3 \times 4$ matrix $\mathbf{P} = A\left(R^t \quad -R^t\vec{T}\right)$ is called the *projective matrix*, which directly connects the 3D point in space and the corresponding 2D point in the image. To estimate all extrinsic and intrinsic parameters, the first step is to estimate the components of **P**. Then, decomposing **P**, we can obtain the intrinsic matrix $A$ and the extrinsic parameters $R$ and $\vec{T}$.

### 5.1.1 Solving for the Projective Matrix

In Eq.5.8, the number of unknown parameters seem to be $12 + 1$. These are 12 elements of the $3 \times 4$ matrix **P** and $\kappa$. We can't determine the value of $\kappa$, which is called the *projective depth* and differs from point to point. This means we can't determine the scale of an object only by watching its image. Therefore, we use the following equation derived from Eq.5.8.

$$\begin{aligned} \tilde{\mathbf{m}}_a &\propto \mathbf{P}\tilde{W} \\ \therefore\ \tilde{\mathbf{m}}_a &\times \mathbf{P}\tilde{W} = \vec{0} \end{aligned} \tag{5.11}$$

Moreover, there is ambiguity in the scale of **P**'s components. We can't determine the absolute value of the components. The number of unknown parameter is, therefore, only 11. It corresponds to the 5 intrinsic parameters and 6 extrinsic parameters (3 in rotation and 3 in translation).

If we know the coordinate value of 3D point $(X, Y, Z)$ and its corresponding 2D point $(u_a, v_a)$, we can derive two equations from Eq.5.11. For example, suppose

$$\mathbf{P} = \begin{pmatrix} P_1 & P_2 & P_3 & P_4 \\ P_5 & P_6 & P_7 & P_8 \\ P_9 & P_{10} & P_{11} & P_{12} \end{pmatrix} \tag{5.12}$$

then we obtain two equations as follows:

$$\begin{cases} XP_1 + YP_2 + ZP_3 + P_4 - u_aXP_9 - u_aYP_{10} - u_aZP_{11} - u_aP_{12} & = & 0 \\ XP_5 + YP_6 + ZP_7 + P_8 - v_aXP_9 - v_aYP_{10} - v_aZP_{11} - v_aP_{12} & = & 0 \end{cases} \tag{5.13}$$

If we get more than six pairs of $(X, Y, Z)$-$(u_a, v_a)$ correspondences, we can solve for the vector $\vec{P} = (P_1, P_2, \cdots, P_{12})$ of unknown 12 parameters as a linear system problem.

As mentioned above, we can't determine the absolute length of $\vec{P}$ because of the ambiguity in scale. In our study, we fix $|\vec{P}| = 1$.

### 5.1.2 Solving for the Intrinsic Matrix

Let us consider the leftmost 3×3 part of the matrix **P**. From Eq.5.10, we can obtain the next.

$$\mathbf{P}' = \begin{pmatrix} P_1 & P_2 & P_3 \\ P_5 & P_6 & P_7 \\ P_9 & P_{10} & P_{11} \end{pmatrix} = AR^t \tag{5.14}$$

Here is a significant property with respect to the rotation matrix $R$.

$$R^tR = \begin{pmatrix} \vec{i}^t \\ \vec{j}^t \\ \vec{k}^t \end{pmatrix} (\vec{i} \ \ \vec{j} \ \ \vec{k}) = \begin{pmatrix} 1 & & \\ & 1 & \\ & & 1 \end{pmatrix} = E \tag{5.15}$$

Therefore,

$$\mathbf{P}'(\mathbf{P}')^t = (AR^t)(AR^t)^t = AR^tRA^t = AA^t \tag{5.16}$$

The next task is the decomposition of $\mathbf{P}'(\mathbf{P}')^t$ into the upper triangle matrix $A$. Since $\mathbf{P}'(\mathbf{P}')^t$ is a symmetric matrix, the above decomposition can be attained as follows:

$$if \quad K = \begin{pmatrix} k_1 & k_2 & k_3 \\ k_2 & k_4 & k_5 \\ k_3 & k_5 & 1 \end{pmatrix} = AA^A$$

$$\textit{then} \quad A = \begin{pmatrix} \sqrt{k_1 - {k_3}^2 - \dfrac{(k_2 - k_3 k_5)^2}{k_4 - {k_5}^2}} & \dfrac{k_2 - k_3 k_5}{\sqrt{k_4 - {k_5}^2}} & k_3 \\ & \sqrt{k_4 - {k_5}^2} & k_5 \\ & & 1 \end{pmatrix} \tag{5.17}$$

Consequently, we can estimate all intrinsic parameters.

### 5.1.3 Solving for the extrinsic parameters

Once having estimated the projective matrix **P** and the intrinsic matrix $A$, it is simple to solve for the extrinsic parameters.

$$R = (\mathbf{P}')^t A^{-t} \tag{5.18}$$

$$\vec{T} = -RA^{-1}\vec{p'} \tag{5.19}$$

where $\vec{p'} = (P_4, P_8, P_{12})^t$.

Thus, given more than 6 pairs of 3D-2D corresponding points, we can estimate 5 intrinsic and 6 extrinsic parameters with scale ambiguity. The method mentioned above is called a linear solver since it consists of only linear calculations.

In practice, we apply a non-linear solver after solving by above linear solver. Suppose N ($N \geq 6$) pairs of 3D-2D correspondences $(X_i, Y_i, Z_i) - (u_i, v_i)$, we have to minimize the following cost function:

$$\begin{aligned} F &= \sum_i^N \left[ \left( u_i - u_a(R, \vec{T}, X_i, Y_i, Z_i) \right)^2 + \left( v_i - v_a(R, \vec{T}, X_i, Y_i, Z_i) \right)^2 \right] \\ &= \sum_i^N \left[ \left( u_i - \frac{f x(R, \vec{T}, X_i, Y_i, Z_i) + s y(R, \vec{T}, X_i, Y_i, Z_i) + u_0 z(R, \vec{T}, X_i, Y_i, Z_i)}{z(R, \vec{T}, X_i, Y_i, Z_i)} \right)^2 \right. \\ &\qquad \left. + \left( v_i - \frac{\alpha f y(R, \vec{T}, X_i, Y_i, Z_i) + v_0 z(R, \vec{T}, X_i, Y_i, Z_i)}{z(R, \vec{T}, X_i, Y_i, Z_i)} \right)^2 \right] \end{aligned} \tag{5.20}$$

where the functions $x(R, \vec{T}, X_i, Y_i, Z_i)$ etc. have been defined in Eq.5.3.

Therefore, refined parameters by the non-linear solver are estimated as

$$\{f, s, \alpha, u_0, v_0, R, \vec{T}\} = \arg \min_{f, s, \alpha, u_0, v_0, R, \vec{T}} F \tag{5.21}$$

We adopt Levenberg-Marquardt method for the minimization $F$.

Here, the procedure of solving for camera parameters is summarized as follow.

***Input:*** $N(\geq 6)$ pairs of 3D-2D correspondences extracted manually.

***Output:*** The intrinsic and extrinsic camera parameters.

1. Solve for the projective matrix **P** by using N pairs of Eq.5.13.
2. By using the left $3 \times 3$ part of **P**, solving for the intrinsic matrix $A$.
3. Solving for the rotation matrix $R$ and the camera position $\vec{T}$ by Eq.5.18 and 5.19, respectively.
4. Input these parameters into Eq.5.20 and refine them through the non-linear minimization of the cost function.

Before we come to Levenberg-Marquardt method, let us return to the lens distortions. In our study, we consider only on the radial distortion, which is modeled as

$$\begin{cases} u'_a = (u_a - u_0)\left(1 + k_1 r^2 + k_2 r^4\right) + u_0 \\ v'_a = (v_a - v_0)\left(1 + k_1 r^2 + k_2 r^4\right) + v_0 \end{cases} \tag{5.22}$$
$$where \quad r^2 = (u_a - u_0)^2 + (v_a - v_0)^2$$

The parameter $k_1$ and $k_2$ represent the lens distortions. In order to remove the distortions, the cost function $F$ which includes the parameters $k_1$ and $k_2$ based on Eq.5.21 should be minimized.

### 5.1.4 Levenberg-Marquardt Method for Optimization

Levenberg-Marquardt [Mar63] method is a general non-linear optimization algorithm for parameter fitting when the form and derivatives of the objective function are known. It mixes a gradient descent and Newton method dynamically in each iteration. In this subsection, we explain our implementation of the method briefly.

Let us consider the following situation, where given some observed values $\hat{x}$, we want minimize a function $F(\vec{p}|\hat{x})$ with respect to unknown parameters $\vec{p}$. In other words, we want to estimate the optimal parameters $\vec{p}_{op}$.

$$\vec{p}_{op} = \arg\min_{\vec{p}} F(\vec{p}|\vec{x}) \tag{5.23}$$

In the gradient descent method, a new candidate of the solution $\vec{p}_{t+1}$ is estimated by using the current solution $\vec{p}_t$ as follows:

$$\vec{p}_{t+1} = \vec{p}_t - \lambda \frac{\partial F(\vec{p}|\hat{x})}{\partial \vec{p}}\bigg|_{\vec{p}=\vec{p}_t} \tag{5.24}$$

where $\lambda$ is a positive real number.

The gradient descent method tries to bring the solution close to the global minimum along the direction of the steepest descent at each iteration. The direction of the steepest descent, however, does not coincide with the direction toward the global minimum in the multi-dimensional space.

Then, Newton method, which uses 2nd-order approximation of $F$, was proposed in order to bring the global minimum faster. We apply Taylor-expansion to $f(\vec{p})$ around point $\vec{p}_0$.

$$F(\vec{p}) = F(\vec{p}_0) + (\vec{p} - \vec{p}_0)^t \frac{\partial F(\vec{p}_0)}{\partial \vec{p}} + \frac{1}{2}(\vec{p} - \vec{p}_0)^t H(\vec{p}_0)(\vec{p} - \vec{p}_0) + \cdots \tag{5.25}$$

where $H(\vec{p})$ is called the *Hesse Matrix* or the *Hessian*. With N-dimensional vector $\vec{p} = (p_1, p_2, \cdots, p_N)^t$, the Hessian is defined as

$$H(\vec{p}) = \begin{pmatrix} \dfrac{\partial^2 F}{\partial {p_1}^2} & \dfrac{\partial^2 F}{\partial p_1 \partial p_2} & \cdots & \dfrac{\partial^2 F}{\partial p_1 \partial p_N} \\ \dfrac{\partial^2 F}{\partial p_2 \partial p_1} & \dfrac{\partial^2 F}{\partial {p_2}^2} & \cdots & \dfrac{\partial^2 F}{\partial p_2 \partial p_N} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 F}{\partial p_N \partial p_1} & \dfrac{\partial^2 F}{\partial p_N \partial p_2} & \vdots & \dfrac{\partial^2 F}{\partial {p_N}^2} \end{pmatrix} \tag{5.26}$$

At the extremal point, the derivative of $f$ takes 0.

$$\frac{\partial F}{\partial \vec{p}} \simeq \frac{\partial F(\vec{p}_0)}{\partial \vec{p}} + H(\vec{p}_0)(\vec{p} - \vec{p}_0) = 0$$

$$\therefore \quad \vec{p} = \vec{p}_0 - H(\vec{p}_0)^{-1} \frac{\partial F(\vec{p}_0)}{\partial \vec{p}} \tag{5.27}$$

Comparing Eq.5.24 to 5.27, $\lambda$ is used in the steepest descent method and $H(\vec{p}_0)$ in Newton method respectively as a coefficient of $\dfrac{\partial F(\vec{p}_0)}{\partial \vec{p}}$. When the current approximate solution $\vec{p}_t$ is far from the extremum, the steepest descent method brings the solution to it faster than Newton method. On the other hand, the convergence of the steepest descent method becomes worse near the extremum and Newton method becomes more effective.

Levenberg-Marquardt method adopts advantages of both methods. In Levenberg-Marquardt method, the update formulation of the solution is set as follows:

$$\vec{p}_{t+1} = \vec{p}_t - (\lambda E + H(\vec{p}_t))^{-1} \frac{\partial F(\vec{p}_t)}{\partial \vec{p}} \tag{5.28}$$

Levenberg-Marquardt method modifies the value of $\lambda$ dynamically at each iteration. Equation 5.28 allows the Levenberg-Marquardt method to smoothly switch between the steepest descent method (large $\lambda$) and Newton method (small $\lambda$). In practice, it starts with large $\lambda$. Then, $\lambda$ is reduced when the last iteration gives an improved estimation, i.e. $F(\vec{p}_{t+1}) < F(\vec{p}_t)$.

Here, it is a daunting task to calculate the Hessian (Eq.5.26). For simplicity, instead of 2nd-order derivatives we approximate the Hessian by using 1st-order derivatives as follows. Suppose the cost function is set as

$$F = \sum_{i}^{M} (y_i - f(\vec{p}|\hat{x}_i))^2 \tag{5.29}$$

where $\hat{x}_i$ and $y_i$ are observed values with index $i$. A 1st-order derivatives is

$$\frac{\partial F}{\partial p_k} = -2 \sum_{i}^{M} (y_i - f(\vec{p}|\hat{x}_i)) \frac{\partial f}{\partial p_k} \tag{5.30}$$

The 2nd-order derivative is therefore

$$\frac{\partial^2 F}{\partial p_k \, \partial p_l} = 2 \sum_{i}^{M} \left[ \frac{\partial f}{\partial p_k} \frac{\partial f}{\partial k_l} - (y_i - f(\vec{p}|\hat{x}_i)) \frac{\partial f^2}{\partial p_k \, \partial k_l} \right] \tag{5.31}$$

The second term of the above equation is interpreted as the summation of weighted errors $(y_i - f(\vec{p}|\hat{x}_i))$. Assuming it is close to 0, we can approximate the 2nd-order derivative as

$$\frac{\partial^2 F}{\partial p_k \, \partial p_l} \simeq 2 \sum_{i}^{M} \frac{\partial f}{\partial p_k} \frac{\partial f}{\partial k_l} \tag{5.32}$$

Thus, we can estimate the 2nd-order derivative as the summation of the products of 1st-order derivatives.

### 5.1.5 Robust Estimation of Parameters

In practice, the linear solver mentioned so far is affected the influence of noises strongly. If there are some noises in image or positions of interest points, they affect the accuracy not only on the linear solution but also on the refined parameters by non-linear minimization.

Let us take an object without any geometrical feature for example (see Fig.5.1). The figurine of the cat in Fig.5.1 has a smooth shape and only a few geometric features while there are several features with respect to texture. In the case of manual detection of interest points from a smooth model, noise in inevitable. Figure 5.1

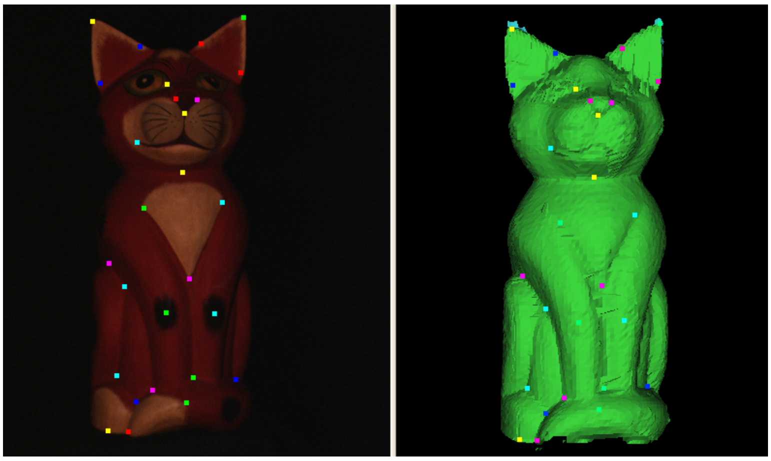


Figure 5.1: Calibration with a known model.

shows 17 corresponding point pairs between the 3D model and its 2D picture. At a glance, the 3D position on the model looks proper with respect to its 2D image. The estimated intrinsic parameters by a conventional linear solver are, however, wrong. The next intrinsic parameters are estimated through the above linear solver.

$$A = \begin{pmatrix} 4219.69 & -694.37 & -1806.50 \\ & 4330.27 & -1571.74 \\ & & 1 \end{pmatrix}$$

First of all, the principal point is located outside of the image[1]! Also the focal length is very large and the value of the skew is incredible. Consequently, the extrinsic parameters include huge errors.

The reason why the linear solver outputs incorrect parameters is considered to be that there are outliers in the input. In the case of an object with smooth surfaces, it is very difficult to specify the locations of interest points on the 3D model. Therefore, we try to get rid of outliers from input data. It is, unfortunately, not easy to make judgments as to which points include error only by watching the input data.

Then we take particular note of the facts that the aspect ratio of the camera intrinsic parameter is close to 1.0 and the skew nearly vanishes in modern digital cameras. Moreover, almost all cameras locate their principal point at the center of their images. That means, given input pairs that output the aspect ratio far from

[1] the image size is $640 \times 480$.

1.0, the skew far from 0.0 or the principal point far from the center of the image, there must be errors in the input data.

Therefore, adopting the RANSAC (Random Sampling Consensus [FB81]) technique, we propose the following algorithm for the estimation of camera parameters.

***Input:*** $N(\geq 6)$ pairs of 3D-2D correspondences extracted manually.

***Output:*** The intrinsic and extrinsic camera parameters.

1. Pick up 6 pairs from N pairs at random.

2. Solve for the projective matrix **P** and estimate the intrinsic parameters.

3. Calculate the next cost function,

$$G = f^2\left((\alpha - 1.0)^2 + s^2\right) + w\left[(u_0 - C_u)^2 + (v_0 - C_v)^2\right] \tag{5.33}$$

   Here, $(C_u, C_v)$ is the center of the image and $w$ is a weight [2].

4. Repeat above procedure, and store the intrinsic parameter set with the minimum $G$. The intrinsic parameter set with the minimum $G$ is considered as the proper intrinsic matrix element.

5. By using 6 input data sets with the minimum $G$, the extrinsic parameters are estimated.

6. Considering the above parameters as the initial solution, entire parameters are refined through Eq.5.21 by using all N pairs.

By using this robust method, we estimate the parameters as follows from the same data of Fig.5.1,

$$A = \begin{pmatrix} 832.44 & 18.36 & 284.99 \\ & 800.98 & 102.08 \\ & & 1 \end{pmatrix}$$

While some noises seem still left in the intrinsic parameters, the above result is better than the previous result achieved by the conventional method.

[2] We set $w = 0.01$.

## 5.2 3D Reconstruction by Images

The method described in Chapters 3 and 4 is based on a calibrated system in which the relative positions are known between the range sensor-oriented coordinate system and the video camera-oriented one. In this section, we describe how we applied our method to an uncalibrated system in which the configuration between the two systems is unknown.

The strategy is as follows. First, we reconstruct the 3D scene with ambiguity in scale from image sequences. In this process, camera positions with scale ambiguity and camera poses are estimated. Then the reconstructed 3D data with scale ambiguity are aligned to the roughly rectified 3D data obtained by the range sensor, which are derived from translational parameters with scale ambiguity. This alignment process removes the scale ambiguity and determines the relationship between the camera and sensor coordinate systems. After obtaining the absolute scale, we apply the refinement method mentioned in the previous section and rectify the shape.

Before describing this process, we will pause here to look briefly at related works on 3D reconstruction.

Three-dimensional reconstruction from images is one of the most significant and interesting field in Computer Vision. Besides factorization, stereopsis is one of the most traditional methods for reconstructing an object shape by using several images. Generally, it is said that there are two problems in stereo vision; *Correspondence* and *Reconstruction*.

Correspondence is determining which token in a image corresponds to another token in other images. The interesting point detector mentioned in Section 3.3 is one on the solutions for this problem. While the reconstructed model is sparse, it can deal with wide view point change by affine invariant feature detectors [Bau00] [PZ98] [STG03]. Besides the feature-based method, many area-based methods have been recently proposed. In [TSR00], [TSR01], the scene is reconstructed by using small patches segmented by color. In [RLSP03], affine patches are utilized. Optical flow [PGPO94] [SG02a] [SG02b], [ADSW02] and Graph cut [IG98] [SC98] [Roy99] [BVZ01] [KZ01] [KZ02] [SZS03] technique are also used dense reconstruction of scenes.

The another problem, Reconstruction, is dealt with in this section. In stereo vision, given the disparity between correspondence tokens, knowledge of the parameters of camera positions and poses enables reconstruction of the shape. If the parameters of all cameras are known in advance (e.g. a parallel stereo), the shape will be easily reconstructed. When we do not know camera parameters, especially extrinsic parameters, we have to estimate the camera configuration by images. In [LH81], [TH84] and [WHA89], they estimated the rotation matrix and the translation vector from the essential matrix $E$. By using the eight-point algorithm [Har97] and the five-point algorithm [Nis03], the fundamental matrix $F$ and the essential matrix $E$ are estimated, respectively, only through images. Ambiguity of scaling, however, remains in these methods. Recently, many researchers have used some sophisticated physical sensors, including gyros and GPS, to obtain the absolute scaling. In particular, for modeling large objects such as buildings and scenes, a great deal of research combining these sensors (sensor fusion) has been undertaken. In [ZNH04], they recovered camera poses and 3D structures of large objects by image sequences from the air by using motion stereo. Then the reconstructed shapes (3D point clouds) are registered to other correct 3D data, and texture images are mapped onto the 3D data.

### 5.2.1 Increment of Track Points

According to our strategy in the uncalibrated system, we need to construct a 3D model. However by using the factorization, only sparse 3D points can be reconstructed because of using the points visible from the whole sequence. Unfortunately, the number of estimated 3D points is small especially in the case of wide camera motion. It is, therefore, difficult to align this sparse 3D model to the dense model obtained by a range sensor. To overcome this problem, we increase the number of tracked points and construct dense 3D model from images. In this section, we use other points that are visible over a certain number of frames while we utilize the points that are trackable over a sequence in the previous factorization. After increasing the number of track points, we estimate their 3D coordinates. To estimate these re-registered 3D points, we use a Maximum Likelihood (ML) estimation method [DHS00].

### 5.2.2 3D Reconstruction by ML Estimation

Let us consider a situation that we are given images taken by a moving camera with known parameter and that we are given a point on each image that corresponds to the same 3D point. Here, we want to determine the 3D position of the point. Theoretically, the position of the point is interpreted as the intersection of all rays that connect optical centers and 2D points on image planes. However in practice, these rays do not intersect at a point because of noises and errors in measurements (Fig.5.2).

Then, where is the most proper 3D position? Here we assume that all the rays go through neat the true 3D point. In this case, the error corresponds to the distance between the ray and the 3D point. In addition, we assume that the error distribution follows a Gaussian function. If we denote the correct 3D point as $\hat{\mathbf{x}}$ and $\vec{x_i}$ as the nearest point on each ray of frame $i$, the distribution of error $\vec{x_i} - \hat{\mathbf{x}}$ follows the Gaussian. All vectors $\vec{x_i}$ and $\hat{\mathbf{x}}$ are described in the world coordinate system.

The error is estimated as follows in each image,

$$\begin{aligned} p(\vec{x_i} \mid \hat{\mathbf{x}}) &= p(\vec{x_i} - \hat{\mathbf{x}}) \\ &= \frac{1}{(2\pi)^{3/2}|\Lambda|^{1/2}} \exp\left[-\frac{(\vec{x_i} - \hat{\mathbf{x}})^t(\vec{x_i} - \hat{\mathbf{x}})}{2\Lambda_i}\right] \end{aligned} \tag{5.34}$$

Here, $\Lambda_i$ is covariance with respect to frame $i$. The probability $p(\vec{x_i} \mid \hat{\mathbf{x}})$ is interrupted as the conditional probability for the closest point $\vec{x_i}$ given the 3D point $\hat{\mathbf{x}}$.

For 3D reconstruction, we must estimate the $\hat{\mathbf{x}}$ with the maximum $p(\hat{\mathbf{x}} \mid \vec{x_i})$. By ML estimation, we maximize the probability $p(\vec{x_i} \mid \hat{\mathbf{x}})$ instead of $p(\hat{\mathbf{x}} \mid \vec{x_i})$ since we have no prior knowledge about the function $p(\hat{\mathbf{x}} \mid \vec{x_i})$.

For all frames, total probability is estimated as

$$p(X \mid \hat{\mathbf{x}}) = \prod_i p(\vec{x_i} - \hat{\mathbf{x}}) \tag{5.35}$$

Therefore, the correct 3D point is estimated as $\hat{\mathbf{x}}_{ML}$ by maximizing the above probability.

$$\begin{aligned} \hat{\mathbf{x}}_{ML} &= \arg\max_{\vec{x_i}} p(X \mid \hat{\mathbf{x}}) \\ &= \arg\max_{\vec{x_i}} \log\left[p(X \mid \hat{\mathbf{x}})\right] \\ &= \arg\max_{\vec{x_i}} \log\left[\prod_i p(\vec{x_i} - \hat{\mathbf{x}})\right] \\ &= \arg\max_{\vec{x_i}} \sum_i \left[-\frac{(\vec{x_i} - \hat{\mathbf{x}})^t(\vec{x_i} - \hat{\mathbf{x}})}{2\Lambda_i}\right] \\ &= \arg\min_{\vec{x_i}} \sum_i (\vec{x_i} - \hat{\mathbf{x}})^t {\Lambda_i}^{-1} (\vec{x_i} - \hat{\mathbf{x}}) \end{aligned} \tag{5.36}$$

In each frame $i$, $\vec{T_i}$ is the position of camera $i$ and $\vec{a_i}$ is the unit ray of frame $i$ described in the world system.

$$\vec{x_i} = \gamma_i \vec{a_i} + \vec{T_i} \tag{5.37}$$

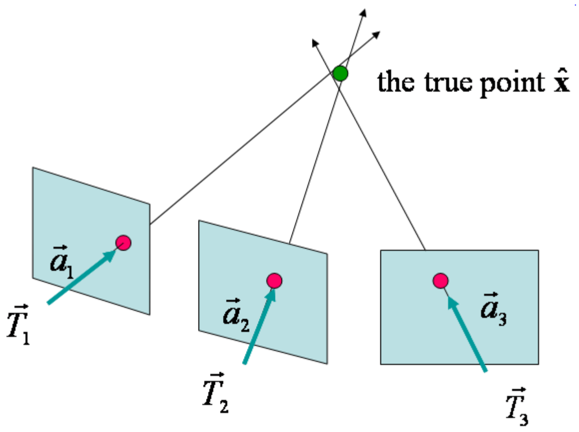


Figure 5.2: The 3D reconstruction of a point.

Here, $\gamma$ is the length between the *i*-th camera center and the 3D point, and

$$\gamma_i = ||\hat{\mathbf{x}} - \vec{T}_i|| = \vec{a}_i{}^t(\hat{\mathbf{x}} - \vec{T}_i) \tag{5.38}$$

therefore, we obtain

$$\vec{x}_i - \hat{\mathbf{x}} = \vec{a}_i\vec{a}_i{}^t(\hat{\mathbf{x}} - \vec{T}_i) + \vec{T}_i - \hat{\mathbf{x}} \tag{5.39}$$

For the minimization of Eq.5.36, the derivative takes 0.

$$\begin{aligned}\frac{d\hat{\mathbf{x}}_{ML}}{d\hat{\mathbf{x}}} &= \frac{d}{d\hat{\mathbf{x}}}\sum_i(\vec{x}_i - \hat{\mathbf{x}})^t\Lambda_i{}^{-1}(\vec{x}_i - \hat{\mathbf{x}}) \\ &= \sum_i 2\Lambda_i{}^{-1}(\vec{x}_i - \hat{\mathbf{x}})^t\frac{d(\vec{x}_i - \hat{\mathbf{x}})}{d\hat{\mathbf{x}}} \\ &= 2\sum_i \Lambda_i{}^{-1}\left(\vec{a}_i\vec{a}_i{}^t(\hat{\mathbf{x}} - \vec{T}_i) + \vec{T}_i - \hat{\mathbf{x}}\right)^t(\vec{a}_i\vec{a}_i{}^t - E) \\ &= 0\end{aligned} \tag{5.40}$$

where, $E$ is the $3 \times 3$ unit matrix.

Then, replacing $\vec{a}_i{}^t\vec{a}_i = A_i$,

$$\sum_i \Lambda_i{}^{-1}\left((A_i - E)\hat{\mathbf{x}} - (A_i - E)\vec{T}_i\right)^t(A_i - E) = 0 \tag{5.41}$$

$$\sum_i \Lambda_i{}^{-1}(A_i - E)^t(A_i - E)\hat{\mathbf{x}} = \sum_i \Lambda_i{}^{-1}(A_i - E)^t(A_i - E)\vec{T}_i \tag{5.42}$$

When we know all the camera parameters, all $\vec{a}_i$s are estimated. In this thesis, we set as $\Lambda_i = E$. Therefore, we have now obtained a linear system in the general form of $\mathbf{Kx} = \mathbf{y}$.

Thus, given camera parameters and an interest point on each frame, we can estimate the proper position of the corresponding 3D point in closed form. In addition, we can construct a dense 3D model by this method.

Here, we apply this method to a sample sequence (the sequence of ”Case3” in Chapter 7. For further details of the sequence, see Chapter 7). First, 140 interest points are detected through the whole sequence (Fig.5.3). On the other hand, the increment method detects 10,119 points, which are visible in over 30 continuous frames.

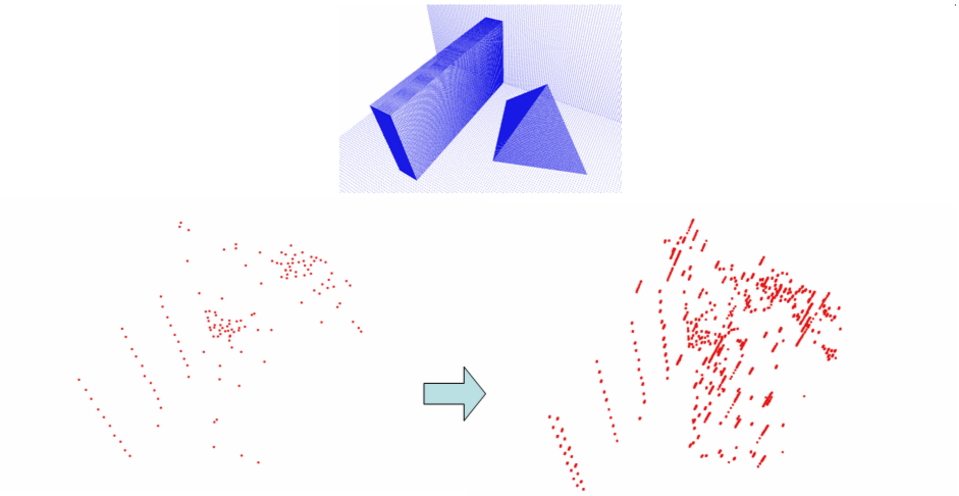


Figure 5.3: Increment of interest points. (top : ground truth)

## 5.3 Refinement by Levenberg-Marquardt Method

The camera positions and poses, and 3D positions of interest points include some amount of noise, so it is very useful to refine these parameters. In the case of 3D reconstruction, it is effective to refine parameters by Levenberg-Marquardt method. For the refinement, the minimization of the next cost function is required (Bundle Adjustment) .

$$\begin{aligned} G &= \sum_{f=1}^{F}\sum_{p=1}^{P} O_{fp}\left(\left(u_{fp} - f\frac{\vec{i}_f^{\,t}\cdot(\vec{S}_p - \vec{T}_f)}{\vec{k}_f^{\,t}\cdot(\vec{S}_p - \vec{T}_f)}\right)^2 + \left(v_{fp} - f\frac{\vec{j}_f^{\,t}\cdot(\vec{S}_p - \vec{T}_f)}{\vec{k}_f^{\,t}\cdot(\vec{S}_p - \vec{T}_f)}\right)^2\right) \\ &= \sum_{f=1}^{F}\sum_{p=1}^{P} O_{fp}\left(g_{fp}^2 + h_{fp}^2\right) \qquad (5.43) \end{aligned}$$

Here, $O_{fp}$ takes 1 if point $p$ is observable in the $f$-th frame: otherwise it takes 0. Note that the above equation has the same form as Eq.4.1 in Chapter 4 and $(u_{fp}, v_{fp})$ is described in the principal point-origin coordinate system.

There are two kinds of parameters to be solved; camera position and pose (6 degree of freedom per frame) and the 3D position of interest points. In the case of F frames and P interest points, the total number of parameters are $6F + 3P$. Describing the unknown parameter vector as $\vec{\theta}$, we divide it into two parts, $\vec{\theta}_{camera}$ of camera motion and $\vec{\theta}_{shape}$ of 3D point.

$$\vec{\theta} = [\vec{T}_1, \mathbf{q}_1, \vec{T}_2, \mathbf{q}_2, \cdots, \vec{S}_1, \vec{S}_2, \cdots] = [\vec{\theta}_{camera},\ \vec{\theta}_{shape}] \tag{5.44}$$

Then, let us consider the Hessian. The Hessian is a huge matrix because it has $(6F+3P)\times(6F+3P)$ elements. In this case, however, the Hessian is a very sparse matrix. For convenience, we divide the Hessian into 4 blocks as Fig.**??**.

First, we shall focus on the top left block, $6F\times 6F$ matrix $U$. Based on Eq.5.32, $(i - j)$ element of $U$ is calculated as

$$[U]_{i,j} = \frac{\partial^2 G}{\partial\theta_i\,\partial\theta_j} = 2\sum_{f=1}^{F}\sum_{p=1}^{P} O_{fp}\left(\frac{\partial g_{fp}}{\partial\theta_i}\frac{\partial g_{fp}}{\partial\theta_j} + \frac{\partial h_{fp}}{\partial\theta_i}\frac{\partial h_{fp}}{\partial\theta_j}\right) \tag{5.45}$$

here, $1 \le i, j, \le 6F$.

For camera motion parameters, each parameter affects only the other parameters in the same frame. Therefore, it is found that $U$ has a diagonal structure of $6 \times 6$ sub-matrices and

$$[U]_{i,j} = 2\sum_{p=1}^{P} O_{fp}\left(\frac{\partial g_{fp}}{\partial\theta_i}\frac{\partial g_{fp}}{\partial\theta_j} + \frac{\partial h_{fp}}{\partial\theta_i}\frac{\partial h_{fp}}{\partial\theta_j}\right) \tag{5.46}$$

Similarly, it is found that $V$ has a diagonal structure of $3 \times 3$ sub-matrices because each shape parameter affects only other parameters of the same point. Therefore,

$$[V]_{i,j} = 2\sum_{f=1}^{F} O_{fp}\left(\frac{\partial g_{fp}}{\partial\theta_i}\frac{\partial g_{fp}}{\partial\theta_j} + \frac{\partial h_{fp}}{\partial\theta_i}\frac{\partial h_{fp}}{\partial\theta_j}\right) \tag{5.47}$$

where $1 \le i, j \le 3P$.

Finally,

$$[W]_{i,j} = 2O_{fp}\left(\frac{\partial g_{fp}}{\partial\theta_i}\frac{\partial g_{fp}}{\partial\theta_j} + \frac{\partial h_{fp}}{\partial\theta_i}\frac{\partial h_{fp}}{\partial\theta_j}\right) \tag{5.48}$$

where $1 \le i \le 6F$, $1 \le j \le 3P$.

From Eq.5.28,

$$\begin{aligned} \Delta\vec{\theta} = \vec{\theta}^{t+1} - \vec{\theta}^{t} &= -(\lambda E + H)^{-1}\frac{\partial G}{\partial\vec{\theta}} \\ \therefore\ (\lambda E + H)\,\Delta\vec{\theta} &= -\frac{\partial G}{\partial\vec{\theta}} \end{aligned} \tag{5.49}$$

Since $\lambda E$ has only diagonal elements, the following displacement does not lose generality.

$$\lambda E + H = \begin{pmatrix} \lambda E + U & W \\ W^t & \lambda E + V \end{pmatrix} \longmapsto \begin{pmatrix} U & W \\ W^t & V \end{pmatrix} \tag{5.50}$$

After the replacement, multiply Eq.5.49 by matrix $\begin{pmatrix} E & -WV^{-1} \\ 0 & E \end{pmatrix}$ from the left side, then

$$\begin{pmatrix} E & -WV^{-1} \\ 0 & E \end{pmatrix}\begin{pmatrix} U & W \\ W^t & V \end{pmatrix}\Delta\vec{\theta} = \begin{pmatrix} U - WV^{-1}W^t & 0 \\ W^t & V \end{pmatrix}\Delta\vec{\theta} = \begin{pmatrix} E & -WV^{-1} \\ 0 & E \end{pmatrix}\frac{\partial G}{\partial\vec{\theta}} \tag{5.51}$$

This transformation can divide whole system into two groups of equations. First,

$$\Delta\vec{\theta}_{camera} = \left(U - WV^{-1}W^t\right)^{-1}\left(\frac{\partial G}{\partial\vec{\theta}_{camera}} - WV^{-1}\frac{\partial G}{\partial\vec{\theta}_{shape}}\right) \tag{5.52}$$

Then, $\Delta\vec{\theta}_{camera}$ can be used to solve the next

$$\Delta\vec{\theta}_{shape} = V^{-1}\left(\frac{\partial G}{\partial\vec{\theta}_{shape}} - W^t\Delta\vec{\theta}_{camera}\right) \tag{5.53}$$

The computation of inverse $V$ is very effective since $V$ has a diagonal structure of $3 \times 3$ sub-matrices. These transformations result in fast calculation of updated parameters.

## 5.4 Alignment-based Calibration

Next, we estimate the configuration between the camera-oriented coordinate system and the range sensor-oriented system. Using the refined camera parameters in the previous subsection, camera rotation matrix $R$ with respect to time f is represented as

$$R_f = (\vec{i}_f\ \vec{j}_f\ \vec{k}_f) \tag{5.54}$$

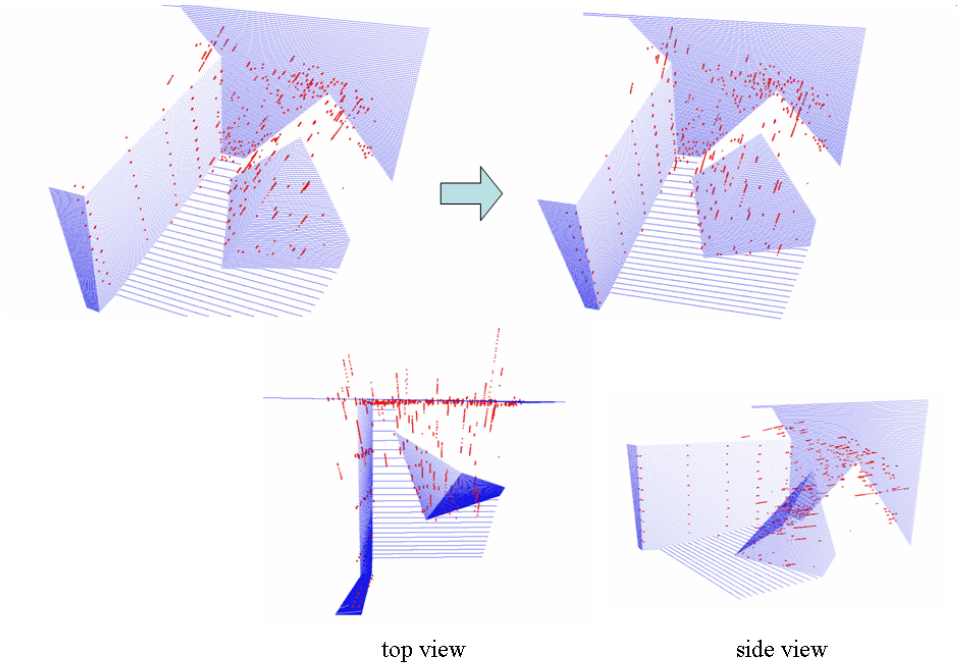
Here, the configuration between the video camera and the range sensor is described by rotation matrix $R_{intra}$ and translation vector $\vec{T}_{intra}$. Solving for $R_{intra}$ and

$\vec{T}_{intra}$ is calibration. Using scale factor $s$, the minimization of the next function (Eq.5.55 uses M-estimator, which is explained in the next chapter in detail) leads to the estimation of the relation between two coordinates.

$$\arg \min_{R_{intra},\vec{T}_{intra},s} \sum_{p=1}^{P} \log(1 + \frac{z_{fp}^2}{2\sigma^2}) \tag{5.55}$$

$$z_{fp} = R_{intra}(R_f\vec{x} + s\vec{T}_f) + \vec{T}_{intra} - s\vec{S}_p \tag{5.56}$$

There is scale ambiguity in $\vec{T}_f$ and $\vec{S}_p$, which are derived only from images. The scale factor $s$ is, therefore, multiplied by these parameters. The above function takes a small value when the shape of a cloud $s\vec{S}_p$ is aligned to the rectified range data $R_f\vec{x} + s\vec{T}_f$. After the estimation of $s$, $R_{intra}$ and $\vec{T}_{intra}$, we can use the same method described in Chapter 4.



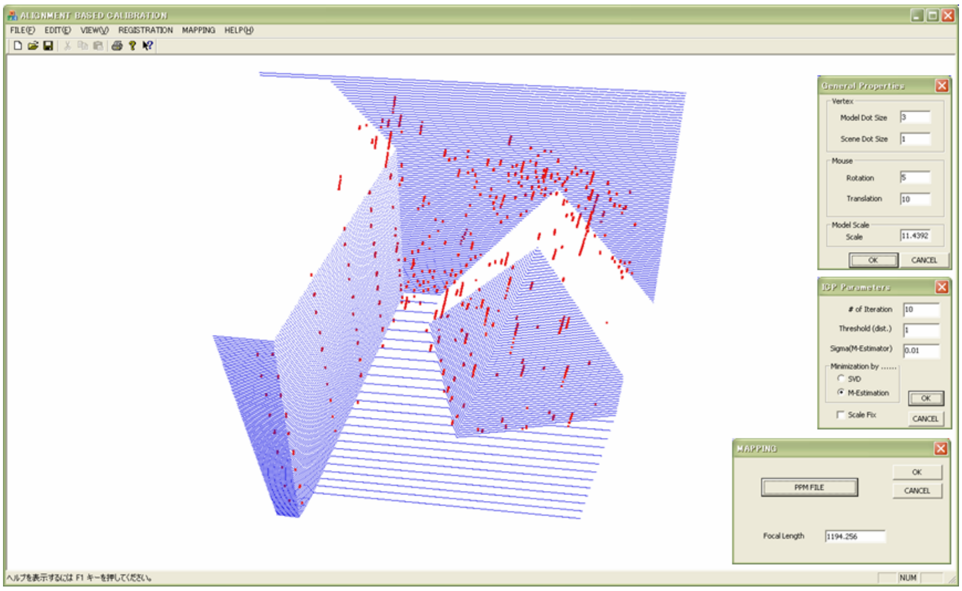
top view

side view

Figure 5.4: The result of alignment with scale factor in translation

Here, Fig.5.4 shows the result of the alignment in the sample sequence. The top left figure shows the initial solution by manual operations. The top right figure shows the result by Eq.5.55. It is found that the rectified range data (blue model) are well-fitted to the point clouds constructed by images (red points).

Incidentally, we build a software of this algorithm and it makes calibration easy (Fig.5.5).


ALIGNMENT BASED CALIBRATION
FILE(F) EDIT(E) VIEW(V) REGISTRATION MAPPING HELP(H)
OK
CANCEL
PPM FILE
ヘルプを表示するには F1 キーを押してください。
NUM

General Properties

Vertex

Model Dot Size 3

Scene Dot Size 1

Mouse

Rotation 5

Translation 10

Model Scale

Scale 11.4392

ICP Parameters

# of Iteration 10

Threshold (dist.) 1

Sigma(M-Estimator) 0.01

Minimization by ......

○ SVD

● M-Estimation

□ Scale Fix

MAPPING

Focal Length 1194.256

Figure 5.5: The GUI of the alignment based calibration method.

# Chapter 6

# Shape Rectification without Images

The method mentioned so far does not need another range data set. We can rectify distorted range data by using only a single range image and an image sequence.

In actual cases, however, there should be some available range data sets taken by another range sensor fixed on the ground. Our FLRS is originally devised to complement the measurement for the region that is invisible from the ground.

Some parts of a range image taken by the FLRS are also taken by another range sensor fixed on the ground. Based on these overlapping regions, we propose another algorithm which rectifies the distorted range data obtained from the FLRS. In this method, we do not use any image sequences.

## 6.1 Basic Idea

Originally ICP(Iterative Closest Point) algorithm [BM92] [CM92] was developed to align two shapes. In a range image, coordinates of 3D points are described in the sensor-oriented coordinate system. Two range images from different viewpoints, therefore, have different coordinate systems. To unify two shapes, two data sets have to be described in the unified system. In order to do that, we apply a coordinate conversion to one data set. When there are some overlapping regions in the two data sets, we apply a transformation of the coordinate system in order to coincide them.

To simplify the transform procedure, we assume that one shape is fixed and another can move. We call the fixed shape the ”model shape” and the movable one the ”data shape”. Rotating and translating the data shape aligns two shapes. In overlapping region, a point on the model shape has a corresponding point on the data shape. Which point is the corresponding point, however, is usually unknown. We resolve this correspondence problem by an iterative method. Initially a temporal corresponding point is assumed. A movement is determined so as to minimize an objective function, which is defined by distances between the corresponding points. The temporal correspondences are changed after the movement. Then a new movement is determined under the new temporal correspondence. This procedure is repeated until the total distance converges. The objective function, which should be minimized for the alignment, is defined as

$$f\left(R, \vec{T}\right) = f\left(\mathbf{q}, \vec{T}\right) = \sum_i \| R(\mathbf{q})\vec{x_i} + \vec{T} - \vec{y_i} \|^2 \tag{6.1}$$

This objective function indicates the summation of distances between all pairs of corresponding points. Initially the function takes high values because there are a lot of wrong relationships of correspondences. As iterating calculations, wrong correspondences are improved and the function takes a converged value. If two shapes coincide, the function takes a low value. When the function converges under a threshold, we decide two shapes are similar.

There are many variations of ICP algorithms [RL01]. For example, while we estimate the cost function as the total distances of point-to-point pairwise [BM92] [Zha94], some methods adopt the distance between a point and its mate’s tangent plane [CM92] [Neu97].

For corresponding points, there are several methods to determine them. Some methods search the corresponding point along the ray [BL95]. In this thesis, we adopt the nearest neighbor points as the corresponding points. We speed up searches for the nearest neighbor point by using KD-tree [FBF77] [Whe96] [Nis01] [Mas03].

We use quaternion to minimize the objective function $f$. By substituting quaternion $\mathbf{q}$ to rotate matrix $R$, motion vector $\vec{T}$ can be found as follows.

$$\{\mathbf{q}, \vec{T}, \} = \arg\min_{\mathbf{q},\vec{T}} f\left(\mathbf{q}, \vec{T}\right) = \sum_i \| R(\mathbf{q})\vec{x_i} + \vec{T} - \vec{y_i} \|^2 \tag{6.2}$$

In the conventional ICP algorithm mentioned above, it is assumed that both shapes are obtained by fixed range sensors. On the other hand, in our situation, the model shape is obtained by a fixed range sensor while the data shape is measured by a moving sensor. Therefore we have to take account into the motion of the range sensor.

The motion of the sensor is expected to be smooth, as mentioned in the previous chapter. It is, therefore, proper that the traces of the motion parameters are approximated by some polynomials with respect to time. Consequently, we approximate six parameter, three translational elements and three elements of the quaternion, by following polynomials.

$$\vec{T}(t) = \vec{T}_0 + t\vec{T}_1 + t^2\vec{T}_2 + \cdots = \sum_{n=0} t^n\vec{T}_n \tag{6.3}$$

$$\mathbf{q}(t) = \mathbf{q_0} + t\mathbf{q_1} + t^2\mathbf{q_2} + \cdots = \sum_{n=0} t^n\mathbf{q_n} \tag{6.4}$$

where $\{\vec{T}_0, \vec{T}_1, \cdots, \vec{T}_N, \mathbf{q_0}, \mathbf{q_1}, \cdots, \mathbf{q_N}\}$ are the parameters that describe the sensor motion. Then we formulate a new cost function including the above forms.

## 6.2 Extended ICP Algorithm

Instead of Eq.6.1, we have to set up a new cost function.

First, we will change the index of points of data shape, $\vec{x}_i$. Our sensor measure the distance to a point in the raster scan order. Therefore, all points on the data shape, which are measured by the moving sensor, are distinguishable by time $t$. According to the time factor, the corresponding points on the model shape $\vec{y}_i$, which are obtained by a fixed sensor, are described as functions $\vec{y}\,(\vec{x}(t))$.

Then, the cost function for the extended ICP algorithm is described as follows:

$$f\left(\vec{T}_0, \vec{T}_1, \cdots, \vec{T}_N, \mathbf{q_0}, \mathbf{q_1}, \cdots, \mathbf{q_N}\right) = \sum_t \| \, R\,(\mathbf{q}(t))\,\vec{x}(t) + \vec{T}(t) - \vec{y}(\vec{x}(t)) \, \|^2 \tag{6.5}$$

We take a summation form with respect to time $t$ in spite of the continuity of time. Since it is only necessary to pick up the moments when the point on the data shape is actually scanned.

To minimize the above function, the parameters of the sensor motions are estimated.

$$\begin{aligned} &\{\vec{T}_0, \vec{T}_1, \cdots, \vec{T}_N, \mathbf{q_0}, \mathbf{q_1}, \cdots, \mathbf{q_N}\} = \\ &\qquad \arg \min_{\vec{T}_0, \vec{T}_1, \cdots, \vec{T}_N, \mathbf{q_0}, \mathbf{q_1}, \cdots, \mathbf{q_N}} f\left(\vec{T}_0, \vec{T}_1, \cdots, \vec{T}_N, \mathbf{q_0}, \mathbf{q_1}, \cdots, \mathbf{q_N}\right) \end{aligned} \tag{6.6}$$

If we assume $N$-order polynomials, the number of unknown valuables is $6(N+1)$.

We minimize the cost function through the steepest descent method and Golden section search. The reason why we adopt them is that the many corresponding pairs change at every iteration and the contours in Fig.4.3 change on each iteration. There is no advantage to applying Levenberg-Marquardt nor the conjugate gradient method, which can search the next approximate solution effectively in the fixed contours.

Furthermore we adopt a robust estimation to reject outliers in the minimization.

### 6.2.1 M-Estimator

In the original ICP algorithm, the rigid transformation parameters $R$ and $\vec{T}$ are estimated by minimizing the cost function. In fact, however, there are many situations in which the solutions do not result in conformation of the data shape to the model data because both data sets are contaminated by noise. Moreover, since two data sets are measured from different viewpoints, some parts of the data shape have no corresponding points on the model shape. In the above method, the nearest neighbor points are use as the corresponding points. There are, therefore, many wrong pairs in the correspondences of $\vec{x} \Leftrightarrow \vec{y}(\vec{x})$.

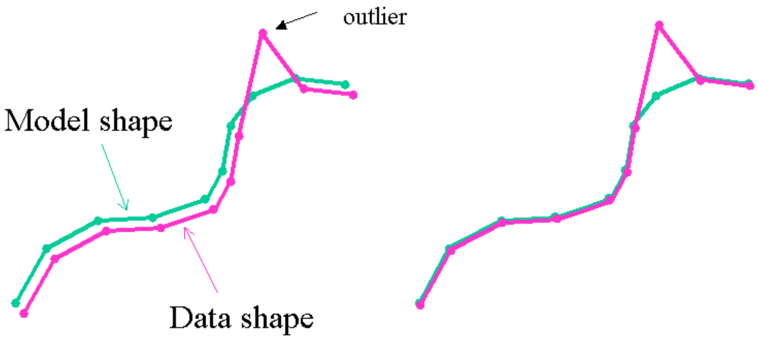


Figure 6.1: Alignment of two shapes with an outlier. left: least-square method. right: an expected fitting with outliers

Because of the above reasons, there are many disadvantages to minimize Eq.6.5 of a simple least-square form. For example, suppose two models on the right side of Fig.6.1, which contains outliers in the data shape. The minimization of a simple least-square leads to the solution on the left side of Fig.6.1.

At the next step, we have to reject outliers robustly and estimate valid parameters. Following are several types of methods that estimate the solution in the case of noisy data sets. Among them, we adopt a technique of M-estimator [PFTV88] [GMW81] [WP97]. In M-estimation, the cost function has the general form as follows:

$$E(z) = \sum_i \rho(z) \tag{6.7}$$

where $\rho(z)$ is an arbitrary function of the errors $z_i$ in the data set. When we adopt the $\rho(z)$ as $\rho(z) = z^2$, it is a simple least-square method. That is, a least-square method is one of the branches in M-estimations. In our implementation, we adopt Lorentzian function as the M-estimator.

$$\rho(z) = \log\left(1 + \frac{1}{2\sigma^2}z^2\right) \tag{6.8}$$

This function is introduced as follows: Suppose that the probability distribution of outliers is in this form;

$$P = \prod_i \exp\left[-\rho(z_i)\right] \tag{6.9}$$

For example, when the probability distribution has a Gaussian form, $\rho(z) = \frac{1}{2}z^2$, we define the deviation of $\rho(z)$ as $\psi(z)$.

$$\psi(z) \equiv \frac{\partial\rho(z)}{\partial z} \tag{6.10}$$

Here, we want to minimize the probability P of Eq.6.9. The errors means the differences between the observed values and the theoretical figures with parameter **a**. Minimizing $P$ equals minimizing $\log P$. Therefore by taking the deviation of $\log P$ with respect to **a** as 0,

$$\frac{\partial \log P}{\partial \mathbf{a}} = -\sum_i \frac{\partial\rho}{\partial z_i}\frac{\partial z_i}{\partial \mathbf{a}} = -\sum_i \psi(z_i)\frac{\partial z_i}{\partial \mathbf{a}} = 0 \tag{6.11}$$

As can be seen in Eq.6.11, a function $\psi(z)$ can serve as a weight for each data sets $z_i$, and M-estimator can be interpreted as a weighted-least-square method.

In the conventional least-square method, $\psi(z) = z$ is applied, in which the greater errors $z$, the greater value takes $\psi(z)$.

$$\psi(z) = \frac{\partial\rho(z)}{\partial z} = z$$
$$\therefore \rho(z) = \frac{1}{2}z^2$$

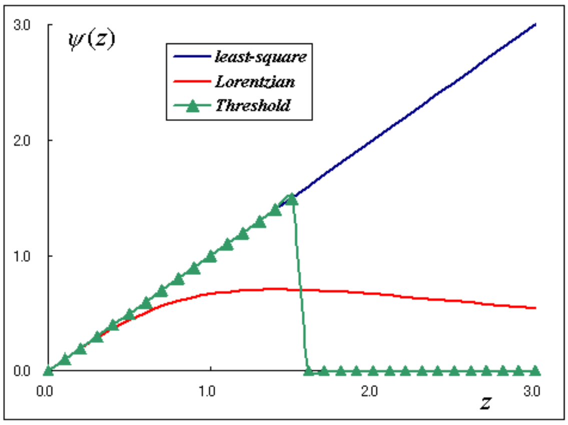


Figure 6.2: Several types of $\psi(z)$. $\sigma = 1.0$ in the Lorentzian function and the threshold is set at 1.5

Consequently some data sets with huge errors prevent the proper estimation of parameters.

In this thesis, we adopt Lorentzian function as M-estimator as follows;

$$\psi(z) = \frac{z}{1 + \frac{1}{2\sigma^2}z^2} \tag{6.12}$$

In Lorentzian function, the weight $\psi(z)$ is increasing as the error $z$ increases within a certain range. As the error increases more than it, the weight is decreased (Fig.6.2). Consequently perfect outliers have less influence on the estimation of the parameters. Integrating Eq.6.12 with respect to $z$ leads to Eq.6.8. And $\sigma$ is interpreted as a parameter that determines the weight for the outliers. The larger $\sigma$, the heavier the weight for the outliers. As seen as Eq.6.12, in the case of $\sigma \to \infty$, M-estimation corresponds to the least-square method.

### 6.2.2 Minimization with M-Estimator

Based on the above considerations, the cost function Eq.6.5 is rewritten as follows:

$$\arg \min_{\vec{T}_0,\vec{T}_1,\cdots,\vec{T}_N,\mathbf{q_0},\mathbf{q_1},\cdots,\mathbf{q_N}} \sum_t \log\left(1 + \frac{1}{2\sigma^2}{z_t}^2\right) \tag{6.13}$$

$$\text{where} \quad z_t = R(\mathbf{q}(t))\vec{x}(t) + \vec{T}(t) - \vec{y}(\vec{x}(t))$$
$$\vec{T}(t) = \vec{T}_0 + t\vec{T}_1 + t^2\vec{T}_2 + \cdots = \sum_{n=0} t^n\vec{T}_n$$
$$\mathbf{q}(t) = \mathbf{q_0} + t\mathbf{q_1} + t^2\mathbf{q_2} + \cdots = \sum_{n=0} t^n\mathbf{q_n}$$

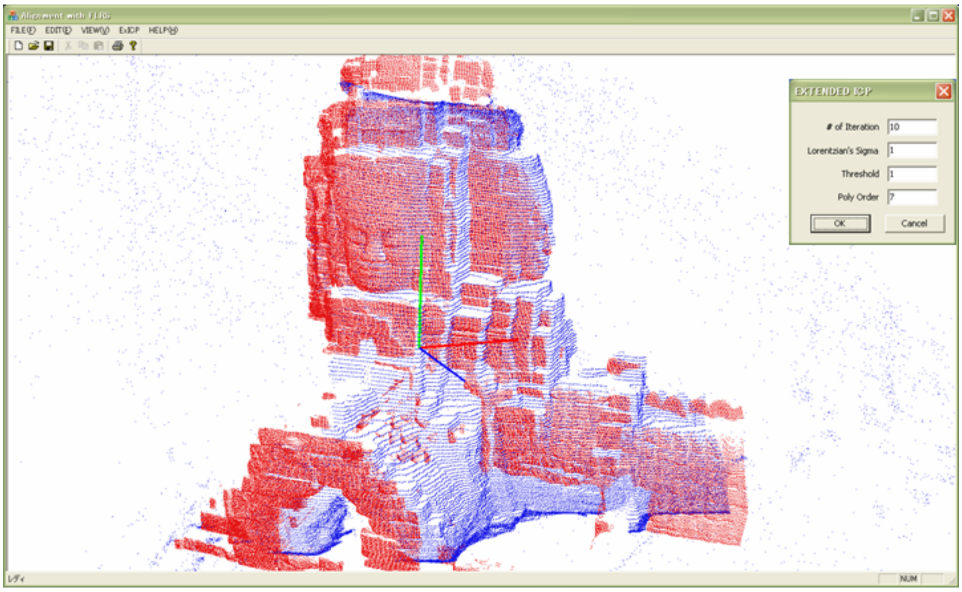
Replacing Eq.6.13 as $F$, a derivative with respect to $\vec{T}_i$ is

$$\begin{aligned}\frac{\partial F}{\partial \vec{T}_i} &= \sum_t \frac{2z_t}{2\sigma^2 + z^2}\frac{\partial z_i}{\partial \vec{T}_i} \\ &= \sum_t \frac{2z_t}{2\sigma^2 + z^2}\ t^i \end{aligned} \tag{6.14}$$

Similarly, a derivative with respect to $\mathbf{q}_i$ is

$$\begin{aligned}\frac{\partial F}{\partial \mathbf{q}_i} &= \sum_t \frac{2z_t}{2\sigma^2 + z^2}\frac{\partial z_i}{\partial \mathbf{q}_i} \\ &= \sum_t \frac{2z_t}{2\sigma^2 + z^2}\frac{\partial R}{\partial \mathbf{q}_i}\vec{x}(t) \\ &= \sum_t \frac{2z_t}{2\sigma^2 + z^2}\frac{\partial R}{\partial \mathbf{q}}\ t^i\ \vec{x}(t) \end{aligned} \tag{6.15}$$

We can easily build a graphic user interface (GUI) onto this method and a practical software (Fig. 6.3).



# of Iteration 10

Lorentzian's Sigma 1

Threshold 1

Poly Order 7

Figure 6.3: The GUI of the extended ICP algorithm

# Chapter 7

# Evaluation

In this Chapter, we evaluate our algorithms by using known CAD models. Constructing a virtual FLRS using a PC, we estimate the accuracy and the limitation of our methods objectively.

## 7.1 Benchmark Shapes

To evaluate our rectification algorithms quantitatively, the most efficient method is to check them for given models in advance.

In order to do that, we construct a virtual FLRS system on a PC and obtain the distorted range data and the image sequences for known model. Motion parameters are know completely. Also, we rectify the distorted range data through our two proposed methods.

The rectified shape data are, eventually, compared with the correct shape data, and the results are evaluated numerically.

We use the following CAD models as a benchmark for the evaluation (Fig.7.1). The benchmark has a large depth, which has a strong perspective effect. For reference, the height of the pyramid is 0.6, that of the side wall is 0.78 and the thickness of the side wall is 0.2. The equation of the back plane is $z = 0$ and that of the floor is $y = 0$.

Then, we map textured pictures onto the surfaces of the benchmark shapes to detect many interest points for tracking. In this chapter, we do not intend to evaluate the performance of the interest point detectors. All we are interested in is to evaluate the performance of rectification and the accuracy of the rectified shapes.

After that, we provide three sensor motions for virtual measurements (Fig.7.2).

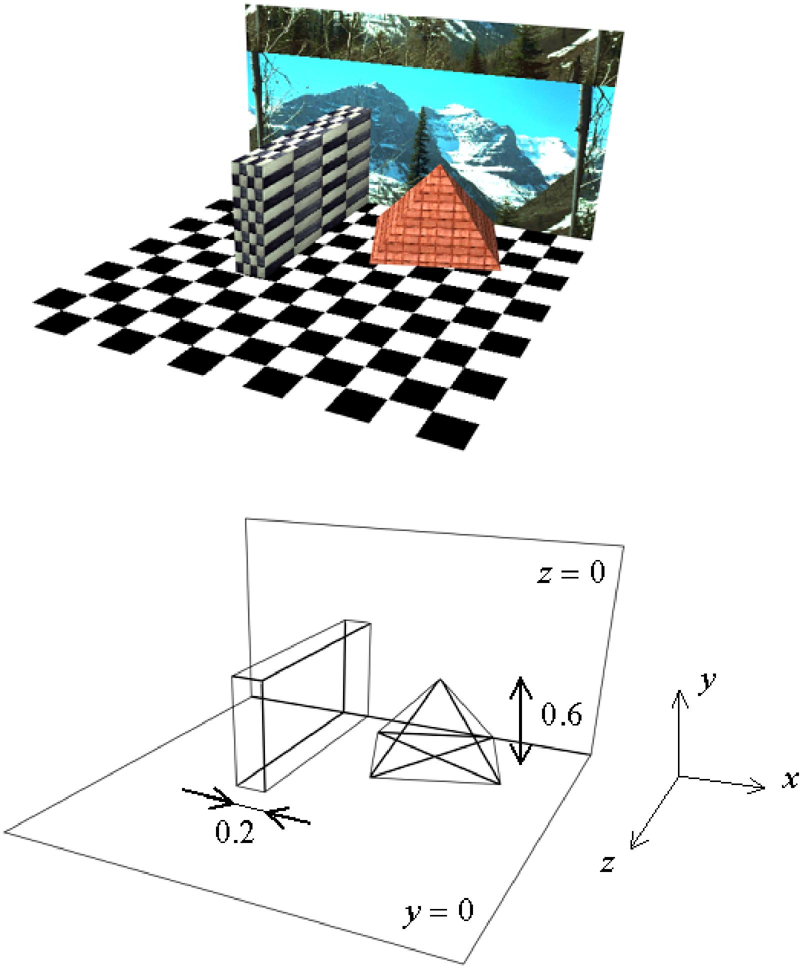


Figure 7.1: The benchmark shape for the evaluation.

1. Pure translation along the $x$ direction (parallel to the image plane).
2. Pure translation along the $-z$ direction (perpendicular to the image plane).
3. Translation and rotation around the $y$ axis.

## 7.2 Evaluation of Our Algorithm with Images

First, let us evaluate the method mentioned in Chapters 3 and 4, which uses image sequences for initial estimation of the shape and motion parameters. This method

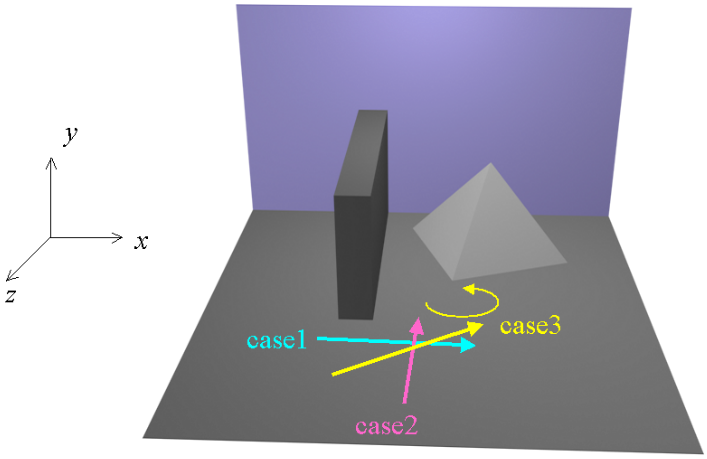


Figure 7.2: The sensor path for the evaluation.

is based on ”Structure from Motion” techniques. In the nature of things, we presuppose that the motion of the sensor has translational components. If the motion has only rotational components and does not have any translational ones, it is impossible to reconstruct 3D shapes or motions by using image sequences only. In this section, it is also assumed that the virtual FLRS use a calibrated camera for the sensor-oriented coordinate system because we are not interested in the accuracy of calibration in this step.

### Coefficients $(w_1, w_2, w_3)$ in the Cost Function

In this thesis, we determine the coefficients $w_1$, $w_2$, and $w_3$ in Eq.4.14 as follows: First, let us approximate the values of $F'_A$, $F'_B$ and $F'_C$, respectively.

The function value of $F_A$ in Eq.4.1 originally means the total distances between the interest points and the re-projected points through the whole sequence. We can expect the distance of each pair in a frame as $10^{-1}$ pixel order, $O(10^{-1})$. Then the number of the interest points is at most $O(10^2)$ and that of the frames is almost $O(10^2)$. The order of $\vec{S}_p - \vec{T}_f$ is considered as $O(10^1)$ since the size of the target for the real FLRS is $O(10^1)$ [m]. Therefore, the value of $F'_A$ is expected as $O(10^4)$.

The function value of $F'_B$ is the squared summation of total velocity and quaternion accelerations. While it depends upon the case, it is expected to be of a small order. We approximate the order of $F'_B$ as $O(10^{-3})$ based on several measurements by the real FLRS.

The value of the third constraint $F'_C$ means the total errors with respect to 3D positions of the interest points. They depend on the accuracy of the range sensor. For our FLRS, it is expected as $O(10^{-2})$ [m]. Therefore, we approximate the value of $F'_C$ as $O(10^{-2})$.

Based on the above considerations, the values of the three functions are considered as $O(10^4)$, $O(10^{-3})$ and $O(10^{-2})$ respectively. Then we set three coefficients as $w_1 : w_2 : w_3 = 1 : 10^7 : 10^6$ so that all constraints have the same weight, which is fixed in all cases.

### Case 1:

In this case, the FLRS simply moves during the measurement process toward the horizontal direction with respect to the camera-oriented coordinate system. The motion path is parallel to the image plane and the back plane of the benchmark model.

Some example images of the sequence are shown in Fig.7.3. These images look like pictures obtained by simple parallel stereo vision since there are not any rotational elements in Case 1.

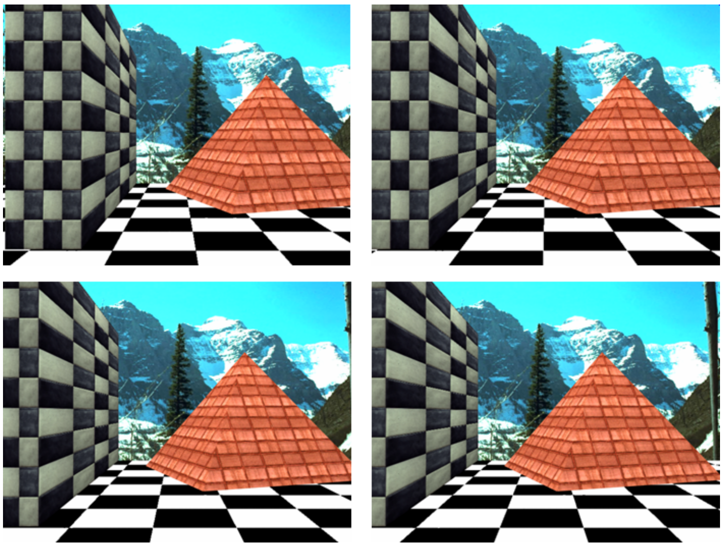


Figure 7.3: Some sample images of the sequence Case 1. (top left → top right → bottom left → bottom right)

The distorted shape which is obtained by the virtual FLRS is shown in the left of Fig.7.4. Especially, it is found that the top region of the side wall is skewed to the right side. On the other hand, in the right shape, which is the rectified shape by our algorithm, the side wall stands perpendicular to the ground. For the time being, the shape seems to be rectified properly by our method. The numerical evaluation for the rectified shape is show at the end of this section.

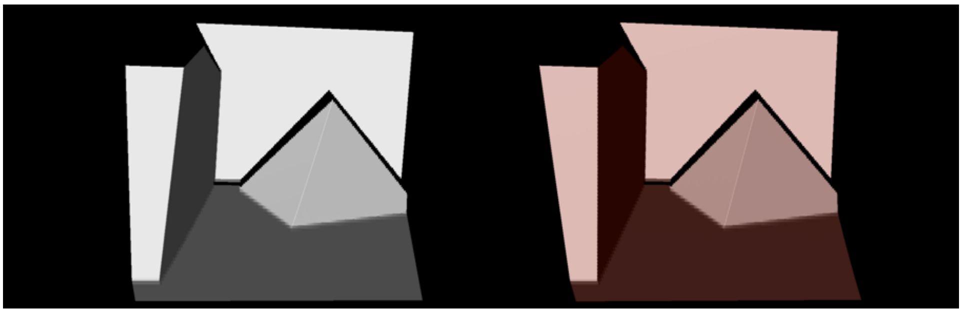


Figure 7.4: The original and rectified model of Case 1.

Figure 7.5 indicates the estimated $x$ position and the ground truth. In Case 1, we set a uniform straightly-line motion and the result shows it. The difference between the estimated velocity and the ground truth is only 6.4%.

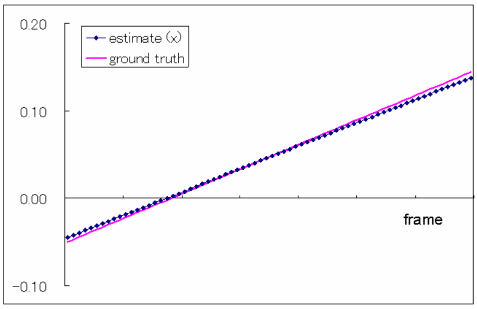
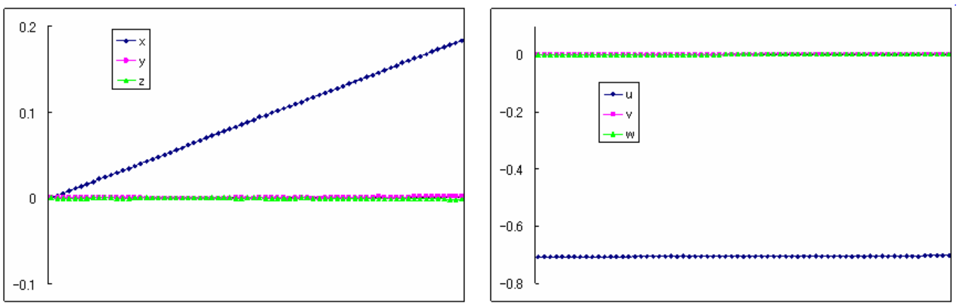


Figure 7.5: The camera path and the ground truth in Case 1.

All parameters, three components of translation and three components of camera pose, through the scanning period are shown in Fig.7.6. As the translational components, the position at $f = 0$ is set as the origin. The left figure shows that the FLRS moved only along the $x$ direction, which corresponds to the ground truth. In addition, the right figure shows that the motion did not have any rotational component, which also corresponds to the ground truth.



translation

rotation

Figure 7.6: The all camera parameters in Case 1.

### Case 2:

In this case, the FLRS moves along the optical axis, which is perpendicular to the image plane. Figure 7.7 shows several images of the sequence. Compared to Case 1 we set a larger moving distance in this case. It is found that the scene is dynamically closing.

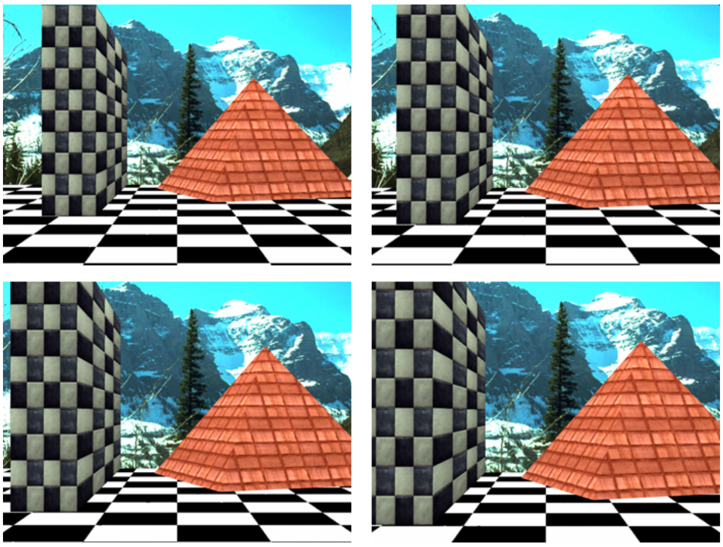


Figure 7.7: Some sample images of the sequence Case 2.

The distorted shape which is obtained by the virtual FLRS is shown in the left of Fig.7.8. When the virtual FLRS scans the top region of the scene it is located far from the scene. Then the closer the FLRS moves, the lower region it scans.

Therefore, the obtained shape seems as though it is skewed backward. As with Case 1, the right side of the figure shows the rectified shape, which looks like the proper shape.

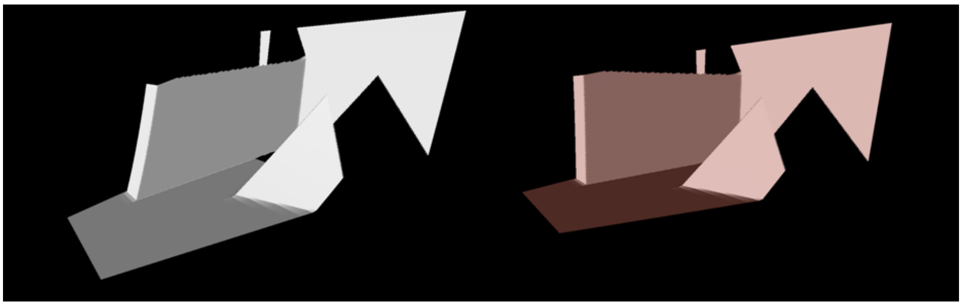


Figure 7.8: The original and rectified model of Case 2.

Figure 7.9 indicates the estimated $z$ position and the ground truth. The difference between the estimated velocity and the ground truth is 13.4 %. While the estimated error is larger than that of Case 1, the motion of Case 2 is wider than that of Case 1. The virtual FLRS's speed in Case 2 corresponds to about 3.0 $m/s$ in terms of the real FLRS scale. It is thought that the our algorithm can rectified the distorted shape in spite of the wide motion.

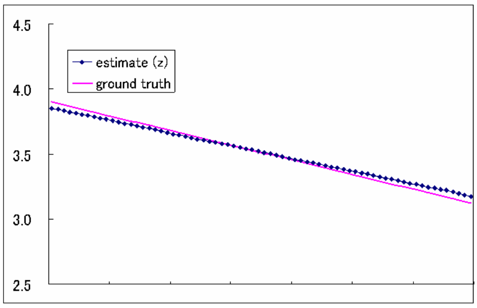
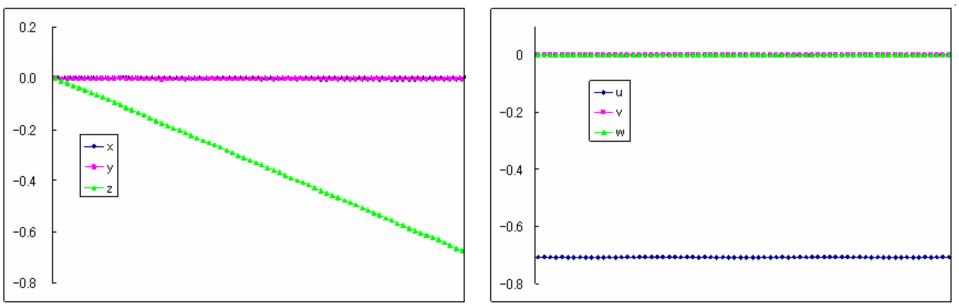


Figure 7.9: The camera path and the ground truth in Case 2.

All motion parameters are shown in Fig.7.10. The left figure which shows the translational components shows that the FLRS moved only along the $z$ direction. And the right figure shows that the FLRS was keeping the same pose during the scanning process. These figures indicate that the parameters are estimated properly.



translation

rotation

Figure 7.10: The all camera parameters in Case 2.

### Case 3:

In this case, the virtual FLRS motion has two translational components, *x* and *z*. In addition, the FLRS rotates 3° around the *y* axis during the scanning process. Figure 7.21 shows several images of the sequence.

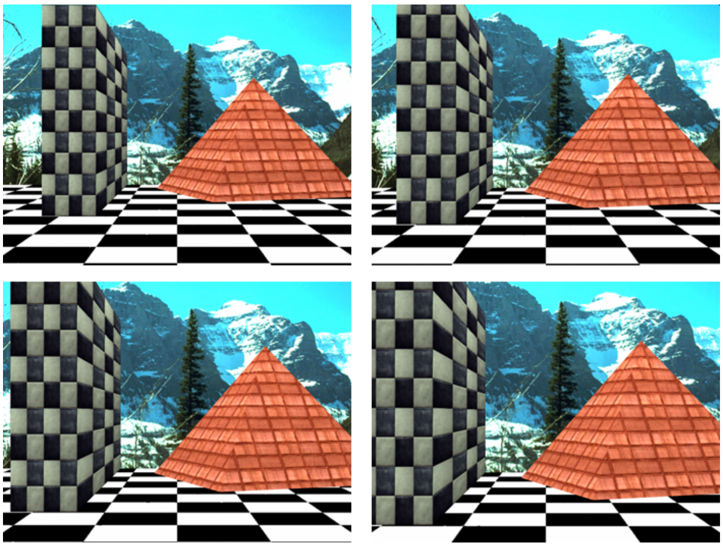


Figure 7.11: Some sample images of the sequence Case 3.

The distorted shape obtained by the virtual FLRS is shown in the left side of Fig.7.12. As in Case 1, it is found that the top region of the side wall is skewed to the right side. The right side of the figure shows the rectified shape, which looks like proper shape.

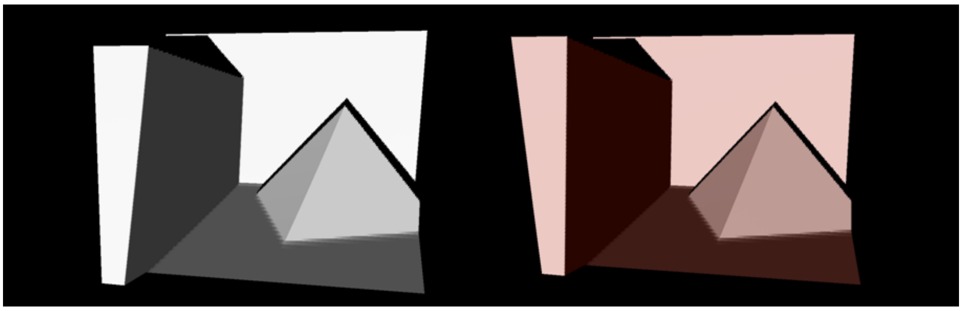
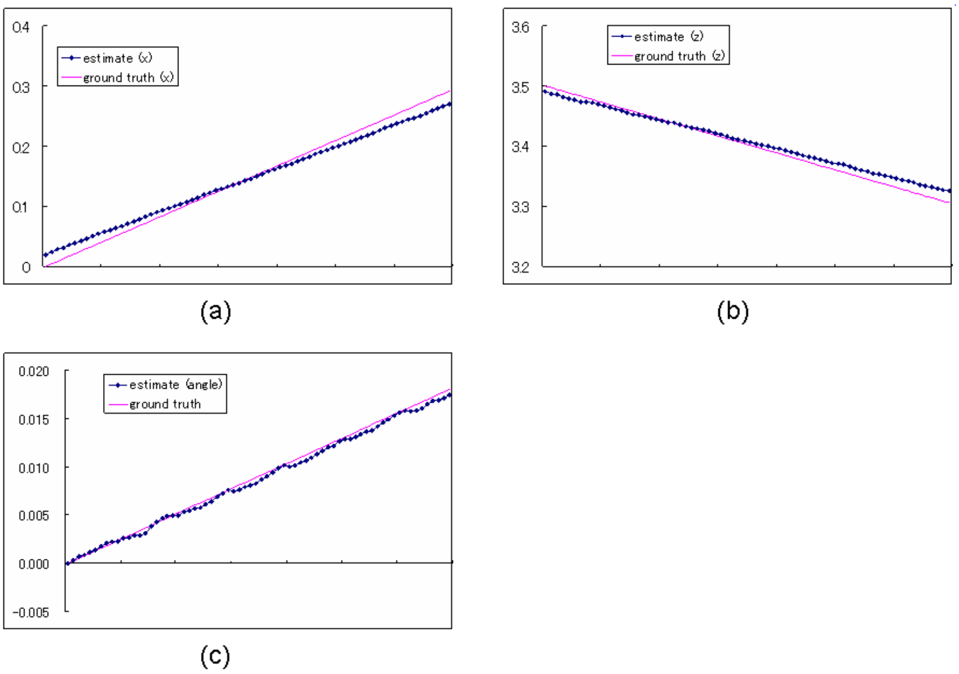


Figure 7.12: The original and rectified model of Case 3.

Figure 7.13 indicates the estimated parameters and the ground truths. In Fig.7.13, three parameters, $x$ position (a), $z$ position (b) and rotational component around $y$ axis are shown. The difference between the estimated velocity and the ground truth is 13.8 % with respect to $x$ and 15.0 % with respect to $z$. But the difference with respect to the rotational angle is within 5.6 %.



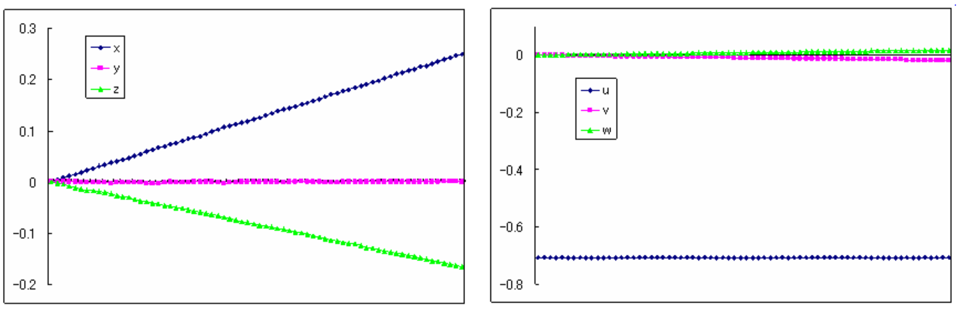
(a)

(b)

(c)

Figure 7.13: The camera path and the ground truth in Case 3. (a) $x$ position (b) $z$ position (c) Rotational component around $y$ axis

All motion parameters are shown in Fig.7.14. These figures show that our algorithm works well on a case with several motion components.



translation rotation

Figure 7.14: The all camera parameters in Case 3.

Finally, Table 7.1 shows the errors in all cases. These values are mean errors by point-to-patch distance. The errors in ”Before Rectification” row are the mean errors between the distorted shapes and the ground truth, which are aligned by ICP algorithm [BM92] [CM92]. On the other hand, the values in ”After Rectification” row are the mean errors between the rectified shapes and the ground truth. It is found that our method could decrease the errors in all cases. In the case of the real 25m FLRS, the maximum distance for scan is at most 25 meters while the distance to the backplane in the benchmark shapes is about 3.5 in the CAD model. Therefore, multiplying the values of Table 7.1 by at most 7 gives the estimated errors in practical measurement. In almost data sets in the Bayon Temple, we measure objects at a distance of 15 $\sim$ 18 meters. For example, the estimated accuracy in Case 2 will be about 3 cm in practice.

Table 7.1: The mean errors of the method with images.

| | case1 | case2 | case3 |
|---|---|---|---|
| Before Rectification | 0.0134202 | 0.066315 | 0.0310331 |
| After Rectification | 0.00499022 | 0.00637914 | 0.00426805 |

## 7.3 Evaluation of Our Algorithm without Images

Next, we evaluate the method mentioned in Chapter 6, which uses correct shapes obtained by other fixed laser sensors without any image sequences. In this section, the data sets are the same as in the previous section. Besides these, Case 4 is added, in which the motion of the sensor contains only rotation without any translational components. In fact, the method with images failed in Case 4 since any disparities could not be detected in images.

### Case 1:

In Case1, the sensor simply moves toward the horizontal direction.

Figure 7.15 shows the rectified model and the ground truth (the original distorted model is shown in Fig.7.11). At a glance, we see that the method could rectified the distorted model properly.

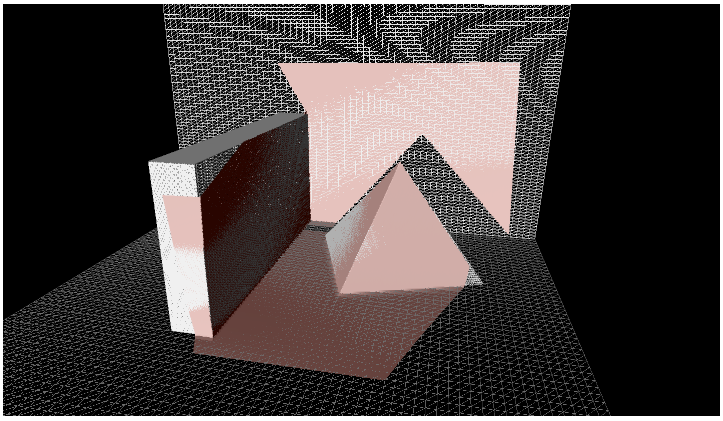


Figure 7.15: The ground truth and rectified model of Case 1.

The following figures show all motion parameters. All translational parameters change in time although the ground truth setting moves the sensor only along the *x* axis. In addition, the estimated velocity is not constant. Comparing it to Fig.7.6, it is found that the graphs, especially in the left figure, differ from those using the method with images.

In spite of these graphs, we can safely state that our method is effective. This method places more emphasis on the minimization of the geometrical error and less on the proper estimation of sensor motion. For example, when the FLRS scans a simple plane, many patterns of motion can be right. Therefore, we consider that our method could rectify the deformed shape properly.

The table of errors in all cases is shown at the end of this section.

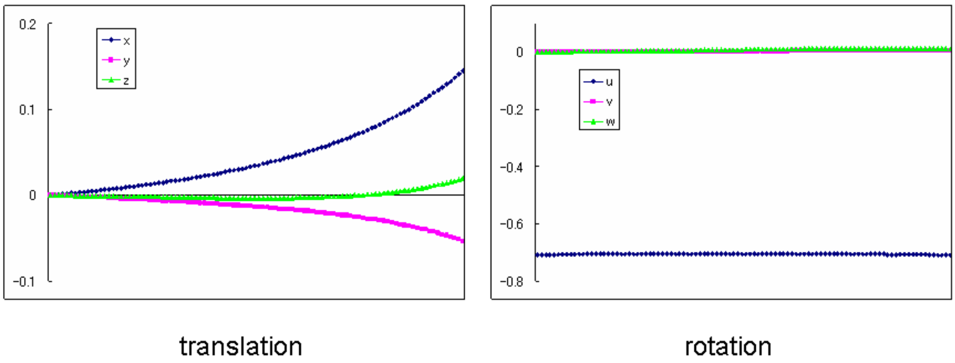


Figure 7.16: The all camera parameters in Case 1.

## Case 2:

In this case, the sensor moves along the optical axis at a fast speed. Figure 7.17 shows the rectified model in Case 2. There are some mismatched parts, especially at the top of the side wall. We consider the high speed motion would cause the mismatches.

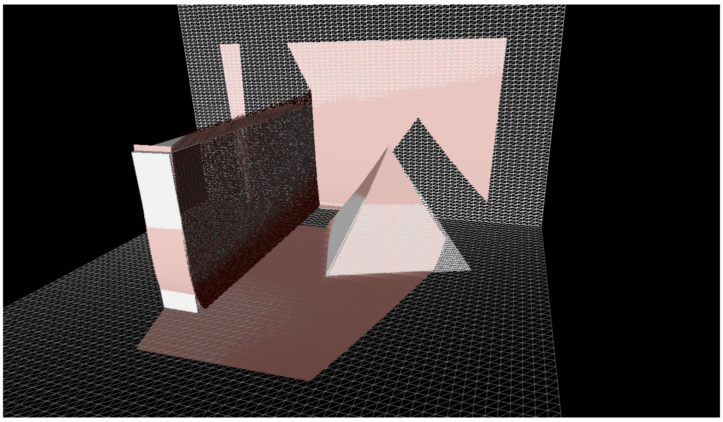
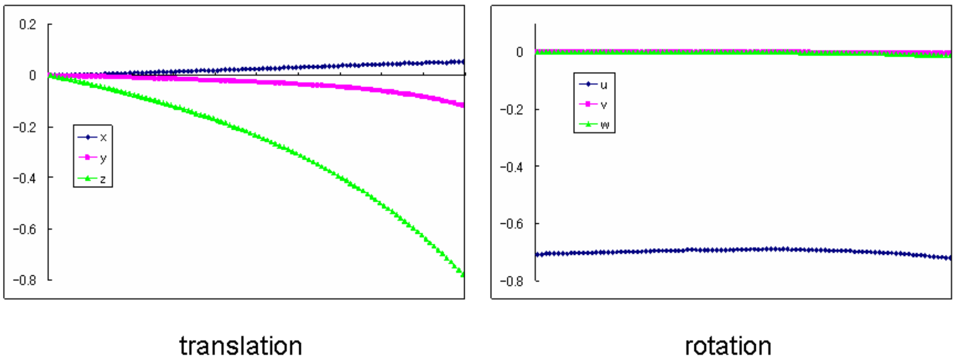


Figure 7.17: The ground truth and rectified model of Case 2.

Figure 7.18 shows the all motion parameters. Under the ground truth configuration, only the *x* translational parameter is supposed to change. In Fig.7.18, it is easily found that almost all parameters fluctuate.



translation

rotation

Figure 7.18: The all camera parameters in Case 2.

## Case 3:

In this case, the sensor motion moves within a plane parallel to $y = 0$ and rotates 3° around the $y$ axis. Figure 7.19 shows the rectified model in Case 3. The rectified model looks like proper because of a relatively moderate sensor motion.

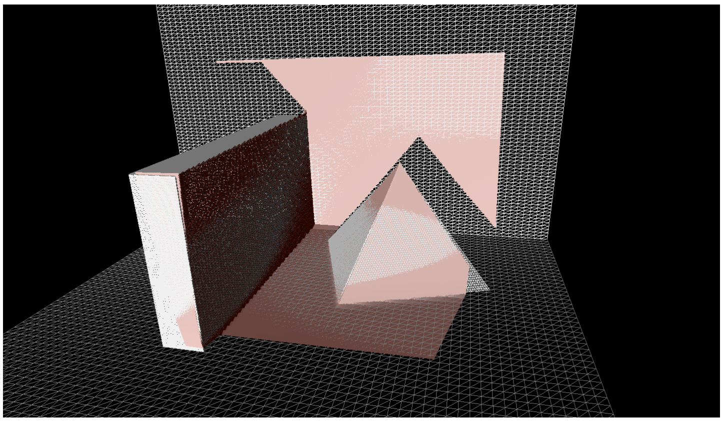
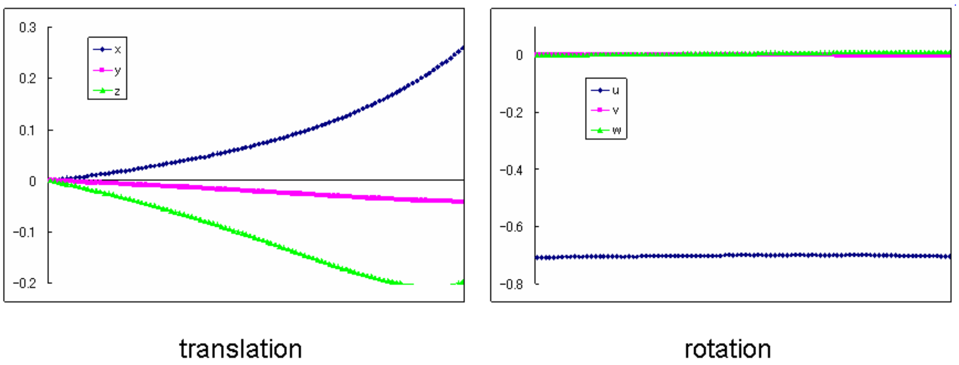


Figure 7.19: The ground truth and rectified model of Case 3.

Figure 7.20 shows the all motion parameters. Comparing it to Fig.7.14, the graphs in Fig.7.20 have similar properties. The translational graphs are, however, curved and the $y$ component, which is supposed to be fixed, is moving.



translation

rotation

Figure 7.20: The all camera parameters in Case 3.

### Additional Case (Case 4):

In this case, while the position of the sensor does not change, it rotates 3° around the *y* axis. As mentioned in the previous section, the method with images can not rectify the distorted model because it is impossible to reconstruct the 3D model from images without disparity (Fig.7.21).

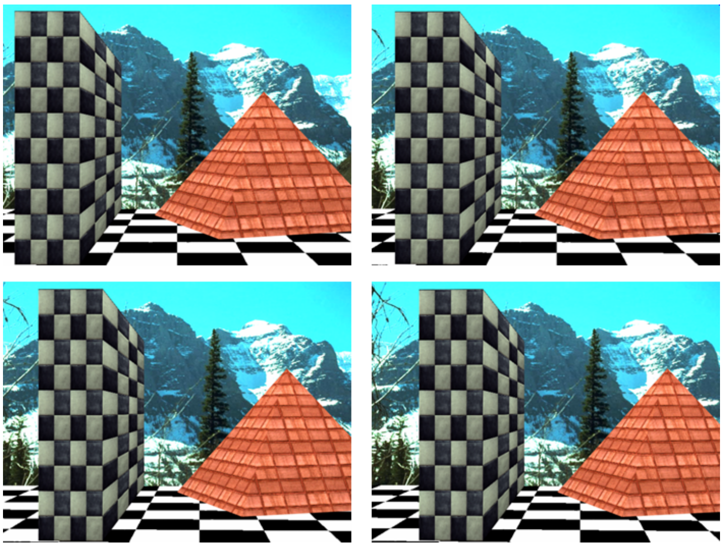


Figure 7.21: Some sample images of the sequence Case 4.

The left side of the figure in Fig.7.22 is a comparison between the ground truth and the original distorted model while the right side of the figure is a comparison between the ground truth and the rectified model. It is found that the method without images can properly rectify distorted models that are obtained from a sensor only with rotation. Thus, this is the strong advantage for this method.

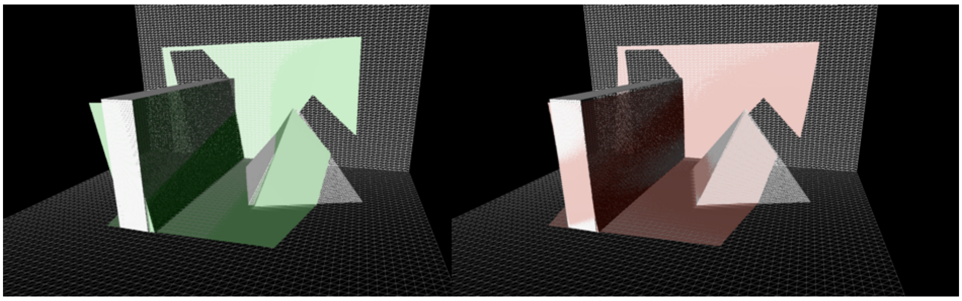


Figure 7.22: The ground truth and rectified model of Case 4.

Figure 7.23 indicates the estimated rotational angle and the ground truth. The difference between the estimated angular speed and the ground truth is 15.4 %.

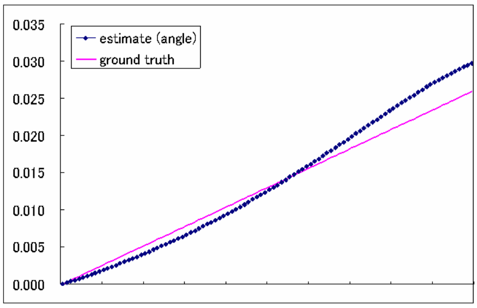


Figure 7.23: The camera path and the ground truth in Case 4.

Figure 7.24 shows the all motion parameters. It is found that the estimated position is moving, especially with respect to the *x* component, although all parameters are supposed not to change.

Table 7.1 shows the errors by the method without images in all cases. These values are also mean errors by point-to-patch distance. Overall, the method with images is superior to the method without images in accuracy. This table shows the worst result is obtained in Case 2, which has a rapid sensor motion, and the accuracy in the practical case is about 10 cm. On the other hand, the accuracy of other test case results, especially in Case 1 and 4, are the same level as those by the

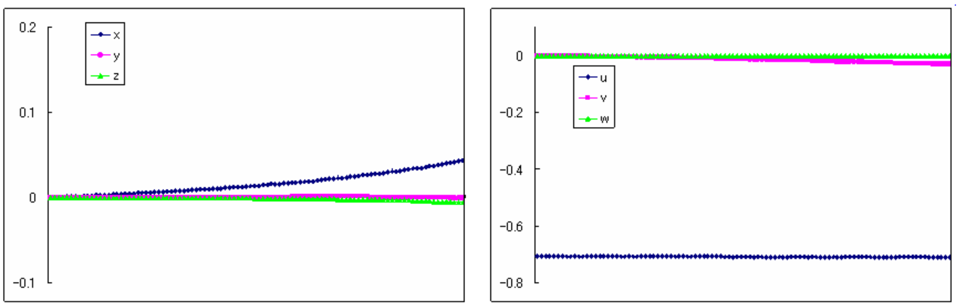


Figure 7.24: The all camera parameters in Case 4.

method with images. This means that the method without images is effective in the case of the sensor motion only with rotation.

Table 7.2: The mean errors of the method without images.

| | case1 | case2 | case3 | case4 |
|---|---|---|---|---|
| Before Rectification | 0.0134202 | 0.066315 | 0.0310331 | 0.0458285 |
| After Rectification | 0.00556148 | 0.014278 | 0.00889398 | 0.00508361 |

Finally, we have used the complete model as the ground truth in this section. In practical cases, it is expected that a correct shape will have many missing parts and that we have to rectify the distorted shape based on an incomplete reference. We are going to demonstrate such cases in the following chapter.

# Chapter 8

# Experiments

We have been conducting the ”Digital Bayon Project”, in which the geometric and photometric information on the Bayon Temple is preserved in digital form. With respect to the acquisition of the geometric data, large parts of the temple visible from the ground are scanned by range sensors placed on the ground. On the other hand, some parts invisible from the ground, for example, roofs and tops of towers, are scanned by our FLRS system.

## 8.1 Shape Rectification with Images

### Case1:

Figure 8.1 shows a sample image of the sequence of Case1.

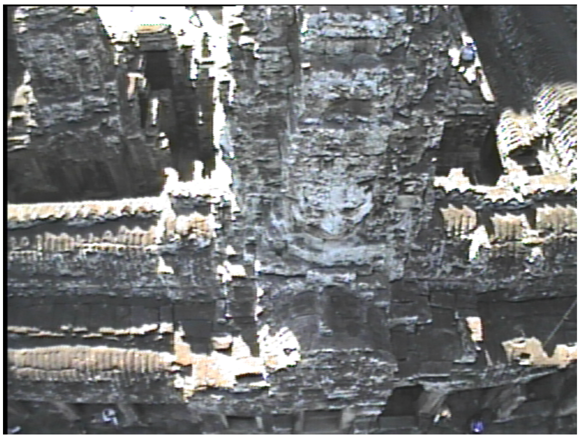


Figure 8.1: A sample shot of the image sequence

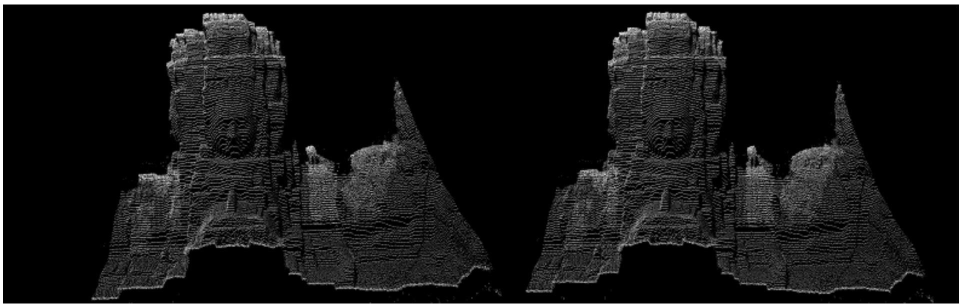


Figure 8.2: The original distorted shape (left) and the rectified shape (right).

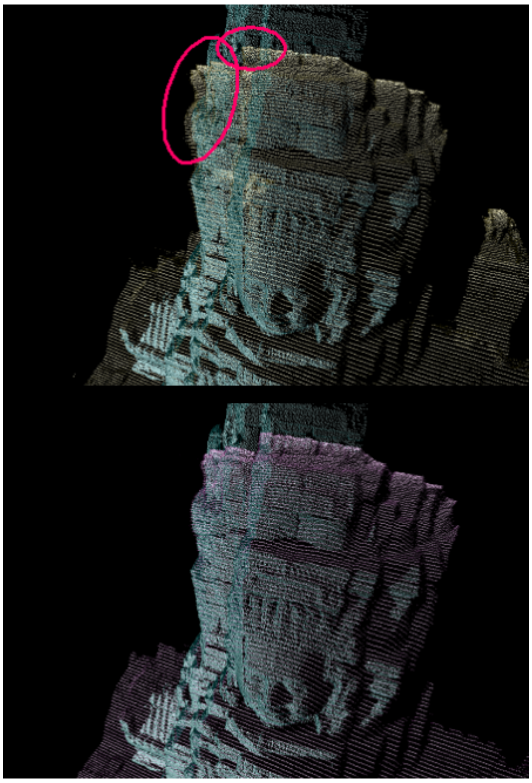


Figure 8.3: Range data before and after the rectification process: the upper figure shows the original distorted shape by the FLRS (white) and the correct shape obtained by the Cyrax-2500 fixed on the ground (blue). The lower figure shows the rectified shape (pink) fitted onto the correct one.

In Fig.8.2, the left figure shows the original shape obtained by the FLRS while the right figure shows the rectified shape by our method.

To evaluate the accuracy of our shape rectification algorithm, we compare the rectified shape with other data, which are obtained by a range finder, the Cyrax-2500 [Lei] [1], positioned on the ground. Aligning two data sets by using the conventional ICP algorithm [BM92] [CM92], we analyze the overlapping area.

The result is shown in Figure 8.3. The fine blue shape in both images is a non-distorted data (the correct data) obtained by the Cyrax-2500. The coarse white shape in the upper figure indicates the original distorted shape obtained by the FLRS, while the pink shape in the lower figure indicates the one rectified by our method. One can easily find that the rectified 3D shape is well-fitted onto the correct shape. In particular, taking notice of the area of ellipses in the upper figure, makes it obvious that our algorithm is effective.

The cross-section, cut off at the forehead of the statue, also shows the effectiveness (8.4).

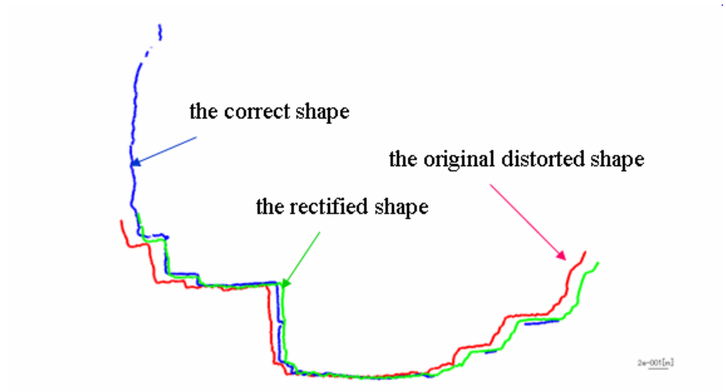


Figure 8.4: The figure shows the cross section at the forehead of the statue.

Figure 8.5 also shows the effectiveness of the method. The figures indicate the point-to-point distances between the correct data and the rectified data. The left image shows a comparison between the correct and the original distorted shapes, while the right shows a comparison between the correct and the rectified shape. The region where the distances between them are less than 6.0 cm is colored green [2]. The area where the distances are further than 6.0 cm is displayed in blue. At a glance, the green region is clearly expanded by the rectification algorithm. Some parts of the rectified shape are colored blue because of the lack of corresponding

[1] Now, Cyrax-2500 scanner is re-labeled as HDS2500 in Leica Geosystems

[2] In the previous chapter, we have approximated the accuracy in the practical case as 3.0cm. Therefore, we set the threshold as 6.0cm, twice of the estimated error.

points. Taking account of the fact the correct shape could not measure the parts invisible from the ground, the method could rectify the 3D shape correctly.

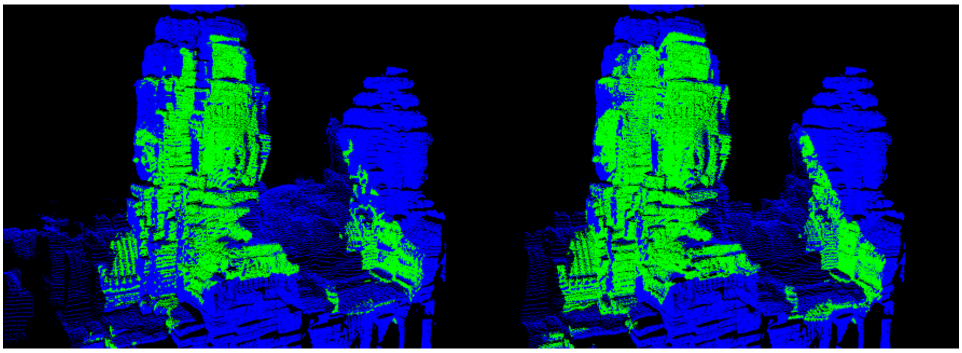


Figure 8.5: The comparison between the Cyrax-2500's (the correct data) and the original distorted data (left), and that between the Cyrax-2500's and the rectified data (right): the green region indicates where the distance of two shapes is less than 6.0 cm.

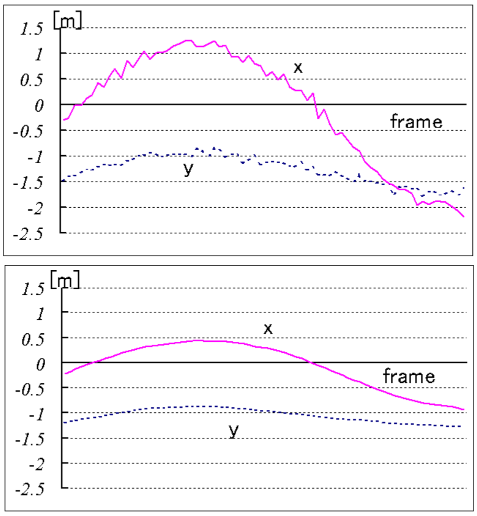


Figure 8.6: The trace of the camera translation: The curves in the upper graph show the parameters (x, y in the world coordinates) obtained by the perspective factorization. The lower graph shows the result of the refinement, in which the camera motion becomes smooth.

Here, to verify the effect of the refinement, we show the trajectory of camera motion parameters. The upper graph of Figure 8.6 shows the trace of the camera's translation obtained by the perspective factorization, while the lower graph shows the results after the refinement.

The curves in the upper graph appear to be globally probable for a camera movement. However, one can see that the they are not smooth locally, and therefore are unacceptable for the motion of a balloon. On the other hand, the curves in the lower graph are smooth and acceptable for the balloon motion.

### Case2:

Figure 8.7 shows a sample image of the sequence of Case2.

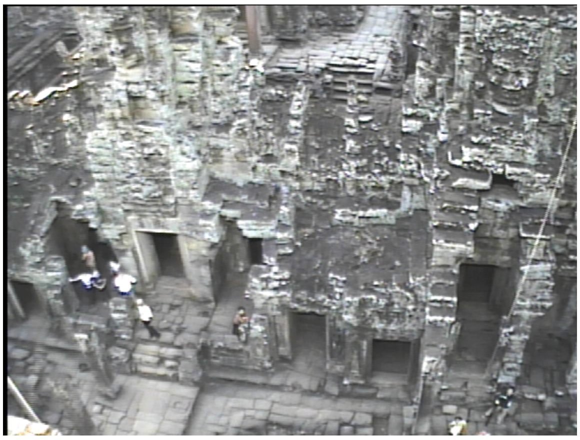


Figure 8.7: A sample shot of the image sequence

Figure 8.8 shows a photo picture of the scanned area. On the right side of Fig.8.8, the dense fine model is the correct shape obtained by the Cyrax-2500 fixed on the ground.

The result is shown in Fig.8.9. The upper shape in Fig.8.9 is the original one obtained from the FLRS. It is found that the shape is widely deformed. In the middle of Fig.8.9, the rectified shape by full-perspective factorization is shown. With respect to motion parameters, the ambiguity in scale is removed manually. At a glance, the factorization seems to rectify the shape properly. In detail, however, the distortion in S shape is still left. Especially, the shape of the entrance is skewed. On the other hand, the lower shape is rectified correctly by our method. It is clear that the distortion in S shape is removed and the shape of the entrance is correctly recovered into a rectangle.

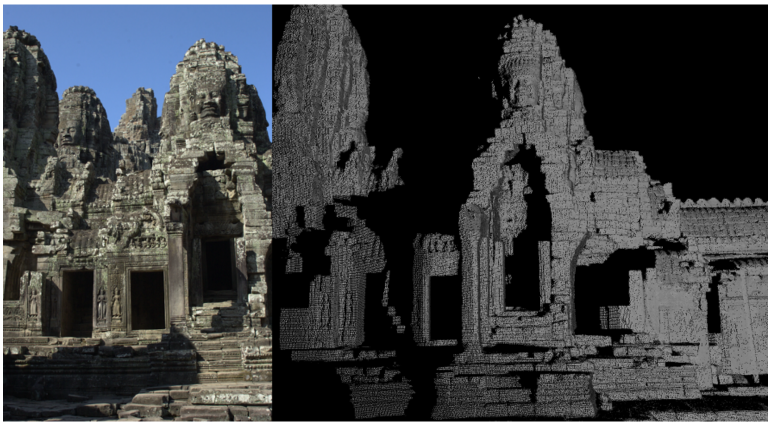


Figure 8.8: A scene for Case 2

Figure 8.10 indicates the point-to-point distances in the ICP algorithm, similar to Case 1. The upper figure shows the comparison between the correct shape and the original distorted one obtained by the FLRS. The middle one shows the rectified shape by the full-perspective factorization without ambiguity in scale. The lower shows the rectified shape by our method. Note that the green region (match region) is expanded by our method.

The upper graphs of Fig.8.11 shows the trace of camera's translation obtained by full-perspective factorization, while the middle one in Fig.8.11(b) shows the results after the refinement. The convergence in the factorization was not very good, therefore, the curves in the upper graph of Fig.8.11 have jagged shapes, which are not acceptable for the balloon's motion. On the other hand, the curves in the middle graph are smooth and acceptable. This result shows that our refinement is effective in lessening the camera motion and leads to the correct motion estimation. The lower graph of Fig.8.11 shows the estimated path by an accelerometer for reference [3].

Figure 8.12 shows several samples of the method with images.

[3] The output of the accelerometer does not possess so higher reliability.

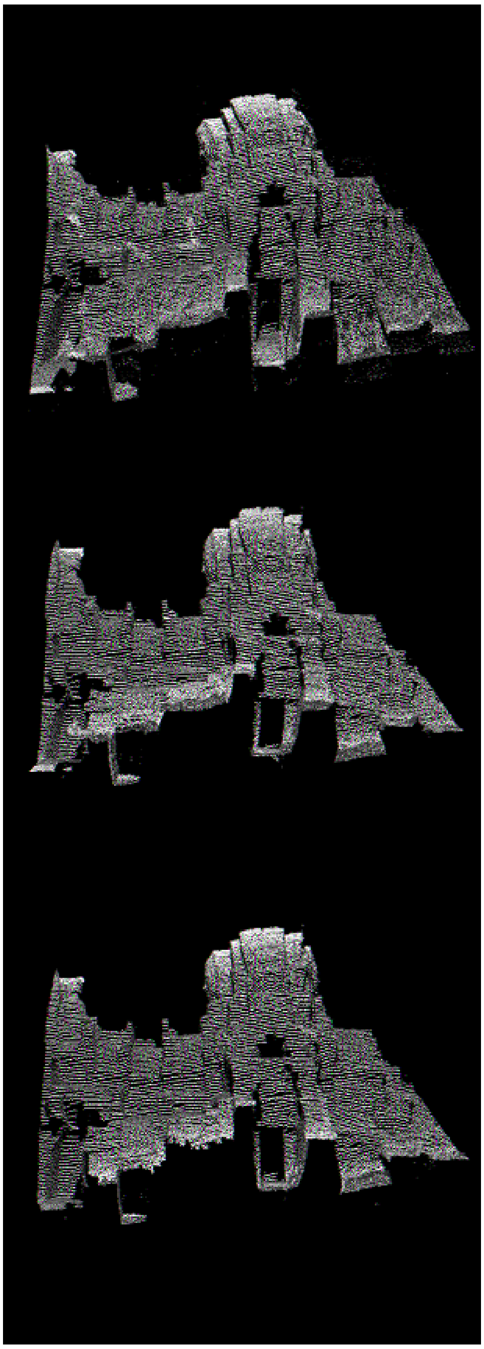


Figure 8.9: The upper figure shows the original distorted shape obtained by the FLRS. The middle one shows the rectified shape by the full-perspective factorization without ambiguity in scale. The lower shows the rectified shape by our method.

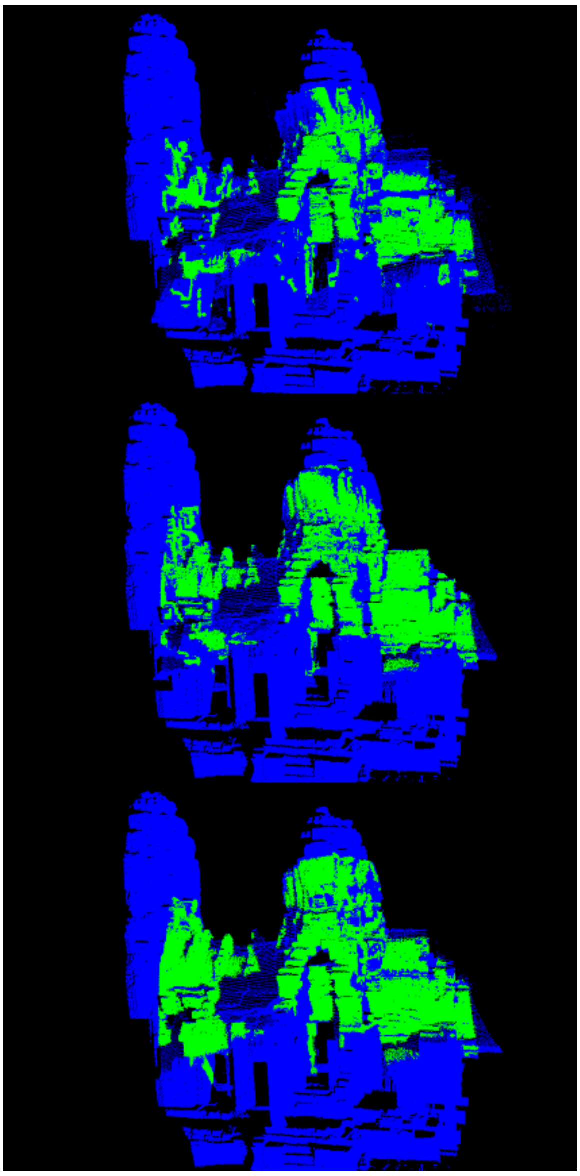


Figure 8.10: The upper figure shows the comparison between the correct shape and the original distorted one obtained by the FLRS. The middle one shows the rectified shape by the full-perspective factorization without ambiguity in scale. The lower shows the rectified shape by our method.

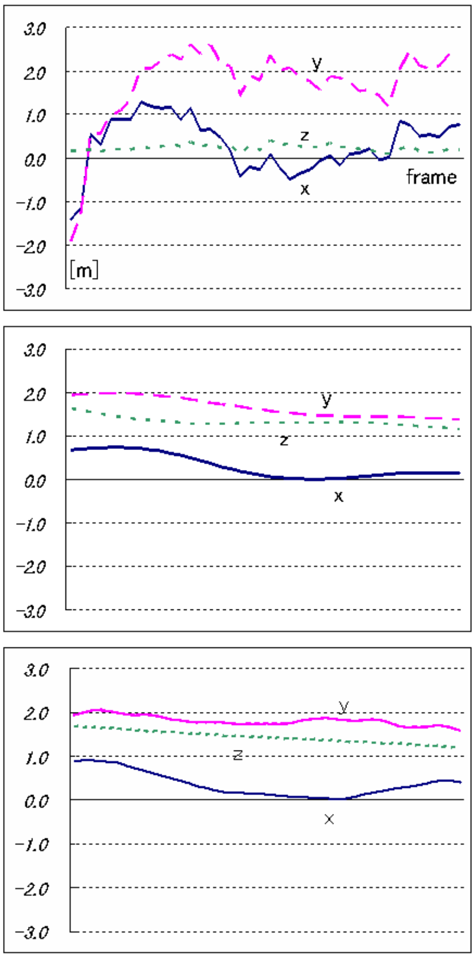


Figure 8.11: The trace of the camera translation: The curves show the parameters(x, y in the world coordinate) estimated by the perspective factorization(a), and by our proposed method(b). In (b), the camera motion becomes smooth and valid.

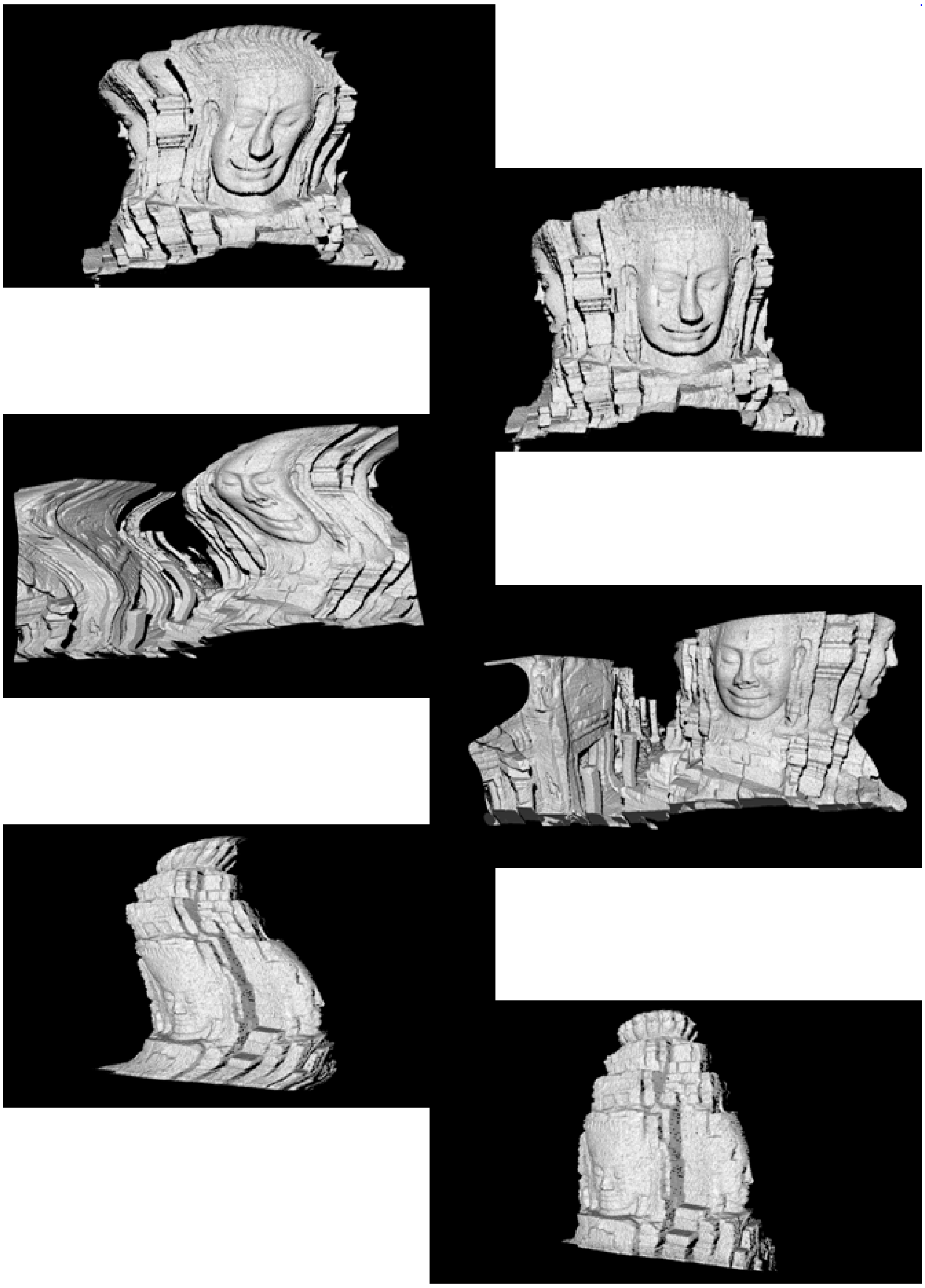


Figure 8.12: The original distorted data sets (left) and the rectified sets (right)

## 8.2 Shape Rectification without Images

We also applied the method without images to the real data set. As the reference shape, we utilize the shape obtained by the Cyrax-2500. There are some blank parts in the reference shape because there are no data set on the part that is invisible from the ground.

Figure 8.13: A sample shot in this case.

In Fig. 8.14, the left figure shows the original shape obtained by the FLRS while the right one shows the rectified shape by our method.

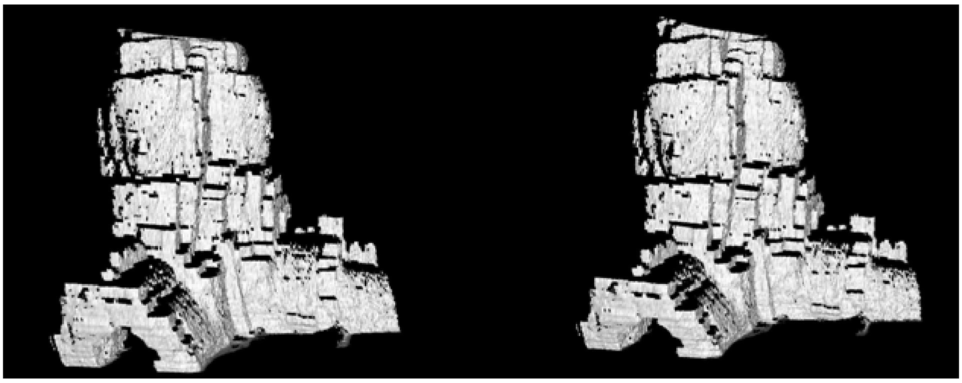


Figure 8.14: The original distorted shape (left) and the rectified shape (right).

Figure 8.15 shows the comparison between the reference shape. The upper figure shows the original distorted shape by the FLRS (white) and the reference shape (blue). The lower figure shows the recovered shape in the lower figure (pink) and the reference one. It is found that the rectified 3D shape is well-fitted onto the reference one, particularly the area of ellipses in the upper figure, in spite of the blanks on the reference shape.

Similarly, we show the trajectory of camera motion parameters. The upper graph of Figure 8.16 shows the trace of the translation parameters, while the lower graph shows the trace of three elements (*u*, *v* and *w*) of the quaternion which represents the camera's pose.

Finally, Figure 8.17 shows several samples of the method without images.

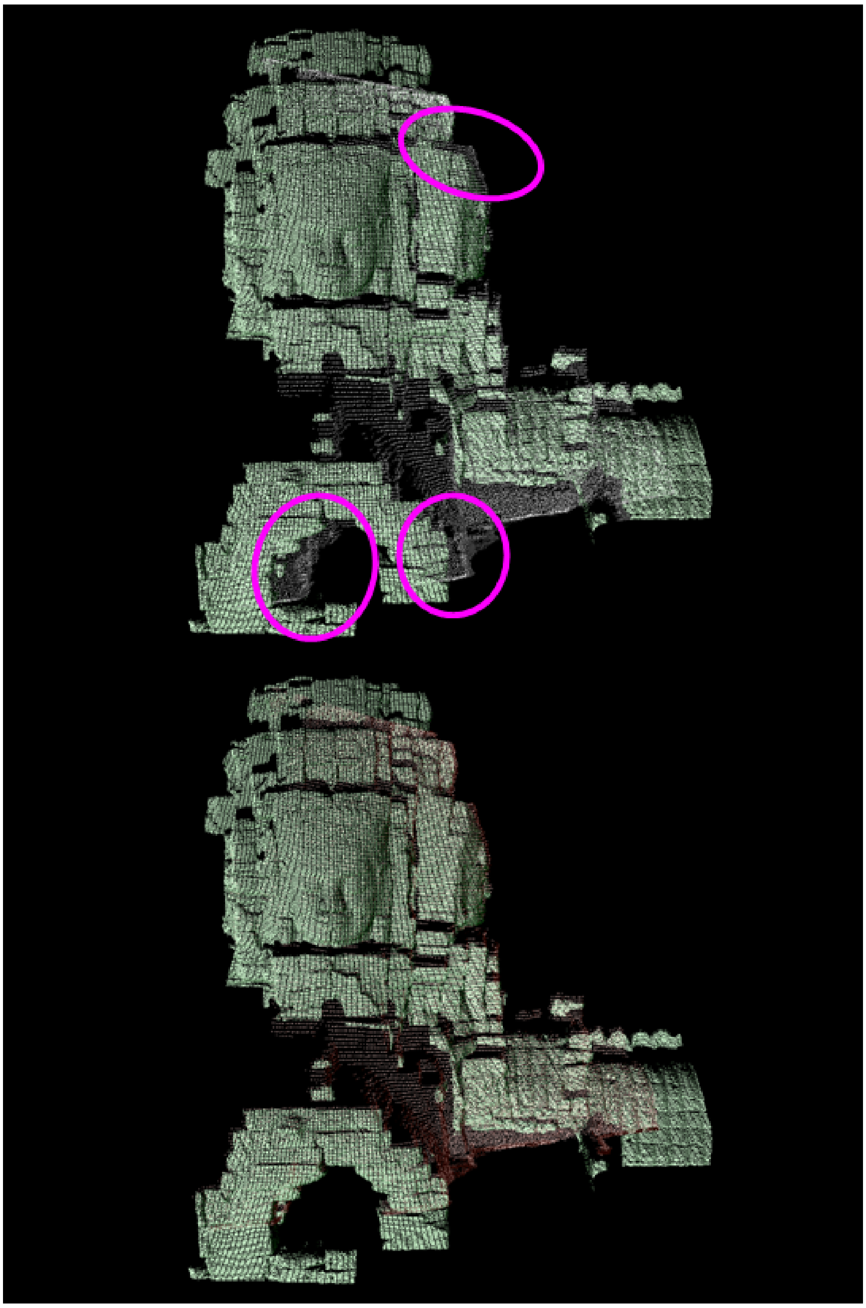


Figure 8.15: Range data before and after the rectification method without images: the upper figure shows the original distorted shape by the FLRS (white) and the reference shape obtained by the Cyrax-2500 fixed on the ground(blue). The lower figure shows the recovered shape in the lower figure (pink) fitted onto the correct one.

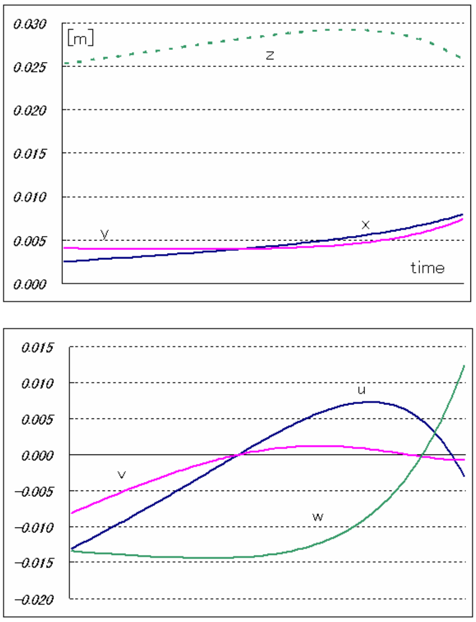


Figure 8.16: The trace of the camera translations and poses: the curves in the upper graph show the translational parameters(x, y and z in the world coordinate), those in the lower graph show the 3 components of the quaternion estimated by the method without images.

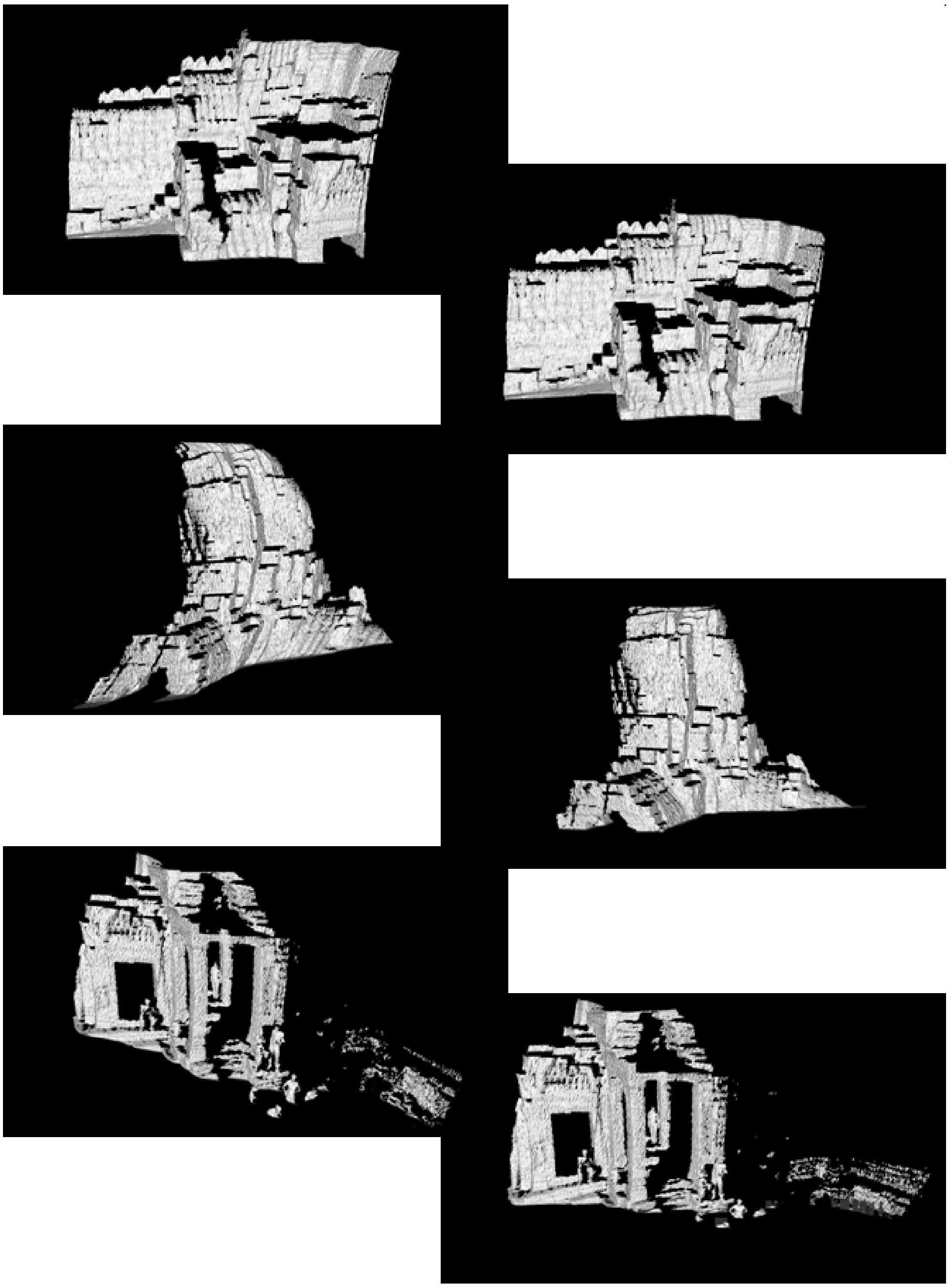


Figure 8.17: The original distorted data sets (left) and the rectified sets (right)

# Chapter 9

# Conclusions

## 9.1 Conclusions

In this thesis, we have described FLRS system and two proposed methods to rectify 3D range data obtained by a moving laser range sensor.

We described how an outstanding measurement system FLRS was built to scan large objects from the air. This system allowed us to measure the large cultural heritage objects by using a balloon. To rectify the distorted shapes obtained from the FLRS, we proposed two methods:

- The rectification method based on the ”Structure from Motion” techniques by using image sequences

- The rectification method based on the extended ICP algorithm by using another data set

In the first case, we described a method based on ”Structure from Motion”. We utilized distorted range data obtained by a moving range sensor and image sequences obtained by a video camera mounted on the FLRS. First, the motion of the FLRS was estimated through full perspective factorization only by the obtained image sequences. Then the more refined parameters were estimated based on an optimization imposing three constraints: the tracking, smoothness and range data constraints. Finally, refined camera motion parameters rectified the distorted range data. For this method, while the calibrated range sensor and camera system was originally assumed, we indicated that the method is also applicable to the uncalibrated system.

In the second case, we proposed an extended ICP algorithm without using any images. Assuming that the motions of the sensor are smooth, we applied them to polynomials. Then, we rectified the distorted range data based on the correct model obtained by other sensors fixed on the ground.

Both methods have shown proper performance and practical utilities. The experiments have shown that the distorted shapes can be rectified with the utmost precision when the images are available. On the other hand, we found that the second method has properly rectified the dataset of only rotation, which cannot be rectified by the first method.

These methods can be generally applied to a framework in which a range sensor moves during the scanning process, and is not limited to our FLRS because we impose only the smooth movement constraint.

## 9.2 Future Works

We have mentioned some methods which rectify distorted range data obtained by a moving range sensor. Originally, we developed these methods in the process of digital archiving of cultural heritage objects.

We point up some future works based on two aspects: hardware and software.

### Hardware

For hardware, there are some improvements we have to make. One of them is the reduction in size and weight of the system. The current weight of the FLRS is about 50 *kg* and we need to lighten it. The weight makes the practical measurement massive. We will attempt to lighten the FLRS so that we can easily scan large objects. In the future, we would also like to utilize some handy device instead of a balloon. We aspire to a hand-held measurement system as our ultimate goal.

### Software

Also, there are a lot of works to do in the future on the software in our system. First, we have to improve the accuracy of rectified shapes by our algorithm. The burning issue is the improvement of the accuracy of the method without images. We want to boost it to the same level as that of using the method with images.

Besides accuracy, there are a few challenging problems in the rectification algorithm without images. Currently, we use a single distorted shape and a single correct shape. As the next step, we are trying to rectify several distorted shapes at the same time by using a single correct shape. Moreover, we plan to rectify and register multi-distorted shapes simultaneously without any correct shapes. We envision a rectification method that utilizes both images and the correct models.

The framework of a moving range sensor during the scanning process is just beginning to be applied to practical missions. We have several scanning missions and many problems in practical scenes. However, we fully expect to overcome these difficulties with ”task-oriented vision”.

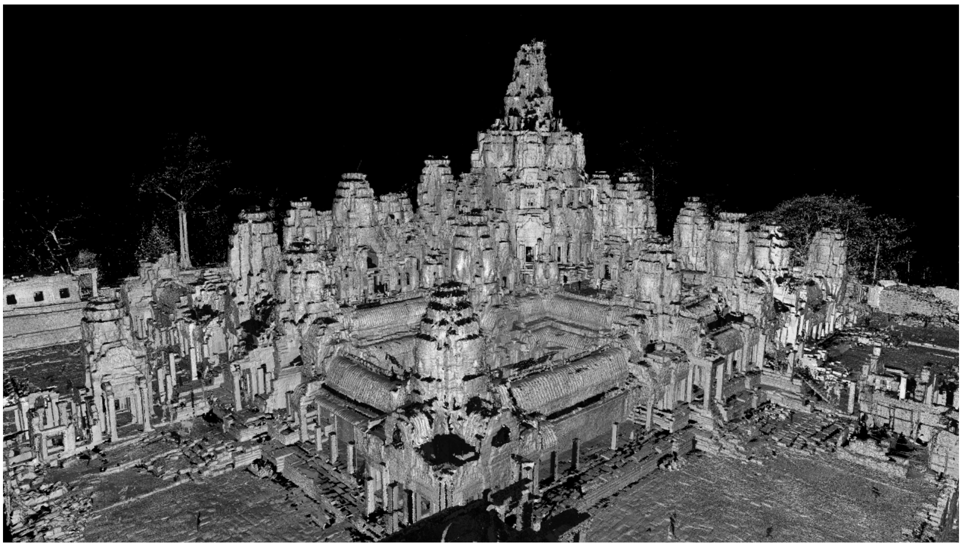


Figure 9.1: The Overview of the ”Digital Bayon”.

# Appendix A

# Solving for the Symmetric Matrix $T$ in the Factorization

We have introduces the cost function (3.40) to estimate the symmetric matrix $T$.

$$\begin{aligned} G &= \sum_{f=1}^{F} \Big( \left( |\vec{m'_f}|^2 - |\vec{n'_f}|^2 \right)^2 + w \left(\vec{m'_f}^t \cdot \vec{n'_f}\right)^2 \Big) \\ &= \sum_{f=1}^{F} \Big( \left( \vec{m'_f}^t T \vec{m'_f} - \vec{n'_f}^t T \vec{n'_f} \right)^2 + w \left(\vec{m'_f}^t T \vec{n'_f}\right)^2 \Big) \qquad \text{(A.1)} \end{aligned}$$

In this section, we show a method for solving $T$.

First, all elements of $T$ are set as follows:

$$T = \begin{pmatrix} T_1 & T_2 & T_3 \\ T_2 & T_4 & T_5 \\ T_3 & T_5 & T_6 \end{pmatrix} \qquad \text{(A.2)}$$

Supposing $\vec{m'_f} = (m'_{fx}, m'_{fy}, m'_{fz})^t$ and $\vec{n'_f} = (n'_{fx}, n'_{fy}, n'_{fz})^t$, we obtain

$$\begin{aligned} \vec{m'_f}^t T \vec{m'_f} &= (m'_{fx})^2 T_1 + 2m'_{fx}m'_{fy}T_2 + 2m'_{fx}m'_{fz}T_3 \\ &\quad +(m'_{fy})^2 T_4 + 2m'_{fy}m'_{fz}T_5 + (m'_{fz})^2 T_6 \qquad \text{(A.3)} \\ \vec{n'_f}^t T \vec{n'_f} &= (n'_{fx})^2 T_1 + 2n'_{fx}n'_{fy}T_2 + 2n'_{fx}n'_{fz}T_3 \\ &\quad +(n'_{fy})^2 T_4 + 2n'_{fy}n'_{fz}T_5 + (n'_{fz})^2 T_6 \qquad \text{(A.4)} \\ \vec{m'_f}^t T \vec{n'_f} &= m'_{fx}n'_{fx}T_1 + (m'_{fx}n'_{fy} + m'_{fy}n'_{fx})T_2 \\ &\quad +(m'_{fx}n'_{fz} + m'_{fz}n'_{fx})T_3 + m'_{fy}n'_{fy}T_4 \\ &\quad +(m'_{fy}n'_{fz} + m'_{fz}n'_{fy})T_5 + m'_{fz}n'_{fz}T_6 \qquad \text{(A.5)} \end{aligned}$$

Then the function (A.1) is rewritten as

$$\begin{aligned} G &= \sum_{f=1}^{F}\Bigg[\bigg\{\Big((m'_{fx})^2-(n'_{fx})^2\Big)T_1+2\Big(m'_{fx}m'_{fy}-n'_{fx}n'_{fy}\Big)T_2 \\ &\qquad +2\Big(m'_{fx}m'_{fz}-n'_{fx}n'_{fz}\Big)T_3+\Big((m'_{fy})^2-(n'_{fy})^2\Big)T_4 \\ &\qquad +2\Big(m'_{fy}m'_{fz}-n'_{fy}n'_{fz}\Big)T_5+\Big((m'_{fz})^2-(n'_{fz})^2\Big)T_6\bigg\}^2 \\ &\quad +w\,\Big\{m'_{fx}n'_{fx}T_1+(m'_{fx}n'_{fy}+m'_{fy}n'_{fx})T_2 \\ &\qquad\qquad +(m'_{fx}n'_{fz}+m'_{fz}n'_{fx})T_3+m'_{fy}n'_{fy}T_4 \\ &\qquad\qquad +(m'_{fy}n'_{fz}+m'_{fz}n'_{fy})T_5+m'_{fz}n'_{fz}T_6\Big\}^2\Bigg] \end{aligned} \tag{A.6}$$

The above function can be simplified by some replacements.

$$\begin{aligned} G &= \sum_{f=1}^{F}\Big\{(C_{1f}T_1+C_{2f}T_2+\cdots+C_{6f}T_6)^2 \\ &\qquad +w\,(D_{1f}T_1+D_{2f}T_2+\cdots+D_{6f}T_6)^2\Big\} \end{aligned} \tag{A.7}$$

To minimize the function $G$, we can derive the next 6 constraints.

$$\begin{aligned} \frac{\partial G}{\partial T_1} &= \sum_{f=1}^{F}\Big\{2C_{1f}\,(C_{1f}T_1+C_{2f}T_2+\cdots+C_{6f}T_6) \\ &\qquad +2w\,D_{1f}(D_{1f}T_1+D_{2f}T_2+\cdots+D_{6f}T_6)\Big\} \\ &= 2\sum_{f=1}^{F}\Big\{C_{1f}\,(C_{kf}T_k)+w\,D_{1f}(D_{kf}T_k)\Big\}=0 \\ \frac{\partial G}{\partial T_2} &= 0 \\ &\vdots \\ \frac{\partial G}{\partial T_6} &= 0 \end{aligned}$$

They are summarized in a matrix form as follows (divided by 2):

$$\left(\sum_{f=1}^{F}[C_{if}C_{jf}+wD_{if}D_{jf}]_{ij}\right)\begin{pmatrix}T_1\\T_2\\T_3\\T_4\\T_5\\T_6\end{pmatrix}=\begin{pmatrix}0\\0\\0\\0\\0\\0\end{pmatrix} \tag{A.8}$$

Therefore, the unknown vector $\vec{T} = (T_1, T_2, T_3, T_4, T_5, T_6)^t$ is calculated as the null space of the $6 \times 6$ matrix $CD = \left(\sum_{f=1}^{F} [C_{if}C_{jf} + wD_{if}D_{jf}]_{ij}\right)$. $\vec{T}$ corresponds to the eigenvector with the minimal eigenvalue of the matrix $(CD)^t(CD)$.

Referring to the replacements,

$$\left\{\begin{array}{lcl} C_{1f} & = & (m'_{fx})^2 - (n'_{fx})^2 \\ C_{2f} & = & 2\left(m'_{fx}m'_{fy} - n'_{fx}n'_{fy}\right) \\ C_{3f} & = & 2\left(m'_{fx}m'_{fz} - n'_{fx}n'_{fz}\right) \\ C_{4f} & = & (m'_{fy})^2 - (n'_{fy})^2 \\ C_{5f} & = & 2\left(m'_{fy}m'_{fz} - n'_{fy}n'_{fz}\right) \\ C_{6f} & = & (m'_{fz})^2 - (n'_{fz})^2 \\ D_{1f} & = & m'_{fx}n'_{fx} \\ D_{2f} & = & m'_{fx}n'_{fy} + m'_{fy}n'_{fx} \\ D_{3f} & = & m'_{fx}n'_{fz} + m'_{fz}n'_{fx} \\ D_{4f} & = & m'_{fy}n'_{fy} \\ D_{5f} & = & m'_{fy}n'_{fz} + m'_{fz}n'_{fy} \\ D_{6f} & = & m'_{fz}n'_{fz} \end{array}\right. \tag{A.9}$$

# Appendix B

# Quaternion

In this thesis, we represent a rotational matrix with a unit quaternion and we have to calculate the derivative with respect to its parameters. We explain quaternion in this section, which can describe rotation and its derivative with respect to the quaternion parameters.

Quaternion has the following parameters:

$$\mathbf{q} = (s, u, v, w) \tag{B.1}$$

Geometrically, when an object is rotated $\theta$ around axis $\vec{u}$, these parameters have the following meaning.

$$s = \cos\frac{\theta}{2} \tag{B.2}$$

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \sin\frac{\theta}{2}\vec{u} \tag{B.3}$$

therefore,

$$s^2 + u^2 + v^2 + w^2 = \cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2}||\vec{u}||^2 = \cos^2\frac{\theta}{2} + \sin^2\frac{\theta}{2} = 1 \tag{B.4}$$

Thus, quaternion is interpreted as a combination of a scalar and a 3-dimensional vector. Here, we explain basis operations of a quaternion. Suppose 2 quaternion as $\mathbf{p} = (a, \vec{\mathbf{u}}^t)^t$ and $\mathbf{q} = (b, \vec{\mathbf{v}}^t)^t$.

Addition (subtraction) of quaternions is defined as

$$\mathbf{p} + \mathbf{q} = \begin{pmatrix} a + b \\ \vec{\mathbf{u}} + \vec{\mathbf{v}} \end{pmatrix} \tag{B.5}$$

And the product is

$$\mathbf{pq} = \begin{pmatrix} ab - \vec{\mathbf{u}}^t \cdot \vec{\mathbf{v}} \\ a\vec{\mathbf{v}} + b\vec{\mathbf{u}} + \vec{\mathbf{u}} \times \vec{\mathbf{v}} \end{pmatrix} \tag{B.6}$$

In addition, the norm of a quaternion is defined as

$$|\mathbf{p}| = \sqrt{a^2 + \vec{\mathbf{u}}^2} \tag{B.7}$$

Conjugate and inverse quaternion of **q** is defined as follows, respectively

$$\mathbf{p}^* = \begin{pmatrix} a \\ -\vec{\mathbf{u}} \end{pmatrix} \tag{B.8}$$

$$\mathbf{p}^{-1} = \frac{\mathbf{p}^*}{|\mathbf{p}|^2} \tag{B.9}$$

In the case of the rotation of angle $\theta$ around axis $\vec{u}$, the rotated vector of $\vec{x}$ is generally described by using the quaternion **p** as

$$\begin{pmatrix} 0 \\ \vec{x}' \end{pmatrix} = \mathbf{p} \begin{pmatrix} 0 \\ \vec{x} \end{pmatrix} \mathbf{p}^{-1} \tag{B.10}$$

here, $\vec{x}'$ is the destination vector of $\vec{x}$.

In the case of the same rotation $\vec{x}' = R\vec{x}$, the rotational matrix $R$ is described by using these parameters as follow.

$$R = \begin{pmatrix} s^2 + u^2 - v^2 - w^2 & 2(uv + sw) & 2(uw - sv) \\ 2(uv - sw) & s^2 - u^2 + v^2 - w^2 & 2(vw + su) \\ 2(uw + sv) & 2(vw - su) & s^2 - u^2 - v^2 + w^2 \end{pmatrix} \tag{B.11}$$

While quaternion has 4 components, it is adequate to consider only 3 components since there are 3 independent variables. Based on Eq. B.4, we are to deal with the parameter $u$, $v$ and $w$. The parameter $s$ is an induced variable of $u$, $v$ and $w$, that means $s = s(u, v, w)$. Then, let us consider the first- and second-order derivatives of $s$ with respect to other parameters.

$$\begin{aligned} \frac{\partial s}{\partial u} &= \frac{\partial}{\partial u}\left(1 - u^2 - v^2 - v^2\right)^{\frac{1}{2}} = \frac{1}{2}(-2u)\left(1 - u^2 - v^2 - v^2\right)^{-\frac{1}{2}} \\ &= -\frac{u}{\sqrt{1 - u^2 - v^2 - v^2}} = -\frac{u}{s} \end{aligned} \tag{B.12}$$

$$\frac{\partial s^2}{\partial u} = \frac{\partial}{\partial u}\left(1 - u^2 - v^2 - v^2\right) = -2u \tag{B.13}$$

similarly,

$$\frac{\partial s}{\partial v} = -\frac{v}{s} \tag{B.14}$$

$$\frac{\partial s^2}{\partial u} = -2v \tag{B.15}$$

$$\frac{\partial s}{\partial w} = -\frac{w}{s} \tag{B.16}$$

$$\frac{\partial s^2}{\partial u} = -2w \tag{B.17}$$

Here, it is found that the derivatives of the rotational matrix $R$ with respect to $u$, $v$ and $w$ are written as follows:

$$\frac{\partial R}{\partial u} = \begin{pmatrix} 0 & 2(v - \frac{uw}{s}) & 2(w + \frac{uv}{s}) \\ 2(v + \frac{uw}{s}) & -4u & 2(s - \frac{u^2}{s}) \\ 2(w - \frac{uv}{s}) & 2(\frac{u^2}{s} - s) & -4u \end{pmatrix} \tag{B.18}$$

$$\frac{\partial R}{\partial v} = \begin{pmatrix} -4v & 2(u - \frac{vw}{s}) & 2(\frac{v^2}{s} - s) \\ 2(u + \frac{vw}{s}) & 0 & 2(w - \frac{uv}{s}) \\ 2(s - \frac{v^2}{s}) & 2(w + \frac{uv}{s}) & -4v \end{pmatrix} \tag{B.19}$$

$$\frac{\partial R}{\partial w} = \begin{pmatrix} -4w & 2(s - \frac{w^2}{s}) & 2(u + \frac{vw}{s}) \\ 2(\frac{w^2}{s} - s) & -4w & 2(v - \frac{uw}{s}) \\ 2(u - \frac{vw}{s}) & 2(v + \frac{uw}{s}) & 0 \end{pmatrix} \tag{B.20}$$

# Appendix C

# Removal of Specular with EPI

This appendix describes methods that removes specularities from image sequences taken by a video camera in a uniform straightly-line motion. Specular components, especially strong highlights, raise some problems in object recognition. We propose two methods to remove specular component based on spatio-temporal image analysis and to reconstruct original texture on the body as diffuse components. In the first method, analyzing the motion of specular components in EPIs (Epipolar Plane Images), we can distinguish specularities from ordinary texture. In the second method, by using a segmentation technique with Markov Random Field (MRF), we remove specular components. Some experiments have been conducted using our methods, and the results show the effectiveness of the method in removing specularities from image sequences. Even if the texture on a body is hidden by strong highlights, these methods recover the original texture.

## C.1 EPI (Epipolar Plane Image)

Image sequence is a collection of images taken at certain sampling interval. A box that consists of these images accumulating in time is a ”spatio-temporal volume” (Fig.C.1). When the sampling interval is enough dense or when the motion of the camera or the photogenic objects is slow, the spatio-temporal volume forms images with strong correlations on the cross-sections. The motion of the camera or the photogenic objects is detected by analyzing the cross-sections.

In this appendix, we suppose the situation where the camera moves in a uniform straight line to the direction parallel to the optical axis taking stationary objects (Fig.C.2).

Let us consider the horizontal cross-sections of a spatio-temporal volume. This

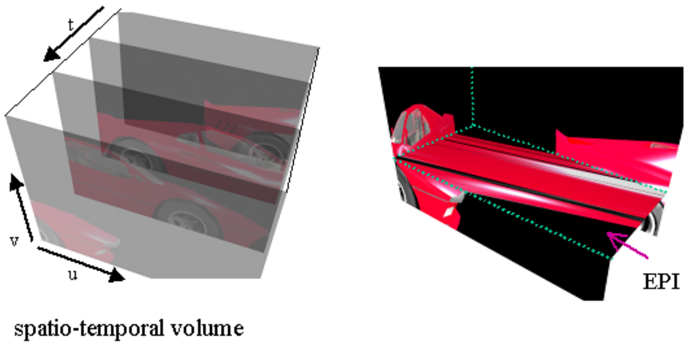


Figure C.1: The spatio-temporal volume and EPI.

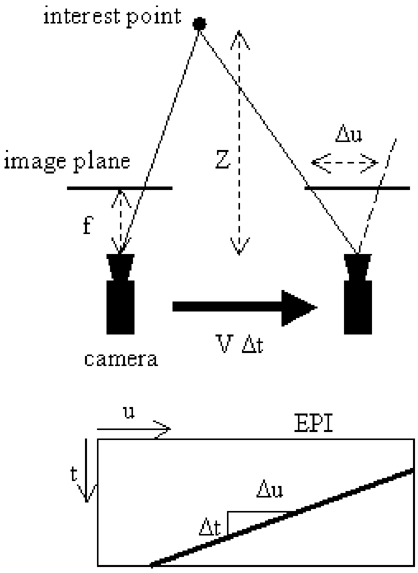


Figure C.2: EPI as temporal stereo vision

type of image is called an EPI (Epipolar Plane Image) [BBM87]. In an EPI, we can observe an interest point in space as a continuous trajectory. In our situation a camera in a uniform straightly line motion, the trajectory of a stationary point in space forms a line. In addition, a moving camera that is interrupted forms a stereopsis configuration with time difference. Therefore, the following relation exists between the depth of the 3D point and the slope of the trajectory in EPI:

$$\frac{\Delta u}{\Delta t} = f\frac{V}{Z} \tag{C.1}$$

Here, $f$ is the confocal length of the camera, $V$ is the velocity of the camera and $Z$ is the depth of the interest 3D point. From the above equation, it is easily found that the further the 3D point, the steeper the slope, and that the nearer the point, the gentler the slope in the constant velocity.

## C.2 Characteristics of specularity

A dichromatic reflection model is generally utilized for the description of an appearance by our visual perception. In dichromatic reflection, the model consists of two components: specular and diffuse component. While many reflection models based on the dichromatic model, such as Phone [Pho75] and Torrance-Sparrow [TS67] etc., have been proposed, they assume the next two points.

- The strength of the diffuse component is determined only by the incident angle, the angle between the normal vector of the object surface and the vector towards the light source (Fig. C.3). That means the strength of the diffuse component does not depend on the view position.
- The strength of the specular component is greatest when the incident angle equals the reflection angle. That means the strength of the specular component depends on the view position.

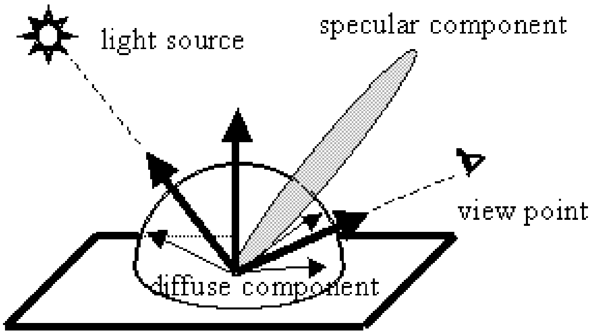


Figure C.3: Two reflectance components.

Based on the above considerations, we assume the situation where the objective scene and the light source remain stationary and the camera (view position) moves. Watching a certain point on the surface, we find that what changes during observation is the specular component. Therefore, when the view position is far away from the region of mirror reflection and the specular component is negligible, we observe only diffuse components.

## C.3 Removal of specular

### C.3.1 Removal by Line Search

In [SI94], they decomposed two components by observing each RGB values of each on the surface. Based on a similar consideration, in [SKS+02], they proposed a method to detect and remove strong highlight.

Strong highlight is interpreted as reflection of the light source on the object surface. Therefore, the observation of the highlight means the observation of the imaginary light source on the opposite side of the camera (Fig. C.4). That means the object is near and the imaginary light source is far from the camera. Consequently, an EPI shows that the trajectory of the object is steep and that the trajectory of the highlight is gentle. Generally, the trajectory of a far object is fragmented by that of a near object because of occlusions. The trajectory of highlight of a far object is, however, not fragmented. That is, the object far from the camera is not occluded by a near object. Thus, the EPI with specularity has inconsistency and we can determine that the slope without consistency is the trajectory of specularity.

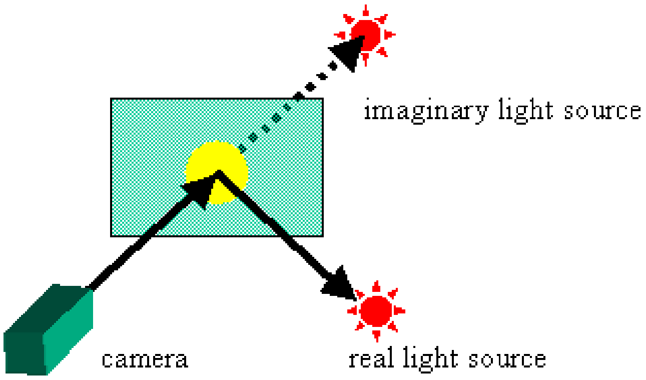


Figure C.4: Observing the specular components means observing the imaginary light source.

Based on the above considerations, specularity can be removed by the following procedure (Fig.C.5).

1. Derive EPIs at all cross sections from the spatio-temporal volume.

2. Extract the gentlest slope from at each EPI by using Canny operator and Hough transform. Then an affine transform rectifies the EPI by which the slop becomes vertical. In the rectified image, a vertical line means the trajectory of a certain point on the object surface.

3. Derive the minimum RGB values along a vertical line. The RGB values on a vertical point consist of specularity and constant diffuse component. The specular component is added to by viewpoint change. Therefore, without specular component, the RGB values equal the diffuse component. Then, we assume the minimum RGB values as the diffuse component in study.

4. Replace all RGB values along a vertical line with the minimum RGB values on the line. The inverse affine transformation unkinks the rectified image into the original shape.

### C.3.2 Removal by Image Segmentation

In EPIs, the edges of specularity regions are blurred and it is difficult to segment the specular regions by a region grow method. This means that a specular region is taken a part in the diffusion region. Then representing the minimum GRB in each segment leads to the specularity removal. Taking account of this property, we propose the next procedure.

1. Noise removal by using anisotropic diffusion.

2. Segmentation of color region by a region growing method.

3. Complement by using Markov Random Field (MRF).

First, the input EPIs are smoothened by an anisotropic diffusion over the color images [PM90] which preserves edges. Then the region growing method maps all pixels into the color space proposed in [OKS80].

$$\begin{cases} I_1 = \dfrac{R+G+B}{3} \\ I_2 = \dfrac{R-B}{2} \\ I_3 = \dfrac{2G-R-B}{4} \end{cases} \tag{C.2}$$

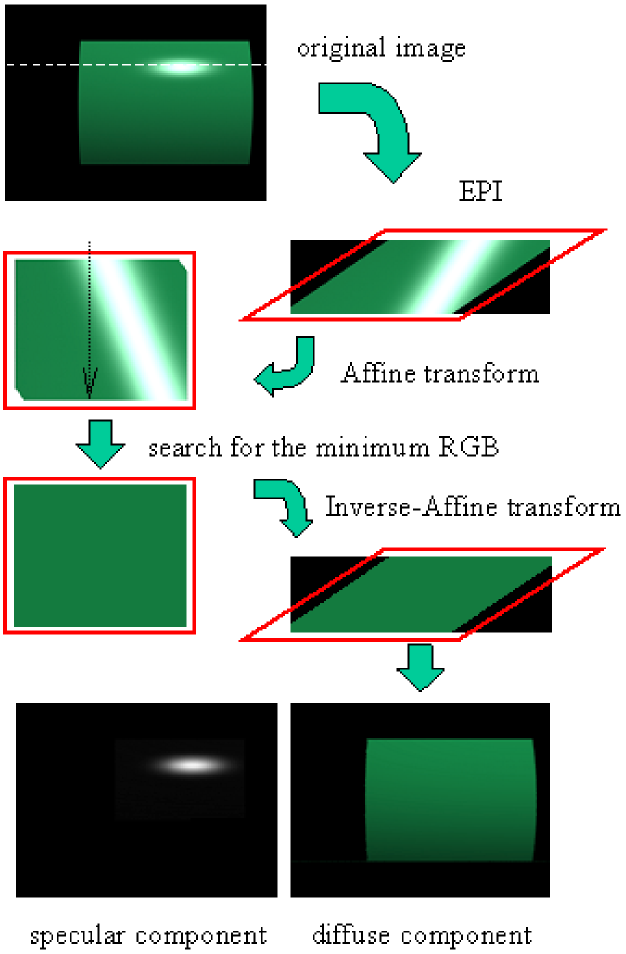


Figure C.5: Flowchart of the removal by EPI.

If the Euclidean distance in the color space between the adjacent pixels is less than a threshold, these two pixels are classified into the same label.

The minimum RGB values of each segment are representative of the labeled region. If the area of a segment is less than a threshold, the label is stripped because it might be noise. Large parts of pixels of EPIs are labeled and large parts of specularity are removed at this step. For the unlabeled regions, we complement them to label by using Markov Random Field [GG84] [Muk02].

Markov Random Field gives unlabeled pixels labels at random in order to minimize a local energy taking account into the adjacent pixels. Supposing there is an unlabeled pixel ”p” and the MRF gives it label ”1”, the local energy around the pixel is defined as follows based on the adjacent pixels ”q”.

$$U_p^l = \alpha \sum_q \rho(\mu_l, \mu_{\Theta(q)}) + \beta \rho(\mu_l, I(p)) \tag{C.3}$$

Here, $\mu_l$ represents the point of label ”l” at the color space and $\rho(\mu_l, \mu_k)$ is the Euclidean distance between label ”l” and label ”k” in the color space. $\Theta(q)$ means the label of the adjacent pixel q and $I(p)$ equals the original color of p. $\alpha$ and $\beta$ are parameters.

The Gibbs distribution of whole labeling configuration $\omega$ with temperature T is defined as follows:

$$\pi(\omega) = \frac{1}{Z} \exp\left(-\frac{U(\omega)}{T}\right) \tag{C.4}$$

When the local energy of each unlabeled pixel is decreased, the property of the whole configuration of labels is increase. In our algorithm, we use a simulated annealing technique to decrease the temperature T avoiding local minimums.

The entire algorithm of the complement by using MRF is as follows:

1. Given a high temperature $T$.
2. Select an unlabeled pixel p at random and assume label ”l”. ($l \in \Theta(q)$ : ”l” is the label of a adjacent pixel q)
3. Calculate the local energy $U_p^l$ around the pixel p.
4. If $U_p^l$ decreases, the pixel p is given label ”l”. On the other hand, if $U_l^p$ increases, the pixel p is given label ”l” with a probability of $\exp\left(\frac{-\Delta U_p^l}{T}\right)$.
5. Repeat step 2 ∼ 4 with a certain number iteration.
6. Decrease the temperature $T$ and repeat the above procedure until $T < T_{small}$.

There would be some unlabeled pixels in spite of the complement by using Markov Random Field. The original RGB values are given to these unlabeled pixels.

## C.4 Experiments

First, we show a result of a CG image(Fig.C.6). In the original image (the upper of Fig.C.6), we can find the reflection of the circular cylinder in the picture. By observing EPIs, the specular components are easily detected (the middle image) since the slope of the specular component differs from that of the diffuse components. Then, the specular-free image is obtained by line search for the minimum RGB (the lower).

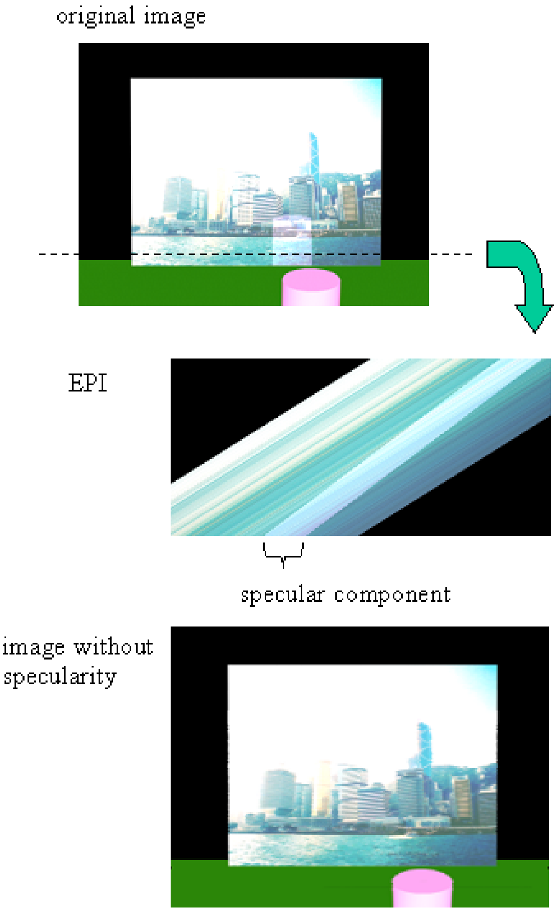


Figure C.6: Reults of a CG images by EPI.

We apply the line search method to real images. The left side images of Fig.C.7 are taken from a video camera on a moving car. There are some strong highlights on the surfaces of parked cars. We assume that the strong highlight exists on the nearest surface in pictures. The results are shown on the right side of Fig.C.7. As in the CG images, it is found that the specular components are removed in the case of real images. The textured surface which is occluded by the strong highlight (especially the letters on the surface of the taxi) is recovered.

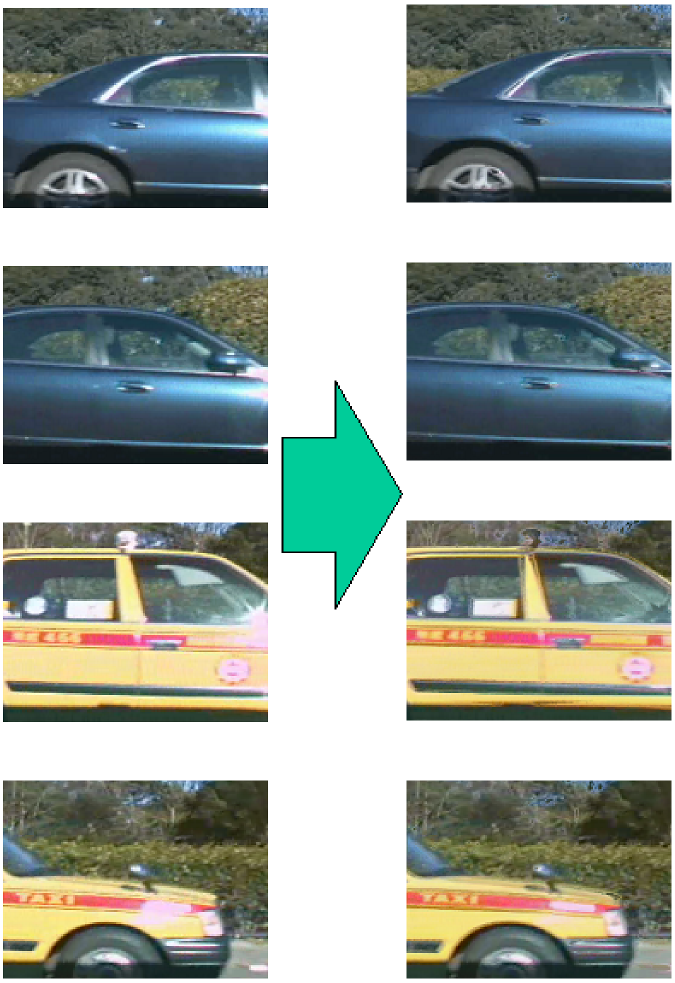


Figure C.7: Reults of real images by EPI.

Finally, we apply the method by image segmentation to CG images. Figure C.8 shows the results of segmentation at each step. Some specular components are found in the top left image of an original EPI. The right figure shows the EPI applied the anisotropic diffusion smoothing and the region growing segmentation. It is found that almost all specularities are removed, and there are many unlabeled pixels. The bottom left figure is the EPI after the complement by Markov Random Field; a great number of unlabeled pixels are given labels.

The results of specular removal are shown in Fig.C.9. The highlights on the pyramid and on the map are removed while some edges are blurred. It is confirmed that the recovered color of the pyramid corresponds to the ground truth.

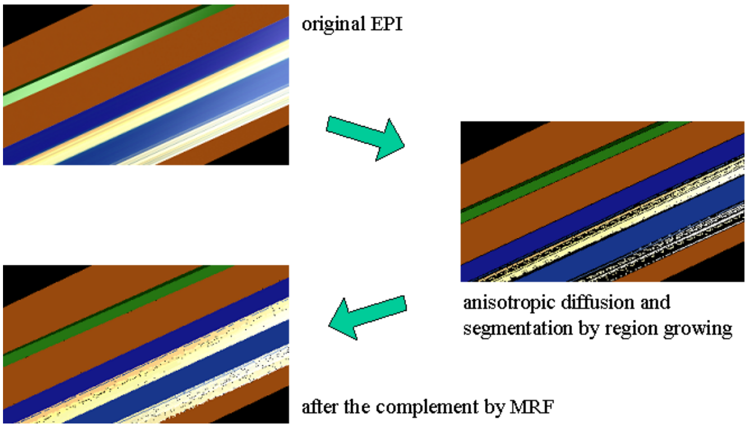

original EPI
anisotropic diffusion and
segmentation by region growing
after the complement by MRF

Figure C.8: Specular removal in EPI by segmentation.

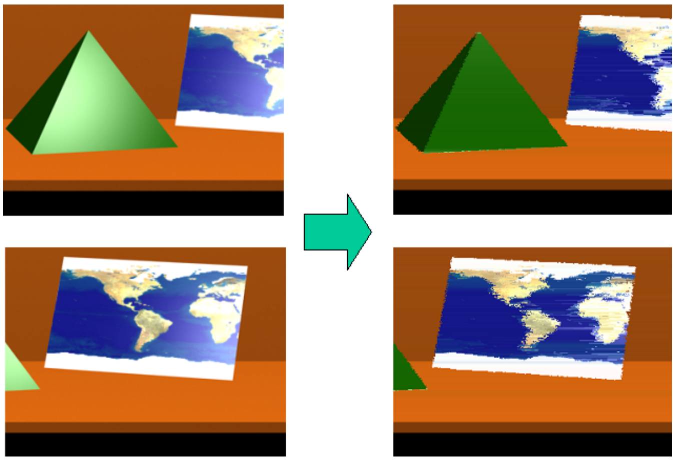


Figure C.9: Results of a CG image by Markov Random Field.

## C.5 Conclusions

We describe two methods which remove the specular component from image sequences taken by a video camera in uniform straightly-line motion. Both methods utilize EPIs to detect the diffuse components. In the first method, assuming that the specular components exist on the nearest surface to the camera, we derive the diffuse components along the gentlest slope in the EPIs. The results shows that our method has removed the specular components. Moreover it has recovered the original texture on the surface, which was occluded by the strong highlights. In the second method, based on the fact that the edges of the specularities are blurred, we remove the specular components by using image segmentation. The greatest advantage of the second method is that the method does not require the assumption of a camera in uniform straightly-line motion. We are, therefore, planning to apply this method to an image sequence taken by a camera in general motion. Moreover we intend to apply it to more complex scenes in the future.

# Appendix D

# 3D Identification of Fired Bullets

## D.1 Introduction

Many striation and impression marks caused by various ordinary tools, such as a screwdriver, a crowbar and a hummer, are left at crime scenes. These marks are significant evidences. In particular, striation marks on a fired bullet are important for identifying the suspicious firearm(Fig.D.1). Forensic scientists identify these striations mainly by using optical tools such as comparison microscopes, CCD cameras and photos. The surfaces of striation have three-dimensional roughness intrinsically. By using optical devices, we compare reflectance images instead of 3D shapes. Appearances of striations through these devices, however, depend on location of light and viewpoint [Leo97]. In other words, it is possible that the same striation has will look different under different lighting conditions. Besides these appearance-based methods [HL03], we are also able to exploit 3D geometric data of striations. That is model-based methods. The measurement of small elevations on a striation had been difficult in aspect of hard ware. However many sophisticated 3D measurement devices are developed recently and we can easily obtain fine 3D maps of striation surfaces. The shape of striation surface is expected to be printed intrinsic shapes of the tool that caused the striation marks. Moreover, 3D data are independent of lighting condition.

In addition, there is another difficulty in identifying striations. That is, a perfect correspondence of two striation patterns is rarely encountered, even if the two are on non-deformed bullets and have been fired from the same firearm(Fig.D.2). We, therefore, need an algorithm which is robust with respect to minute changes of patterns.

Although there are some researches on 3D surfaces of bullets and tool marks

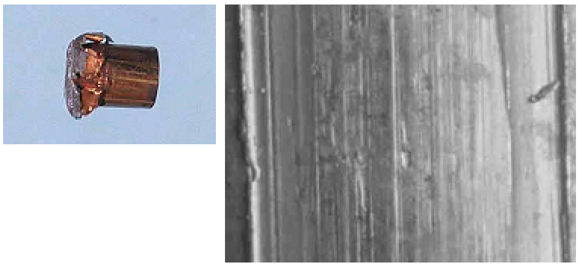


Figure D.1: A bullet and a striation mark

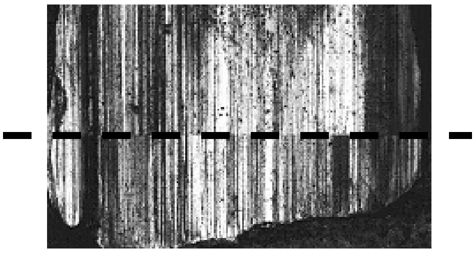


Figure D.2: An image by a comparison microscope. The correspondence of two striations is ”pretty” good.

[GZH+01] [KB99], they had not led to shape comparisons by using 3D surface data directly. In the field of Japanese archaeology, Masuda et al. [MIF+02] have analyzed shape difference of ancient bronze mirrors with a method of computer vision. In this study, we apply this method to identification of bullets, especially landmark impressions. Moreover, by using neural networks, we have developed a robust identification algorithm [BMI04]. Neural networks [RM86] are modeled after the structure of the human brain, and the human brain has an advantage over a computer in terms of pattern recognition [KC92]. In this study, neural networks have appeared to overcome minute changes of striation patterns.

At first, we acquired 3D data of striations surfaces and compared global 3D shapes numerically. The distance of two surfaces is calculated for the evaluation of global shape matching. Then neural networks compare local elevation patterns. This two-stage method enabled us to construct a robust identification algorithm of striation patterns.

## D.2 Global shape comparison

### D.2.1 Alignment of 3D data

We obtained 3D data of striations surfaces by a confocal microscope. To compare two shapes, we must move one shape in order to coincide two surfaces better. If the two striations are derived from the same origin, the shapes will be similar. Furthermore, if we could calculate the distance of the two shapes' difference, similarity of two shapes would be estimated according to the distance.

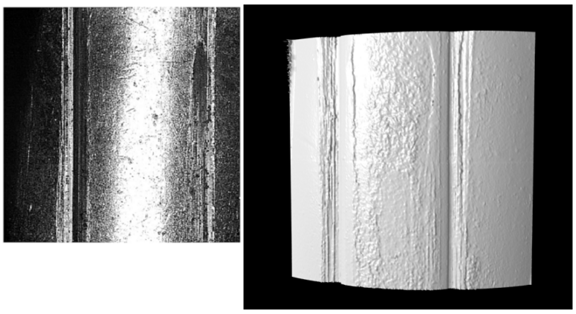


Figure D.3: A real striation mark and it 3-D model.

We adopted the alignment method [NI02], which is a kind of ICP method[BM92] for shape matching. If two shapes have the same origin, a point on one shape has the corresponding point on the other shape. The location of the corresponding point, however, is usually unknown. Then, we resolve this correspondence problem by iterative method. The objective function, which should be minimized for the alignment, is defined as:

$$f\left(R,\vec{t}\right)=\sum_{i,\,j}\|\,R\vec{x_i}+\vec{t}-\vec{y_{ij}}\,\|^2 \tag{D.1}$$

$R$ is a Rotation matrix, $\vec{t}$is a translation vector, $\vec{x_i}$ is the $i$-th point in one data and $\vec{y_{ij}}$is the corresponding $j$-th point in the another data for $\vec{x_i}$.

This objective function indicates the summation of distances between all pairs of corresponding points. When the function converges under a threshold, we decide two shapes are similar [Ban04].

We use quaternion to minimize the objective function. By substituting quaternion $\mathbf{q}$ to rotate matrix $R$, motion vector $\mathbf{p}$ can be found as follows.

$$\mathbf{q}=\arg\min_{R,\,\vec{t}}\ f\left(R,\vec{t}\right)=\arg\min_{\mathbf{q},\,\vec{t}}\ f\left(\mathbf{q},\vec{t}\right) \tag{D.2}$$

Motion vector **p**, that is **q** and $\vec{t}$, is solved by the conjugate gradient method and line minimization with golden section search. The solutions are the ones that minimize the objective function Eq.(D.1).

### D.2.2 Shape difference

Above alignment determines the relationship of corresponding points. Therefore, the distance between each pair of corresponding points can be calculated. We regard these distances between the corresponding points will be a cue of shape matching. If the distance of a pair is less than a threshold, the correspondence is regarded as right. Otherwise, the pair does not have correspondence, namely two shapes do not match at this part.

In terms of shape matching of two surfaces, wide region of non-matching indicates that two shapes are different.

## D.3 Local shape comparison

### D.3.1 Character extraction

The shape of a striation is usually uniform along the direction of the scratch. To input into neural networks, elevations on the surface should be converted into a binary signal. The method of binarization is simple(Fig.D.4); at first, gradients of all patches are calculated. Then, shapes of striation are converted into binary images by a threshold for these gradients. Finally, we derive a binary signal from a binary image by using morphological processings.

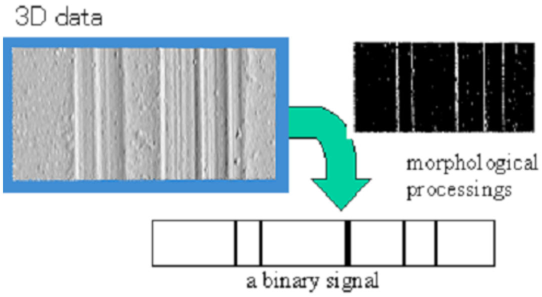


Figure D.4: The binarization method, which converts a surface shape into a binary signal.

### D.3.2 Neural network model

In this study, a multi-layer network that contains three layers is used(Fig.D.5). There are 96 neurons in the input layer, 15 neurons in the middle layer and only 1 neuron in the output layer. The neurons in the input layer are divided into two blocks: input blocks A and B. Each input block contains 48 neurons. There are two patterns to be compared in terms of their similarity. Two patterns are inputted into the two input blocks A and B separately.

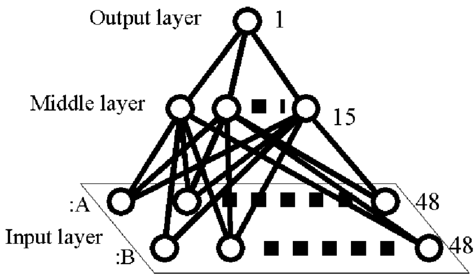


Figure D.5: The structure of the three-layer network model with two input block.

### D.3.3 Learning

Two training patterns to be compared are inputted into each block, which contains 48 neurons. The training patterns are binary signals with a 48-bit length. Each signal consists of only one element with a value of ”1” and forty-seven elements with a value of ”0”. Namely, in the learning process, only one neuron in each block has an input value of ”1” (this neuron is referred to as an ”excited neuron”), and the other 47 neurons in each block have an input value of ”0”. Supposing the *i*-th neuron of a block and the *j*-th neuron of the another block are excited, the teaching signal is given in the following form.

$$T(i,\ j) = \exp\left(-\frac{(i-j)^2}{\sigma^2}\right) \tag{D.3}$$

That is, if two patterns are the same, the output value of this network is ”1”. In addition, the closer together two positions of the excited neurons are, the closer to ”1” the output value will be. On the other hand, the further apart the two positions are, the closer to ”0” is the value.

## D.4 Experiments

### D.4.1 Shape difference

The shape difference is visualized according to the distances of corresponding pairs(Fig.D.6). If the distances are within a threshold (in this study, it is 0.015mm), the area is displayed in pink region. While the distances are further than it, the area is colored blue. In the left side of Fig.D.6, almost all part of overlapped region is colored pink. It indicates that the two shapes are matched well because two images in the left side are results of comparisons that compare two pairs from the same origins.

On the other hand, a result, which compares two shapes in different origin, is shown in the right side. Blue region are wider than in the left side. It indicates the number of corresponding pairs is fewer even in overlapped region. In addition, the shape of non-coincide region spreads out along the direction of the scratch (a blue region sandwiched between pink regions). This is an obvious feature when two shapes have different origins.

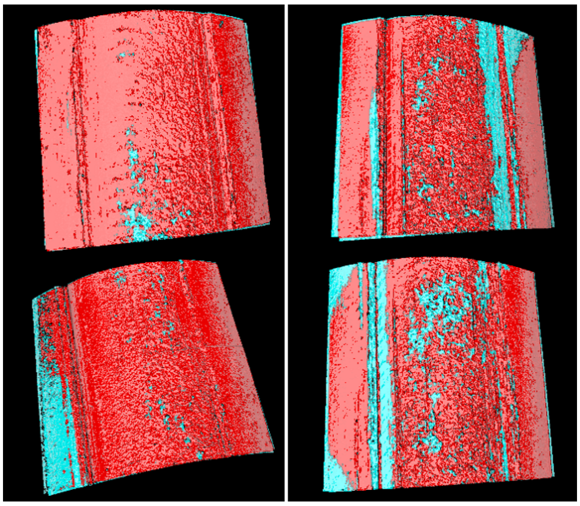


Figure D.6: Shape differences of landmark impressions. The left side pairs are comparisons of impressions by the same landmarks, and the right side pairs are by different ones.

### D.4.2 Simulation by neural networks

The neural network was used to identify 300 artificial patterns produced at random. These patterns are stored as a database. Unidentified patterns are slightly deformed database patterns. The deformed patterns are compared with the database. According to the output score, the Neural Network determines the ranking of all patterns in the database. A deformed pattern resembles the original. Therefore, if the original pattern ranks high, this simulation is proved successful.

The deformed patterns are produced on the following 4 systems;

(A) All elements transfer to 3-element.

(B) Elements on a certain part(=20%) disappear.

(C) Elements tend to gather around the center.

(D) Elements transfer on a sine wave.

The results of the simulation are also shown in Fig.D.7. In deformed systems (A), (C) and (D), over 91% of the original patterns were ranked within the top 5. Over 96% of the patterns were ranked within the top 10. The percentage of the patterns that failed within the top 20 was only 2%. This indicates that if an examiner searches at least 20 striations in the 300 database striations, we should be able to find the answer with a probability of more than 98%. On the other hand, the accuracy was worse for the deformed system (B) than for the others. Only 85% of the original patterns were ranked within the top 10 and 7% patterns failed to be included in the top 20. In many cases, many excited neurons corresponding to failure patterns are located in the erased part.

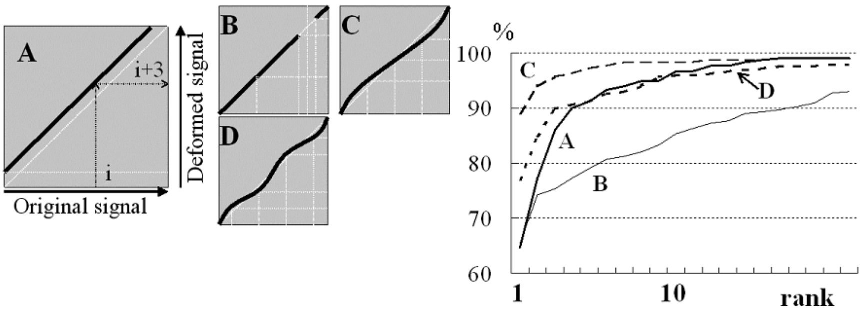


Figure D.7: The 4 deformation systems and the result of query simulation with 300 artificial patterns.

### D.4.3 Two-stage evaluation

Finally, we want to calculate a combined evaluation that contains both global and local shape similarities. We then introduce a combined score. When two data $Z_1$(database striation shape) and $Z_2$(unidentified striation shape) are given, a score that presents two striation shapes have the same origin is defined as follow,

$$S(Z_1, Z_2) = S_{local}(Z_1, Z_2) \cdot S_{global}(Z_1, Z_2) \tag{D.4}$$

The similarity score about global shape matching $S_{global}$ is represented as the area ratio defined by

$$S_{global} = \alpha \frac{areaofpinkreagion}{overlappedare} \tag{D.5}$$

Here, $\alpha$ is a coefficient that takes a low value (in this study 0.5) when there are any non-coincide regions spreading out along the direction of the scratch. Otherwise, it takes 1.

On the other hand, we regard the local shape similarity score $S_{local}$ as the score by the neural networks. The score $S_{local}$ is the averaged score evaluated in eight local regions chosen among the whole surface at random.

We have compared 100 pairs of real striations on fired bullets. Figure D.8 shows the results. Ten pairs of them have the same origins and others have different ones. This 2-stage method shows a good performance, since the result clearly shows the difference between by the same origins and by different origins. All pairs of same origins have scores over 0.4, while pairs of different ones have under 0.3. We could consider that a value of a threshold for identification is between 0.3 and 0.4.

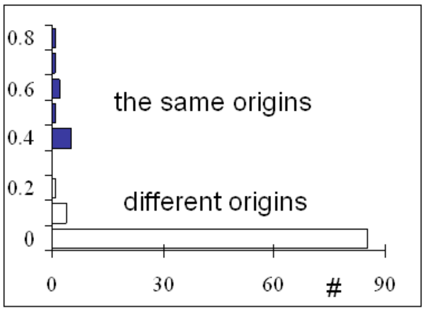


Figure D.8: Experimental results

## D.5 Conclusions

In this study, we presented a 2-stage algorithm for a shape comparison of impressions on bullets, by using 3D shape data. Firstly, we measured surface topography and compared the global shapes of two impressions. Neural networks were used for similarity evaluation of local elevations.

Our goal is to propose a 3-Dimensional identification method. To extend this method into rigid bullet identification, we have to compare numerous pairs of bullets to determine the rigid parameters. This is one of the most important future works about this method. We used striations on fired bullets mainly. It is not to say that this algorithm can be applied to other tool marks and other shapes.

At present, we compared two shapes by global curvatures and by local small elevations. Since elevations on striations of bullets are very small, it takes much time to measure striations. Moreover, it takes huge memory to store many striation shape data. As the future works, we are going to compress huge 3D data and build a practical system for tool mark identification.

# Bibliography

[ADSW02] L. Alvarez, R. Deriche, J. Snchez, and J. Weickert. Dense disparity map estimation respecting image discontinuities : A PDE an scale-space based approach. *Journal of Visual Communication and Image Representation*, Vol.13, pp.3–21, 2002.

[Agr03] M. Agrawal. Camera calibration using spheres : A semi-definite programing approach. In *Proceedings of the International Conference on Computer Vision (ICCV2003)*, pp.782–789, 2003.

[Alo90] J. Y. Aloimonos. Perspective approximations. *Image and Vision Computing*, Vol.8, No.3, pp.177–192, 1990.

[Aut] usa.autodesk.com.

[Ban96] A. Banno. A study on the starting transition of supersonic through-flow fan. Master's thesis, Department of Aeronautics and Astronautics, the University of Tokyo, 1996. (in Japanese).

[Ban04] A. Banno. Estimation of bullet similarity by using neural networks. *Journal of Forensic Sciences*, Vol.49, pp.500–504, 2004.

[Bau00] A. Baumberg. Reliable feature matching across widely separated views. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2000)*, Vol.1, pp.1774–1781, 2000.

[BBM87] R. C. Bolles, H. H. Baker, and D. H. Marimont. Epipolar plane image analysis: and approach to determining structure from motion. *International Journal of Computer Vision*, Vol.1, pp.7–55, 1987.

[BCS01] E. Bayro-Corrochano and G. Sobczyk. *Geometric algebra with applications in science and engineering*. Birkhäuser, Boston, 2001.

[BL95] G. Blais and M. D. Levine. Registering multiview range data to crate 3D computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.17, No.8, pp.820–824, 1995.

[BM92] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, pp.239–256, 1992.

[BMI04] A. Banno, T. Masuda, and K. Ikeuchi. Three dimensional visualization and comparison of impressions on fired bullets. *Forensic Science International*, Vol.140, pp.233–240, 2004.

[Bro66] D. C. Brown. Decentering distortion of lenses. *Photogrammetric Engineering*, Vol.32, No.3, pp.444–462, 1966.

[Bro76] D. Brown. The bundle adjustment – progress and prospect. In *XIII Congress of the ISPRS*, Helsinki, 1976.

[BVZ01] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cut. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.23, No.11, pp.1222–1239, 2001.

[CH96] S. Christy and R. Horaud. Euclidean shape and motion from multiple perspective views by affine iterations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.18, No.11, pp.1098–1104, 1996.

[CK95] J. Costeira and T. Kanade. A multi-body factorization method for motion analysis. In *Proceedings of the International Conference on Computer Vision (ICCV1995)*, pp.1071–1076, 1995.

[CM92] 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.

[Dav97] E. R. Davies. *Machine Vision: Theory, Algorithms, Practicalities*. Academic Press, London, 1997.

[DHS00] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Willey Interscience, second edition, 2000.

[DSH04] Y. Dufournaud, C. Schmid, and R. Horaud. Image matching with scale adjustment. *Computer Vision and Image Understanding*, Vol.93, pp.175–194, 2004.

[Fau93] O. Faugeras. *Three-dimensional computer vision*. The MIT press, Cambridge, Massachusetts, 1993.

[FB81] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, Vol.24, No.6, pp.381–395, 1981.

[FBF77] J. H. Friedman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best-matches in logarithmic time. *ACM Transactions on Mathematical Software*, Vol.3, No.3, pp.209–226, 1977.

[FL01] O. Faugeras and Q. T. Loung. *The geometry of multiple images*. The MIT press, Cambridge, Massachusetts, 2001.

[FP02] D. A. Forsyth and J. Ponce. *Computer vision –A modern approach*. Prentice Hall, 2002.

[GG84] S. Geman and D. Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.6, No.6, pp.721–741, 1984.

[GL96] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, 1996.

[GMW81] P. Gill, W. Murray, and M. Wright. *Practical Optimization*. Academic Press, London, 1981.

[GW04] A. Gruber and Y. Weiss. Multibody factorization with uncertainty and missing data using the em algorithm. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2004)*, Vol.1, pp.707–714, 2004.

[GZH$^+$01] Z. Geradts, D. Zaal, H. Hardy, J. Lelieveld, I. Keereweer, and J. Bijhold. Pilot investigation of automatic comparison of striation marks with structured light. In *Proceedings of SPIE*, Vol.4243, pp.49–56, 2001.

[Har97] R. I. Hartley. In denfence of the eight-point algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, No.6, pp.580–593, 1997.

[HHO+04] K. Hasegawa, Y. Hirota, K. Ogawara, R. Kurazume, and K. Ikeuchi. Laser range sensor suspended beneath balloon -flrs(flying laser range sensor)-. In *Meeting on Image Recognition and Understanding(MIRU2004)*, Vol.1, pp.739–744, 2004. (in Japanese).

[HHO+05] K. Hasegawa, Y. Hirota, K. Ogawara, R. Kurazume, and K. Ikeuchi. Laser range sensor suspended beneath balloon: Flrs(flying laser range sensor). *The IEICE Transactions on Information and Systems, PT.2 (Japanese Edition)*, Vol.J88-D-II, No.8, pp.1499–1507, 2005. (in Japanese).

[HK99] M. Han and T. Kanade. Perspective factorization methods for euclidean reconstruction. Technical Report:CMU–RI–TR–99–22, Robotics Institute, Carnegie Mellon University, 1999.

[HL03] M. Heizmann and F. P. Len. Imaging and analysis of forensic striation marks. *Optical Engineering*, Vol.42, No.12, pp.3423–3432, 2003.

[HMK+04a] Y. Hirota, T. Masuda, R. Kurazume, K. Ogawara, K. Hasegawa, and K. Ikeuchi. Designing a laser range finder which is suspended beneath a balloon. In *Proceedings of the 6th Asian conference on Computer Vision (ACCV2004)*, Vol.2, pp.658–663, 2004.

[HMK+04b] Y. Hirota, T. Masuda, R. Kurazume, K. Ogawara, K. Hasegawa, and K. Ikeuchi. Flying laser range finder and its data registration algorithm. In *Proceeding of the International Conference on Robotics and Automation (ICRA2004)*, pp.3155–3160, 2004.

[Hor86] B. K. P. Horn. *Robot Vision*. The MIT press, Cambridge, Massachusetts, 1986.

[Hr96] A. Heyden and K. Åström. Euclidean reconstruction from constant intrinsic parameters. In *Proceedings of the International Conference on Pattern Recognition (ICPR96)*, pp.339–343, 1996.

[HS88] C. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of Alvey Vision Conference*, pp.147–152, 1988.

[HZ04] R. I. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge University Press, second edition, 2004.

[IG98] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *Proceedings of European Conference on Computer Vision (ECCV1998)*, Vol.1, pp.232–248, 1998.

[IHN$^+$04] K. Ikeuchi, K. Hasegawa, A. Nakazawa, J. Takamatsu, T. Oishi, and T. Masuda. Bayon digital archival project. In *Proceedings of International Conference on Visual Systems and Multimedia (VSMM2004)*, pp.334–343, 2004.

[INHO03] K. Ikeuchi, A. Nakazawa, K. Hasegawa, and T. Ohishi. The great buddha project: Modeling cultural heritage for VR systems through observation. In *Proceedings of the 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR2003)*, 2003.

[Int] www.interface.co.jp.

[Jac77] D. A. Jacobs. *The State of the Art in Numerical Analysis*. Academic Press, London, 1977.

[KB99] J. D. Kinder and M. Bonfanti. Automated comparisons of bullet striations based on 3D topography. *Forensic Science International*, Vol.101, No.2, pp.85–93, 1999.

[KC92] C. Kingston and D. Crim. Neural networks in forensic science. *Journal of Forensic Science*, Vol.37, No.1, pp.252–264, 1992.

[KZ01] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions via graph cuts. In *Proceedings of the International Conference on Computer Vision (ICCV2001)*, pp.508–515, 2001.

[KZ02] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proceedings of European Conference on Computer Vision (ECCV1998)*, Vol.3, pp.82–96, 2002.

[Leo97] F. P. León. Enhanced imaging by fusion of illumination series. In *Proceedings of SPIE*, Vol.3100, pp.297–308, 1997.

[Lei] www.leica-geosystems.com.

[LF97] Q. T. Luong and O. D. Faugeras. Self-calibration of a moving camera from point correspondences and fundamental matrices. *International Journal of Computer Vision*, Vol.22, No.3, pp.261–289, 1997.

[LH81] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, Vol.293, pp.133–135, 1981.

[Low99] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the International Conference on Computer Vision (ICCV1999)*, pp.1150–1157, 1999.

[Low04] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, Vol.60, No.2, pp.91–110, 2004.

[Mar63] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial and Applied Mathematics*, Vol.11, pp.431–441, 1963.

[Mas03] T. Masuda. 3D shape restoration and comparison through simultaneous registration. Master's thesis, the Graduate School of Information Science and Technology, the University of Tokyo, 2003.

[MHNI05] T. Masuda, Y. Hirota, K. Nishino, and K. Ikeuchi. Simultaneous determination of registration and deformation parameters among 3D range images. In *Proceedings of the 5th International Conference on 3-D Digital Imaging and Modeling (3DIM2005)*, pp.369–376, 2005.

[MIF$^+$02] T. Masuda, S. Imazu, T. Furuya, K. Kawakami, and K. Ikeuchi. Shape difference visualization for old copper mirrors through 3D range images. In *Proceedings of the 8th International Conference on Virtual Systems and Multimedia*, pp.942–951, 2002.

[MK97] T. Morita and T. Kanade. A sequential factorization method for recovering shape and motion from image streams. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, No.8, pp.858–867, 1997.

[MNS$^+$00] D. Miyazaki, T. Oishi T. Nishikawa, R. Sagawa, K. Nishino, T. Tomomatsu, Y. Yakase, and K. Ikeuchi. The great buddha project: Modelling cultural heritage through observation. In *Proceedings of the 6th International Conference on Virtual Systems and Multimedia (VSMM2000)*, pp.138–145, 2000.

[Mor77] H. P. Moravec. Towards automatic visual obstacle avoidance. In *Proceedings 5th International Joint Conference on Artificial Intelligence*, page584, 1977.

[MS01] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proceedings of the International Conference on Computer Vision (ICCV2001)*, pp.525–531, 2001.

[MS02] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the European Conference on Computer Vision (ECCV2002)*, Vol.1, pp.1228–142, 2002.

[MS03] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2003)*, Vol.2, pp.257–263, 2003.

[MS04] K. Mikolajczyk and C. Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, Vol.60, No.1, pp.63–86, 2004.

[Muk02] J. Mukherjee. MRF clustering for segmentation of color images. *Pattern Recognition Letters*, Vol.23, pp.917–929, 2002.

[Neu97] P. Neugebauer. Geometrical cloning of 3D objects via simultaneous registration of multiple range images. In *Proceedings of the International Conference on Shape Modeling and Application*, pp.130–139, 1997.

[NI02] K. Nishino and K. Ikeuchi. Robust simultaneous registration of multiple range images. In *Proceedings on the 5th Asian Conference on Computer Vision (ACCV2002)*, Vol.2, pp.454–461, 2002.

[Nis01] K. Nishino. *Photometric object modeling -Rendering from a dense/sparse set of images-*. PhD thesis, the Graduate school of the University of Tokyo, 2001.

[Nis03] D. Nistér. An efficient solution to the five-point relative pose problem. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2003)*, Vol.2, pp.195–202, 2003.

[Ohk03] R. Ohkubo. Simultaneous registration of 2D images onto 3D models for texture mapping. Master's thesis, the Graduate School of Information Science and Technology, the University of Tokyo, 2003.

[OKS80] Y. Ohta, T. Kanade, and T. Sakai. Color information for region segmentation. *Computer Graphics and Image Processing*, Vol.13, No.3, pp.222–241, 1980.

[PFTV88] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C*. Cambridge University Press, 1988.

[PG99] M. Pollefeys and L. Van Gool. Stratified self-calibration with the modulus constraint. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.21, No.8, pp.707–724, 1999.

[PGPO94] M. Proesmans, L. Van Gool, E. Pauwels, and A. Oosterlinck. Determination of optical flow and its discontinuities using non-linear diffusion. In *Proceedings of European Conference on Computer Vision (ECCV1994)*, Vol.2, pp.295–304, 1994.

[Pho75] B. T. Phong. Illumination for computer generated pictures. *Communications of the ACM*, Vol.18, No.6, pp.311–317, 1975.

[PK97] C. Poelmann and T. Kanade. A paraperspective factorization method for shape and motion recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, No.3, pp.206–218, 1997.

[PKG99] M. Pollefeys, R. Koch, and L. Van Gool. Self-calibration and metric reconstruction in spite of varying and unknown intrinsic camera parameters. *International Journal of Computer Vision*, Vol.32, No.1, pp.7–25, 1999.

[PM90] P. Perona and J. Malik. Scale space and edge deduction using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.12, No.7, pp.629–639, 1990.

[Pol71] E. Polak. *Computational Methods in Optimization*. Academic Press, New York, 1971.

[Pol02] M. Pollefeys. Visual 3D modeling from images (tutorial notes). Technical report, University of North Carolina, Chapel Hill, USA, 2002.

[PZ98] P. Pritchett and A. Zisserman. Wide baseline stereo matching. In *Proceedings of the International Conference on Computer Vision (ICCV1998)*, pp.754–760, 1998.

[RL01] S. Rusinkiewicz and M. Levoy. Efficient variant of the ICP algorithm. In *Proceedings of the 3rd International Conference on 3-D Digital Imaging and Modeling (3DIM2001)*, pp.145–152, 2001.

[RLSP03] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3D object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2003)*, Vol.2, pp.272–280, 2003.

[RM86] D. E. Rumelhart and J. L. McClelland. *Parallel Distributed Processing*. The MIT Press, New York, 1986.

[Roy99] S. Roy. Stereo without epipolar lines: A maximum-flow formulation. *International Journal of Computer Vision*, Vol.34, No.2/3, pp.147–161, 1999.

[SB97] S. M. Smith and M. Brady. SUSAN - a new approach to low level image processing. *International Journal of Computer Vision*, Vol.23, No.1, pp.45–78, 1997.

[SC98] Royand S and I. J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *Proceedings of the International Conference on Computer Vision (ICCV1998)*, pp.492–502, 1998.

[SG02a] C. Strecha and L. Van Gool. Motion - stereo integration for depth estimation. In *Proceedings of European Conference on Computer Vision (ECCV2002)*, Vol.2, pp.170–185, 2002.

[SG02b] C. Strecha and L. Van Gool. Pde-based multi-view depth estimation. In *Proceedings of 1st International Symposium on 3D Data*

*Processing Visualization and Transmission (3DPVT 2002)*, pp.416–427, 2002.

[SI94] Y. Sato and K. Ikeuchi. Temporal-color space analysis of reflection. *Journal of the Optical Society of America*, Vol.11, No.11, pp.2990–3002, 1994.

[SKS+02] R. Swaminathan, S. B. Kang, R. Szeliski, A. Criminisi, and S. K. Nayar. On the motion and appearance of specularities in image sequences. In *Proceedings of the 7th European Conference on Computer Vision (ECCV2002)*, Vol.2, pp.508–523, 2002.

[SM97] C. Schmid and R. Mohr. Local grayvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.19, No.5, pp.530–535, 1997.

[SM99] P. F. Strum and S. J. Maybank. On plane-based camera calibration : A general algorithm, singularities, applications. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR1999)*, Vol.1, pp.432–437, 1999.

[SMB98] C. Schmid, R. Mohr, and C. Bauckhage. Comparing and evaluationg interest points. In *Proceedings of the International Conference on Computer Vision (ICCV1998)*, pp.230–235, 1998.

[SN00] R. Swaminathan and S. K. Nayar. Nonmetric calibration of wide-angle lenses and polycameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22, No.10, pp.1172–1178, 2000.

[SR80] J. Stoer and R.Bulirsh. *Introduction to Numerical Analysis*. Springer-Verlag, New York, 1980.

[STEY05] R. Sagawa, M. Takatsuji, T. Echigo, and Y. Yagi. Calibration of lens distortion by structured-light scanning. In *IEEE/RSJ International Conference on Intelligent Robots & Systems (IROS2005)*, 2005.

[STG03] C. Strecha, T. Tuytelaars, and L. Van Gool. Dense matching of multiple wide-baseline views. In *Proceedings of the International Conference on Computer Vision (ICCV2003)*, pp.1194–1201, 2003.

[SZS03] J. Sun, N-N. Zheng, and H-Y. Shum. Stereo matching using belief propagation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.25, No.7, pp.787–800, 2003.

[TDH03] S. Thrun, M. Diel, and D. Haehnel. Scan alignment and 3-D surface modeling with a helicopter platform. In *Proceedings of the 4th International Conference on Field and Service Robotics*, 2003.

[TH84] R. Y. Tsai and T. S. Huang. Uniqueness and estimation of three-dimensional motion parameters of rigid objects with curved surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.6, pp.13–27, 1984.

[TK92] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, Vol.9, No.2, pp.137–154, 1992.

[TS67] K. Torrance and E. Sparrow. Theory for off-specular reflection from roughened surfaces. *Journal of the Optical Society of America*, No.57, pp.1105–1114, 1967.

[Tsa86] R. Y. Tsai. An efficient and accurate camera calibration technique for 3D machine vision. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR1986)*, pp.364–374, 1986.

[Tsa87] R. Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Journal of Robotics and Automation*, Vol.3, pp.323–344, 1987.

[TSR00] H. Tao, S. Sawhney, and Kumar R. Global matching criterion and color segmentation based stereo. In *Proceedings of the Workshop on the. Application of Computer Vision (WACV2000)*, pp.246–253, 2000.

[TSR01] H. Tao, S. Sawhney, and Kumar R. A global matching framework for stereo computation. In *Proceedings of the International Conference on Computer Vision (ICCV2001)*, pp.532–539, 2001.

[TV98] E. Trucco and A. Verri. *Introductory techniques for 3-D computer vision*. Prentice Hall, 1998.

[UOS05] T. Ueshiba, T. Okatani, and T. Sato. A survey of camera calibration techniques. *IPSJ SIG Technical Report, 2005–CVIM–148*, Vol.2005, No.18, pp.1–18, 2005. (in Japanese).

[UT03] T. Ueshiba and F. Tomita. Plane-based calibration algorithm for multi-camera systems via factorization of homography matrices. In *Proceedings of the International Conference on Computer Vision (ICCV2003)*, pp.966–973, 2003.

[VZG01] J. Visnovcova, L. Zhang, and A. Gruen. Generating a 3D model of a bayon tower using non-metric imagery. In *Proceedings of the International Workshop Recreating the Past –Visualization and Animation of Cultural Heritage*, 2001.

[WCH92] J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.14, No.10, pp.965–980, 1992.

[WHA89] J. Weng, T. S. Huang, and N. Ahuja. Motion and structure from two perspective views: algorithms, error analysis and error estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.11, No.5, pp.451–476, 1989.

[Whe96] M. D. Wheeler. *Automatic modeling and localization for object recognition*. PhD thesis, School of Computer Science, Carnegie Mellon University, 1996.

[WP97] E. Walter and L. Prontazo. *Identification of Parametric Models from Experimental Data*. Springer, 1997.

[WZHW04] Y. Wu, H. Zhu, Z. Hu, and F. Wu. Camera calibration from the quasi-affine invariance of two parallel circles. In *Proceedings of European Conference on Computer Vision (ECCV2004)*, Vol.1, pp.190–202, 2004.

[Z+F] www.zf-lase.com.

[Zha94] Z. Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, Vol.13, pp.119–152, 1994.

[Zha99] Z. Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *Proceedings of the International Conference on Computer Vision (ICCV1999)*, pp.666–673, 1999.

[Zha00] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.22, No.11, pp.1330–1334, 2000.

[Zha04] Z. Zhang. Camera calibration with one-dimensional objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol.26, No.7, pp.892–899, 2004.

[ZNH04] W. Zhao, D. Nister, and S. Hsu. Alignment of continuous video onto 3D point clouds. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR2004)*, Vol.2, pp.964–971, 2004.