

攝影機校準及影像轉換技術
與其應用之研究

**A Study on Camera Calibration and Image
Transformation Techniques and Their
Applications**

研 究 生： 鄭 勝 文
指 導 教 授： 蔡 文 祥 博 士

Student: Sheng-Wen Jeng
Advisor: Dr. Wen-Hsiang Tsai

國 立 交 通 大 學 資 訊 學 院
資 訊 科 學 與 工 程 研 究 所
博 士 論 文

A Dissertation Submitted to
Institute of Computer Science and Engineering
College of Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy
in Computer and Information Science
June 2007
Hsinchu, Taiwan, 300
Republic of China

中華民國 九十六 年 六 月

國家圖書館博碩士論文電子檔案 上網授權書

ID:GT008623815

本授權書所授權之論文為授權人在國立交通大學資訊學院資訊工程系資訊科學與工程研究所 95 學年度第 二 學期取得博士學位之論文。

論文題目：攝影機校準及影像轉換技術與其應用之研究

A Study on Camera Calibration and Image Transformation
Techniques and Their Applications

指導教授：蔡文祥

茲同意將授權人擁有著作權之上列論文全文（含摘要），非專屬、無償授權國家圖書館，不限地域、時間與次數，以微縮、光碟或其他各種數位化方式將上列論文重製，並得將數位化之上列論文及論文電子檔以上載網路方式，提供讀者基於個人非營利性質之線上檢索、閱覽、下載或列印。

※ 讀者基於非營利性質之線上檢索、閱覽、下載或列印上列論文，應依著作權法相關規定辦理。

授權人：鄭勝文

親筆簽名：

鄭勝文

民國 96 年 06 月 27 日

1. 本授權書請以黑筆撰寫，並列印二份，其中一份影印裝訂於附錄三之二(博碩士紙本論文著作權授權書)之次頁；另一份於辦理離校時繳交給系所助理，由圖書館彙總寄交國家圖書館。

國立交通大學

博碩士論文全文電子檔著作權授權書

(提供授權人裝訂於紙本論文書名頁之次頁用)

本授權書所授權之學位論文，為本人於國立交通大學資訊工程系資訊科學與工程研究所，95 學年度第二 學期取得博士學位之論文。

論文題目：攝影機校準及影像轉換技術與其應用之研究

A Study on Camera Calibration and Image Transformation Techniques
and Their Applications

指導教授：蔡文祥

同意 不同意

本人茲將本著作，以非專屬、無償授權國立交通大學與台灣聯合大學系統圖書館：基於推動讀者間「資源共享、互惠合作」之理念，與回饋社會與學術研究之目的，國立交通大學及台灣聯合大學系統圖書館得不限地域、時間與次數，以紙本、光碟或數位化等各種方法收錄、重製與利用；於著作權法合理使用範圍內，讀者得進行線上檢索、閱覽、下載或列印。

論文全文上載網路公開之範圍及時間：

本校及台灣聯合大學系統區域網路	<input checked="" type="checkbox"/> 中華民國96年08月01日公開
校外網際網路	<input checked="" type="checkbox"/> 中華民國96年08月01日公開

授權人：鄭勝文

親筆簽名： 鄭勝文

中華民國 96 年 06 月 27 日

國立交通大學
資訊工程系博士班

論文口試委員會審定書

本校 資 訊 工 程 系 鄭勝文 君

所提論文 攝影機校準及影像轉換技術其應用之研究

合於博士資格水準、業經本委員會評審認可。

口試委員：陳永昌 葉仁輝

范國清 楊谷暉

陳淑媛 蔡文祥

王玲玲

指導教授：蔡文祥

系主任：李代

中華民國九十六年六月二十二日

Department of Computer Science
College of Computer Science
National Chiao Tung University
Hsinchu, Taiwan, R.O.C.

Date: Jun. 22, 2007

We have carefully read the dissertation entitled **A Study on Camera Calibration and Image Transformation Techniques and Their Applications** submitted by **Sheng-Wen Jeng** in partial fulfillment of the requirements of the degree of Doctor of Philosophy and recommend its acceptance.

Ying Chen

Huo-Chin Jan

Shu-Yuan Chen

Ling-Ling Wang

Jun-H. Cheng

Yung-Yen

Wen-hsiung Tsai

Thesis Advisor:

Wen-hsiung Tsai

Chairman:

Chin-Hao Teng

攝影機校準及影像轉換技術 與其應用之研究

研究生： 鄭勝文

指導教授： 蔡文祥博士

國立交通大學資訊學院


資訊工程系

資訊科學與工程研究所

摘要

在電腦視覺領域中，抽取與分析攝影機攝取之影像內所含資訊是由演算法則實作之電腦程式來完成，其中一種資訊為空間中的物體幾何(形狀或位置等)。此類應用之演算法則中一般包含一重要之程序稱為“攝影機校準(camera calibration)”，其目的是在建立影像平面與物體空間座標系統間之對應關係，此對應關係以“數學函數(mathematical function)”表示，並在校準後得到一組參數，用以代表此攝影機在物體空間座標系統中之“特性(characteristics)”表現。此特性對於不同攝影機的光學系統架構(光學焦距、元件擺置等)是唯一的。而不同的攝影機種類(光學系統設計不同)則需要用不同的“數學函數”模式來描述。在完成“攝影機校準”程序後，演算法則才能將攝取影像之資訊轉換至物體空間中，以符合人腦之理解模式。本博士論文內容在探討不同種類攝影機之校準及影像轉換技術與其應用，共提出三種有關全方位(omni-directional)攝影機之全新影像轉換方法及兩種傳統攝影機之全新校準方法及應用。

全方位攝影機因有接近(甚至超過)半球形之廣大視野而被廣泛的應用於視覺監控、機器人視覺或自動導航，其擷取之全方位影像(omni-image)最後必須轉正為一般正常之透視影像(perspective-view image)或環場影像(panoramic image)，以利人眼觀察或影像證據保存。目前之研究注重單一視點(single view point, SVP)攝影機之影像轉正，對於非單一視點(non-SVP)全方位攝影機之影像轉正，因其困難度高而少有研究。但因非單一視點攝影機具有比單一視點較好之平均徑向解析度(radial resolution)及視野較大之優點，使其更適合應用於上述之領域。本博士論文冀能領先全球研究，發展出並探討適用於非單一視點全方位攝影機之影像轉正方法，補足其先天缺點，使其更適於實際之應用。我們總結提出的方法如下：

- 
- (a) 提出一個針對非單一視點之雙曲下折攝影機(hypercatadioptric camera)的影像轉正(image unwarping)方法。此方法擴展了目前既存(單一視點)之方法，可容許透鏡/反射鏡(lens/mirror)之不精確組裝。此問題在大部份的實際應用中是難以克服的。
- (b) 提出一個稱為“雙層八方向之邊緣保存權重式插補(edge-preserving 8-directional two-layered weighting interpolation)”方法，可用以插補從一個非單一視點之全方位攝影機攝取之全方位影像(omni-image)轉正到透視(perspective-view)或環場(panoramic)影像之未填充像點(unfilled pixel)。此方法能解決內含許多不均勻分佈之未填充像點的影像插補問題。
- (c) 提出一個全方位影像轉正到環場或透視影像之統合方法，此方法是基植於一個環場轉換表(pano-mapping table)之新觀念。此表由一針對任何型式之全方位攝影機做一次簡單之學習程序而產生，隱含了所有攝影機參數之資訊。

此外，針對傳統攝影機在指標系統應用上的缺點，我們提出兩種方法：

- (d) 提出一個顯示螢幕與其影像間座標轉換之強韌且精確的校準方法，此方法經由消除接近影像邊界之位移誤差而改善了座標轉換之精確度。
- (e) 提出一個以視訊為基礎之控制電腦游標的相機滑鼠(camera mouse)，此滑鼠使用一個視訊攝影機，可手持於空中操作。此方法之主要優點是不需要複雜之攝影機校準。

以上本論文提出之方法(a)至(e)，皆為創新之作。同時多方面的實驗結果顯示所提方法可行實用，與其他方法比較結果，亦具有相當的優越性。



ABSTRACT

In the field of computer vision, extracting and analyzing the information contained in the image captured by a camera are performed by a computer program implementing a certain algorithm. One kind of such information is the geometry (its shape or pose) about an object in the space. The algorithm to extract such a kind of information usually includes an important procedure called “camera calibration.” The purpose of camera calibration is to construct the relationship between the image plane of the camera and the coordinates system of the object space. The relationship is usually represented by a “mathematical function.” After calibration, a set of parameters representing the “characteristics” of the camera in the coordinate system of the object space is obtained. The characteristics are unique for each distinct optical structure (the focus length of optics, the component lay-out, etc.) of the camera. Different cameras with different optics designs need different “mathematical function” models to describe their features. After completing the “camera calibration” procedure, an algorithm is utilized to transform the information contained in a captured image into the object space to conform the realization model of the human brain.

In this dissertation study, the investigation of camera calibration and image transformation techniques, as well as their applications is conducted. Three new methods are proposed for related topics of image transformations for the omni-directional camera (or just *omni-camera*) and two novel methods are proposed for the purpose of camera calibration.

Because the field of view (FOV) of an omni-camera is almost near or even beyond a full hemisphere, it is popularly applied in the fields of visual surveillance,

and vision-based robot or autonomous vehicle navigation. The captured omni-directional image (or just *omni-image*) should be rectified into a normal perspective-view or panoramic image for convenient human viewing or image-proof preservation. Current studies focus on the image rectification or image unwarping for a single-view-point (SVP) omni-camera. Studies on non-SVP ones are limited because of the difficulty to analyze their structures. But a non-SVP omni-camera is superior, compared with an SVP one in the aspects of possessing uniform radial resolutions and larger FOVs. These merits make it more suitable in the above application cases. In this study, we develop some suitable solutions to the issue of image unwarping for non-SVP omni-cameras to compensate their inherent deficiencies for fitting the requirement of practical applications. Proposed methods in this study are summarized in the following.

- (a) An analytic image unwarping method is proposed for a non-SVP hypercatadioptric camera. The method has extended the image unwarping capability of the existing methods for SVP omni-cameras to tolerate lens/mirror assembly imprecision, which is difficult to overcome in most real applications.
- (b) A new method called “edge-preserving 8-directional two-layered weighting interpolation” is proposed for interpolating unfilled pixels in a perspective-view or panoramic image resulting from unwarping an omni-image taken by a non-SVP omni-camera. This method can solve the problem of edge preserving in interpolating the input image which has many irregularly distributed unfilled pixels.
- (c) A unified approach to unwarping of omni-images into panoramic or perspective-view images is proposed. The approach is based on a new concept of pano-mapping table, which is created once forever by a simple learning process

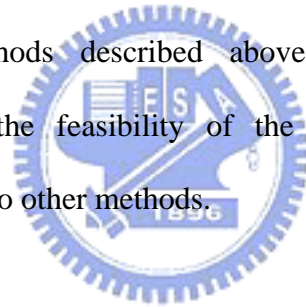
for an omni-camera of any kind as a summary of the information conveyed by all the camera parameters.

Moreover, for resolving the deficiency of traditional camera applications in the pointing system, we propose two methods.

(d) A robust and accurate calibration method for coordinate transformation between display screens and their images is proposed. The method improves the accuracy of the coordinate transformation to eliminate the shift errors near the image border.

(e) A camera mouse with a vision-based method for computer cursor control using a video camera held in hand in the air is proposed. The main merit of this method is that it requires no complicated camera calibration.

All the proposed methods described above are innovative. Meanwhile, experimental results show the feasibility of the proposed methods, and their effectiveness and superiority to other methods.



ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my advisor, Professor Wen-Hsaing Tsai, for his kind patience and guidance throughout in the course of this dissertation study and the invaluable training. Thanks are also extended to the administrative assistant Miss Lee in the Computer Vision Laboratory at National Chiao Tung University for her valuable help during this study.

Finally, I am so grateful to my wife, my parents, and my children for their love, support, and tolerance during the dissertation study. This dissertation is dedicated to them.



Table of contents

摘要 (中文)	i
Abstract in English	iv
ACKNOWLEDGEMENTS	vii
Table of contents	viii
List of figures	xii
Chapter 1 Introduction	1
1.1 Survey of Related Works	6
1.1.1 Concept of Camera Calibration	6
1.1.2 Survey of Omni-directional Cameras	8
1.1.3 Survey of Approaches of Unwarping Omni-image into Normal-view Image	10
1.1.4 Survey of pointing methods using video cameras	13
1.2 Contributions of This Study	14
1.3 Dissertation organization	16
Chapter 2 Analytic image unwarping for omni-directional cameras with hyperbolic-shaped mirrors	17
2.1 introduction	17
2.2 Review of Previous Works	19
2.3 Proposed Method for Calibrating Camera Pose with Respect to Mirror	22
2.3.1 System Configuration and Coordinate System Relationships	22
2.3.2 Proposed Calibration Process for Estimating Pose Parameters with Respect to Camera	25
2.3.3 Proposed Calibration Process for Deriving Pose Parameters with Respect to	

Mirror	28
2.4 Back-Projection of Image Point	30
2.4.1 Derivation of Unit Normal Vector \vec{n}	31
2.4.2 Use of Co-Planarity Constraint	33
2.4.3 Use of Reflection Constraint	35
2.4.4 Calculating Direction of Incident Ray	36
2.4.5 Calculating Coordinates of Mirror Surface Point in Terms of Image Point Coordinates	37
2.5 Experimental Results	39
2.5.1 Unwarping of A Pseudo-image into Perspective Views	39
2.5.2 Unwarping of A Real Image into Perspective Views.....	44
2.6 Summary	44
Chapter 3 Improving quality of unwarped omni-images by a new edge-preserving interpolation technique.....	47
3.1 introduction	47
3.2 Image Interpolation by Edge Preserving and Pixel Value Weighting	52
3.2.1 Non-Edge-Preserving Version of Proposed Method.....	52
3.2.2 Edge-Preserving Version of Proposed Technique.....	55
3.3 Experimental Results	58
3.4 Summary	65
Chapter 4 Using Pano-Mapping Tables for Unwarping of Omni-Images into Panoramic and Perspective-View Images	68
4.1 introduction	68
4.2 Proposed Method Using Pano-Mapping Table	70

4.2.1 Landmark Learning Procedure	73
4.2.2 Table Creation Procedure	74
4.2.3 Image Unwarping Procedure	77
4.3 Experimental Results	83
4.4 Summary	85
Chapter 5 A Robust and Accurate Calibration Method for Coordinate	
Transformation between Display Screens and Their Images.....	88
5.1 introduction	88
5.2 Proposed Method	90
5.2.1 Use of Calibration Patterns.....	91
5.2.2 Feature Extraction.....	93
5.2.3 Coordinate transformation from image coordinates to screen coordinates	101
5.3 Experimental Results	103
5.3.1 Plane-to-plane calibration by Tsai [13].....	103
5.3.2 Comparison of results of proposed method and Tsai's method.....	105
5.4 Summary	106
Chapter 6 A Camera Mouse for Computer Cursor Control.....	108
6.1 introduction	108
6.2 Proposed Method	111
6.2.1 Locating landmarks and computer monitor screen border in the detection stage	113
6.2.2 Computing cursor location in the tracking stage.....	115
6.2.3 Computing corner coordinates of corrected rectangular monitor screen shape	117
6.2.4 Dynamic tracking of landmarks for continuous cursor position computation .	118

6.2.5 Tracking with missing landmarks.....	120
6.3 Experimental Results	120
6.4 Summary	122
Chapter 7 Conclusions and Suggestions for Future Research	125
7.1 Conclusions.....	125
7.2 Suggestions for future research.....	130
References	132
Publication List	141
Vita	143



List of figures

Fig. 1.1	Concept of camera calibration & image transformation.....	1
Fig. 1.2	Structures of. dioptric (left) and catadioptric (right) omni-cameras.....	3
Fig. 1.3	Structure of an SVP hypercatadioptric camera.....	4
Fig. 1.4	Caustic surfaces of hypercatadioptric cameras.....	4
Fig. 1.5	A laser pointer pointing system.....	5
Fig. 1.6	Calibration work ranges for different types of camera design.....	7
Fig. 1.7	FOVs of different type of cameras.....	8
Fig. 1.8	Unwarped perspective view images using Mashita’s calibration method [11].....	9
Fig. 1.9	Example of unwarping image of a fish-eye camera.....	10
Fig. 1.10	Paraboloid (left), hyperboloid (middle), and spheroid (right) mirror systems..	10
Fig. 1.11	A paracatadioptric camera.....	12
Fig. 1.12	The Camera Mouse user playing with educational software.....	13
Fig. 2.1	An SVP hypercatadioptric camera.....	20
Fig. 2.2	The configuration of a hypercatadioptric camera used in this study... ..	23
Fig. 2.3	The calibration pattern designed for use in this study.....	26
Fig. 2.4	An omni-image of the calibration pattern.....	26
Fig. 2.5	The calibration result of a hypercataoptric camera used in this study.	28
Fig. 2.6	The image projection model.	31
Fig. 2.7	The unit normal vector \vec{n}	33
Fig. 2.8	The co-planar vectors and the cross product.....	34
Fig. 2.9	A pseudo target of size 20m × 20m with an L-shaped wall at the center position.	40
Fig. 2.10	The warped image of the pseudo target in Fig. 2.9.....	40

Fig. 2.11	View planes defined in the real world for unwarped images.....	41
Fig. 2.12	Unwarped images of Fig. 2.10 (Top view)..	42
Fig. 2.13	Unwarped images of Fig, 2.10 from 4 side views..	43
Fig. 2.14	Unwarped images of a real scene in Fig. 2.4 (Top view)..	45
Fig. 2.15	Unwarped images of a real scene in Fig. 2.4 (Side view).....	46
Fig. 3.1	An SVP hypercatadioptric camera.....	48
Fig. 3.2	Examples of images for illustration.....	50
Fig. 3.3	Illustration of proposed interpolation.....	53
Fig. 3.4	Creation of unwarped edge map from original omni-image.....	57
Fig. 3.5	Illustration of proposed on-edge decision.....	58
Fig. 3.6	Comparison of results of different methods.....	59
Fig. 3.7	Results using different edge threshold values at constant angle threshold ($\theta_t = 120^\circ$).....	61
Fig. 3.8	Results using different angle threshold values at constant edge threshold ($e_t = 0.35$).....	63
Fig. 3.9	Detailed edge maps obtained by using different angle thresholds and an identical edge threshold.....	64
Fig. 3.10	Detailed calculation of the span angle of an unfilled pixel.....	64
Fig. 3.11	Interpolation results with different window sizes.....	66
Fig. 3.12	Comparison of results of different interpolation methods for image expansion.....	67
Fig. 4.1	An example image unwarping.....	68
Fig. 4.2	System configuration.....	71
Fig. 4.3	Mapping between pano-mapping table and omni-image.....	72
Fig. 4.4	Landmark learning interface.....	74

Fig. 4.5	Lateral-view configuration for generating a panoramic image.....	78
Fig. 4.6	Top-view configuration for generating a perspective-view image.	82
Fig. 4.7	Landmark calibration.....	84
Fig. 4.8	Examples of image unwarping.....	85
Fig. 4.9	Unwarped perspective images using Mashita’s calibration method.	86
Fig. 4.10	Perspective-view images generated from an omni-image video sequence taken by a tracking system.	87
Fig. 5.1	Calibration patterns - white-rectangle, black-rectangle, and dot-matrix patterns.	91
Fig. 5.2	Examples of captured images of Fig. 5.1.....	91
Fig. 5.3	A TDM image created by images in Figs. 5.2(a) and (b).	93
Fig. 5.4	An example of image analysis of dot-matrix pattern image.....	94
Fig. 5.5	Polygonal model of a screen region.....	95
Fig. 5.6	Extracting the vertex P_1 from the TDM.....	96
Fig. 5.7	Extracting P_2, L_1 , and L_2 from TDM image.....	96
Fig. 5.8	A 5-dot cross shape near the center of binary dot-matrix pattern image.	98
Fig. 5.9	Rough CPM superimposed on the binary dot matrix pattern image.....	99
Fig. 5.10	Deformable template matching.....	100
Fig. 5.11	Extracting Landmark point P_i at left side.....	101
Fig. 5.12	Calculating screen coordinates (X_j, Y_j) in a basic region R_j and in border area R_g	102
Fig. 5.13	Comparison using calibration image points. The precise landmark position, the computed position by proposed method, and that by Tsai’s method are listed under each dot.....	106
Fig. 5.14	Comparison using re-captured image points. The precise landmark position,	

the computed position by proposed method, and that by Tsai's method are listed under each dot.	107
Fig. 6.1 Illustration of proposed camera mouse system.	111
Fig. 6.2 An example of computer monitor images.	112
Fig. 6.3 Illustration of detecting candidate landmark circles.	115
Fig. 6.4 Detecting of border lines.	115
Fig. 6.5 Affine transformation and cursor position computation.	117
Fig. 6.6 Eight sampling bars at a candidate landmark.	119
Fig. 6.7 Some snapshots extracted from a video with 412 frames of images.	121
Fig. 6.8 Some snapshots of cursor locations on a notebook PC extracted from a video with 412 frames of images.	122
Fig. 6.9 Some snapshots of cursor locations on a desktop PC extracted from a video with 880 frames of images.	123
Fig. 6.10 Layout of four square targets.	124
Fig. 6.11 Images of access results.	124

LIST OF TABLES

Table 4.1. An example of pano-mapping table of size $M \times N$	71
---	----

Chapter 1

Introduction

It is a kernel concept in computer vision to identify an object in a working space by analyzing an image captured by a camera. The technique of building the relationship between the coordinate system of a working space and that of a captured image is called *camera calibration*. The work of transforming a pixel in a captured image into a corresponding point in a working space is called *image transformation*. The image transformation work relies on the completion of camera calibration by using the *calibrated parameters* created by the calibration procedure. Fig. 1.1 shows the concept described above, where, multiple n known pairs (u_k, v_k) and (x_{wk}, y_{wk}, z_{wk}) , $k = 1, 2, \dots, n$, are used to get a set of calibrated parameters. With help of these calibrated parameters, an image pixel (u, v) is transformed into a point (x_w, y_w, z_w) in the working space.

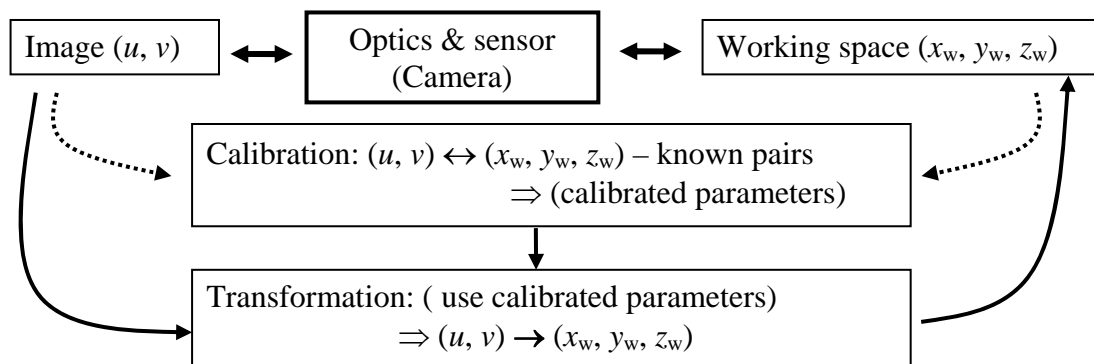


Fig. 1.1 Concept of camera calibration & image transformation.

For a traditional perspective camera, calibration is a well-known and mature technique [1]. But for a kind of so-called omni-directional camera, the camera calibration work is a new and developing technique, and many practical problems need to be resolved. Even in the field of traditional perspective camera calibration, there are still some problems in real applications. In this study, we investigate the *image unwarping* problem for omni-directional cameras and propose some new solutions to such a kind of problem. To conduct image unwarping which is an image transformation problem, we have to conduct the camera calibration work in advance. For some real applications using perspective cameras, for example, applying an image-based pointing device, we will propose some new ideas for improving the accuracy of the calibration result.

It is well known in computer vision that enlarging the field of view (FOV) of a camera enhances the visual coverage, reduces the blind area, and saves the computation time, of the camera system, especially in applications like visual surveillance and vision-based robot or autonomous vehicle navigation. An extreme way is to expand the FOV to be beyond a full hemisphere by the use of some specially designed optics. A popular name for this kind of camera is *omni-directional camera* [2], and an image taken by it is called an *omni-directional image*.

Omni-cameras can be categorized into two types according to the involved optics, namely, *dioptric* and *catadioptric*. A dioptric omni-camera captures incoming light going directly into the imaging sensor to form omni-images. Examples of image unwarping works for a kind of dioptric omni-camera called *fish-eye camera* can be found in [3][4]. A catadioptric omni-camera [5] captures incoming light reflected by a mirror to form omni-images. Fig. 1.2 shows the different structures of these two kinds of cameras. The mirror surface of a catadioptric omni-camera may be in various shapes, like conic,

parabolic, hyperbolic, etc. If all the reflected light rays pass through a common point, the camera is said additionally to be of the *single-view-point* (SVP) type [6]; otherwise, of the *non-single-view-point* (non-SVP) type.

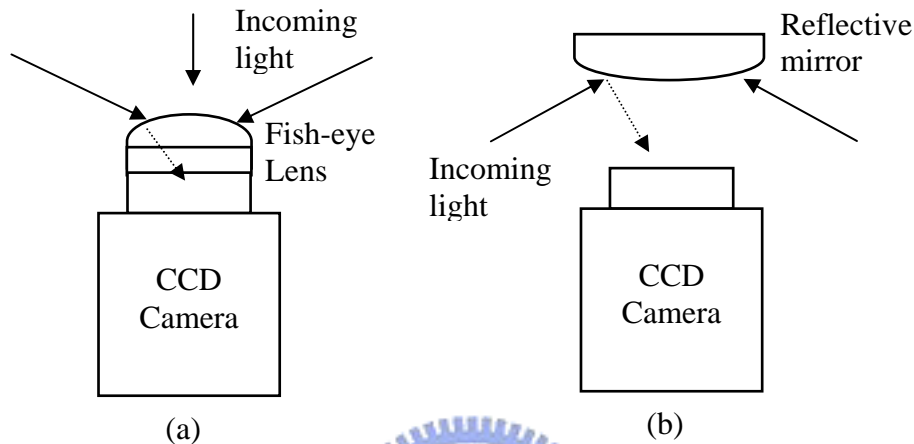


Fig. 1.2 Structures of omni-cameras. (a) Dioptric. (b) Catadioptric.

Fig. 1.3 shows the structure of an SVP hypercatadioptric omni-camera [7][8], where the focus point of the perspective camera is located at the outer focus point of the hyperbolic curve of the mirror surface. In this perfectly aligned structure, all incoming light rays will pass through the common point O_c . If the structure is misaligned, incoming light rays will not pass through a common point but form a “caustic” surface [9] which is the locus of viewpoints, and this omni-camera is a non-SVP one. Fig. 1.4 shows the caustic surfaces of two hypercatadioptric omni-cameras, where Fig. 1.4(a) is an SVP type, and Fig. 1.4(b) is a non-SVP one. We can see the caustic surface merges into a single point in Fig. 1.4(a). Grossberg et. al. [57] proposed an imaging model to represent an arbitrary imaging system by using a Caustic Raxel Model, but there lacks discussions about the detail of how to conduct image unwarping between a captured image and a defined view plane in the world space.

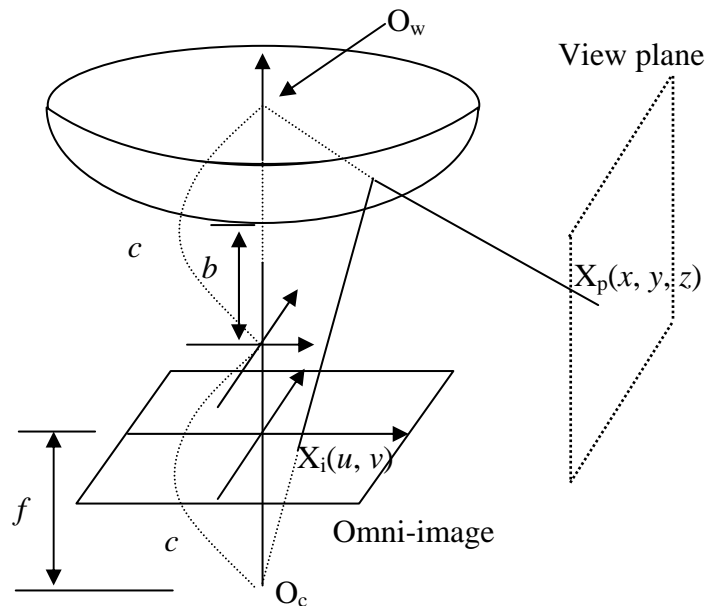


Fig. 1.3 Structure of an SVP hypercatadioptric camera. Where O_w is the origin of the world coordinate system (also one focus of the hyperbolic curve), and O_c is the optical center (another focus of the hyperbolic curve).

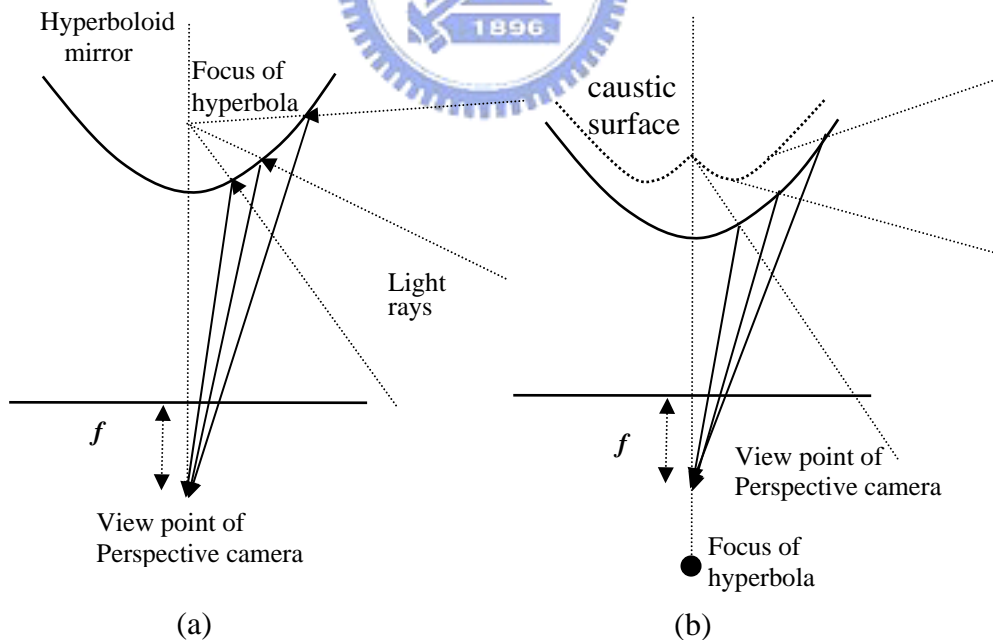


Fig. 1.4 Caustic surfaces of hypercatadioptric cameras. (a) SVP (merge into one point). (b) non-SVP.

The image unwarping work for an omni-camera is an image transformation

process from a portion of an omni-image to a view plane (as showed in Fig. 1.3) defined in the world space after finishing the camera calibration work. The difficulty of image unwarping depends on the type of omni-camera and its structure. Usually, the SVP type is easier to handle [7][10], while the non-SVP ones are more difficult to deal with. If the non-SVP omni-camera is treated as an SVP one to unwarp the omni-image into a perspective view image, the resulting image, which we call an *unwarped image*, will suffer a serious geometric distortion [11], especially when the structure misalignment is large in some camera design cases for extension of the FOV. So, in the case of non-SVP unwarping, we need another way to conduct the unwarping work effectively.

Another example of requiring perspective camera calibration in real applications is the design of an image-based laser-pointer pointing system [14][15], as shown in Fig. 1.5, where the image sequence of a projection screen is analyzed to find the image position of the laser spot. This position is then transformed into a position of the display coordinate system to represent the location of the screen cursor. So, we can use the laser pointer instead of the function of a mouse. In this system, the accuracy of the pointed location

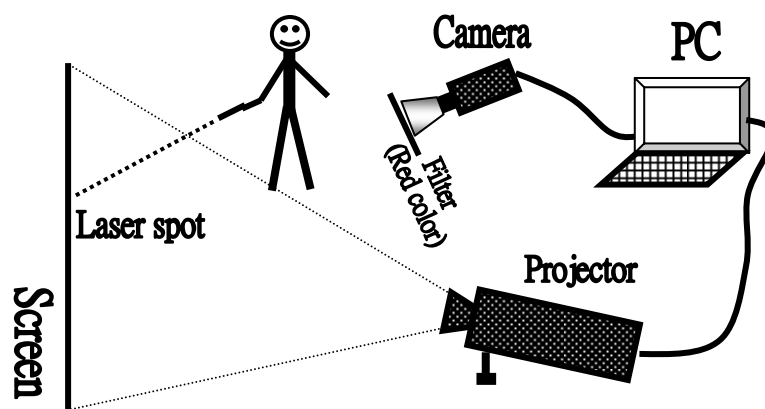


Fig. 1.5 A laser pointer pointing system.

relies on the precision of camera calibration. In this study, we propose a simple, accurate, and robust calibration method by using three different calibration patterns.

Another more example of perspective camera application for pointing system is trying use the camera directly as a computer cursor control tool. In this study, we propose a so called “camera mouse” with vision-based method using a video camera held in hand and operated in the air.

In the following sections, surveys of related researches are given in Section 1.1. The contributions of this study and the organization of this dissertation are reported in Sections 1.2 and 1.3, respectively.

1.1 Survey of Related Works

In Section 1.1.1, concepts of camera calibration are presented. In Section 1.1.2, structures of various omni-cameras are introduced. Approaches to unwarping an omni-image into a normal view image are described in Section 1.1.3. In Section 1.1.4, some pointing methods operated in the air are introduced for comparisons with our proposed camera mouse.

1.1.1 Concept of Camera Calibration

The technique of building the relationship between the coordinate system of a working space and the coordinate system of a captured image is called camera calibration, as mentioned previously. For different purposes, images are captured by different camera designs. Fig. 1.6 shows the calibration works involved in different types of camera designs.

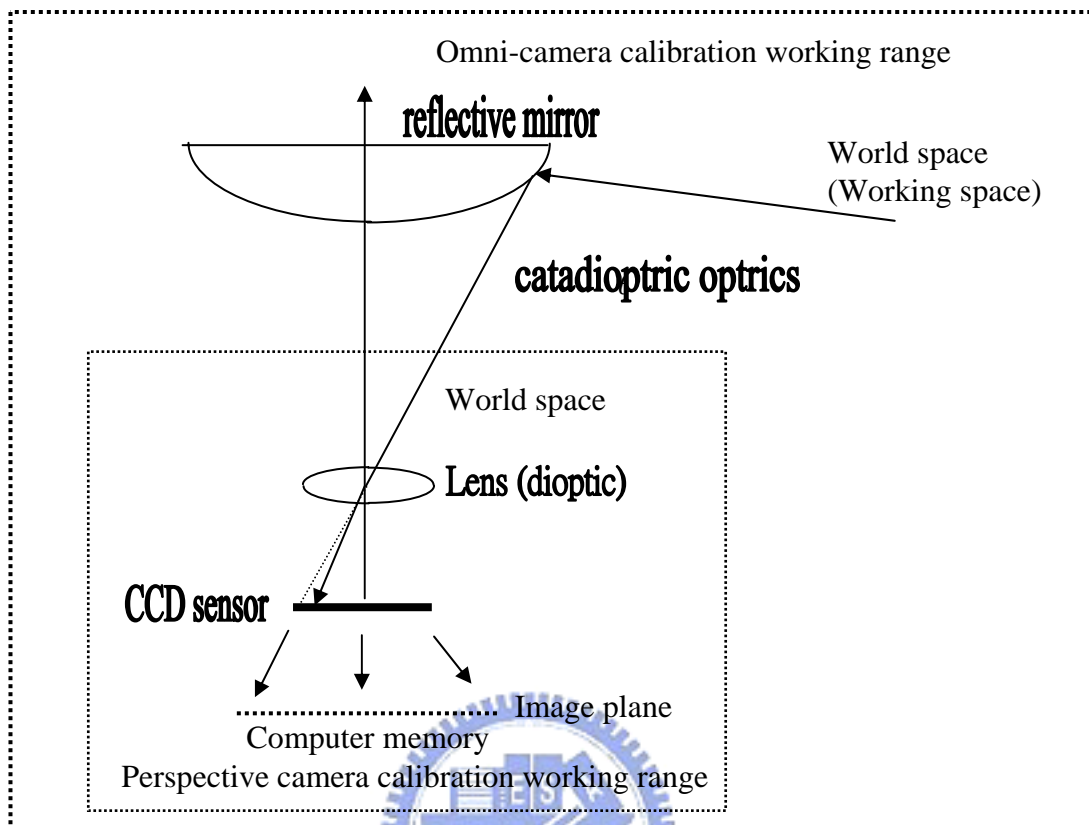


Fig. 1.6 Calibration work ranges for different types of camera design.

As mentioned above, an omni-camera has a large FOV and is suitable for applications in visual surveillance. A perspective camera is enough for applications in a laser-pointer pointing system. The calibration methods are different for the above two types of cameras, but from the top view of concept, the purposes of camera calibration are the same and are to build the relationship between the working space and the image plane. For example, in Fig. 1.3, the relationship is between the view plane and the omni-image. In Fig. 1.5, the relationship is between the display coordinate system on the screen and the image captured by the camera.

After calibration, a set of constant values called *calibrated parameters* (or *system parameters*) can be used for identifying any corresponding point pair between the image and the working space. We call this identifying work as *image transformation* in this study.

So, unwarping of an omni-image and transformation of a laser spot location by the captured image both belong to the work of image transformation.

More specifically, omni-camera calibration is an extension of traditional calibration of a video camera by including the consideration of the reflective mirror effect. In this study, new approaches are proposed for image unwarping for omni-cameras and for cursor location for pointing systems.

1.1.2 Survey of Omni-Cameras

As mentioned before, an omni-camera [2] is a specially designed camera system with its FOV beyond a hemisphere, which captures the camera view to form a highly geometrically distorted omni-image. Fig. 1.7 shows the different FOVs of a traditional perspective camera, a fish-eye camera, and a catadioptric camera, respectively. The image captured by a perspective camera is a normal geometrical distortion-free picture, but limited in a narrow range of FOVs. The images captured by a fish-eye or a catadioptric camera have large FOVs but with highly geometric distortion. So, they are not suitable for human observation. Usually, in visual surveillance applications, we use the benefit of the large FOV of the omni-image to locate interesting objects and unwarped a portion of the omni-image containing this object into a virtual perspective-view image.

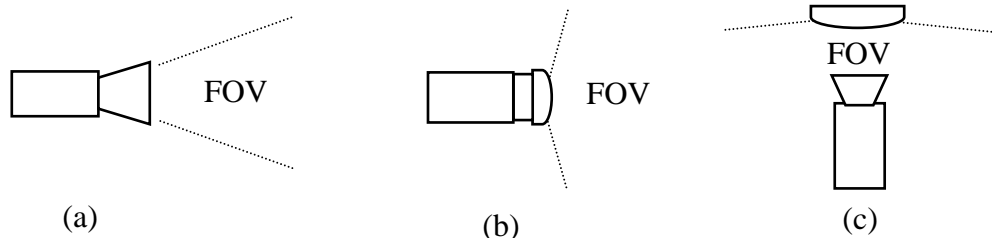


Fig. 1.7 FOVs of different type of cameras. (a) Perspective camera. (b) Fish-eye camera. (c) Catadioptric camera.

Fig. 1.8 shows an example of image unwarping for a non-SVP catadioptric camera in Mashita [11]. Here in this study, an interesting region of an omni-image is unwarped into a perspective-view image. Fig. 1.9 shows an example of image unwarping for a fish-eye camera found in [3]. Here the entire portion of an omni-image is unwarped into a perspective-view image by a method proposed in [3] with some calibrated parameters.

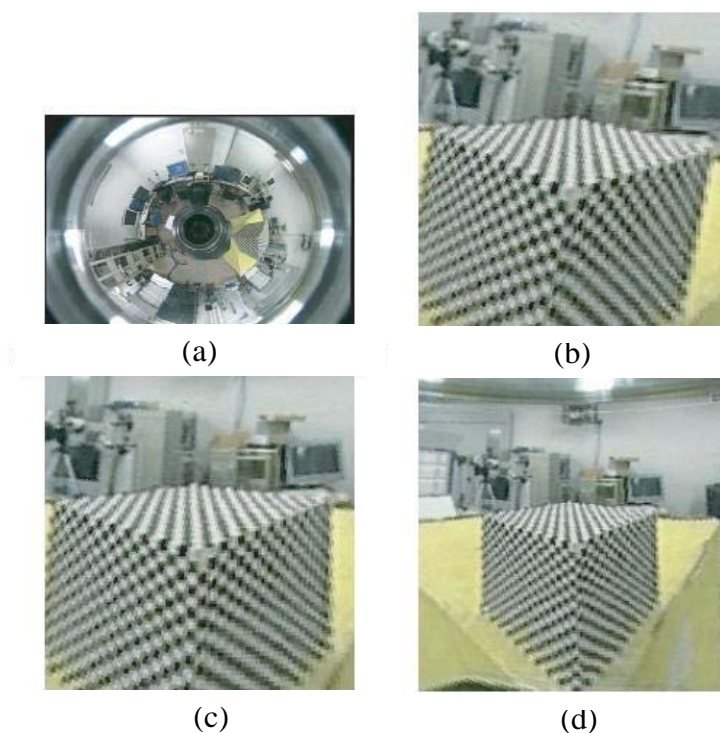


Fig. 1.8 Unwarped perspective view images using Mashita's calibration method [11].
(a) Original omni-image. (b) An unwarping result by treating the camera as an SVP omni-camera. (c) Best result by manually adjusting some parameters in the case of (b). (d) An unwarping result using calibrated parameters.

In this study, we focus on the study of image unwarping for catadioptric omni-cameras. A catadioptric omni-camera is a combination of a reflective mirror and a CCD camera as shown in Fig. 1.2 (b). The mirror surface of a catadioptric omni-camera may be in various shapes. The lens of the CCD camera may be of a perspective or orthographic projection type. As mentioned previously, if all the reflected light rays pass

through a common point, the omni-camera is of the SVP type; otherwise, of the non-SVP type. Only some combinations of mirror and lens designs can fit the SVP condition, and the others are non-SVP. Fig. 1.10 shows three types of catadioptric cameras, in which the omni-camera with the spherical mirror always exhibits the non-SVP property.

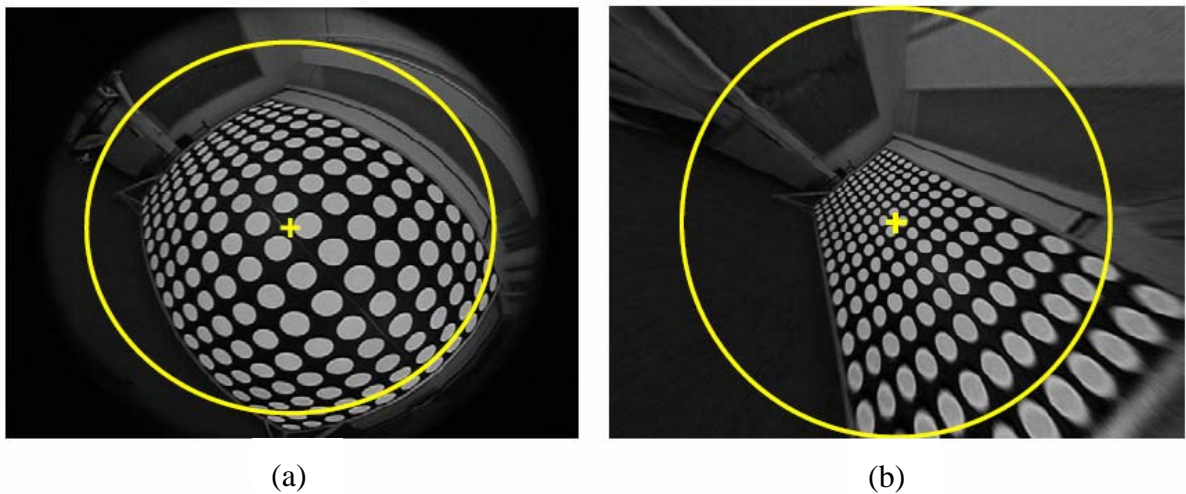


Fig. 1.9 Example of unwarping image of a fish-eye camera. (a) Original image. (b) Perspective view image unwarping from (a).
(Graph source: J. Kannala and S. Brandt [3])

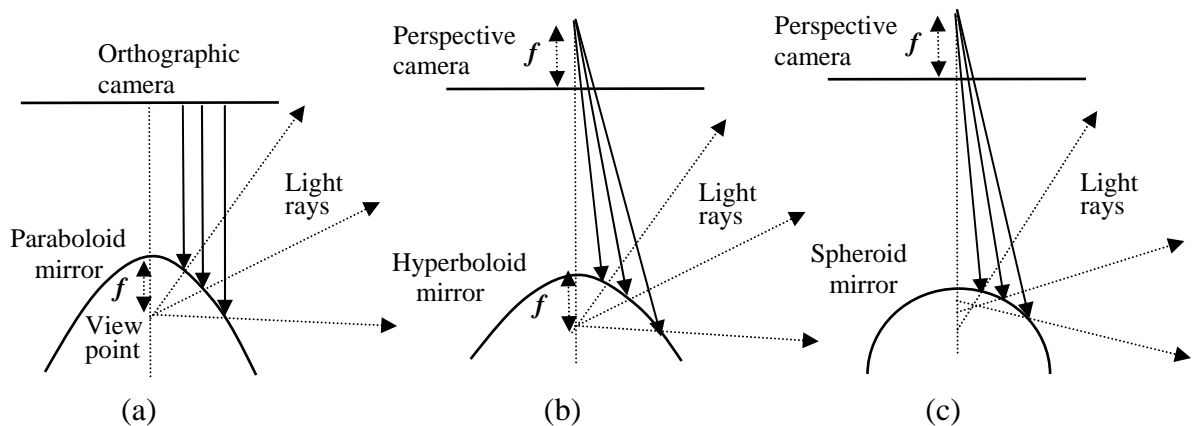


Fig. 1.10 Three types of catadioptric cameras. (a) Paraboloid mirror system. (b) Hyperboloid mirror system. (c) Spheroid mirror system. (The spheroid does not have a single viewpoint.)

1.1.3 Survey of Approaches to Unwarping Omni-images into Normal-view Images

The objective of unwarping an omni-image into a perspective or panoramic one is to provide a normal-view image for comfortable human observation, especially in the application of visual surveillance.

(a) Unwarping of SVP-type omni-image

Unwarping an omni-image taken with an SVP catadioptric camera into a normal-view image is a process of forward projection from a point X_p in a certain view scope in the world space to an omni-image point X_i , which can be described by $X_i = h(X_p)$ with h being a one-to-one mapping function from the world space to the omni-image plane. Eq. (1.1) below is the detailed description of this mapping for an SVP-type hypercatadioptric camera [7][8] as shows in Fig. 1.3:

$$u = \frac{f(b^2 - c^2)x}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}}, v = \frac{f(b^2 - c^2)y}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}} \quad (1.1)$$

where f is the focal length of the camera lens, and a , b and c are the parameters of the hyperbolic curve of the mirror surface described as follows:

$$z = -c + b\sqrt{1 + \frac{r^2}{a^2}}, \quad r^2 = x^2 + y^2 \quad (1.2)$$

with $c = \sqrt{a^2 + b^2}$.

Consequently, after scanning all the points (x, y, z) in the view scope in the world space to get the corresponding points (u, v) in the omni-image in the unwarping process, there will be no unfilled pixel in the resulting unwarped image because of the one-to-one mapping property.

For an SVP type of paracatadioptric camera [5] as shown in Fig. 1.11, the mapping may be described as follows:

$$u = \frac{-fa^2x_p}{(b^2 + c^2)z_p - 2bc\sqrt{x_p^2 + y_p^2 + z_p^2}}, v = \frac{-fa^2y_p}{(b^2 + c^2)z_p - 2bc\sqrt{x_p^2 + y_p^2 + z_p^2}} \quad (1.3)$$

where $f = h/2$, and $c = \sqrt{a^2 + b^2}$, and a and b are the parameters of the parabolic curve of the mirror.

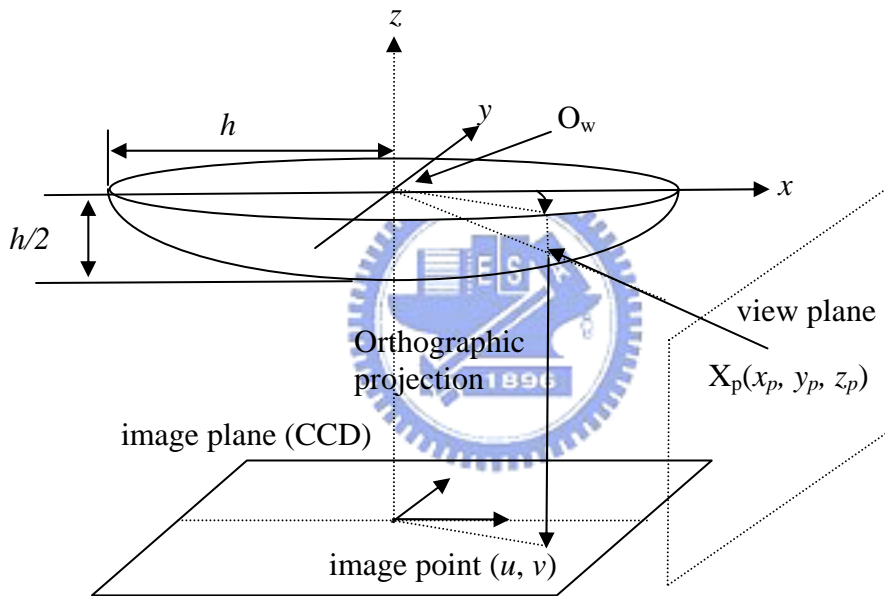


Fig. 1.11 A paracatadioptric camera, where O_w is the origin of the world coordinate system (also the focus of the parabolic curve).

(b) *Unwarping of non-SVP-type omni-image*

For a non-SVP catadioptric camera, there exists no direct one-to-one mapping $X_i = h(X_p)$ from X_p to X_i , and the relationship between X_p and X_i instead is $X_p = g(f(X_i))$ with f being a mapping from the omni-image plane to the mirror surface and g another mapping from the mirror surface to the world space. The inverse forms g^{-1} and f^{-1} of the mappings are too complicated to obtain such that the direct mapping $X_i = h(X_p) = f^{-1}(g^{-1}(X_p))$ is

unavailable. Therefore, it is impossible to conduct the same unwarping task as in the SVP case. Usually, some direct ray tracing techniques are used to induce the relationships between an incident ray in the world space and a point in the image plane in the non-SVP case. Mashita et. al. [11] proposed a calibration method for misaligned hypercatadioptric camera by using optical reflection law and a technique of ray tracing of optics, which is similar to the idea in [12]. Some experimental results of image unwarping in [11] are shown in Fig. 1.8.

1.1.4 Survey of pointing methods using video cameras

One way to implement a computer cursor function for pointing purpose is using video camera to track object motion or posture, for example, hand or head of a user. Nesi et. al. [49] proposed a vision-based, glove- and mark-free hand tracking system which is based on stereo vision. The system allows the simultaneous hand-position tracking and the recognition of some hand postures, thus implementing a true non-constrictive 3-D mouse. Betke et. al. [58] proposed a system which can track the computer user's movements with a video camera and translates them into the movements of the mouse pointer on the screen. Body features such as the tip of the user's nose or finger can be tracked. Fig. 1.12 shows an example application of proposed method in [58].



Fig. 1.12 The Camera Mouse user playing with educational software. [58]

The above methods use fixed cameras and track the object motion to calculate the corresponding cursor position in the display screen. We usually call these types of systems as outside-in vision-based mice. Another way to implement a computer cursor function is use a hand-held camera to view some fixed features of object (marks) near the display screen side to get the cursor position. Yang and Tsai [51] proposed an inside-out vision-based 3D mouse, which is a camera held by hand to view a square mark in front of the mouse. For many applications, 3D information is not necessary, and a “vision-based 2D mouse” is sufficient. In this study, we concentrate on the design of such a kind of mouse. More specifically, we hold a web camera in hand as the mouse and let it look at a computer monitor screen. After taking an image of the display screen, we use image processing techniques to detect and track appropriate features of certain artificially-attached landmarks attached on the monitor and the monitor screen corners, thus achieving the function of locating the cursor which is controlled by the in-air movement of the hand-held camera.

1.2 Contributions of This Study

The main contributions of this dissertation study are summarized in the following.

- (1) A systematic method is proposed to derive a set of new and general analytic equations for unwarping images taken from an omni-directional camera with a hyperbolic-shaped mirror. The generality of the proposed method so has extended the image-unwarping capability of the existing methods for the hypercatadioptric camera to tolerate lens/mirror assembly imprecision, which is difficult to overcome in most real applications.

- (2) A new method called “edge-preserving 8-directional two-layered weighting interpolation” is proposed for interpolating unfilled pixels in a perspective-view or panoramic image resulting from unwarping an omni-image taken by a non-single-view-point hypercatadioptric camera. This method can solve the problem of edge preserving in interpolating the input image which has many irregularly distributed unfilled pixels.
- (3) A unified approach to unwarping of omni-images into panoramic or perspective-view images is proposed. The approach does not adopt the conventional technique of calibrating the related parameters of an omni-camera. Instead, it is based on a new concept of pano-mapping table, which is created once forever by a simple learning process for an omni-camera of any kind as a summary of the information conveyed by all the camera parameters. With the help of the pano-mapping table, any panoramic or perspective-view image can be created from an input omni-image taken by the omni-camera according to an analytic computation process proposed in this study.
- (4) A robust and accurate calibration method for coordinate transformation between display screens and their images is proposed. We focus on improving the accuracy of the coordinate transformation to eliminate the shift errors near the image border. Also, we simplify the algorithms to avoid complicated calculations in the calibration and coordinate transformation processes.
- (5) A camera mouse with vision-based method for computer cursor control using a video camera held in hand in the air is proposed. The proposed method computes the cursor position with the help of four landmarks attached on the corners of the outer frame of the computer monitor. Some merits of the proposed method are: (a) the method requires no complicated camera calibration; (b) the method is reliable because of the

combined use of display feature detection and tracking; (c) the method is robust against loss of one or two landmark points in the tracking, which allows the user to have high freedom and space for hand movement; (d) the method allows unintended device rotation by affine transformation to correct the rotation error.

1.3 Dissertation Organization

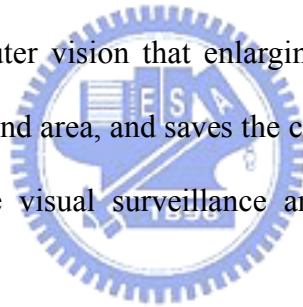
In the remainder of this dissertation, the proposed systematic method to derive a set of new and general analytic equations for unwarping images taken from an omni-directional camera with a hyperbolic-shaped mirror (called a hypercatadioptric camera) is described in Chapter 2. In Chapter 3, the proposed new method of “edge-preserving 8-directional two-layered weighting interpolation” for interpolating unfilled pixels in a perspective-view or panoramic image resulting from unwarping an omni-image taken by a non-SVP hypercatadioptric camera is described. In Chapter 4, the proposed unified approach to unwarping of omni-images into panoramic or perspective-view images is described. In Chapter 5, the proposed robust and accurate calibration method for coordinate transformation between display screens and their images is presented. In Chapter 6, the proposed camera mouse with the vision-based method for computer cursor control using a video camera held in hand in the air is described. Finally, conclusions and some suggestions for future research appear in Chapter 7.

Chapter 2

Analytic image unwarping for omni-directional cameras with hyperbolic-shaped mirrors

2.1 Introduction

It is well known in computer vision that enlarging the FOV of a camera enhances the visual coverage, reduces the blind area, and saves the computation time, of the camera system, especially in applications like visual surveillance and vision-based robot or autonomous vehicle navigation.



There are many ways to design a camera system consisting of CCD sensors, lenses, and mirrors to increase the FOV of the system [5]. An extreme way is to expand the FOV to a full hemisphere by the use of a *catadioptric camera*, which is an integration of a CCD sensor chip, a convex reflection mirror, and a projection lens. A popular name for this kind of camera, as mentioned previously, is *omni-camera*, and that for an image taken by it is *omni-image*. The surface curve of the reflection mirror in such a kind of camera may be conical, spherical, parabolic, or hyperbolic, and the lens may be of the type of orthographic or perspective projection. To simplify the process for unwarping omni-images into commonly-used perspective ones, it is usually desired to design an omni-directional camera in such a way that

the SVP constraint is satisfied [6]. Only some of the possible mirror/lens combinations can fit the SVP constraint, for examples, a combination of a parabolic mirror and a orthographic lens or that of a hyperbolic mirror and a perspective lens [6]. However, because of the difficulty in the alignment of the mirror and the camera lens, many commercial products do not satisfy the SVP constraint. When this constraint is not met, the resulting locus of viewpoints will form a so-called *caustic curve* [9]. In such a case, the image unwarping work is very complicated. On the other hand, when the parabolic mirror/orthographic lens combination is used; the resulting system is called a *paracatadioptric camera* [16]. Following this idea of naming the camera system, when the hyperbolic mirror/perspective lens combination is used, the resulting system is called a *hypercatadioptric camera* in this study. We deal with the image-unwarping problem for a hypercatadioptric camera in a non-SVP system in this study.

More specifically, we propose in this study a systematic method to calibrate the system parameters of a hypercatadioptric camera and derive accordingly a set of equations for accurate image unwarping. In the proposed calibration process, a calibration pattern of the shape of a thin ring is designed and attached at the border of the mirror as an aid. Next, mirror reflection laws as well as system geometry constraints are utilized to derive a set of mapping equations between a pixel in the image coordinate system and a point in the world space. The calibrated system parameters are used as known parameters in the derivation. The derived equations are then used to unwarped accurately an omni-image taken by a hypercatadioptric camera into a perspective-view image from any viewpoint.

A major contribution of this study is that the derived image unwarping equations are *analytic*. This is achieved for the first time. With these equations, unwarping of omni-images taken by a hypercatadioptric camera into perspective-view images will not be confined to the

SVP constraint. And this makes the applicability of the hypercatadioptric camera much wider to various computer vision problems.

The remainder of this chapter is organized as follows. In Section 2.2, we review some basic concepts about SVP omni-directional cameras and some previous works for omni-directional camera calibration. The camera calibration process proposed in this study is described in Section 2.3. In Section 2.4, the corresponding analytic image-unwarping equations are derived. In Section 2.5, some experimental results using simulation data as well as real images are given. Finally, we made a summary of this chapter in Section 2.6.

2.2 Review of Previous Works

For an SVP catadioptric camera, unwarping an omni-image into a perspective version is a process of *forward projection* from a point X_p on a certain perspective-view plane in the world space to an omni-image point X_i , which can be described by $X_i = h(X_p)$ with h being a *one-to-one* mapping function from the world space to the omni-image plane [7][21]. For example, for a SVP hypercatadioptric camera, the mapping relation between a point $X_p(x, y, z)$ in a world space and its projection point $X_i(u, v)$ in the image plane, as illustrated in Fig. 2.1, is as follows:

$$u = \frac{f(b^2 - c^2)x}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}}, v = \frac{f(b^2 - c^2)y}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}} \quad (2.1)$$

where f is the focal length of the camera lens, and a , b and c are the parameters of the hyperbolic curve of the mirror surface described as follows:

$$z = -c + b\sqrt{1 + \frac{r^2}{a^2}}, \quad r^2 = x^2 + y^2 \quad (2.2)$$

with $c = \sqrt{a^2 + b^2}$.

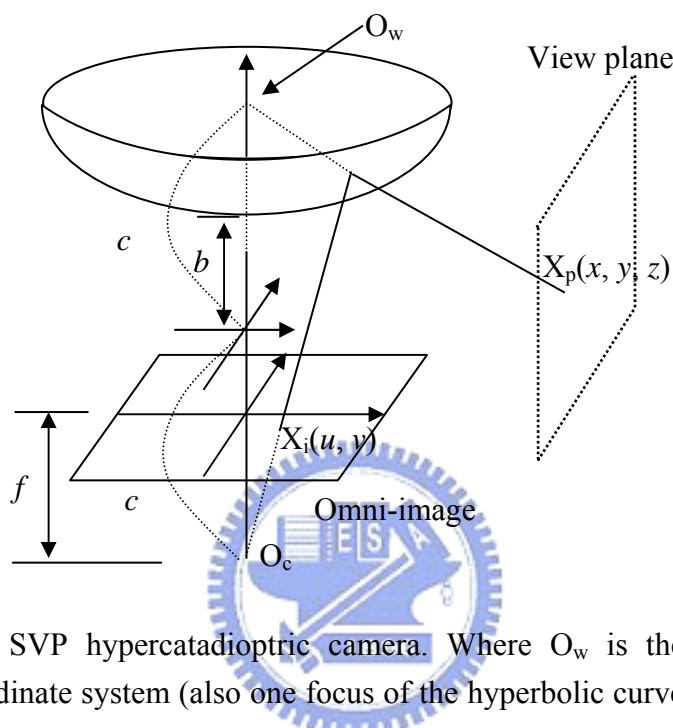


Fig. 2.1 An SVP hypercatadioptric camera. Where O_w is the origin of the world coordinate system (also one focus of the hyperbolic curve), and O_c is the optical center (another focus of the hyperbolic curve).

In practical situations, because of the existence of the geometric lens distortion, the projection point $X_i(u, v)$ in the image plane might be shifted erroneously. So, the real position of the point X_i in the image coordinate system should be calibrated by proper geometric correction even in the SVP case for accurate image unwarping. Some techniques about this can be found in [1][13] and are followed in this study. The details are omitted.

On the other hand, to estimate the intrinsic parameters of a paracatadioptric omni-directional camera system, a calibration procedure should be performed before unwarping omni-images into perspective-view ones. In [16][17], using a single view of three lines, Geyer et al. derived analytic calibration solutions for the focal length, the image center,

and the aspect ratio of a paracatadioptric camera. In [18], Kang used the consistency condition of pair-wise tracked point features across a sequence of paracatadioptric images to calibrate the same parameters. These approaches basically deal with the calibration problem of an SVP paracatadioptric camera, and misalignment between the mirror and the camera components (including the lens and the CCD sensor) was not considered. That is, the image plane was assumed to be parallel to the base plane of the mirror in these approaches, and only the intrinsic parameters of the cameras were taken into account in the calibration. The quality of the unwarped image is severely degraded when equations derived from a system configuration not meeting such an SVP assumption are used in the unwarping process, although the intrinsic parameters of the camera have been calibrated.

On the contrary, when a non-SVP camera is used, for example, for the reason to increase the FOV, system configuration parameters related to the pose of the mirror relative to the camera, in addition to the intrinsic camera parameters, need be calibrated. In [19], Aliaga developed a calibration model using a beacon-based pose estimation algorithm for a catadioptric camera which includes a parabolic mirror and a perspective lens. The mirror/lens combination in [19] is a non-SVP design, and the adopted camera model, like Tsai's [13], has eleven parameters (5 intrinsic and 6 extrinsic). But the physical meanings of Aliaga's extrinsic parameters are different from those of Tsai's, with the translation vector representing the offset between the center point of the mirror base plane and that of the image plane, and the rotation vector representing the orientation of the mirror base plane with respect to a world space system. Also, the mirror base plane is assumed to be parallel to the image plane. The calibrated data were used to estimate the pose of the camera with respect to the world space system.

A more complete calibration procedure for a catadioptric camera with a parabolic mirror and a perspective lens, which estimates the intrinsic camera parameters and the pose of the mirror relative to the camera, appeared in Fabrizio et al. [20]. The images of two circles on two planes existing in the mirror were used to calibrate the intrinsic camera parameters and the system configuration parameters. But no discussion was made about how to use the calibrated parameters to modify the mapping described by Eqs. (2.1) to get an accurate unwarped perspective-view image from an omni-image.

2.3 Proposed Method for Calibrating Camera Pose with Respect to Mirror

In this section, the proposed method for calibrating the camera pose with respect to the mirror of a hypercatadioptric camera system is described. The system configuration and the relationships among the involved coordinate systems are described first, and the proposed calibration process is presented next. The camera pose with respect to the mirror is derived finally, using the calibrated system parameters.

2.3.1 System Configuration and Coordinate System Relationships

The configuration of a hypercatadioptric camera and the related coordinate systems used in this study are depicted in Fig. 2.2. First, we define a world coordinate system with its origin W taken to be the middle point between the foci of the two arms of the hyperbolic curve defined by the mirror surface. Let b be the distance from W to the tip T_m of the mirror, c the distance from W to a focus O_m of an arm of the hyperbolic curve, h the height of the mirror (measured at T_m), and m the radius of the circular-shaped mirror base. Then, a point $M(x_m, y_m,$

z_m) on the mirror surface with respect to W can be described by the following equations according to Eq. (2.2):

$$z_m = b\sqrt{1 + \frac{r_m^2}{a^2}}, \quad r_m^2 = x_m^2 + y_m^2 \quad (2.3)$$

where $a = \sqrt{c^2 - b^2}$. The optical center O_c of the camera lens is taken to be the origin of the 3D camera coordinate system, and the optical axis of the camera is assumed to align with the z -axis of the world coordinate system. Accordingly, the center $O_i(u_0, v_0)$ of the 2D image coordinate system, which is the projection point of the optical axis on the image plane described by $z = f$, is $(0, 0)$. The mirror parameters a, b, h , and m , and the physical size of the CCD sensor may be obtained from the specifications of the hypercatadioptric camera.

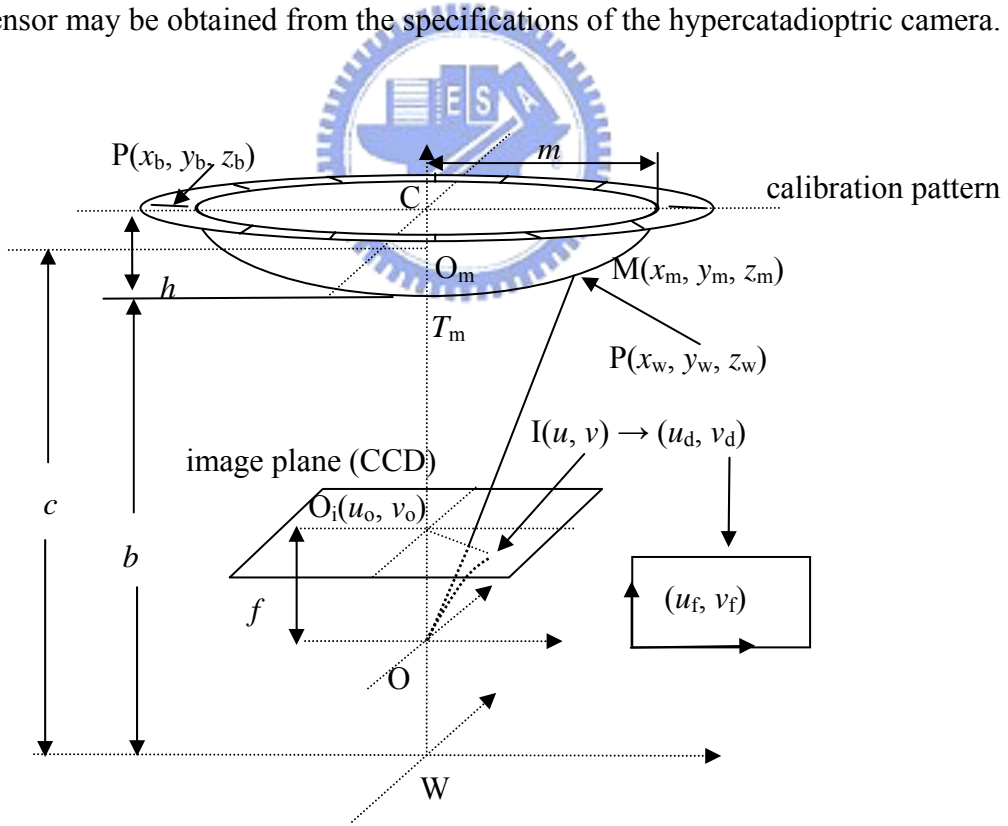


Fig. 2.2 The configuration of a hypercatadioptric camera used in this study.

Next, we define a *base coordinate system* on the mirror with its origin taken to be the

center C of the bottom circle of the mirror. The base plane of the mirror is located at the plane $z = 0$ of the base coordinate system. A point $P(x_b, y_b, z_b)$ on the ring-shaped calibration pattern on the base plane with respect to the origin of the camera coordinate system can be expressed as follows:

$$[x \ y \ z]^T = R[x_b \ y_b \ z_b]^T + T \quad (2.4)$$

where R is a 3×3 rotation matrix with three rotation angles ϕ (pitch), θ (yaw), and ψ (tilt) around the x -, y -, and z -axes of the base coordinate system, respectively, and T is a translation vector described by $T = [T_x \ T_y \ T_z]^T$. Eq. (2.4) represents a relationship from the base coordinate system to the camera coordinate system. We will transform the relationship into one from the camera coordinate system to the base coordinate system in Section 2.3.3, which represents the pose of the camera with respect to the mirror.

On the other hand, the location of the projection point $I(u, v)$ in the image plane of a point $P(x, y, z)$ in the camera coordinate system can be described as follows:

$$u = f \frac{x}{z}, \quad v = f \frac{y}{z}. \quad (2.5)$$

To correct possible geometric distortion of the lens in the radial direction, the following distortion model [13] is adopted in this study:

$$u_d = u + D_x, \quad v_d = v + D_y \quad (2.6)$$

where u_d and v_d are the shifted versions of u and v in the image coordinate system, and D_x and D_y are the amounts of distortion estimated, according to [13], by

$$D_x = \kappa u_d r^2, \quad D_y = \kappa v_d r^2 \quad (2.7)$$

with $r^2 = u_d^2 + v_d^2$ and κ being the *radial distortion factor* of the lens. Combining the above

equations, we get the following equations:

$$u = (1 - \kappa r^2)u_d, \quad v = (1 - \kappa r^2)v_d. \quad (2.8)$$

In the sequel, (u, v) will be called *ideal image coordinates*, and (u_d, v_d) *distorted image coordinates*. Finally, since the unit of the image coordinates (u_f, v_f) used in the computer, called *computer image coordinates* hereafter, is “pixel” for discrete images kept in the computer, additional relations between the distorted image coordinates (u_d, v_d) and the computer image coordinates (u_f, v_f) must be specified, which may be described by:

$$u_f = S_x u_d + C_x, \quad v_f = S_y v_d + C_y \quad (2.9)$$

where S_x and S_y are the *coordinate scaling factors* for the x and y directions, respectively, and (C_x, C_y) are the coordinates of the *origin* of the computer image coordinate system. Here, S_x and S_y , and (C_x, C_y) are some parameters related to the physical properties of the CCD sensors and the computer memory, respectively.

2.3.2 Proposed Calibration Process for Estimating Pose Parameters with Respect to Camera

As mentioned previously, we draw a calibration pattern on a paper ring and attach the ring on the mirror mount around the mirror border for use in the subsequent calibration process. The shape of the calibration pattern consists of an inner circle with a diameter equal to that of the mirror, as well as 16 black marks of short line segments evenly distributed around the circle border. Each short line segment has an end point on the inner circle of the ring, which we call a *calibration point*. The configuration is shown in Fig. 2.3. An image of this calibration pattern is shown in Fig. 2.4. It is noted that only 12 marks are visible in the FOV of the camera.

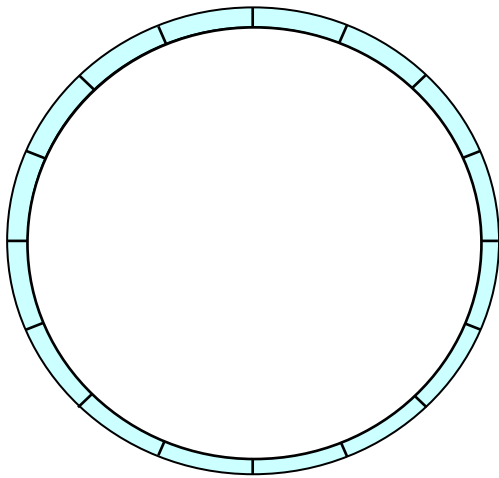


Fig. 2.3 The calibration pattern designed for use in this study.



Fig. 2.4 An omni-image of the calibration pattern.

The proposed calibration process in this study includes the following major steps.

(1) Acquisition of calibration pattern images

At the beginning of the calibration process, an image of the calibration pattern is taken. An example of calibration pattern images is shown in Fig. 2.4.

(2) Identification of calibration points

The calibration points on the base plane of the calibration pattern are then identified in the image. Let the coordinates of their projection points in the computer image coordinate system be denoted as (u_{fi}, v_{fi}) , $i = 0, 1, \dots, n$. On the other hand, the base coordinates (x_{bi}, y_{bi}, z_{bi}) of the calibration points are known in advance, with all the values of z_{bi} being equal to zeros because the points are located on the base plane.

(3) Computation of physical parameters

Let the image size in the computer image coordinate system be $w_i \times h_i$ and the CCD sensor size be $w_s \times h_s$. Then the parameters S_x, S_y and (C_x, C_y) in Eqs. (2.9) are calculated in this study

in the following way:

$$S_x = \frac{w_i}{w_s}, \quad S_y = \frac{h_i}{h_s}, \quad C_x = \frac{w_i}{2}, \quad C_y = \frac{h_i}{2}. \quad (2.10)$$

(4) Computation of intrinsic and extrinsic parameters

The extrinsic parameters R and T in Eq. (2.4), the intrinsic parameters f in Eqs. (2.5), and the radial distortion factor κ in Eqs. (2.8) should be estimated by a certain calibration method. This is accomplished in this study according to the method proposed in [13]. The steps are sketched here. First, from Eqs. (2.9) we get the distorted image coordinates (u_{di}, v_{di}) of a calibration point in the computer image coordinate system as follows:

$$u_{di} = \frac{u_{fi} - C_x}{S_x}, \quad v_{di} = \frac{v_{fi} - C_y}{S_y} \quad (2.11)$$

where (u_{fi}, v_{fi}) are the corresponding computer image coordinates. Next, we combine Eqs. (2.4) through (2.11) to derive the following equations:

$$u_{di}(1 + \kappa r_i^2) = \frac{(r_{11}x_{bi} + r_{12}y_{bi} + r_{13}z_{bi} + T_x)f}{r_{31}x_{bi} + r_{32}y_{bi} + r_{33}z_{bi} + T_z}, \quad (2.12-1)$$

$$v_{di}(1 + \kappa r_i^2) = \frac{(r_{21}x_{bi} + r_{22}y_{bi} + r_{23}z_{bi} + T_y)f}{r_{31}x_{bi} + r_{32}y_{bi} + r_{33}z_{bi} + T_z}, \quad (2.12-2)$$

where $r_i^2 = u_{di}^2 + v_{di}^2$. With sufficient known pairs of (u_{di}, v_{di}) and (x_{bi}, y_{bi}, z_{bi}) , $i = 0, 1, \dots, n$, we can solve R, T, κ from Eqs. (2.12) by Tsai's single view coplanar calibration method [13]. The parameter f is assumed available from the camera specifications.

Fig. 2.5 shows a calibration result of the pose of the base plane with respect to the camera, which includes the values $(-2.99, 0.96, 88.67)$ of the translation vector T in the unit of mm

and the values (0.013, 0.035, 0.007) of the three rotation angles ϕ , θ , and ψ of the rotation matrix R in the unit of radian. The real coordinates of the 12 calibration points are described by the square-bracketed coordinates $[x_i, y_i]$ in Fig. 2.5. After the calibration, the detected image coordinates of the calibration points are back-projected onto the base plane, the results are described by the angle-bracketed coordinates $\langle x_i, y_i \rangle$, which are also shown in Fig. 2.5.

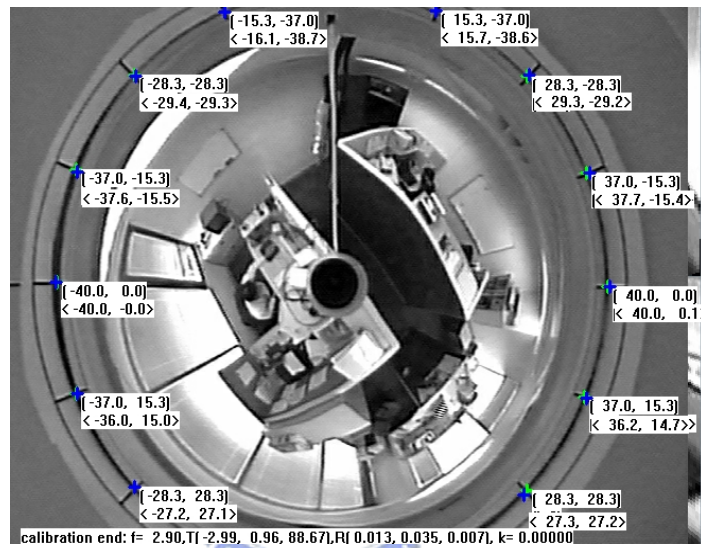


Fig. 2.5 The calibration result of a hypercatadioptric camera used in this study.

2.3.3 Proposed Calibration Process for Deriving Pose Parameters with Respect to Mirror

The pose of the base plane with respect to the camera is composed of the rotation matrix R and the translation vector T derived above. To obtain the pose of the camera with respect to the mirror, we have to transform Eq. (2.4) into a form similar to those specified in Eqs. (2.1). The origin of the mirror coordinate system is defined at one focus of the hyperbolic mirror surface (denoted by O_m in Fig. 2.2). The mirror plane $z = 0$ is taken to be parallel to the base plane at a distance of $d = (b + h) - c$. The z -axis of the mirror coordinate system is aligned with the z -axis of the base coordinate system.

It is known that R has the rotation angles (ϕ, θ, ψ) with respect to the x -, y -, and z -axes

respectively, and T has the values (T_x, T_y, T_z) . To map a point (x, y, z) in the camera coordinate system into a point (x_b, y_b, z_b) in the base coordinate system, the following equation may be applied:

$$\begin{bmatrix} x_b \\ y_b \\ z_b \end{bmatrix} = \begin{bmatrix} r'_{11} & r'_{12} & r'_{13} \\ r'_{21} & r'_{22} & r'_{23} \\ r'_{31} & r'_{32} & r'_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} - \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (2.13)$$

where the new rotation matrix

$$R' = \begin{bmatrix} r'_{11} & r'_{12} & r'_{13} \\ r'_{21} & r'_{22} & r'_{23} \\ r'_{31} & r'_{32} & r'_{33} \end{bmatrix} \quad (2.14)$$

is obtained by reversing the signs of (ϕ, θ, ψ) in Eq. (2.4).

Because the base plane and the mirror plane are apart with a distance of d , the coordinates (x_b, y_b, z_b) of a point in the base coordinate system with origin O_b are related to the coordinates (x_w, y_w, z_w) of a point in the mirror coordinate system with origin O_m by the following equalities:

$$x_w = x_b = xr'_{11} + yr'_{12} + zr'_{13} - T_x, \quad (2.15-1)$$

$$y_w = y_b = xr'_{21} + yr'_{22} + zr'_{23} - T_y, \quad (2.15-2)$$

$$z_w = z_b + d = xr'_{31} + yr'_{32} + zr'_{33} - T_z + (b + h) - c. \quad (2.15-3)$$

So, the position (x_{cw}, y_{cw}, z_{cw}) of the camera origin O_c in the mirror coordinate system may be derived from that of the mirror origin O_m by setting (x, y, z) in Eqs. (2.15) to $(0, 0, 0)$:

$$x_{cw} = -T_x, \quad y_{cw} = -T_y, \quad z_{cw} = -T_z + (b + h) - c. \quad (2.16)$$

Finally, given the coordinates (u, v, f) of a point I in the camera coordinate system where (u, v) is the ideal image coordinates of I, the corresponding coordinates (x_i, y_i, z_i) of I in the mirror coordinate system, according to Eqs. (2.15), may be derived to be:

$$x_i = ur'_{11} + vr'_{12} + fr'_{13} - T_x, \quad (2.17-1)$$

$$y_i = ur'_{21} + vr'_{22} + fr'_{23} - T_y, \quad (2.17-2)$$

$$z_i = ur'_{31} + vr'_{32} + fr'_{33} - T_z + (b + h) - c. \quad (2.17-3)$$

2.4 Back-Projection of Image Point

As a summary of the discussions in Section 2.3, we redraw the camera model as shown in Fig. 2.6 from the viewpoint of image projection. In Fig. 2.6, the angles of pitch ϕ_c , yaw θ_c , and tilt ψ_c are respectively the negative values of the calibrated rotation angles in Eq. (2.4). When the pose of the camera with respect to the mirror is determined in a way as described in Section 2.3.3, a point I(u, v) in the image plane can uniquely determine a reflective ray R_r from the mirror surface and so a corresponding mirror surface point M(x_m, y_m, z_m). In turn, at point M there will be an incident ray R_i corresponding to R_r with its incident orientation being determined by the mirror surface geometry. Let the direction of R_i be specified by a unit vector denoted by $\vec{w}_u = [w_x \ w_y \ w_z]^T$. In this section, we will derive a set of equations to specify a mapping F from the coordinates (u, v) of point I to the elements (w_x, w_y, w_z) of the unit vector \vec{w}_u . To be simple, we denote this mapping as $\vec{w}_u = F(I)$. This mapping is constrained, according to the optical reflection principle, by the following two rules.

(1) *Co-planarity constraint*: the unit normal \vec{n} of the mirror surface at point M and the two

rays, R_i and R_r , are co-planar.

(2) *Reflection constraint*: the *incident angle* of R_i is equal to the *reflection angle* of R_r .

In the sequel, all the derived formulas are based on the mirror coordinate system.

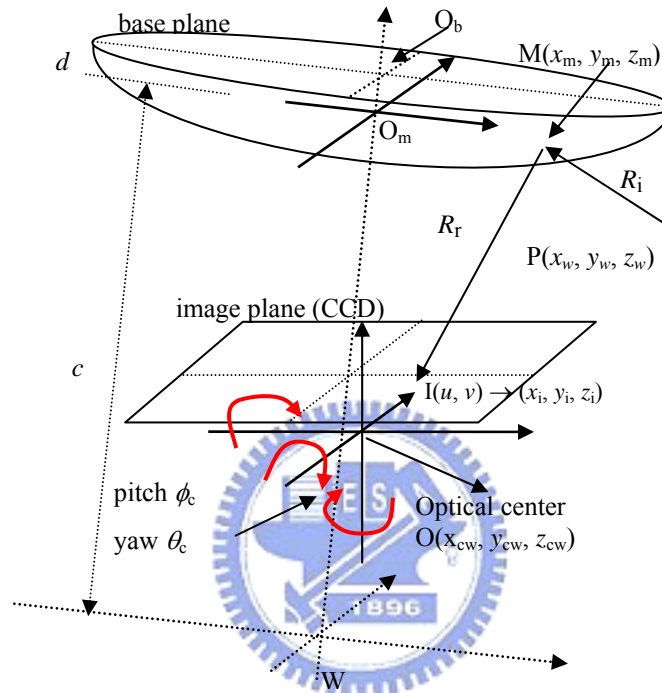


Fig. 2.6 The image projection model, where O_m is the origin of the mirror coordinate system, and O_b is the origin of the base coordinate system.

2.4.1 Derivation of Unit Normal Vector \vec{n}

Fig. 2.7 depicts the unit normal vector \vec{n} at point $M(x_m, y_m, z_m)$ on a plane passing through the z -axis of the mirror coordinate system with the tilt angle φ , denoted as P_n . The vector \vec{n} can be decomposed into two orthogonal vectors \vec{n}_m and \vec{n}_z , where the vector \vec{n}_m is on a plane P_M perpendicular to P_n located at $z = z_m$, and \vec{n}_z is parallel to the z -axis of the mirror coordinate system. The tilt angle by definition is equal to

$$\varphi = \tan^{-1} \frac{y_m}{x_m}. \quad (2.18)$$

On the other hand, we want to derive the equation of the mirror surface in the mirror coordinate system. Eqs. (2.3) describes the mirror surface in the world coordinate system. So, a shift $-c$ should be added to the z -value in Eqs. (2.3), resulting in

$$z_m = -c + b\sqrt{1 + \frac{r_m^2}{a^2}}, \quad r_m^2 = x_m^2 + y_m^2 \quad (2.19)$$

where r_m may be regarded as a polar coordinate composed of the coordinates of x_m and y_m .

Because the mirror surface is rotationally symmetric in the x - and y -directions, we can consider the *polar coordinates* (r_m, z_m) only, i.e., point M may be thought to be located at (r_m, z_m) . Also, let the tangent plane at point M perpendicular to \vec{n} be denoted as P_T , and let the intersection line of P_T and P_n be denoted as T_M . Now, the value of the angle δ of T_M with respect to the plane P_M at point M with polar coordinates (r_m, z_m) on the mirror surface may be derived, by taking the inverse tangent value of a partial derivative of z_m in Eqs. (2.19) with respect to r_m , to be

$$\delta = \tan^{-1} \frac{\partial z_m}{\partial r_m} = \tan^{-1} \frac{br_m}{a\sqrt{r_m^2 + a^2}} = \tan^{-1} \frac{b^2 r_m}{a^2 z_m}. \quad (2.20)$$

Accordingly, we can derive the values $\sin\delta$ and $\cos\delta$ as follows:

$$\sin \delta = \frac{br_m}{\sqrt{a^4 + c^2 r_m^2}}, \quad \cos \delta = \frac{a\sqrt{a^2 + r_m^2}}{\sqrt{a^4 + c^2 r_m^2}}. \quad (2.21)$$

Finally, it is not difficult to derive the unit normal vector at point $M(x_m, y_m, z_m)$ to be $\vec{n} = [\sin \delta \cos \varphi \quad \sin \delta \sin \varphi \quad -\cos \delta]^T$ according to the geometry shown in Fig. 2.7.

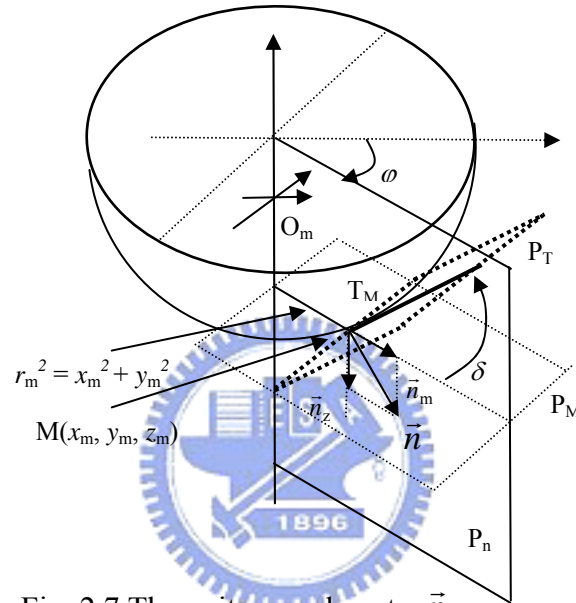


Fig. 2.7 The unit normal vector \vec{n} .

2.4.2 Use of Co-Planarity Constraint

The co-planarity constraint on the unit normal \vec{n} of the mirror surface at point M and the two rays, R_i and R_r , is shown in Fig. 2.8, and can be described by the following equality according to vector analysis:

$$(\vec{o} \times \vec{n}) \cdot \vec{w} = 0, \quad (2.22)$$

or equivalently,

$$\begin{bmatrix} i & j & k \\ (x_{cw} - x_m) & (y_{cw} - y_m) & (z_{cw} - z_m) \\ \sin \delta \cos \varphi & \sin \delta \sin \varphi & -\cos \delta \end{bmatrix} \begin{bmatrix} (x_w - x_m) \\ (y_w - y_m) \\ (z_w - z_m) \end{bmatrix} = 0, \quad (2.23)$$

where “ \times ” and “ \cdot ” denote the cross and inner product operators for vectors, respectively; $\vec{w} = [(x_w - x_m) \ (y_w - y_m) \ (z_w - z_m)]^T$ specifies the direction of the incident ray R_i ; $[i \ j \ k]^T$ is a unit vector; and $\vec{o} = [(x_{cw} - x_m) \ (y_{cw} - y_m) \ (z_{cw} - z_m)]^T$ specifies the direction of the reflection ray R_r . By computing the above matrix product and substituting the result with the following notations

$$K_{m1} = (y_m - y_{cw}) \cos \delta + (z_m - z_{cw}) \sin \delta \sin \varphi, \quad (2.24-1)$$

$$K_{m2} = (x_m - x_{cw}) \cos \delta + (z_m - z_{cw}) \sin \delta \cos \varphi, \quad (2.24-2)$$

$$K_{m3} = (x_{cw} - x_m) \sin \delta \sin \varphi - (y_{cw} - y_m) \sin \delta \cos \varphi, \quad (2.24-3)$$

$$x_n = (x_w - x_m), \quad y_n = (y_w - y_m), \quad z_n = (z_w - z_m), \quad (2.24-4)$$

we get

$$K_{m1}x_n - K_{m2}y_n + K_{m3}z_n = 0. \quad (2.25)$$

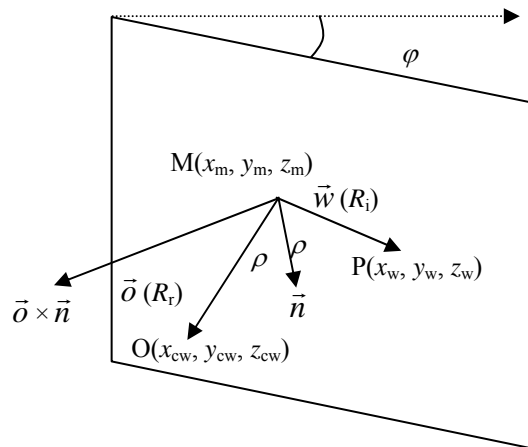


Fig. 2.8 The co-planar vectors and the cross product.

2.4.3 Use of Reflection Constraint

The aforementioned reflection constraint, which indicates the identicalness of the incident angle to the reflection angle, may be expressed by the following equalities:

$$\frac{\vec{w} \cdot \vec{n}}{\|\vec{w}\| \|\vec{n}\|} = \cos \rho, \quad \cos \rho = \frac{\vec{o} \cdot \vec{n}}{\|\vec{o}\| \|\vec{n}\|} \quad (2.26)$$

where ρ denotes the two identical angles. The second equality in Eqs. (2.26) may be expanded to be

$$\cos \rho = \frac{(x_{cw} - x_m) \sin \delta \cos \varphi + (y_{cw} - y_m) \sin \delta \sin \varphi - (z_{cw} - z_m) \cos \delta}{\sqrt{(x_{cw} - x_m)^2 + (y_{cw} - y_m)^2 + (z_{cw} - z_m)^2}}. \quad (2.27)$$

On the other hand, the three components of the unit vector $\vec{w}_u = \frac{\vec{w}}{\|\vec{w}\|} = [w_x \ w_y \ w_z]^T$, by definition, can be calculated as follows:

$$w_x = \frac{x_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}}, \quad w_y = \frac{y_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}}, \quad w_z = \frac{z_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}}. \quad (2.28)$$

Then, the first equality in Eqs. (2.26) can be derived to be as follows:

$$\begin{aligned} \frac{\vec{w} \cdot \vec{n}}{\|\vec{w}\| \|\vec{n}\|} &= \vec{w}_u \cdot \vec{n} \\ &= [w_x \ w_y \ w_z]^T \cdot [\sin \delta \cos \varphi \quad \sin \delta \sin \varphi \quad -\cos \delta]^T \\ &= w_x \sin \delta \cos \varphi + w_y \sin \delta \sin \varphi - w_z \cos \delta \\ &= \cos \rho \end{aligned} \quad (2.29)$$

where $\cos \rho$ can be computed by Eq. (2.27) above.

2.4.4 Calculating Direction of Incident Ray

If the values (x_n, y_n, z_n) are not equal to $(0, 0, 0)$, Eq. (2.25) may be rewritten as

$$K_{m1} \frac{x_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}} - K_{m2} \frac{y_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}} + K_{m3} \frac{z_n}{\sqrt{x_n^2 + y_n^2 + z_n^2}} = 0, \quad (2.30)$$

which is equivalent to

$$K_{m1} w_x - K_{m2} w_y + K_{m3} w_z = 0. \quad (2.31)$$

On another hand, the norm of the unit vector \vec{w}_u is equal to 1, i.e.,

$$w_x^2 + w_y^2 + w_z^2 = 1. \quad (2.32)$$

Using Eqs. (2.30), (2.31), and (2.32), we can solve the three unknown parameters w_x , w_y , and w_z in the following way.

First, we eliminate the unknown w_z in Eqs. (2.30) and (2.31) to get

$$w_y = A_m w_x + B_m \quad (2.33)$$

where

$$A_m = \frac{K_{m1} \cos \delta + K_{m3} \sin \delta \cos \varphi}{K_{m2} \cos \delta - K_{m3} \sin \delta \sin \varphi}, \quad B_m = \frac{-K_{m3} \cos \rho}{K_{m2} \cos \delta - K_{m3} \sin \delta \sin \varphi}. \quad (2.34)$$

Next, we eliminate the unknown w_y in Eqs. (2.30) and (2.31) to get

$$w_z = C_m w_x + D_m. \quad (2.35)$$

where

$$C_m = \frac{(K_{m1} \sin \varphi + K_{m2} \cos \varphi) \sin \delta}{K_{m2} \cos \delta - K_{m3} \sin \delta \sin \varphi}, \quad D_m = \frac{-K_{m2} \cos \rho}{K_{m2} \cos \delta - K_{m3} \sin \delta \sin \varphi}. \quad (2.36)$$

Finally, substituting Eqs. (2.33) and (2.34) into Eq. (2.32) and reducing the result, we get

$$w_x = \frac{-(A_m B_m + C_m D_m) \pm \sqrt{(A_m B_m + C_m D_m)^2 - (1 + A_m^2 + C_m^2)(B_m^2 + D_m^2 - 1)}}{(1 + A_m^2 + C_m^2)}. \quad (2.37)$$

There are two possible solutions for w_x , and using the relationship between the coordinates (x_i, y_i, z_i) of the image point and the coordinates (x_{cw}, y_{cw}, z_{cw}) of the optical center, all in the mirror coordinate system, we can determine one of them as the correct solution. The details are omitted here. After w_x is obtained, w_y and w_z can be computed accordingly by Eqs. (2.33) and (2.35).

2.4.5 Calculating Coordinates of Mirror Surface Point in Terms of Image Point Coordinates

In Sections 2.4.1 through 2.4.4, we have derived the elements (w_x, w_y, w_z) of the unit vector \vec{w}_u in terms of the coordinates (x_m, y_m, z_m) of the mirror surface point M. Here we further want to derive (x_m, y_m, z_m) in terms of the coordinates (u, v) of the image point I to complete the derivations of the formulas for specifying the mapping $\vec{w}_u = F(I)$. The coordinates (u, v) can be calculated from a point (u_f, v_f) in the computer image coordinate system by Eqs. (2.11) and (2.8). Also, the coordinates (x_i, y_i, z_i) of the image point I in the mirror coordinate system can be calculated from Eqs. (2.15) which are repeated in the following:

$$x_i = ur'_{11} + vr'_{12} + fr'_{13} - T_x, \quad (2.17-1)$$

$$y_i = ur'_{21} + vr'_{22} + fr'_{23} - T_y, \quad (2.17-2)$$

$$z_i = ur'_{31} + vr'_{32} + fr'_{33} - T_z + (b + h) - c. \quad (2.17-3)$$

Now, referring to Fig. 2.8, we see that both the tilt angle of the point I and that of its back-projection point M on the mirror surface relative to the camera coordinate system are equal. Let both angles be denoted by ϕ . Then, it is easy to see from the geometry in the figure that

$$\tan \phi = \frac{y_i - y_{cw}}{x_i - x_{cw}} = \frac{y_m - y_{cw}}{x_m - x_{cw}},$$

or equivalently, that

$$y_m = y_{cw} - x_{cw} \tan \phi + x_m \tan \phi. \quad (2.38)$$

Combining Eqs. (2.15) and (2.38) and using the following notations

$$K_1 = y_{cw} - x_{cw} \tan \phi, \quad (2.39-1)$$

$$K_2 = \frac{(z_i - z_{cw})}{x_i - x_{cw}}, \quad (2.39-2)$$

$$K_3 = z_{cw} + c - x_{cw} K_2, \quad (2.39-3)$$

$$K_4 = b^2 (1 + \tan^2 \phi) - a^2 K_2^2, \quad (2.39-4)$$

$$K_5 = b^2 K_1 \tan \phi - a^2 K_2 K_3, \quad (2.39-5)$$

$$K_6 = a^2 b^2 + b^2 K_1^2 - a^2 K_3^2, \quad (2.39-6)$$

we get, after some derivations and reductions, the following result for x_m :

$$x_m = \frac{-K_5 \pm \sqrt{K_5^2 - K_4 K_6}}{K_4}. \quad (2.40)$$

There are two possible solutions for x_m in the above equation, and we can get the correct one by checking the condition that x_m and x_i are at the same side with respect to x_{cw} , or

equivalently, that the value of the product $(x_m - x_{cw})(x_i - x_{cw})$ is larger than or equal to zero. Also, using Eq. (2.38), we can get y_m . And finally the value of z_m can be calculated from Eqs. (2.19) which are repeated as follows:

$$z_m = -c + b\sqrt{1 + \frac{r_m^2}{a^2}}, \quad r_m^2 = x_m^2 + y_m^2. \quad (2.19)$$

2.5 Experimental Results

We show in this section the experimental results of two unwarping cases with two different omni-images as inputs, one being a pseudo-image and the other a real image taken by our hypercatadioptric camera.

2.5.1 Unwarping of A Pseudo-image into Perspective Views

We first describe how we create the pseudo-image for the first unwarping experiment. For this purpose, we used the calibration data obtained in Section 2.3, which include the translation parameters $(-2.99, 0.96, 88.67)$ (in the unit of mm); the rotation angles $(0.013, 0.035, 0.007)$ (in the unit of radian); the radial distortion factor $\kappa = 0.0$; and the focal length $f = 2.9\text{mm}$. Then, we used the mapping equations obtained in Section 2.4 to warp a pseudo target as shown in Fig. 2.9 into the image plane to get the desired pseudo omni-image as shown in Fig. 2.10. The procedure was mentioned in Section 2.5.2, and the details are described in the following.

The pseudo target includes two parts, namely, a ground region with the area of $20 \times 20\text{m}^2$ and consisting of 400 grids with each being of the size of $1 \times 1\text{m}^2$, as well as an L-shaped wall with the height of 1.0m and a side width of 2.8m. The L-shaped wall is placed near the center

of the ground region. In simulating the image taking work, the target was laid under our hypercatadioptric camera and the normal vector of the ground region at the region center aligns with the z -axis of the mirror coordinate system. The distance of the region center from the origin of the mirror coordinate system is 2m. The resulting pseudo-image of Fig. 2.10 is of the size of 640×480 pixels.

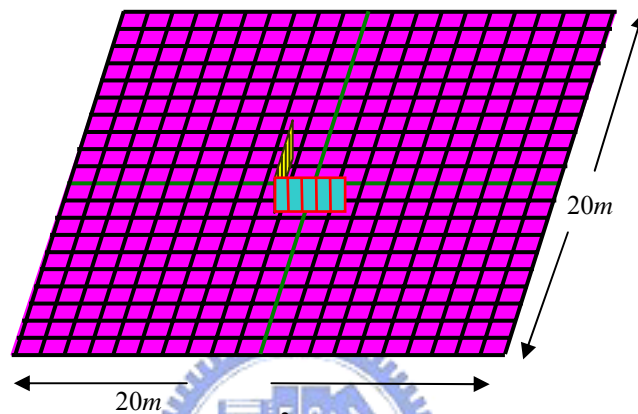


Fig. 2.9 A pseudo target of size $20 \times 20m^2$ with an L-shaped wall at the center position.

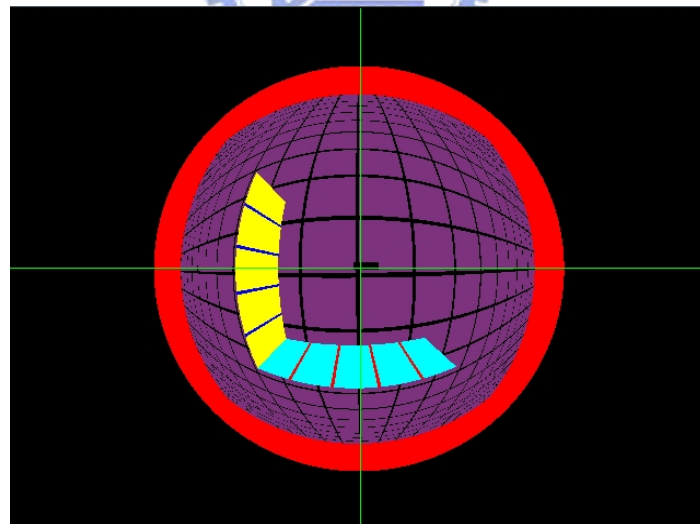


Fig. 2.10 The warped image of the pseudo target in Fig. 2.9.

We then describe how we unwarp the pseudo-image into perspective-view images. We selected two perspective-view planes, one being from a side view and the

other from the top view. As shown in Fig. 2.11, each perspective-view plane is a rectangular region, which was used to capture the rays back-projected from the image plane. Each rectangular region was divided into 320×240 units representing a 320×240 image.

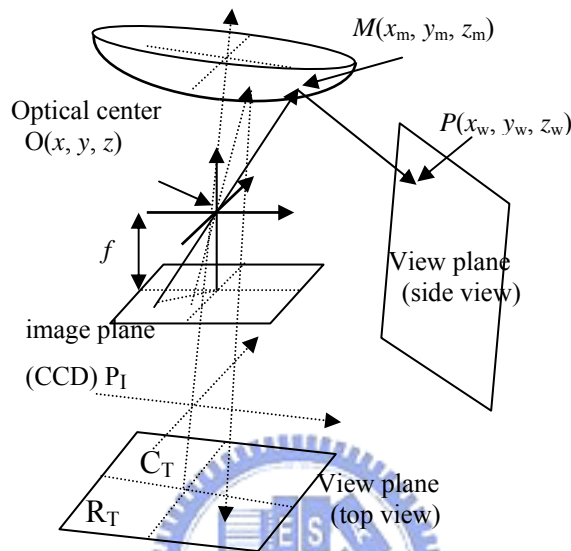


Fig.2.11 View planes defined in the real world for unwarped images.

The top-view region R_T is $4 \times 4 \text{m}^2$ in size, parallel to the x - y plane of the mirror coordinate system. The normal vector of R_T at the center C_T of R_T aligns with the z -axis of the mirror coordinate system, and C_T is 2m below the origin of the mirror coordinate system. Fig. 2.12(a), (b), and (c) are the unwarping results in R_T with different calibration parameter settings. Here, Fig. 2.12(a) was produced using the same translation and rotation parameters as those used in yielding Fig. 2.10. In Fig. 2.12(b), the three rotation angles were all set to be zero. And in Fig. 2.12(c), we further set the two translation parameters T_x and T_y to be zero (meaning perfect alignment of the camera with respect to the mirror). We can see in Fig. 2.12(a) that our derived equations can be used to unwarped the pseudo-image of Fig. 2.10 perfectly within the top-view region R_T . Fig. 2.12(b) and (c) tell us that insufficient calibration of the hypercatadioptric camera will produce distorted unwarping results.

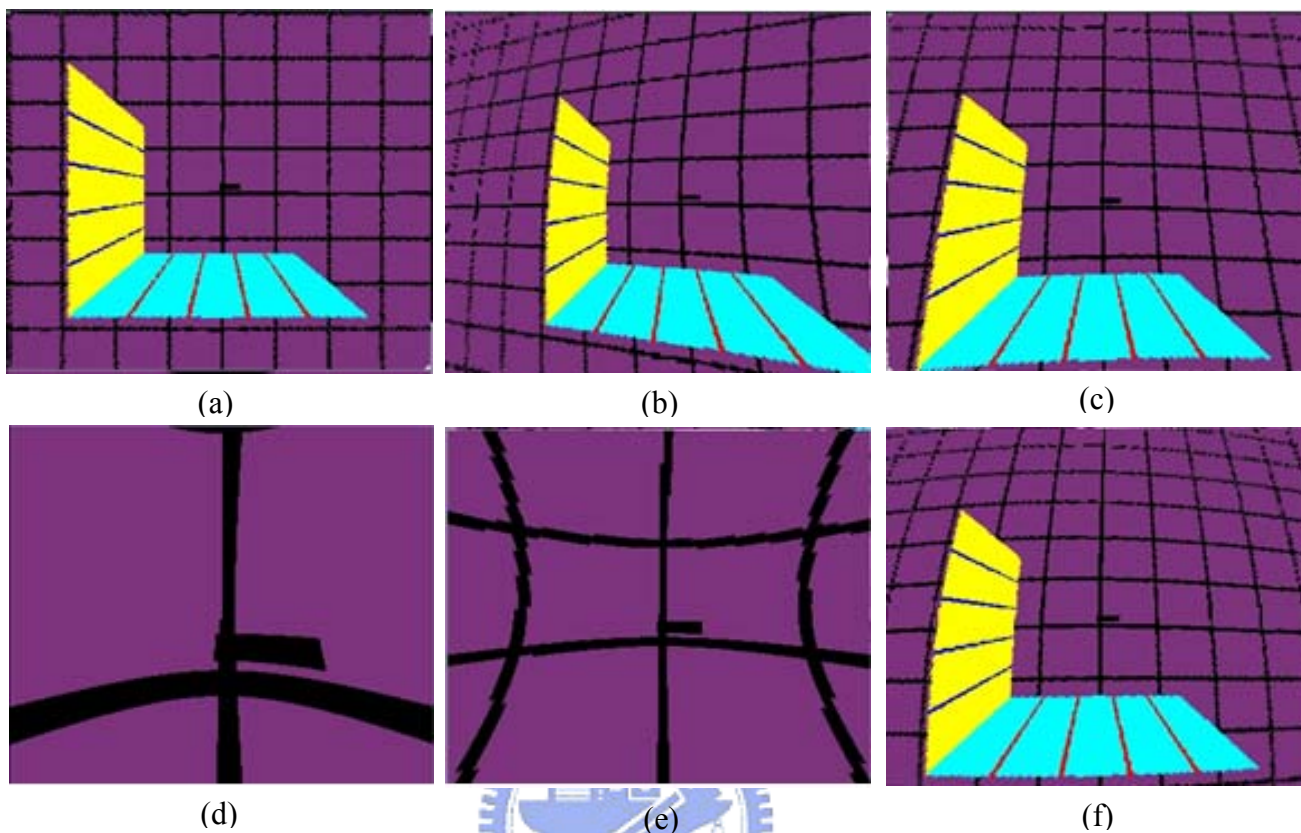


Fig. 2.12 Unwarped images of Fig. 2.10 (Top view). (a) has the same $T(T_x, T_y, T_z)$ and $R(R_x, R_y, R_z)$ as those for yielding Fig. 2.10, (b) has the setting $R_x = R_y = R_z = 0$, and (c) has the further setting $T_x = T_y = 0$. (d), (e) and (f) are the results using Eqs. (2.1) with different CCD sensor sizes of $3.2 \times 2.4 \text{ mm}^2$, $1.6 \times 1.2 \text{ mm}^2$, and $0.8 \times 0.6 \text{ mm}^2$, respectively.

On the other hand, we show some results coming from inappropriate unwarping of the input pseudo-image Fig. 2.10 using Eqs. (2.1) under the erroneous assumption that the camera is an SVP system. They are shown in Fig. 2.12(d), (e), and (f), which are the results coming from the uses of three different CCD sensors of sizes 3.2×2.4 , 1.6×1.2 , and $0.8 \times 0.6 \text{ mm}^2$, respectively. The actual size of our CCD camera sensor is the first one, namely, $3.2 \times 2.4 \text{ mm}^2$ and the corresponding unwarping result is the image shown in Fig. 2.12(d). But the visible scope of the image region in the figure is too small to show the entire unwarping result. For the reason of comparison, we therefore assume the other two sensor sizes for our camera to

yield Fig. 2.12(e) and (f) for the purpose of showing the unwarping results more clearly. Note that the sensor size settings of Fig. 2.12(e) and (f) are unreasonable for a real CCD sensor. From Fig. 2.12(f), it is obviously seen that the result is worse than the perfect one shown in Fig. 2.12(a).

Fig. 2.13(a) through (d) show the unwarping results on four perspective-view planes. Each perspective-view plane is set parallel to the z-axis of the mirror coordinate system with a view-angle span of 90° in the x - y plane and at a distance of $\sqrt{2}$ m from the z-axis of the mirror coordinate system. The unwarping result is projected into a region in each perspective-view plane with a height of 2m and a width of $2\sqrt{2}$ m. Fig. 2.13(a) through (d) are the unwarping results using our derived equations in the four perspective-view planes. The

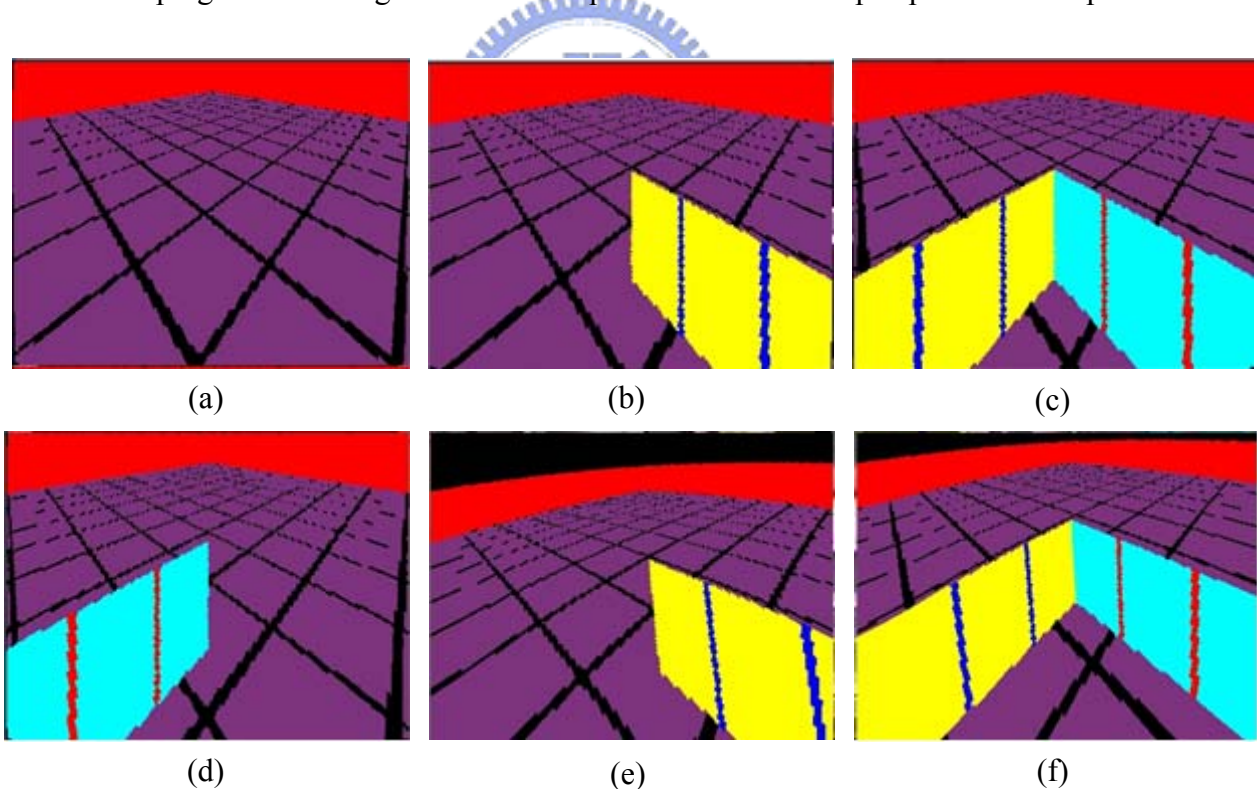


Fig. 2.13 Unwarped images of Fig. 2.10 from 4 side views. (a) to (d) are the results using the proposed method with the same T and R as those for yielding Fig. 2.10. (a) through (d) show the results with different span of view angle, $1.5\pi \sim 2.0\pi$, $1.0\pi \sim 1.5\pi$, $0.5\pi \sim 1.0\pi$, and $0.0\pi \sim 0.5\pi$, respectively. (e) and (f) are the results using Eq. (2.1) with different spans of viewing angle, as (b) and (c), respectively.

settings of the translation parameters and the rotation angles are the same as those for Fig. 2.12(a) through (d). Fig. 2.13(e) and (f) are the unwarping results by Eqs. (2.1) in the same perspective -view planes as those used for Fig. 2.13(b) and (c) but under the SVP assumption and with the CCD sensor size of $0.8 \times 0.6 \text{mm}^2$. Note especially that the vertical lines in Fig. 2.13(e) and (f) can be seen to be slanted erroneously.

2.5.2 Unwarping of A Real Image into Perspective Views

Fig. 2.14 and Fig. 2.15 are the unwarping results of a real image shown previously in Fig. 2.4 taken by our camera. The parameter settings used for computing Fig. 2.14 and Fig. 2.15 are the same as those used to produce Fig. 2.12 and Fig. 2.13, except that the pseudo-image is now replaced by the real image. Or more specifically, Fig. 2.14(a) through (f) correspond to Fig. 2.12(a) through (f), respectively, and Fig. 2.15(a) through (f) to Fig. 2.13(a) through (f), respectively. Comparing Fig. 2.14(a) with Fig. 2.14(f), and Fig. 2.15(b) and Fig. 2.15(c) with Fig. 2.15(e) and Fig. 2.15(f), respectively, we can see that the unwarping results obtained by our methods are better than those obtained under the SVP assumption. Especially, the vertical lines in Fig. 2.15(e) and (f) as well can be seen to be slanted. Such situations are not seen in our results in Figs. 2.15(b) and (c).

It is noted that the images of the above-mentioned experiments were obtained using some methods proposed in our previous paper [22], and the details are omitted here.

2.6 Summary

An approach to systematic calibration and analytic image unwarping for omni-directional non-SVP hypercatadioptric cameras with hyperbolic-shaped mirrors has been proposed. We used the calibrated parameters of the camera to derive precise unwarping equations. The derived equations have been validated to yield the same unwarping results as those yielded by a perfectly designed SVP camera by adjusting the calibrated parameters to fit the SVP constraint. Furthermore, we have shown the advantages of our method over the SVP-constrained method for real cameras by some simulation and experimental results.

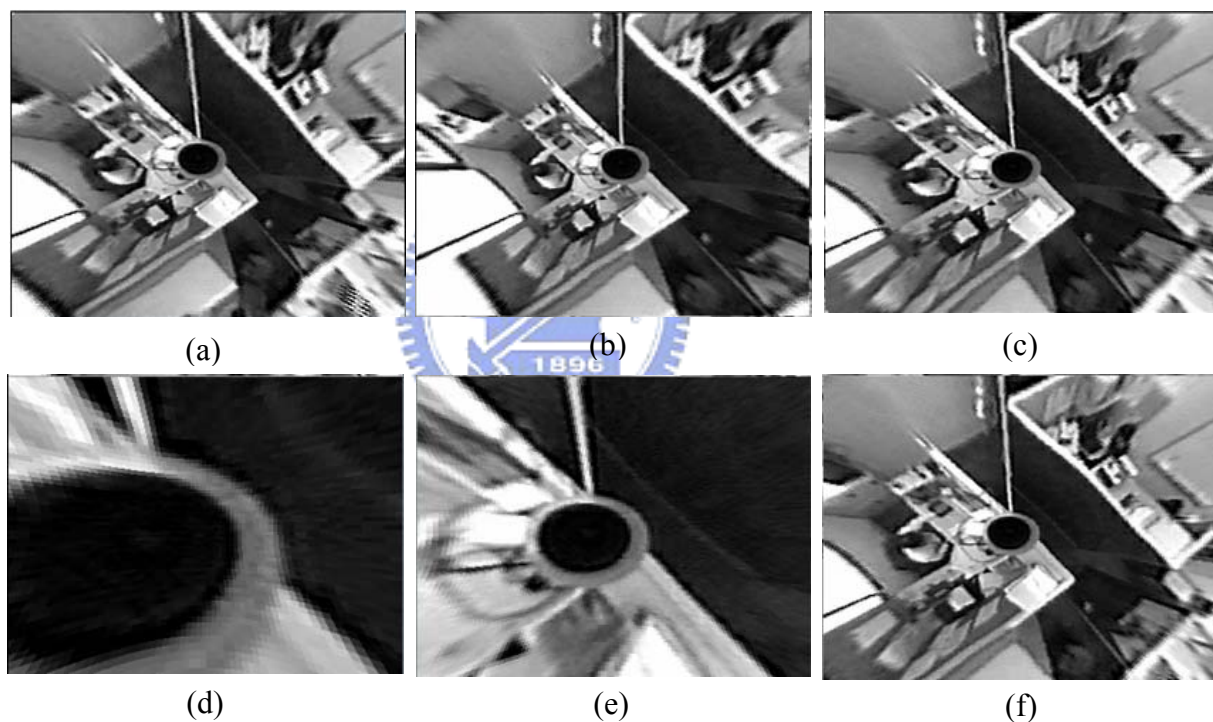


Fig. 2.14 Unwarped images of a real scene in Fig. 2.4 (Top view). (a) has the $T(T_x, T_y, T_z)$ and $R(R_x, R_y, R_z)$ mentioned in Section 3.2 after calibration, (b) has the setting $R_x = R_y = R_z = 0$, and (c) has the further setting $T_x = T_y = 0$. (d), (e) and (f) are the results using Eqs. (2.1) with different CCD sensor size of $3.2 \times 2.4 \text{mm}^2$, $1.6 \times 1.2 \text{mm}^2$, and $0.8 \times 0.6 \text{mm}^2$ respectively.

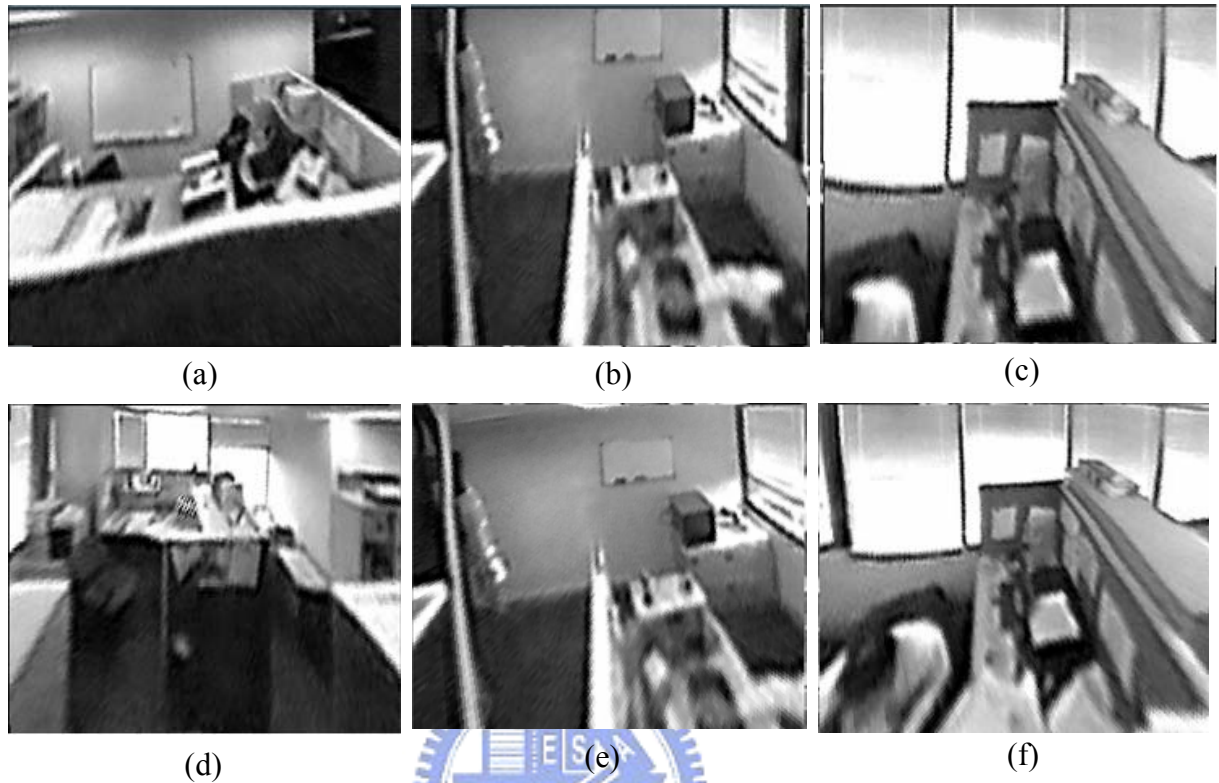


Fig. 2.15 Unwarped images of a real scene in Fig. 2.4 (Side view). (a) to (d) are the results using the proposed method with the same T and R as those for yielding Fig. 2.14. (a) to (d) show the results with different span of view angle, $1.5\pi \sim 2.0\pi$, $1.0\pi \sim 1.5\pi$, $0.5\pi \sim 1.0\pi$, and $0.0\pi \sim 0.5\pi$, respectively. (e) and (f) are the results using Eq. (2.1) with different span of viewing angle as (b) and (c), respectively.

Chapter 3

Improving quality of unwarped omni-images by a new edge-preserving interpolation technique

3.1 Introduction

A catadioptric omni-camera [2][5], which is a combination of a reflective mirror and a conventional camera, captures the incoming light to form an omni-image. The mirror surface may be of various shapes, like conic, parabolic or hyperbolic, etc., as mentioned previously. On the other hand, a hypercatadioptric camera is a type of catadioptric omni-camera with a hyperbolic-shaped mirror [7][8]. Fig. 3.1 shows the structure of an SVP hypercatadioptric camera, where the focus point of the camera is located at the outer focus point O_c of the hyperbolic curve of the mirror surface. That is, in this perfectly aligned structure, all the incoming light rays pass through O_c . If the structure is instead mis-aligned, the incoming light rays will not pass through O_c , but form a so-called “caustic” surface [9], and the camera becomes a non-SVP one.

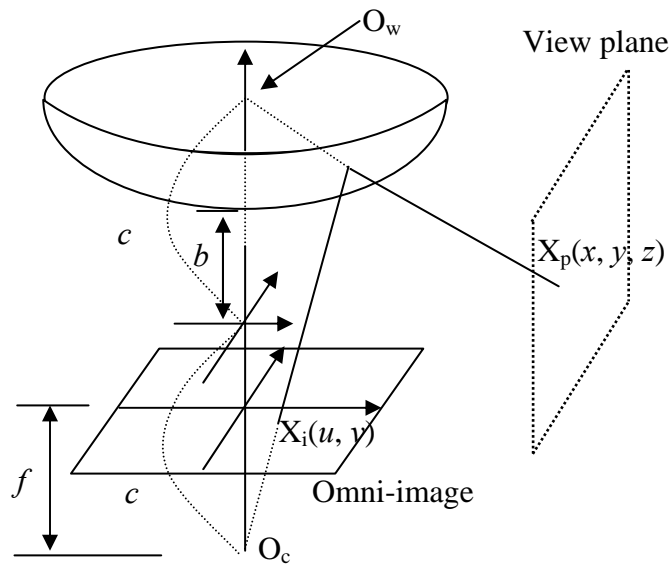


Fig. 3.1 An SVP hypercatadioptric camera. Where O_w is one focus of the hyperbolic curve taken to be the origin of the world coordinate system, and O_c is the optical lens center as well as the other focus of the hyperbolic curve.

It is difficult to unwarped an omni-image into a normal-view one, and the degree of difficulty depends on the type of omni-camera and the alignment of the camera structure. SVP omni-images are generally easier to handle [10][21][23] than non-SVP ones. If a non-SVP omni-image is treated as an SVP one and unwarped accordingly, the resulting perspective-view image, called *unwarped image* in the sequel, will suffer a serious geometric distortion [11], especially when the structural misalignment is *large* in some camera designs aiming to extending the field of view (FOV) of the camera. Fig. 3.2(b) shows this case. So, in the case of *non-SVP image unwarping*, we need another way to do the unwarping job.

More specifically, unwarping an omni-image, which is taken with an SVP hypercatadioptric camera, into a perspective version is a process of *forward projection* from a point $X_p = (x, y, z)$ on a certain perspective-view plane in the world space to an

omni-image pixel $X_i = (u, v)$, which can be described by $X_i = h(X_p)$ with h being a *one-to-one* mapping function from the world space to the omni-image plane [21][22]. Eq.

(3.1) below is a description of this mapping from (x, y, z) to (u, v) :

$$u = \frac{f(b^2 - c^2)x}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}}, v = \frac{f(b^2 - c^2)y}{(b^2 + c^2)z - 2bc\sqrt{x^2 + y^2 + z^2}} \quad (3.1)$$

where f is the focal length of the camera lens, and a, b and c are the parameters of the hyperbolic curve of the mirror surface described by the following equations:

$$z = -c + b\sqrt{1 + \frac{r}{a}}, r^2 = x^2 + y^2, c = \sqrt{a^2 + b^2}. \quad (3.2)$$

Consequently, after scanning all the points (x, y, z) in the perspective-view plane in the world space to get the coordinates of the corresponding pixel (u, v) in the omni-image in the unwarping process using Eq. (3.1), there will be *no unfilled pixel* in the resulting image because of the one-to-one mapping property.

However, for a non-SVP hypercatadioptric camera, there exists no *direct* one-to-one mapping $X_i = h(X_p)$ from X_p to X_i , and the relationship between X_p and X_i instead is $X_p = g(f(X_i))$ with f being a mapping from the omni-image plane to the mirror surface and g another mapping from the mirror surface to the world space [22]. The inverse forms g^{-1} and f^{-1} of the mappings are too complicated to obtain such that the direct mapping $X_i = h(X_p) = f^{-1}(g^{-1}(X_p))$ is unavailable. Therefore, it is impossible to conduct the same type of unwarping task as in the SVP case. Jeng and Tsai [22] solved this unwarping problem by a *back projection* method using the original mapping $X_p = g(f(X_i))$, yielding in the unwarped image many *irregularly-distributed unfilled pixels* to which no omni-image data have been

mapped. Fig. 3.2(c) shows an example of the unwarping result. The unfilled pixels should be filled by some interpolation method.

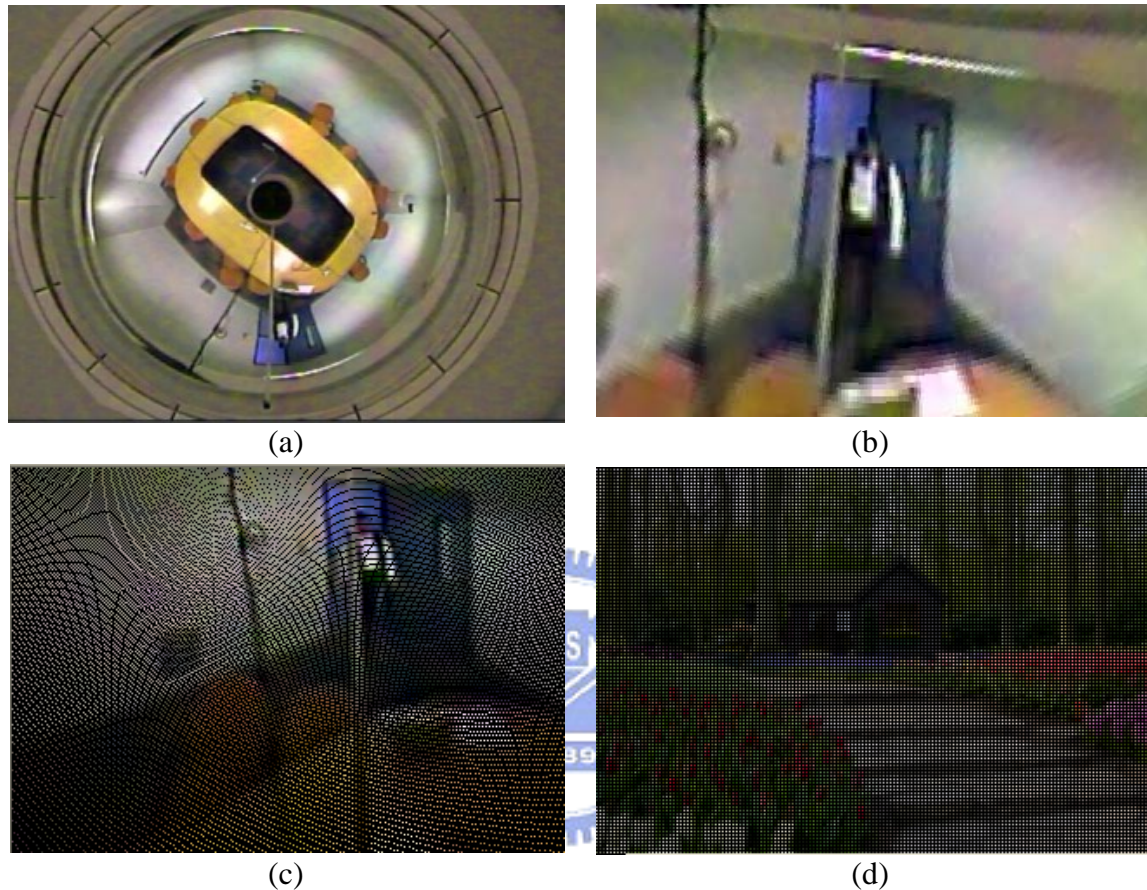


Fig. 3.2 Examples of images for illustration. (a) An omni-image taken by a non-SVP hypercatadioptric camera. (b) An unwarped perspective-view image when (a) is treated as an SVP camera. (c) Raw image resulting from unwarping the omni-image of (a) using a back projection method proposed by Jeng and Tsai [9]. (d) Raw enlarged image of a scene image with regularly-tessellated unfilled pixels before interpolation.

Conventional interpolation methods [24][25] deal mainly with *regularly-tessellated unfilled pixels*, resulting from conventional image transformations like image scaling (see Fig. 3.2(d) for an example), and so are unsuitable for handling the *irregularly-distributed* unfilled pixels in the unwarped image. In [26], Li and Orchard proposed a new edge-directed interpolation by using the geometric duality between the low-resolution

covariance and the high-resolution one of the image, but it is applicable only to images with *regularly-distributed* unfilled pixels created by image expansion. In our case here, the property of geometric duality does not exist, so the method is inapplicable. Jeng and Tsai [22], in addition to the back projection method, have proposed a so-called *8-directional-regions interpolation technique* to do this work. However, there is a drawback in their method, that is, *edge blurring* caused by the averaging operation in the method will occur in the interpolation result.

In this study, we propose a new method called “edge-preserving 8-directional two-layered weighting interpolation” to conduct the desired interpolation work, which can preserve local edges in images to avoid the edge-blurring effect. The method is thus more suitable to the unwarped image which has many irregularly distributed unfilled pixels. An edge-line detection scheme is proposed first for deciding if an unfilled pixel lies on an edge line. If the unfilled pixel is found to be on an edge line, then the pixel value is assigned by a scheme of inverse-distance weighting of two filled pixels on the edge line. Otherwise, a *two-layered weighting* interpolation scheme is applied, which considers *three* concepts of weighting, namely, inverse-distance weighting, pixel-count weighting, and region-wise weighting. This scheme uses the ratio of filled-pixel counts of each 8-directional sub-region to the total filled-pixel counts in all sub-regions as the first-layer weighting factor, and then uses the inverse distances between the filled pixels in each sub-region and the currently-processed unfilled pixel as the second-layer weighting factors to complete an even pixel-value contribution effect in the interpolation. The quality of the interpolation result is greatly improved in the aspect of edge preserving. It is noted here that this type of interpolation work for processing unwarped non-SVP omni-images, to the knowledge of

the authors, has not been investigated before. Although the idea of edge preserving can be found in Ramponi and Carrato [27], which investigated irregular sampling for image coding, here we utilize the edge information in a different way to process the more complicated unwarped image *with irregularly-distributed pixels at unknown positions*. Also, the weighting scheme in utilizing neighboring pixel values adopted in [9] is modified to improve the interpolation accuracy.

In the remainder of this chapter, we describe the proposed method in Section 3.2, show the experimental results in Section 3.3, and finally made a summary of this chapter in Section 3.4.

3.2 Image Interpolation by Edge Preserving and Pixel Value Weighting



The proposed method is an *edge-preserving 8-directional two-layered weighting interpolation technique* which, as mentioned previously, is based on three types of weighting for interpolation: (1) inverse-distance weighting; (2) pixel-count weighting; and (3) region-wise weighting. We first describe in Section 3.2.1 the core of the proposed technique involving *no edge preserving*, and then the proposed *edge-preserving version* of the proposed interpolation technique in Section 3.2.2.

3.2.1 Non-Edge-Preserving Version of Proposed Method

As illustrated in Fig. 3.3, the non-edge-preserving version of the proposed technique is based on an $n \times n$ region R (e.g., a 7×7 region) centered at an unfilled pixel P_u whose value is to be interpolated. Region R is divided into eight sub-regions S_1, S_2, \dots, S_8 . Filled

pixels in each sub-region, called *support pixels* hereafter, are used in calculating the weights for interpolation.

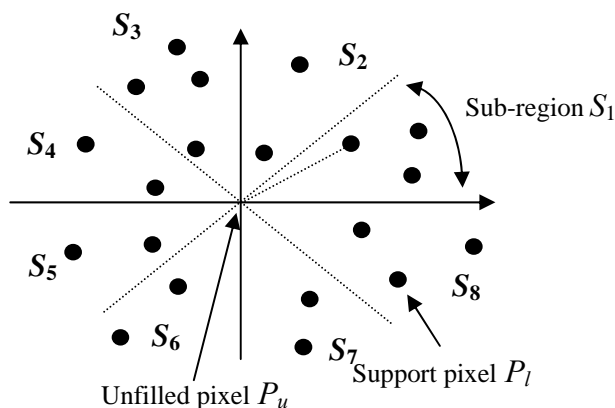


Figure 3.3 Illustration of proposed interpolation.

Let the support pixels in sub-region S_i ($i = 1, 2, \dots, 8$) be denoted as $P_{i1}, P_{i2}, \dots, P_{im_i}$ and their respective pixel values denoted as $I_{i1}, I_{i2}, \dots, I_{im_i}$, where m_i is the number of such pixels in S_i . Let the distance from a support pixel P_{ij} to P_u be denoted as d_{ij} where $j = 1, 2, \dots, m_i$. Let the number of all the support pixels in the neighborhood R of P_u be denoted as M , which is equal to

$$M = \sum_{i=1}^8 m_i . \quad (3.3)$$

The *region weighting factor* s_i for sub-region S_i based on the pixel numbers is defined as

$$s_i = \frac{m_i}{M} . \quad (3.4)$$

And the *distance weighting factor* d_i based on *inverse pixel distances* for sub-region S_i is defined as

$$d_i = \sum_{j=1}^{m_i} (1/d_{ij}) \cdot \quad (3.5)$$

Then, the weight factor w_{ij} of support pixel P_{ij} for use in the interpolation is calculated as

$$w_{ij} = \frac{(1/d_{ij})}{d_i} \times s_i \cdot \quad (3.6)$$

Now, the proposed two-layered weighted interpolation can be described as follows.

First, we compute the *contribution* I_i of sub-region S_i to the desired interpolation value I_u of P_u as

$$I_i = \sum_{j=1}^{m_i} w_{ij} I_{ij} \quad (3.7)$$

where I_{ij} , as described previously, is the pixel value of support pixel P_{ij} . And then, the desired pixel value of P_u can be computed by

$$I_u = \sum_{i=1}^8 I_i \cdot \quad (3.8)$$

Note that in case there exists no support pixel in one or more of the eight sub-regions, the weights of such sub-regions will be ignored automatically according to the above process, and only those of the other sub-regions with existing support pixels will be used in calculating the value I_u of the unfilled pixel P_u . This can be seen from Eq. (3.4) where s_i will be equal to 0 when $m_i = 0$. In the extreme case that no support pixel can be found in all the sub-regions, then we adopt the approach of enlarging the $n \times n$ region R , e.g., from 7×7 to 9×9 , and so on, until at least one support pixel is found in the enlarged R .

A merit of the above proposed approach of *two-layered interpolation* is that the contribution of a sub-region S_i , as can be figured out from the above formula, can be controlled by the corresponding region weighting factor s_i which is computed from the

number of support pixels in S_i . This will limit the magnitude of the contribution of the sub-region, compared with the traditional scheme of one-layered inverse-distance weighting which will cause uneven or dominant contribution of certain pixels at very short distances to the currently-processed unfilled pixel P_u . Another merit is that more contribution will be yielded by a sub-region with more support pixels, as can be seen again from Eq. (3.4) for computing s_i . This is reasonable because more information is conveyed by more pixels and such information should be utilized properly by giving larger weights to them. These two merits are advantageous to the work of unwarping the *irregularly-distributed and spatially-sparse pixels* in the unwarped image which we are dealing with, as proved by our experimental results described later.

3.2.2 Edge-Preserving Version of Proposed Technique

In the above non-edge-preserving version of the proposed technique, the weight of a support pixel basically is determined by its distance to the currently-processed unfilled pixel P_u . All the support pixels are used to accomplish the interpolation work. In the edge-preserving version of the proposed technique, each support pixel instead will be classified first as an *edge pixel* or not, using the edge pixel information found in the original omni-image. And the spatial relationship between P_u and the edge pixels are analyzed to determine which support pixels should be used to conduct the interpolation. The essence of the idea behind the proposed technique is to ignore non-edge pixel contribution to the interpolation if an edge is found to go through P_u , thus reducing the blurring effect in the unwarped image.

Now the issue is how to find the edge pixels for use in the above process. Although edge pixels can be easily found by traditional edge detection methods in the original

omni-image which is rectangularly-tessellated in nature, what we want is their corresponding pixels in the unwarped image. Since such pixels in the unwarped image are irregularly and sometimes even sparsely distributed, traditional edge detection methods are inapplicable. To solve this problem, we compute first an *edge-value image* from the original omni-image, and then unwarped it into a second image and threshold the resulting image values to get an *unwarped edge map*. Finally, we detect edges in the unwarped edge map by a way of checking edge lines formed by triple pixels. The details are described in the following. Let the original omni-image be denoted as I_o .

(1) *Creation of unwarped edge map*

First, we compute an edge-value image I_e by the Sobel operator with I_o as input and normalize the edge values in I_e into the range of 0.0 to 1.0. Fig. 3.4(a) shows an input omni-image I_o and Fig. 3.4(c) shows the resulting I_e computed from Fig. 3.4(a).

Second, the original omni-image and the edge-value image are unwarped with the non-SVP forward-projection method proposed in [22] to get a raw unwarped image and an unwarped edge-value image, respectively. The unwarped edge-value image is then thresholded into an unwarped edge map by a threshold value e_t . Fig. 3.4(b) and Fig. 3.4(d) show the results, respectively. There are many unfilled pixels in the raw unwarped image, whose values are what we want to interpolate in the sequel. And the unwarped edge map will be used to assist this task.

(2) *Performing local edge analysis to detect support pixels on edges*

An unfilled pixel P_u is regarded to be *on an edge line* if P_u and two support pixels P_k and P_l form *approximately* a straight line, where P_k and P_l lie respectively in two sub-regions S_i and S_j which are opposite in directions (e. g., S_1 and S_5 , S_2 and S_6 , and so

on). To detect such *on-edge* support pixels, every possible pair of oppositely-directed sub-regions S_i and S_j and every pair of support pixels P_k and P_l located respectively in them are selected and analyzed in the following way:

- (a) calculate the angle θ_m formed by the two vectors $\mathbf{P_uP_k}$ and $\mathbf{P_uP_l}$;
- (b) if θ_m is close to 180° (determined by checking if it is larger than a threshold θ_t), then regard P_u to be on an edge line and record the pair of support pixels P_k and P_l as a *candidate pair*. Fig. 3.5 illustrates this idea.

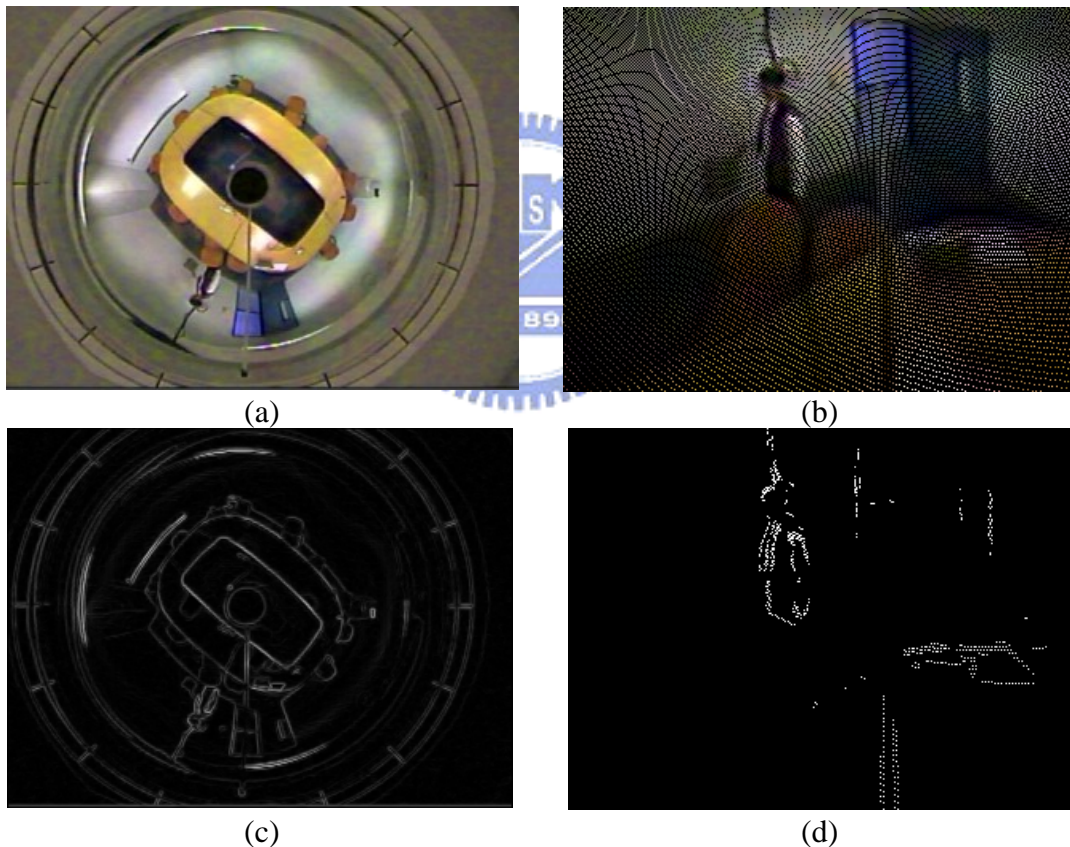


Fig. 3.4 Creation of unwarped edge map from original omni-image. (a) Original omni-image. (b) Raw unwarped image. (c) Edge-value image of (a). (d) Unwarped edge map (with $e_t = 0.3$).

(3) Interpolation using on-edge support pixels

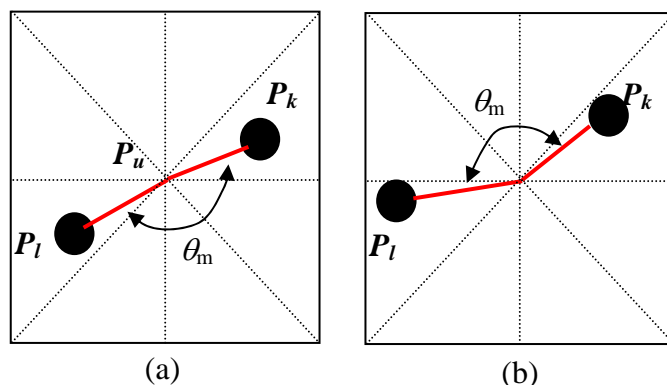


Fig. 3.5 Illustration of proposed on-edge decision. (a) Edge detected (θ_m close to 180°). (b) No edge detected (θ_m not close to 180°).

We then take from the set of all candidate pairs the one with support pixels P_a and P_b which forms an angle θ_m closest to 180° , and use their distances d_a and d_b to P_u to complete the desired interpolation for the pixel value I_u of P_u in the following way:

$$I_u = [(1/d_a) \times I_a + (1/d_b) \times I_b] / [(1/d_a) + (1/d_b)]. \quad (3.9)$$

That is, we compute the pixel value I_u by an interpolation from those of P_a and P_b weighted inversely by their respective distances d_a and d_b to P_u .

(4) Interpolation by non-edge support pixels

If no candidate pair is found in the above step, then we conduct interpolation of the value of I_u for pixel P_u by the non-edge-preserving version of the proposed technique described in Section 3.2.1.

3.3 Experimental Results

In the field of image interpolation, most existing objective metrics of image quality are unsuitable to evaluate the quality of the results of the applied methods, as mentioned in

[26]. Subjective evaluation via human eyes is more practical, and the improvements of the proposed method may be easily observed in the figures of the following experimental results.

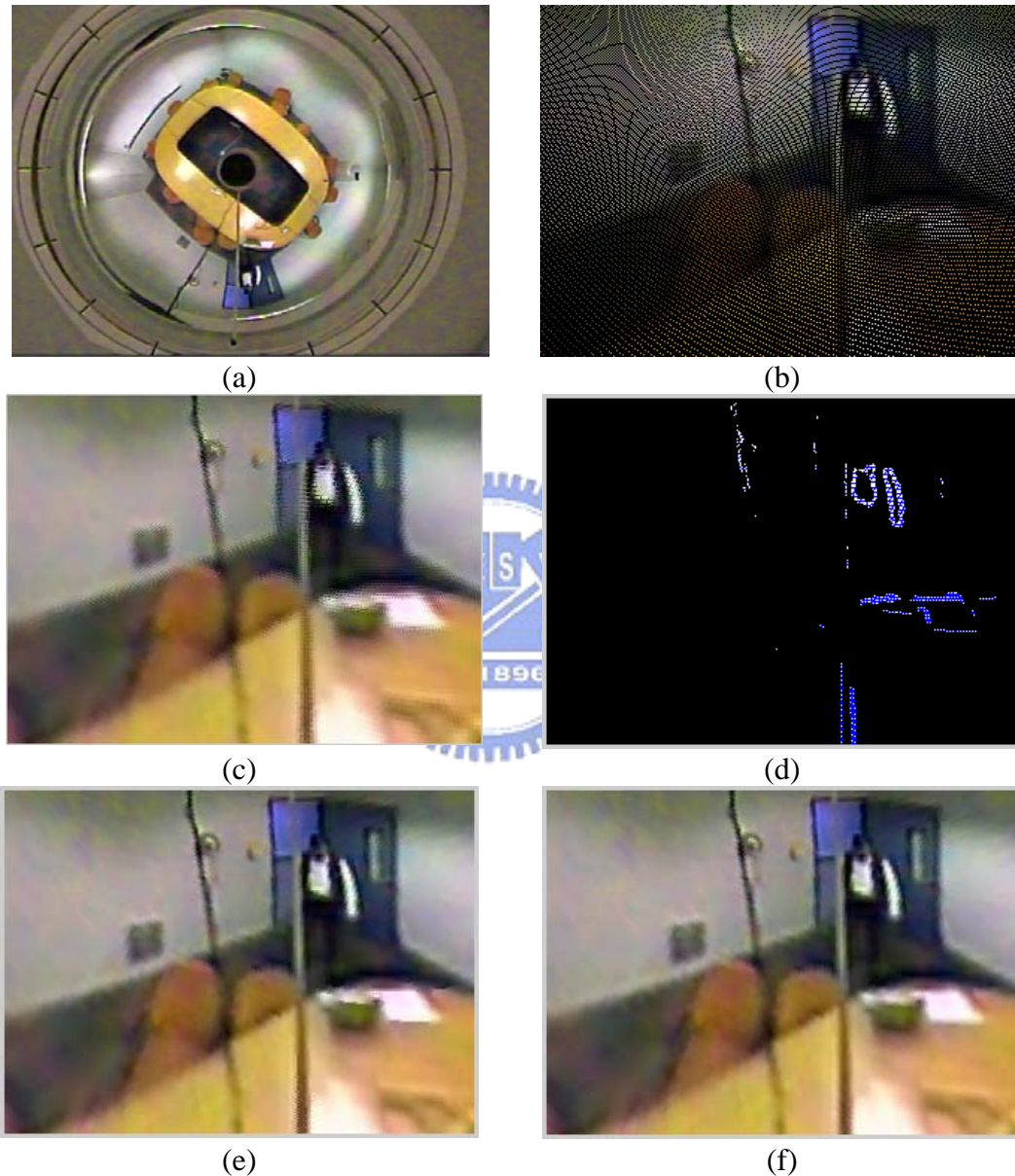


Fig. 3.6 Comparison of results of different methods. (a) Original image. (b) Raw unwarped image with unfilled pixels. (c) Interpolation result using 4NN method (processing time = 90 msec). (d) Detected edge pixels (white) & on-edge pixels (blue). (e) Result of proposed method with edge preserving (processing time = 218 msec). (f) Result of proposed method without edge preserving (processing time = 94 msec).

In our implementation of the proposed method, there are two threshold parameters, the edge value threshold e_t and the angle span threshold θ_t , which can be adjusted for defining edge points and checking the on-edge condition, respectively, as described previously. The value of e_t is set between 0 and 1 representing normalized edge intensity values. The value of θ_t is set between 0° and 180° representing the angle spanned by two support pixels around an unfilled pixel. Fig. 3.6 shows a comparison of an experimental result of the proposed method by setting $e_t = 0.35$, $\theta_t = 120^\circ$ with that of a conventional method. Fig. 3.6(a) shows the original omni-image. Fig. 3.6(b) shows the raw unwarped image which contains many unfilled pixels to be interpolated. Fig. 3.6(c) is the result using the traditional 4-nearest-neighbor (4NN) interpolation method with 7×7 window size. Fig. 3.6(d) shows the unwarped edge map in which the white points are the edge pixels obtained from thresholding the unwarped edge-value image (called *unwarped edge pixels*) and the blue points are unfilled pixels detected to be on edge lines in the unwarped edge map (called *on-edge pixels*). Fig. 3.6(e) is the result using the edge-preserving version of the proposed method, and Fig. 3.6(f) is the result using the non-edge-preserving version of the proposed method. The overall image quality improvement can be found obvious by comparing the results of Figs. 3.6(e) and (f) with that of Fig. 3.6(c). Furthermore, we can see the edge quality improvement in Fig. 3.6(e) near the locations of the edge pixels in Fig. 3.6(d). The processing times for Figs. 3.6(c), (e) and (f) of the size 320×240 using a PC with an Intel P4 3GHz CPU and a 1GB RAM are 90 msec., 218 msec., and 94 msec., respectively. The need of longer processing time for Fig. 3.6(e) than those for Figs. 3.6(c) and (f) is owing to the time-consuming step of computing the edge map of Fig. 3.6(d). Note that the computations of Figs. 3.6(c) and (f) do not involve the edge map, but the cost

of saving time in this aspect is the blurring effect at edges.

Fig. 3.7 shows the results of applying the proposed method to a raw unwarped image, Fig. 3.6(b), using different values for the threshold e_t and a constant value 120° for the threshold θ_t . Lower values of e_t will yield more edge points and so more on-edge unfilled

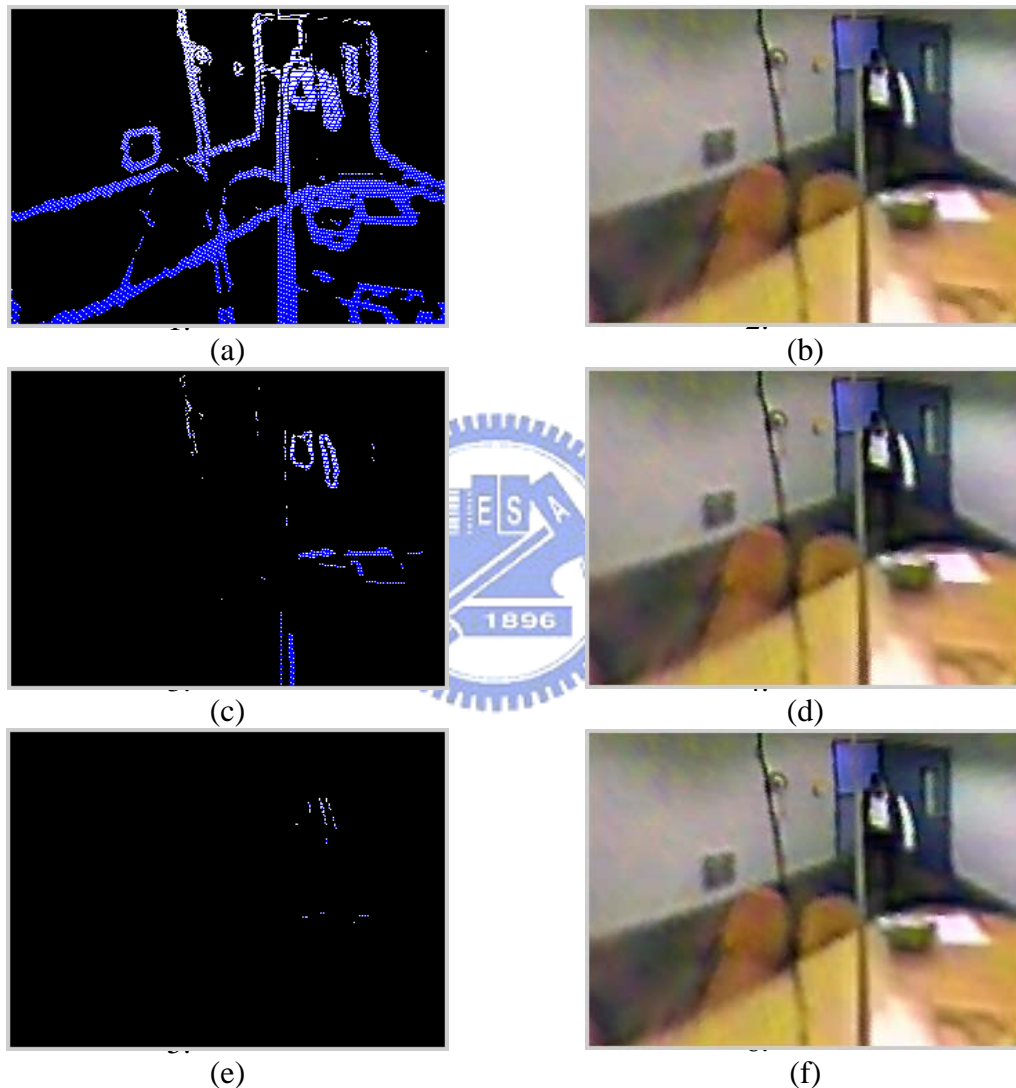


Fig. 3.7 Results using different edge threshold values at constant angle threshold ($\theta_t = 120^\circ$). (a) Unwarped edge pixels (white) and detected on-edge pixels (blue), $e_t = 0.1$. (b) Resulting image of proposed method, $e_t = 0.1$. (c) Unwarped edge pixels and detected on-edge pixels, $e_t = 0.35$. (d) Resulting image of proposed method, $e_t = 0.35$. (e) Unwarped edge pixels and detected on-edge pixels, $e_t = 0.6$. (f) Resulting image of proposed method, $e_t = 0.6$.

pixels, as can be seen in Figs. 3.7(a) and (b), causing improper blurring effects near the edge locations. Higher values of e_t will reduce the number of resulting edge points, yielding a result of Fig. 3.7(f) which is almost the same as that obtained from using the non-edge-preserving version of the proposed method. After many experimental trials, we suggest the use of the edge threshold value range of $0.2 < e_t < 0.4$ for use in unwarping non-SVP omni-images.

Fig. 3.8 shows the results of using different values for θ_t , ranging from 120° to 180° , and a constant value 0.35 for e_t , with the raw unwarped image of Fig. 3.6(b) as input. It is difficult to see visual differences among the global image qualities of these results. But we can inspect the detailed distribution of the on-edge pixels by two enlarging portions of the images of Figs. 3.8(a) and (e), as shown in Figs. 3.9(a) and (b), respectively.

From Fig. 3.9(b), we can see that a very *tight* “on-edge” condition check, using the extreme value of $\theta_t = 180^\circ$, will result in losing some real on-edge pixels because at the “digital pixel” level, a straight edge-line formed by two different pixels does not always pass precisely the middle pixel of the currently-processed window. So, we should *relax* the “on-edge” checking condition to allow more reasonable pixels (for example, the point P_c in Fig. 3.9(b)) to be accepted as on-edge pixels, as Fig. 3.9(a) shows.

More specifically, referring to Fig. 3.10, we calculate the span angle θ_m of the unfilled pixel P_c by the following equation according to trigonometry:

$$\theta_m = \cos^{-1}\left(\frac{a^2 + b^2 - c^2}{2 \times a \times b}\right). \quad (3.10)$$

By substituting proper values of a , b , and c in Fig. 3.10, namely, $a = \sqrt{2}$, $b = 1$, and c

$=\sqrt{5}$, into Eq. (3.10), we get $\theta_m = \cos^{-1}\left(\frac{2+1-5}{2 \times \sqrt{2} \times 1}\right) = \cos^{-1}(-\sqrt{0.5}) = 2.356$ which is

equal to 135° . So, the range of the span angle θ_m is taken to be larger than 120° and smaller than 180° . In this sense, P_c is an on-edge pixel in Fig. 3.9(a) because the threshold θ_t is taken to be 120° , and it is not in Fig. 3.9(b) because of the tight threshold value $\theta_t = 180^\circ$.

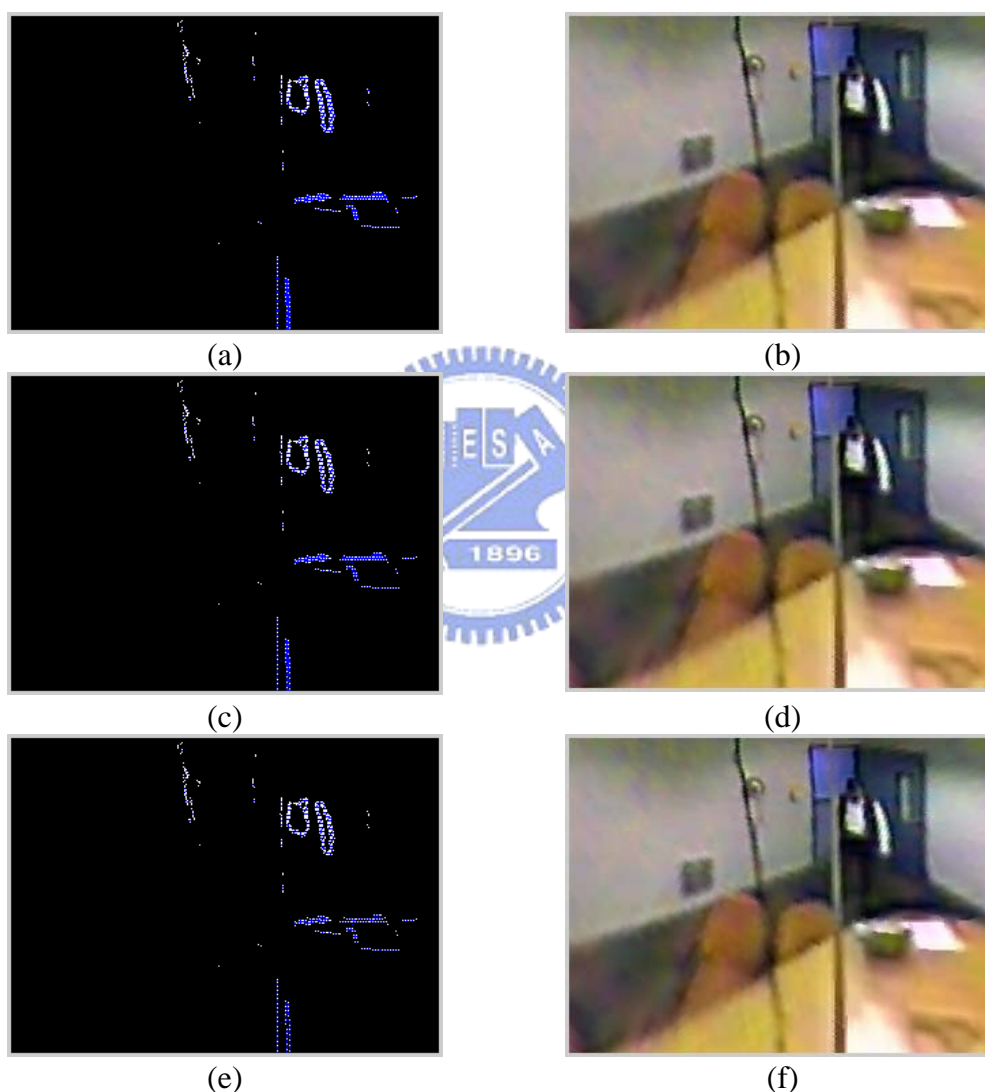
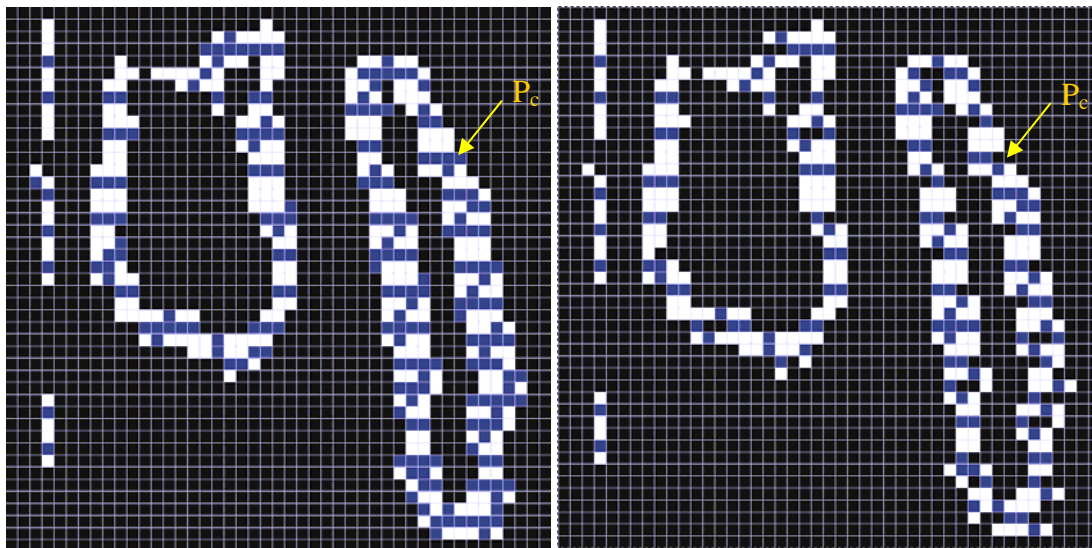


Fig. 3.8 Results using different angle threshold values at constant edge threshold ($e_t = 0.35$). (a) Unwarped edge pixels (white) and detected on-edge pixels (blue), $\theta_t = 120^\circ$. (b) Resulting image of proposed method, $\theta_t = 120^\circ$. (c) Unwarped edge pixels and detected on-edge pixels, $\theta_t = 150^\circ$. (d) Resulting image of proposed method, $\theta_t = 150^\circ$. (e) Unwarped edge pixels and detected on-edge pixels, $\theta_t = 180^\circ$. (f) Resulting image of proposed method, $\theta_t = 180^\circ$.



(a)

(b)

Fig. 3.9 Detailed edge maps obtained by using different angle thresholds and an identical edge threshold. (a) Angle threshold $\theta_t = 120^\circ$. (b) Angle threshold $\theta_t = 180^\circ$. Note that P_c is an on-edge pixel in (a) (marked by blue color), but is not in (b) (appearing as black).

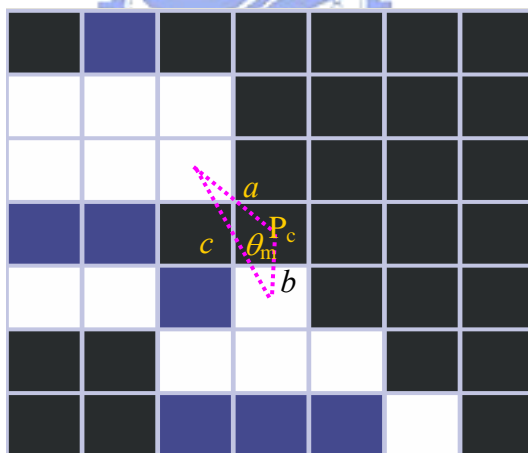


Fig. 3.10 Detailed calculation of the span angle of an unfilled pixel.

Furthermore, in Fig. 3.11, we try the proposed method using different window sizes to a raw unwarped image, Fig. 3.6(b). The processing time increases with the window size, as expected. When the window size is reduced to 5×5 , some unfilled pixels become to have no support pixel and leave holes (black dots) in the image (for example, see the region in

the lower part of Fig. 3.11(a) enclosed by the dotted circle). On the contrary, when the window size is larger than 7×7 , the image quality is almost kept the same. So, it is not necessary to use window sizes larger than 7×7 .

Finally, Fig. 3.12 shows a comparison of the results of different interpolation methods for the case of interpolating *regularly*-distributed unfilled pixels in images. Fig. 3.12(a) is an image produced from one by duplicating the pixel lines without filling the duplicated pixels. And Figs. 3.12(b), (c), and (d) are the interpolation results of three methods including the proposed one. What we want to show here is that the proposed method is also applicable to the conventional image expansion problem, yielding again results of better quality than those of other methods.

3.4 Summary



We have proposed in this paper a method of “edge-preserving 8-directional two-layered weighting interpolation” for processing an unwarped image with irregularly distributed unfilled pixels which is the result of unwarping an omni-image taken from a non-SVP hypercatadioptric camera. In addition to possessing the edge-preserving capability, the method takes into consideration three concepts of weighting for the involved interpolation scheme: (1) inverse-distance weighting; (2) pixel-count weighting; and (3) region-wise weighting. This method has reasonable processing speed and produces good image quality, compared with other conventional methods, as seen from the experimental results.

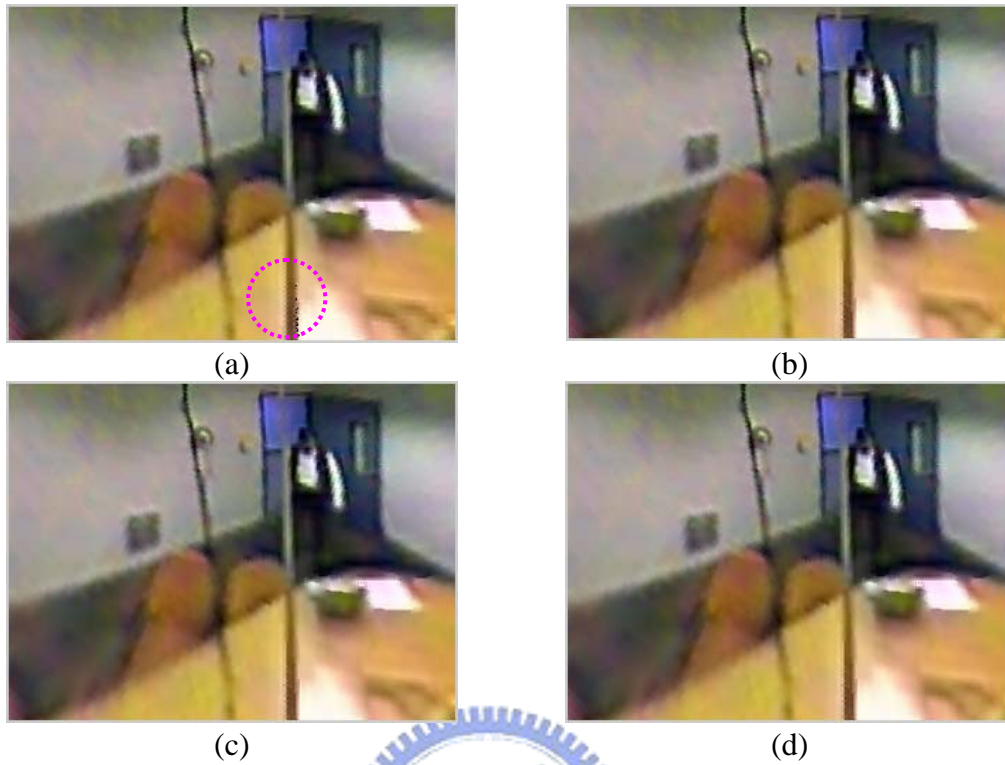


Fig. 3.11 Interpolation results with different window sizes. The processing time increases with the window size. (a) Window size 5×5 (processing time = 172 msec). (b) Window size 7×7 (processing time = 234 msec). (c) Window size 9×9 (processing time = 266 msec). (d) Window size 11×11 (processing time = 313 msec).

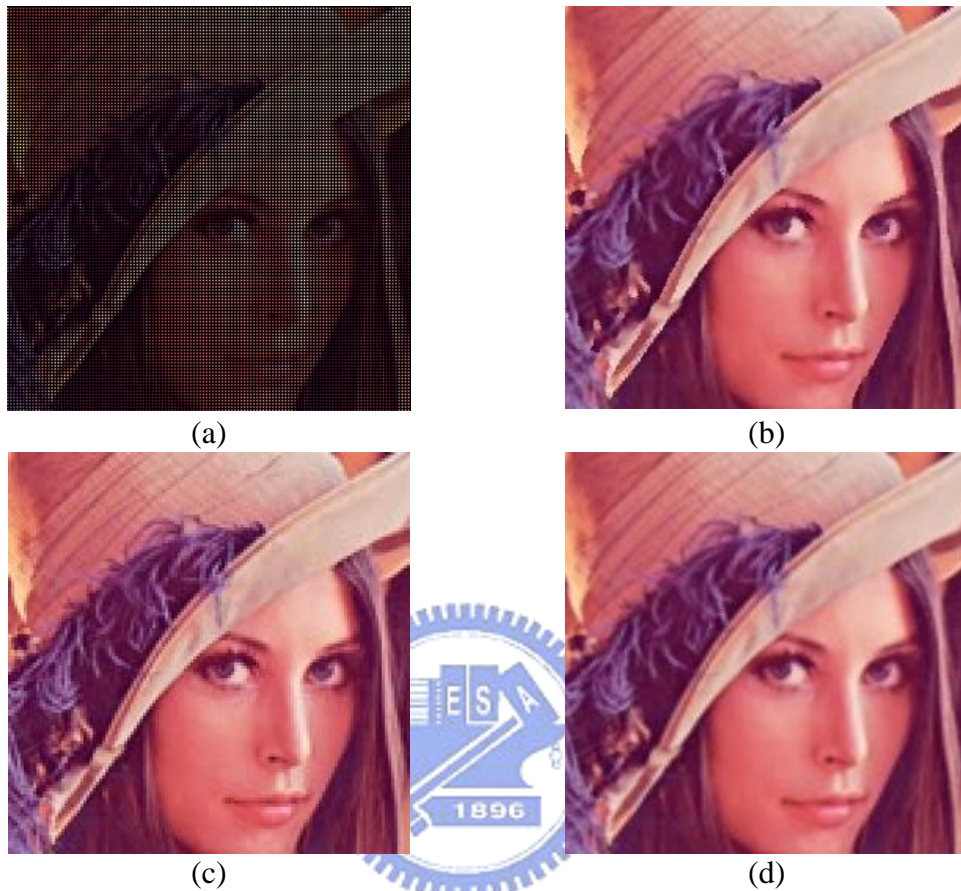


Fig. 3.12 Comparison of results of different interpolation methods for image expansion. (a) Raw enlarged image. (b) Result of proposed method. (c) Result of simple pixel duplication. (d) Result of 4NN interpolation method. Note that the original image size is 256×256 ; which is enlarged to 512×512 in this experiment. The cropped versions of the enlarged image are shown here.

Chapter 4

Using pano-mapping tables for unwarping of omni-images into panoramic and perspective-view images

4.1 Introduction

Omni-cameras are used in many applications such as visual surveillance and robot vision [28][29][30][31] for taking omni-images of camera surroundings. Usually, there exists an extra need to create perspective-view images from omni-images for human comprehensibility or event recording. This image unwarping work usually is a complicated work. Fig. 4.1 shows an example where Fig. 4.1(a) is an omni-image and Fig. 4.1(b) a perspective-view image obtained from unwarping the image of Fig. 4.1(a) by a method proposed by Jeng and Tsai [22].

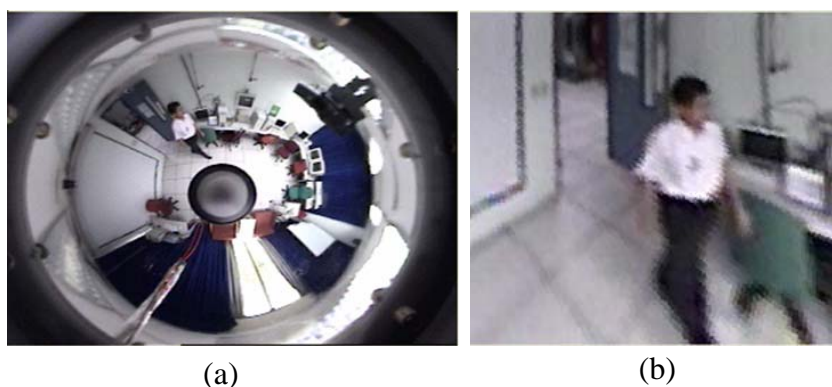


Fig. 4.1 An example image unwarping. (a) An omni-image. (b) A perspective view image resulting from unwarping a portion of the omni-image in (a).

Omni-cameras can be categorized into two types according to the involved optics, namely, dioptric and catadioptric, as mentioned previously [5]. A dioptric

omni-camera captures incoming light going directly into the imaging sensor to form omni-images. An example of dioptric omni-cameras is the fish-eye camera. A catadioptric omni-camera captures incoming light reflected by a mirror to form omni-images. The mirror surface may be in various shapes, like conic, hyperbolic, etc.

The method for unwarping omni-images is different for each distinct type of omni-camera. Generally speaking, omni-images taken by the SVP catadioptric camera [5][21] as well as the dioptric camera [33] are easier to unwarped than those taken by the non-SVP catadioptric camera [12][22]. Conventional methods for unwarping omni-images require the knowledge of certain camera parameters, like the focal length of the lens, the coefficients of the mirror surface shape equation, etc to do calibration before the unwarping can be done [10][11][17][34][35]. In the last chapter, we have proposed a method of this kind. But in some situations, we cannot get the complete information of the omni-camera parameters. Then the unwarping work cannot be conducted. It is desired to have a more convenient way to deal with this problem.

In this chapter, we propose a unified approach for unwarping of omni-images taken by *all* kinds of omni-cameras. It is unnecessary to know the camera parameters in advance. This is made possible by the use of a *pano-mapping table* proposed in this study, which may be regarded as a summary of the information conveyed by all the camera parameters. The pano-mapping table is created *once forever* for each omni-camera. And given an omni-image taken by the camera, the table may be utilized to unwarped the image in analytic ways to create panoramic or perspective-view images from *any viewpoints* in the world space. The pano-mapping table is invariant in nature with respect to the camera position, that is, it is not changed even when

the camera is moved around. The table is created by a calibration process making use of certain selected points in the world space with known coordinates and their corresponding pixels in an omni-image.

The remainder of this chapter is organized as follows. In Section 4.2, we describe the proposed approach based on the use of the pano-mapping table in detail. In Section 4.3, we show some experimental results, and finally make a summary of this chapter in Section 4.4.

4.2 Proposed Method Using Pano-Mapping Table

The proposed method consists of three major stages: (1) landmark learning, (2) table creation, and (3) image unwarping.

A. Landmark learning

The first step, *landmark learning*, is a procedure in which some pairs of selected world space points with known positions and their corresponding pixels in a taken omni-image are set up. More specifically, the coordinates of at least five points, called *landmark points* hereafter, which are easy to identify in the world space (for example, a corner in a room), are measured manually with respect to a selected origin in the world space. Then the corresponding pixels of such landmark points in the taken omni-image are segmented out. A world space point and its corresponding image pixel so selected together are said to form a *landmark point pair*.

B. Table creation

The second step, *table creation*, is a procedure in which a *pano-mapping table* is built using the coordinate data of the landmark point pairs. The table is 2-dimensional in nature with the horizontal and vertical axes specifying respectively the range of the azimuth angle θ as well as that of the elevation angle ρ of all possible incident light rays going through the mirror center. An illustration is shown in Fig. 4.2, and an example of the pano-mapping table of size $M \times N$ is shown in Table 4.1.

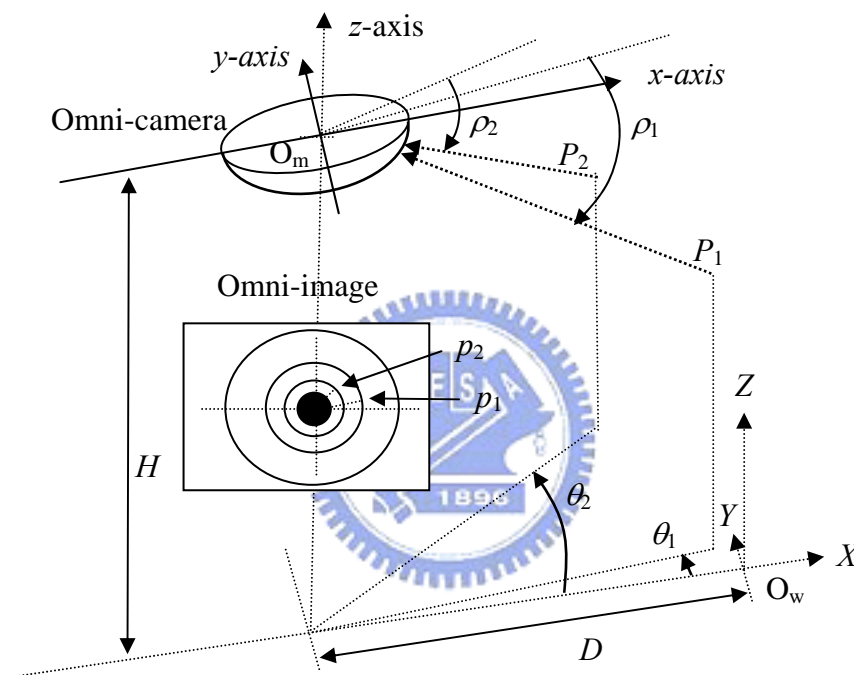


Fig. 4.2 System configuration.

Table 4.1. An example of pano-mapping table of size $M \times N$.

	θ_1	θ_2	θ_3	θ_4	...	θ_M
ρ_1	(u_{11}, v_{11})	(u_{21}, v_{21})	(u_{31}, v_{31})	(u_{41}, v_{41})	...	(u_{M1}, v_{M1})
ρ_2	(u_{12}, v_{12})	(u_{22}, v_{22})	(u_{32}, v_{32})	(u_{42}, v_{42})	...	(u_{M2}, v_{M2})
ρ_3	(u_{13}, v_{13})	(u_{23}, v_{23})	(u_{33}, v_{33})	(u_{43}, v_{43})	...	(u_{M3}, v_{M3})
ρ_4	(u_{14}, v_{14})	(u_{24}, v_{24})	(u_{34}, v_{34})	(u_{44}, v_{44})	...	(u_{M4}, v_{M4})
...
ρ_N	(u_{1N}, v_{1N})	(u_{2N}, v_{2N})	(u_{3N}, v_{3N})	(u_{4N}, v_{4N})	...	(u_{MN}, v_{MN})

Each entry E_{ij} with indices (i, j) in the pano-mapping table specifies an

azimuth-elevation angle pair (θ_i, ρ_j) , which represent an infinite set S_{ij} of world space points passing through by the light ray with azimuth angle θ_i and elevation angle ρ_j . These world space points in S_{ij} are all projected onto an identical pixel p_{ij} in any omni-image taken by the camera, forming a *pano-mapping* f_{pm} from S_{ij} to p_{ij} . An illustration is shown in Fig. 4.3. This mapping is shown in the table by filling entry E_{ij} with the coordinates (u_{ij}, v_{ij}) of pixel p_{ij} in the omni-image. The table as a whole specifies the nature of the omni-camera, and may be used to create any panoramic or perspective-view images, as described subsequently.

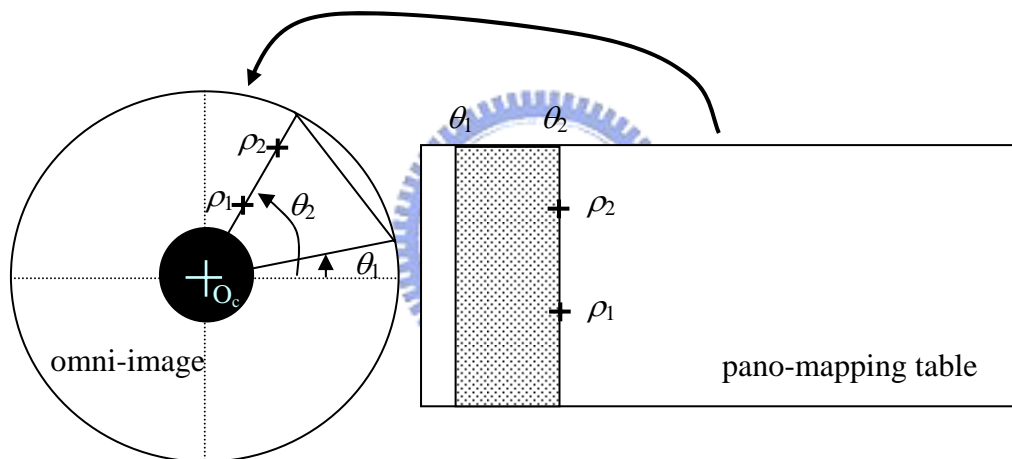


Fig. 4.3 Mapping between pano-mapping table and omni-image.

C. Image unwarping

The third step, *image unwarping*, is a procedure in which the pano-mapping table T_{pm} of an omni-camera is used as a media to construct a panoramic or perspective-view image Q of any size for any viewpoint from a given omni-image I taken by the omni-camera. The basic concept in the procedure is to *map* each pixel q in Q to an entry E_{ij} with coordinate values (u_{ij}, v_{ij}) in the pano-mapping table T_{pm} and to assign the color value of the pixel at coordinates (u_{ij}, v_{ij}) of image I to pixel q in Q .

The detail of each stage is elaborated in the following.

4.2.1 Landmark Learning Procedure

Before describing the landmark learning procedure, we briefly explain the system configuration as shown in Fig. 4.2. A “downward-looking” omni-camera is attached “horizontally” at the ceiling center of a room with both its mirror base plane and omni-image plane parallel to the floor, which is just the X - Y plane of the world coordinate system with its coordinates denoted by (X, Y, Z) and its origin by O_w . The mirror center of the omni-camera, denoted as O_m , is located at $(-D, 0, H)$ with respect to O_w in the world coordinate system. A camera coordinate system is set up at O_m with its coordinates denoted by (x, y, z) and its three axes all parallel to those of the world coordinate system. Also shown in the figure as an illustration of the definitions of the azimuth and elevation angles are two points P_1 and P_2 in the world space with corresponding image points p_1 and p_2 in the omni-image. The elevation angles of P_1 and P_2 with respect to the horizontal base plane of the mirror are ρ_1 and ρ_2 , respectively, and their azimuth angles with respect to the x -axis of the camera coordinate system are θ_1 and θ_2 , respectively.

The landmark learning procedure proceeds at first by selecting a sufficient number (≥ 5) of landmark point pairs with the world space points being easy to identify. The coordinates of the world space points are then measured. Fig. 4.4 shows the interface we have designed for acquiring the data of the landmark point pairs easily. Especially, note that in Fig. 4.3 the mirror center O_m of the camera with known world coordinates (X_0, Y_0, Z_0) just appears to be the image center O_c with known coordinates (u_0, v_0) . This image center can be automatically extracted by a simple image analysis scheme [18][19]. We skip the detail here, and take the coordinate data of the point pair (O_c, O_m) as the first set of the learned data. After learning, assume that we have n sets of landmark point pair data, each set including the coordinates (u_k, v_k) and (X_k, Y_k, Z_k) of

the image point and the corresponding world space point, respectively, where $k = 0, 1, \dots, n - 1$.

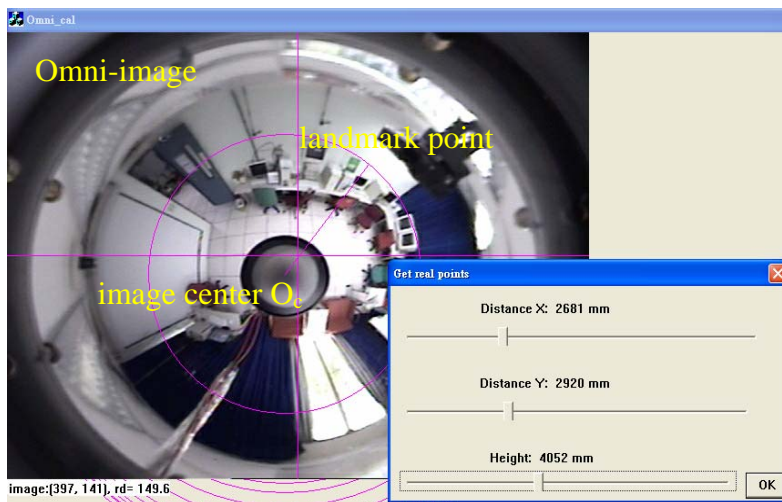


Fig. 4.4 Landmark learning interface.

4.2.2 Table Creation Procedure

The pano-mapping table is a two-dimensional array for use as a media for unwarping omni-images after it is constructed. We may imagine the table as a longitude and latitude system with a horizontal θ -axis and a vertical ρ -axis, specifying the azimuth and elevation angles of incident light rays through the mirror center, as mentioned previously. We divide the range 2π (360°) of the azimuth angles equally into M units, and the range of the elevation angles, say from ρ_s to ρ_e , into N units, to create a table T_{pm} of $M \times N$ entries. Each entry E with corresponding angle pair (θ, ρ) in T_{pm} maps to a pixel p with coordinates (u, v) in the input omni-image I . This mapping f_{pm} may be decomposed into two separate mappings, one in the azimuth direction and the other in the radial direction, called *azimuth-directional mapping* and *radial-directional mapping*, respectively.

Because of the rotation-invariant property of the omni-camera, the azimuth angle θ of each world space point through which the light ray passes is actually identical to

the angle ϕ of the corresponding pixel p with respect to the u -axis in the input image I .

That is, the azimuth-directional mapping is just an identity function f_a such that $f_a(\theta) = \phi = \theta$.

On the other hand, because of the nonlinear property of the mirror surface shape, the radial-directional mapping should be specified by a nonlinear function f_r such that the radial distance r from each image pixel p with coordinates (u, v) in I to the image center O_c at (u_0, v_0) may be computed by $r = f_r(\rho)$. And based on the two mappings f_a and f_r , we can regard the pairs $(r, \phi) = (f_r(\rho), \theta)$ of all the image pixels to form a *polar coordinate system* with the image coordinates (u, v) specified by

$$u = r \times \cos \phi = f_r(\rho) \times \cos \theta, \quad (4.1-1)$$

$$v = r \times \sin \phi = f_r(\rho) \times \sin \theta. \quad (4.1-2)$$

In this study, we also call f_r a *radial stretching function*. And we propose to describe it by the following 4th-degree polynomial function:

$$r = f_r(\rho) = a_0 + a_1 \times \rho^1 + a_2 \times \rho^2 + a_3 \times \rho^3 + a_4 \rho^4, \quad (4.2)$$

where a_0 through a_4 are five coefficients to be estimated using the data of the landmark point pairs, as described in the following algorithm. A similar idea of approximation can be found in Scotti, et al. [32]. Let the data of the n selected landmark point pairs be denoted as $(P_0, p_0), (P_1, p_1), \dots, (P_{n-1}, p_{n-1})$, where $n \geq 5$.

Algorithm 1. Estimation of coefficients of radial stretching function.

Step 1. (*Coordinate transformation in world space*) Transform the world coordinates (X_k, Y_k, Z_k) of each selected landmark point $P_k, k = 1, 2, \dots, n - 1$, with respect to O_w into coordinates with respect to O_m by subtracting from (X_k, Y_k, Z_k) the coordinate values $(X_0, Y_0, Z_0) = (-D, 0, H)$ of O_m . Hereafter, (X_k, Y_k, Z_k) will

be used denote this coordinate transformation result.

Step 2. (*Elevation angle and radial distance calculation*) Use the coordinate data of each landmark point pair (P_k, p_k) , including the world coordinates (X_k, Y_k, Z_k) and the image coordinates (u_k, v_k) , to calculate the elevation angle ρ_k of P_k in the world space and the radial distance r_k of p_k in the image plane by the following equations:

$$\rho_k = \tan^{-1} \left[\frac{Z_k}{D_k} \right]; \quad (4.3)$$

$$r_k^2 = u_k^2 + v_k^2, \quad (4.4)$$

where D_k is the distance from the landmark point P_k to the mirror center O_m in the X - Y plane of the world coordinate system, computed by $D_k = \sqrt{X_k^2 + Y_k^2}$.

Step 3. (*Calculation of coefficients of the radial stretching function*) Substitute all the data $\rho_0, \rho_2, \dots, \rho_{n-1}$ and r_1, r_2, \dots, r_{n-1} computed in the last step into Eq. (4.2) to get n simultaneous equations:

$$r_0 = f_r(\rho_0) = a_0 + a_1 \times \rho_0^1 + a_2 \times \rho_0^2 + a_3 \times \rho_0^3 + a_4 \rho_0^4;$$

$$r_1 = f_r(\rho_1) = a_0 + a_1 \times \rho_1^1 + a_2 \times \rho_1^2 + a_3 \times \rho_1^3 + a_4 \rho_1^4;$$

.

$$r_{n-1} = f_r(\rho_{n-1}) = a_0 + a_1 \times \rho_{n-1}^1 + a_2 \times \rho_{n-1}^2 + a_3 \times \rho_{n-1}^3 + a_4 \rho_{n-1}^4,$$

and solve them to get the desired coefficients $(a_0, a_1, a_2, a_3, a_4)$ of the radial stretching function f_r by a numerical analysis method[36].

Now, the entries of the pano-mapping table T_{pm} can be filled with the corresponding image coordinates using Eqs. (4.1) and (4.2) by the following

algorithm. Note that there are $M \times N$ entries in the table.

Algorithm 2. Filling entries of pano-mapping table.

Step1. Divide the range 2π of the azimuth angles into M intervals, and compute the i th azimuth angle θ_i by

$$\theta_i = i \times (2\pi/M), \text{ for } i = 0, 1, \dots, M - 1. \quad (4.5)$$

Step 2. Divide the range $[\rho_e - \rho_s]$ of the elevation angles into N intervals, and compute the j -th elevation angle ρ_j by

$$\rho_j = j \times [(\rho_e - \rho_s)/N] + \rho_s, \text{ for } j = 0, 1, \dots, N - 1. \quad (4.6)$$

Step 3. Fill the entry E_{ij} with the corresponding image coordinates (u_{ij}, v_{ij}) computed according to Eqs. (4.1) and (4.2) as follows:

$$u_{ij} = r_j \times \cos \theta_i; \quad (4.7-1)$$

$$v_{ij} = r_j \times \sin \theta_i; \quad (4.7-2)$$

where r_j is computed by

$$r_j = f_r(\rho_j) = a_0 + a_1 \times \rho_j^1 + a_2 \times \rho_j^2 + a_3 \times \rho_j^3 + a_4 \rho_j^4 \quad (4.8)$$

with $(a_0, a_1, a_2, a_3, a_4)$ being those computed by Algorithm 1.

4.2.3 Image Unwarping Procedure

Now, we are ready to show how to reconstruct a panoramic or perspective-view image from an omni-image with the aid of a pano-mapping table. Three cases can be identified, as described in the following.

A. Generation of a generic panoramic image from a given omni-image

Given an input omni-image G and a pano-mapping table T_{pm} , we may generate from G a corresponding *generic panoramic image* Q which is exactly of the same size $M \times N$ of T_{pm} . The steps are as follows. First, for each entry E_{ij} of T_{pm} with azimuth

angle θ_i and elevation angle ρ_j , take out the coordinates (u_{ij}, v_{ij}) filled in E_{ij} . Then, assign the color values of the pixel p_{ij} of G at coordinates (u_{ij}, v_{ij}) to the pixel q_{ij} of Q at coordinates (i, j) . After all entries of the table are processed, the final Q becomes a generic panoramic image corresponding to G . In this process, we may regard Q as the output of the pano-mapping f_{pm} described by the pano-mapping table T_{pm} with G as the input, i. e., $f_{pm}(G) = Q$.

B. Generation of a specific panoramic image

With the aid of a pano-mapping table T_{pm} with $M \times N$ entries, we may also generate from a given omni-image G a panoramic image Q of any size, say $M_Q \times N_Q$, which is the *panoramic projection* of the original scene appearing in G at any distance D with respect to the mirror center O_m with a projection band of any height H . An illustration of such an imaging configuration from a lateral view is shown in Fig. 4.5.

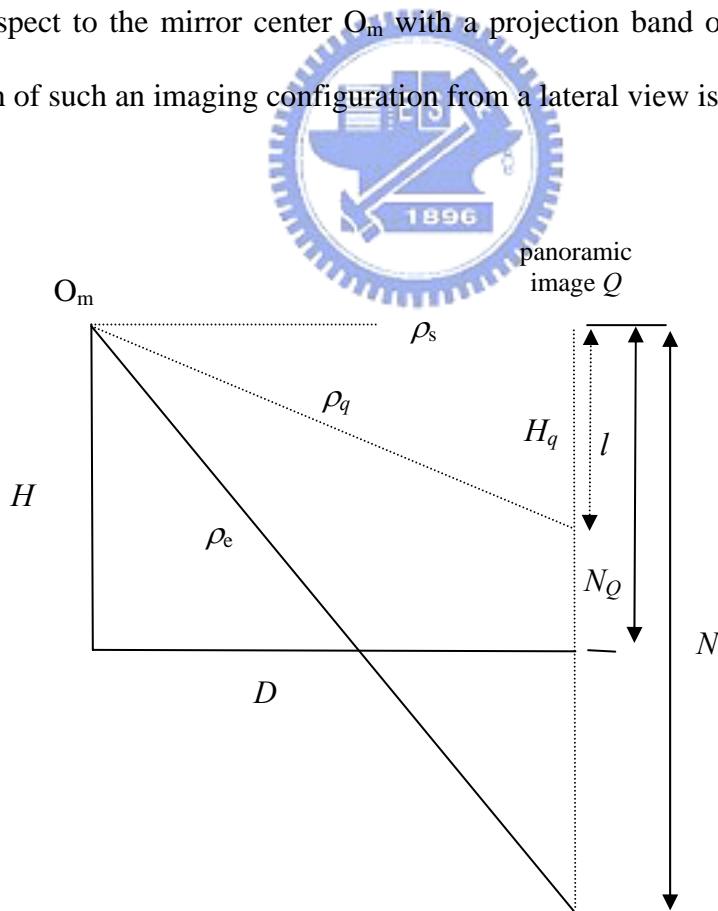


Fig. 4.5 Lateral-view configuration for generating a panoramic image.

The process for generating such an image is similar to that for generating the generic panoramic image described previously. First, we map each image pixel q_{kl} in Q at coordinates (k, l) to an entry E_{ij} in T_{pm} filled with coordinates (u_{ij}, v_{ij}) . Then, we assign the color value of the pixel p_{ij} of G at (u_{ij}, v_{ij}) to q_{kl} . Mapping of q_{kl} to E_{ij} is based on the use of the knowledge of the parameters M_Q , N_Q , D , and H as well as some related principles like triangulation, proportionality, etc.

In more detail, since the azimuth angle range 2π is divided into M_Q intervals in image Q , the image pixel q_{kl} at coordinates (k, l) is, by linear proportionality, the projection result of a light ray R_q with an azimuth angle $\theta_q = k \times (2\pi/M_Q)$. Since each azimuth angle interval in T_{pm} is $2\pi/M$, the index i of the corresponding entry E_{ij} in T_{pm} with the azimuth angle of θ_q is just

$$i = \theta_q / (2\pi/M) = [k \times (2\pi/M_Q)] / (2\pi/M) = k \times \frac{M}{M_Q} \quad (4.9)$$

where we assume M is a multiple of M_Q . In case not, the right-hand side of Eq. 4.9 should be replaced with its integer floor value.

Next, we have to compute the elevation angle ρ_q of the above-mentioned light ray R_q projecting onto pixel q_{kl} to decide the index j of E_{ij} . For this, since the height of the projection band is H and image Q is divided into N_Q intervals, by linear proportionality again, we may compute the height of R_q at D as

$$H_q = l \times \frac{H}{N_Q}. \quad (4.10)$$

Then, by trigonometry, we have the elevation angle ρ_q as

$$\rho_q = \tan^{-1}\left(\frac{H_q}{D}\right). \quad (4.11)$$

Therefore, we can compute the index j of E_{ij} by proportionality again as

$$j = (\rho_q - \rho_s) / [(\rho_e - \rho_s) / N] = \frac{(\rho_q - \rho_s) \times N}{(\rho_e - \rho_s)} \quad (4.12)$$

since the elevation angle range $\rho_e - \rho_s$ is divided into N intervals. In case the right side of Eq.4.12 is not an integer, it should be replaced by its integer floor value.

With the indices (i, j) of E_{ij} available, the content of E_{ij} , i. e., the coordinates (u_{ij}, v_{ij}) , may be obtained. And finally the color value of the omni-image G at (u_{ij}, v_{ij}) is assigned to the pixel q_{kl} at coordinates (k, l) of Q . After all pixels of Q are processed in the above way, the final content of Q is just the desired panoramic image.

C. Generation of a specific perspective-view image

Given an omni-image G and a pano-mapping table T_{pm} with $M \times N$ entries, we may also generate from G a perspective-view image Q of any size $M_Q \times N_Q$, which is the *perspective projection* of the original scene appearing in G onto a planar rectangular region A_P of any size $W \times H$ at any distance D with respect to the mirror center O_m . A top-view of the configuration for such an image generation process is shown in Fig. 4.6. The idea again is to map each image pixel q_{kl} in Q at coordinates (k, l) to an entry E_{ij} in T_{pm} filled with coordinates (u_{ij}, v_{ij}) , and then to assign the color value of the pixel p_{ij} of G at (u_{ij}, v_{ij}) to q_{kl} . Mapping of q_{kl} to E_{ij} is accomplished via the steps of computing the azimuth and the elevation angles θ_q and ρ_q associated with E_{ij} and corresponding to q_{kl} .

Referring to Fig. 4.6, we first calculate the angle ϕ in the figure. By trigonometry, we have

$$W^2 = D^2 + D^2 - 2 \times D \times D \times \cos \phi \quad (4.13)$$

from which ϕ may be solved to be

$$\phi = \cos^{-1} \left[1 - \frac{W^2}{2 \times D^2} \right]. \quad (4.14)$$

Also, it is easy to see from the figure that

$$\beta = \frac{\pi - \phi}{2}. \quad (4.15)$$

Next, we compute the index i of entry E_{ij} of table T_{pm} corresponding to pixel q_{kl} in image Q . First, let P_{ij} denote the intersection point of the light ray R_q projecting onto q_{kl} and the planar projection region A_p . Note that each entry E_{ij} has a corresponding P_{ij} . Then, we compute the distance ℓ between point P_{ij} and the border point P_r shown in Fig. 4.6 by linear proportionality as

$$\ell = k \times \frac{W}{M_Q} \quad (4.16)$$

since the projection region A_p has a width of W , the image Q has a width of M_Q pixels, and pixel q_{kl} has an index of k in the horizontal direction.

Also, by trigonometry we can compute the distance L between point P_{ij} and the mirror center O_m as

$$L = \sqrt{D^2 + \ell^2 - 2 \times \ell \times D \times \cos \beta} \quad (4.17)$$

and then the distance h from point P_{ij} to the line segment $\overline{O_m P_r}$ connecting O_m and P_r as

$$h = l \times \sin \beta. \quad (4.18)$$

So, the azimuth angle θ_q of point P_{ij} with respect to $\overline{O_m P_r}$ satisfies

$$\sin \theta_q = \frac{h}{L} = \frac{\ell \times \sin \beta}{\sqrt{D^2 + \ell^2 - 2 \times \ell \times D \times \cos \beta}}$$

which leads to

$$\theta_q = \sin^{-1} \left[\frac{\ell \times \sin \beta}{\sqrt{D^2 + \ell^2 - 2 \times \ell \times D \times \cos \beta}} \right]. \quad (4.19)$$

Finally, the index i of entry E_{ij} may be computed by linear proportionality as

$$i = \left\lceil \frac{\theta_q}{2\pi} \right\rceil \times M \quad (4.20)$$

where we assume the right-hand side of the above equality is an integer. In case not, it should be replaced by its integer floor value.

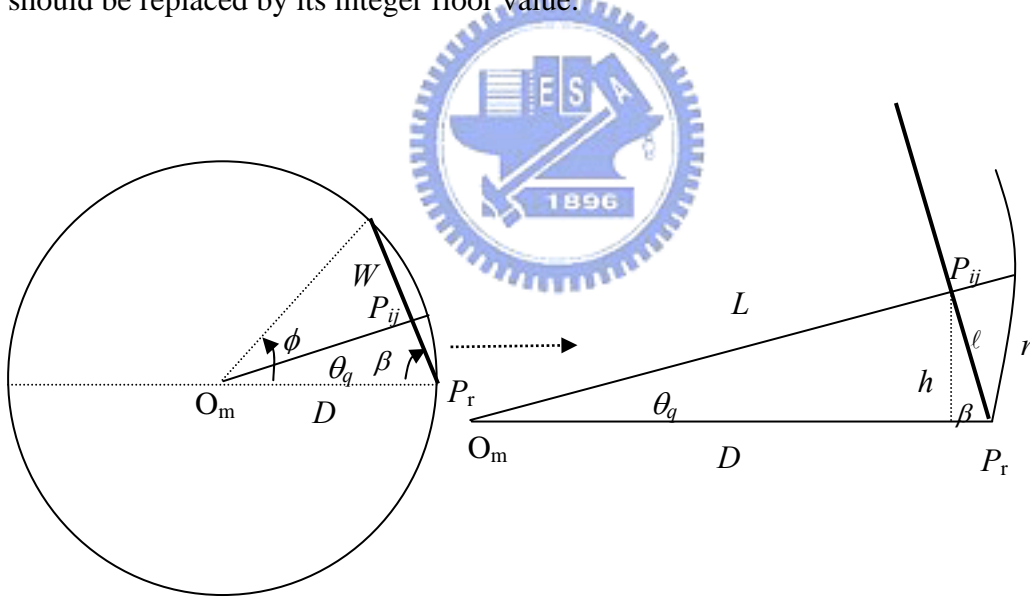


Fig. 4.6 Top-view configuration for generating a perspective-view image.

As to the index j of E_{ij} , it can be computed in a way similar to that for deriving Eqs. (4.10), (4.11) and (4.12) as follows:

$$H_q = l \times \frac{H}{N_Q}; \quad (4.21)$$

$$\rho_q = \tan^{-1} \left[\frac{H_q}{L} \right]; \quad (4.22)$$

$$j = \frac{(\rho_q - \rho_s) \times N}{(\rho_e - \rho_s)}. \quad (4.23)$$

With the indices (i, j) of E_{ij} ready, finally we can obtain the coordinates (u_{ij}, v_{ij}) in E_{ij} and assign the color value of the image pixel p_{ij} of G at coordinates (u_{ij}, v_{ij}) to pixel q_{kl} of Q at coordinates (k, l) . After all pixels of Q are processed, the final content of Q is just the desired perspective-view image.

4.3 Experimental Results

In our experiments, before doing the unwarping procedure, we carry out the landmark learning and the table creation procedures first. For these works, we provide a user-friendly interface as Fig. 4.4 shows, which can be used for identifying appropriate landmark pairs, as described in Section 4.2.1. Fig. 4.7(a) shows the final result of the learning procedure. Ten landmark point pairs were identified, as shown in this figure. Also, the coordinates of the image center, as well as the length of the “cut-off radius” within which the omni-image is invisible because of the self-occlusion of the omni-camera, are extracted automatically in the learning procedure using an algorithm developed in this study[37][38]. The region within the cut-off radius is marked by a circle with the coordinate values of the image center and the radius length printed at the bottom of Fig. 4.7(a). We used the cut-off radius to calculate one end ρ_e of the full range of the elevation angles. The learned landmark point pairs were used to estimate the coefficients of the radial stretching function as described by Eq. (4.2) in Section 4.2.2. Fig. 4.7(b) shows the fitted curve with the 10 learned landmark points superimposed on the drawing (marked with “+”).

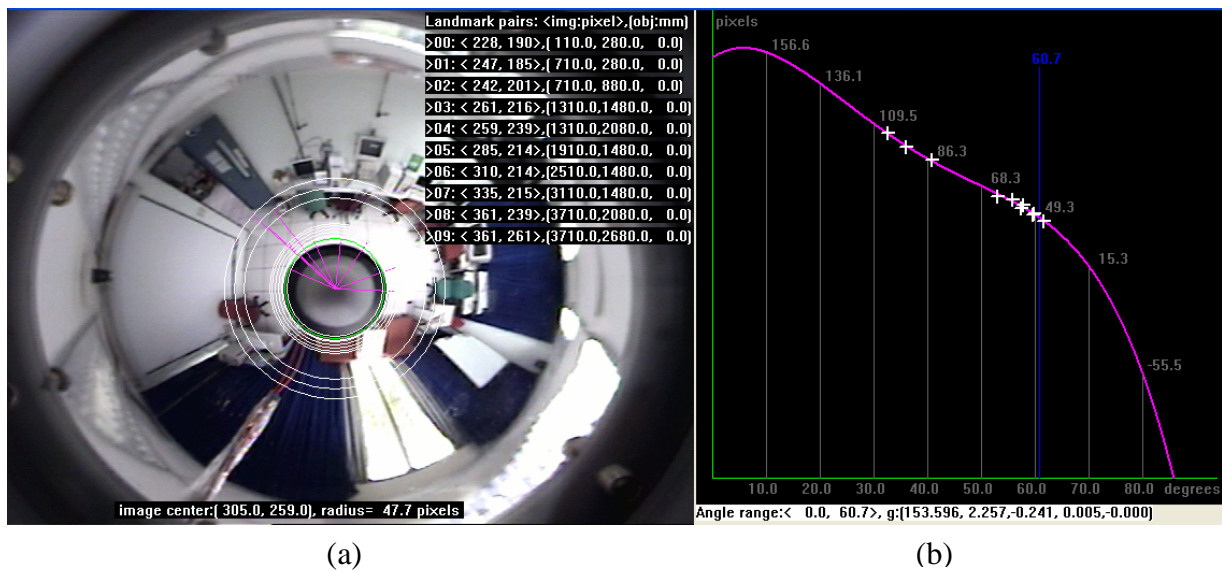


Fig. 4.7 Landmark calibration. (a) Landmark pairs. (b) Fitted radial stretching function.

With the coefficients estimated, Eqs. (4.1) and (4.2) were used to construct a pano-mapping table, as described in Section 4.2.2. The pano-mapping table can be used to unwarped an input omni-image into a panoramic or perspective-view image of any viewpoint, as described in Section 4.2.3. Fig. 4.8 shows some examples of images obtained from unwarping the omni-image shown in Fig. 4.8(c). Fig. 4.8(a) is a generated generic panoramic image. Note that this image is unique for an omni-camera. Fig. 4.8(b) is a panoramic image viewed at distance 184.1 cm with respect to the mirror center O_m of the omni-camera with a projection height of 208.5 cm. Fig. 4.8(d) is a perspective-view image viewed at distance 184.1 cm with respect to the camera in a projection region of $216.3 \times 208.5 \text{ cm}^2$. Note that Figs. 4.8(b) and (d) will look different by changing the relative positions with respect to the omni-camera. For example, Fig. 4.9 shows some perspective-view images generated from an omni-image video sequence at different projection distances.

The results shown in Fig. 4.8 can be compared with those of one conventional calibration method proposed by Mashita et al. [11], as shown in Fig. 4.9. There is

difficulty to tell which of the reconstructed images in Fig. 4.8(d) and Fig. 4.9(d) is better. But we can be sure that the proposed method is a relatively simple and generic solution for all kinds of omni-cameras and no a priori knowledge about the used omni-camera is needed in the proposed method. And this is really a great merit of the proposed method, which is not found in other methods.

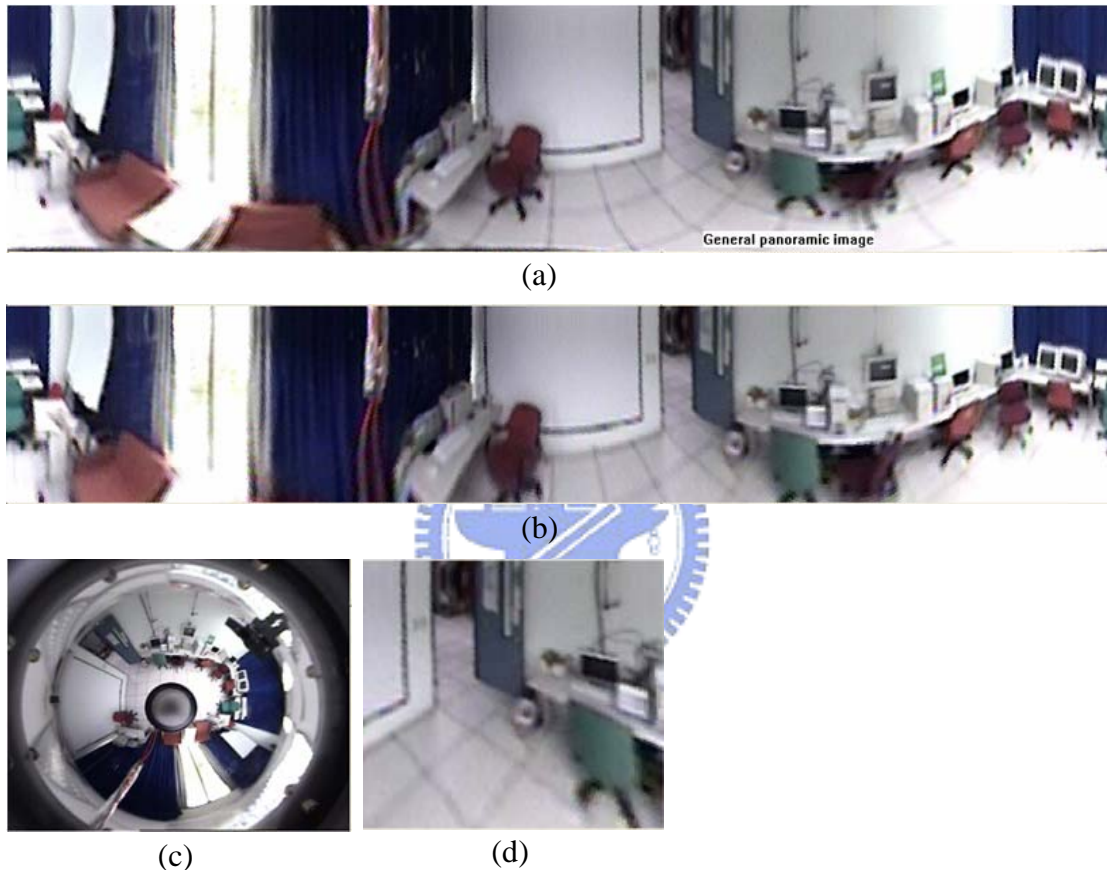


Fig. 4.8 Examples of image unwarping. (a) Generated generic panoramic image. (b) A generated panoramic image at distance 184.1 cm. (c) Original omni-image. (d) A generated perspective-view image.

4.4 Summary

A new approach to unwarping of omni-images taken by all kinds of omni-cameras is proposed. The approach is based on the use of pano-mapping tables proposed in this study, which may be regarded as a summary of the information conveyed by all the parameters of an omni-camera. The pano-mapping table is created

once forever for each omni-camera, and can be used to construct panoramic or perspective-view images of any viewpoint in the world space. This is a simple and efficient way to unwarping an omni-image taken by any omni-camera, when some of the physical parameters of the omni-camera cannot be obtained.

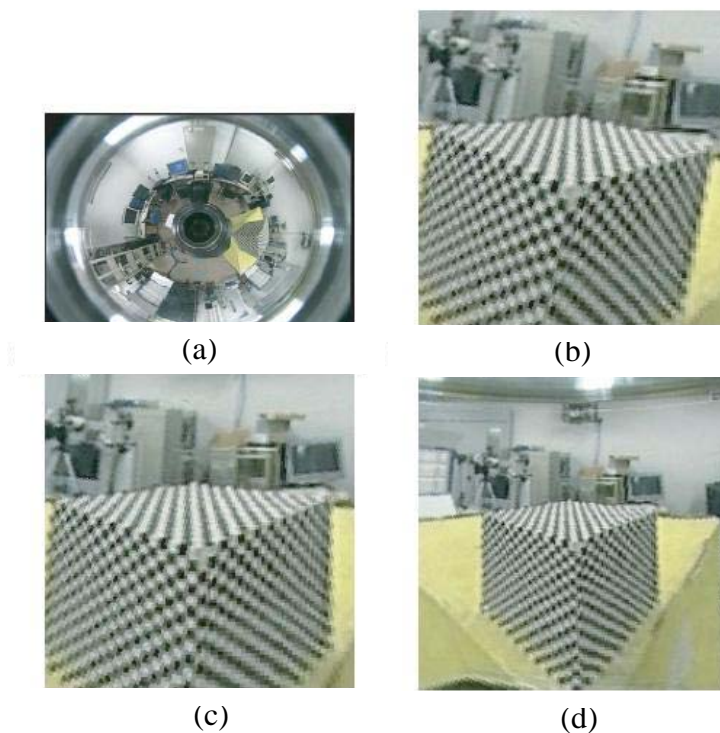


Fig. 4.9 Unwarped perspective view images using Mashita's calibration method [11].

- (a) Original omni-image.
- (b) An unwarping result by treating the camera as an SVP omni-camera.
- (c) Best result by manually adjusting some parameters in the case of (b).
- (d) An unwarping result using calibrated parameters.

A possible application of this approach is panoramic imaging in a visual surveillance system in a room [39]. We tried the idea in this study, and the experimental results shown previously come from this test. Fig. 4.10 shows some perspective-view snapshots generated from the taken videos of the surveillance system, which are useful for specific person or activity monitoring. Because the snapshots are unwarping results from a portion of an omni-image, their image qualities are not so good, compared with images taken by a normal camera. If this

issue is to be solved, we may add a pan-tilt-zoom camera into the surveillance system [32] to capture images of higher quality corresponding to the tracked versions of the perspective-view snapshots.



Fig. 4.10 Perspective-view images generated from an omni-image video sequence taken by a tracking system. (These images can be considered as some snapshots of a video.)



Chapter 5

A robust and accurate calibration method for coordinate transformation between display screens and their images

5.1 Introduction

Projection screens are used commonly in presentations nowadays [14-15] [40-42]. The projected image on the screen includes many types of contents. Usually, we use a laser pointer to point out mentioned targets in the image. It is advantageous to design an intelligent technique to locate the laser spot on the screen automatically by a computer. Possible applications of this technique include game interfacing, computer screen control, presentation paging control, simulation of clicking by a laser pointer, etc. In this study, we want to use the computer vision technique to solve this problem.

More specifically, after using a visual camera to take an image of a screen on which a laser spot appears, it is desired to design a robust and accurate method by computer vision techniques to measure the position of the laser spot, no matter where the spot appears and no matter what type of camera is used. To solve this problem, a *calibration* procedure is needed to build a position relationship between the projection screen and the taken image, followed by a transformation process from the image coordinates to the screen coordinates.

When the display screen is planar, intuitively the resulting issue is a traditional camera calibration problem between two planar planes [13]. When applying a traditional

calibration method, we must know some camera parameters like the focal length of the camera, the dimension of the CCD sensor in the camera, and so on. We also have to build a *lens distortion model* before doing the calibration work [1][13]. Finally, we have to solve some nonlinear equations to compute the solution for coordinate transformation parameters. If there are errors in the computed parameters, such traditional methods will reduce the final coordinate transformation precision. Especially, in the issue of simplifying the calibration computation process, it is usual to use a camera lens distortion model of low-order fitting functions in the radial direction of the lens. The results of the coordinate transformation will usually include unacceptable shift errors near the border of the image. Another kind of traditional method is homographic projection [45], by which the projector may be allowed to be at any pose with respect to the screen plane. However, camera and projector optics are modeled by perspective transformations, and the nonlinear distortion property of the optical lens is not considered.

In this study, we try to remove the above-mentioned drawbacks of traditional calibration methods. In particular, we focus on improving the accuracy of the coordinate transformation to eliminate the shift errors near the image border. Also, we simplify the algorithms to avoid complicated calculations in the calibration and coordinate transformation processes. We design three calibration patterns and an algorithm for robust extraction of geometric feature points from the calibration pattern images. Using the techniques of deformable template matching, local thresholding, and inverse distance-weighted interpolation, we achieve robust feature extraction and accurate coordinate transformations. Good experimental results were obtained, which show the feasibility of the proposed method. The experimental results were also compared with those of a well-known calibration method to show the superiority of the proposed method.

In the remainder of this chapter, we describe the proposed method in Section 5.2,

show some experimental results in Section 5.3, and finally make a summary of this chapter in Section 5.4.

5.2 Proposed Method

The proposed method includes three stages: (1) taking pictures of calibration patterns designed in this study; (2) extracting feature points from the pattern images; and (3) conducting coordinate transformations to accomplish the calibration work. The key idea, which is different from those of traditional calibration methods, is to use *sequentially three* calibration patterns instead of the conventional way of using only one. The three sequential calibration patterns designed for use in this study are shown in Fig. 5.1, including a white rectangular shape, a black rectangular shape, and a matrix of white dots on a black rectangular shape, which are called respectively the *white-rectangle pattern*, the *black-rectangle pattern*, and the *dot-matrix pattern* in the sequel. We capture the images of them, when they are projected on a display screen, with a CCD camera equipped at a proper location in the environment, and apply image analysis techniques to extract relevant image features for later processing. The images of the white-rectangle pattern and the black-rectangle one are used to extract the border information of the display screen effectively and to create a *threshold distribution map* (TDM). The border information and the TDM are then used further to assist the extraction of meaningful feature points, called *landmark points* hereafter, by local thresholding and deformable template matching techniques from the dot-matrix pattern image. The landmark points are then used for coordinate transformations by a technique of inverse distance-weighted interpolation. We describe the details in the following sections.

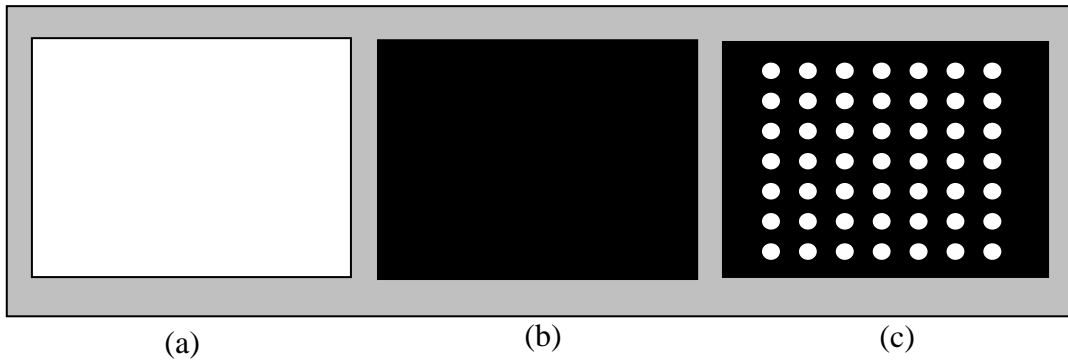


Fig. 5.1 Calibration patterns. (a) white-rectangle. (b) black-rectangle. (c) dot-matrix.

5.2.1 Use of Calibration Patterns

Fig. 5.2 shows the captured images of the three calibration patterns displayed on a projection screen. We can see the grayscale value variations in the images caused by uneven environmental lighting. This problem might cause erroneous feature extraction results and so make the calibration work fail. It is solved in this study by a technique of sequential analysis of three calibration pattern images. The advantages of using the white-rectangle pattern and the black-rectangle one in the calibration process include: (1) making easy and accurate extraction of the border of the display screen from the acquired image; and (2) creating a TDM from these two images for the later effective work of thresholding locally the dot-matrix pattern image into a binary one. The advantages of using the dot-matrix pattern in the calibration process include: (1) providing landmark points in the proposed calibration algorithm; and (2) dividing the display screen images into many small regions for use in the interpolation process which reduces geometric

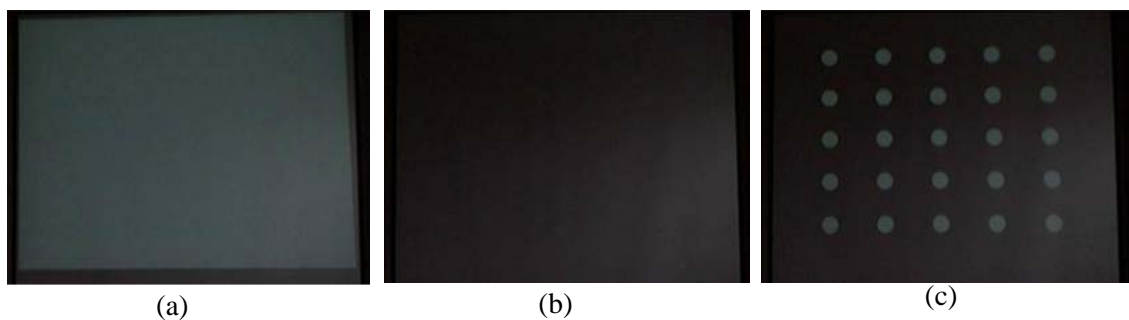


Fig. 5.2 Examples of captured images of Fig. 5.1.

distortions caused by imperfect camera lens optics. If there are $N \times N$ dots in the matrix, then we divide the display screen into $(N+2) \times (N+2)$ small regions, in each of which a coordinate transformation is conducted locally.

The TDM is a grayscale image, each of whose pixels has a threshold value for binarizing the pixel value of the dot-matrix pattern image at the corresponding pixel location. The threshold value at pixel location (i, j) in the TDM can be calculated, according to the concept of *local thresholding*, from the corresponding pixel locations (i, j) in the white-rectangle and in the black-rectangle pattern images in a way described as follows.

Let the grayscale values at pixel location (i, j) in the white-rectangle and in the black-rectangle pattern images be denoted by $w(i, j)$ and $b(i, j)$, respectively. From the two pattern images as shown in Figs. 5.2(a) and (b), we see that the outer parts of the display screen are imaged to be of approximately the same grayscale values, while the white and the black rectangles inside the display screens are imaged to be of great difference in their grayscale values. We define the pixel value $t(i, j)$ of the TDM t at pixel location (i, j) in the following way:

$$\begin{aligned} &\text{if } w(i, j) - b(i, j) > 20, \text{ then set } t(i, j) = b(i, j) + [w(i, j) - b(i, j)]/2; \\ &\text{else set } t(i, j) = 255. \end{aligned} \tag{5.1}$$

Basically, the above rule defines a threshold value at the *middle* of the black and the white pixel values in the two patterns, respectively, when there exists an “effective” pixel whose grayscale difference in the two images is large enough (> 20). Otherwise, we set the threshold value to be 255. Fig. 5.3 shows a TDM created by the images of Figs. 5.2(a) and (b).

The TDM is then used in a process of analyzing a dot-matrix pattern image d by

local thresholding. The aim is to obtain the white dots effectively. The result is a *binary dot-matrix pattern image* d' defined by:

$$\text{if } d(i, j) > t(i, j), \text{ then set } d'(i, j) = 255; \text{ else, } 0. \quad (5.2)$$

An example of using the TDM for extracting the dots from the dot-matrix pattern image is shown in Fig. 5.4(e). We see in the above process that the use of the white-rectangle and the black-rectangle pattern images provides the advantages of removing the outer parts of the display screen and providing local threshold values for binarizing precisely the image of the third calibration pattern, the dot-matrix pattern, into white dots for later processing.



Fig. 5.3 A TDM image created by images in Figs. 5.2(a) and (b).

5.2.2 Feature Extraction

Two kinds of landmark points are then extracted subsequently from the binary dot-matrix pattern image with aids from the original pattern images. The first is the center of each white dot in the image, which we mention as a *dot center* for simplicity in the sequel. The second kind of extracted landmark point is the intersection point of a screen border line with the line formed by a row or a column of the dot centers in the binary dot-matrix pattern image, which we call an *extended dot center* in the sequel. For contrast, the first kind of dot center within the dot-matrix pattern image is mentioned as the *original*

dot center. That is, we now have two types of landmark points now, *original dot center* and *extended one*. These extracted landmark points can be used as anchor points for calibration. They are marked by crosses “+” in Fig. 5.4(f). We also see from the results shown in Fig. 5.4 that the proposed method is quite effective in dealing with non-uniform lighting appearing in the acquired images.

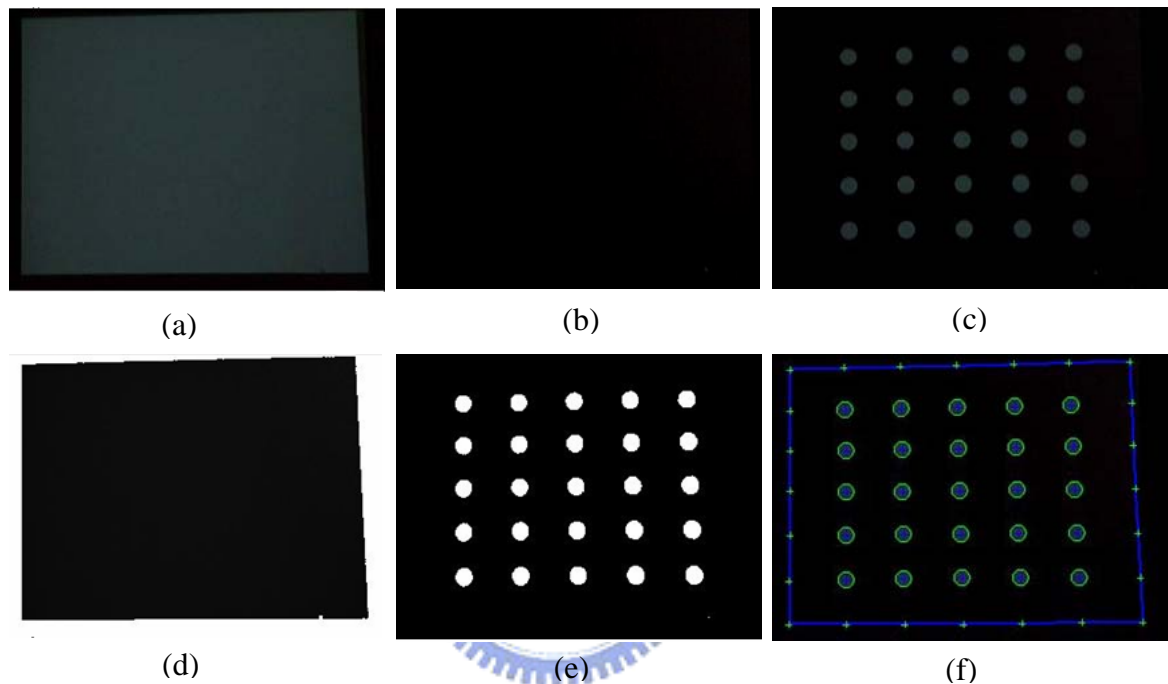


Fig. 5.4 An example of image analysis of dot-matrix pattern image. (a) White-rectangle pattern image; (b) Black-rectangle pattern image. (c) Dot-matrix pattern image. (d) TDM. (e) Binary image of (c). (f) Extracted feature points (with marks “+”).

We now describe how we extract these two types of landmark points in more detail. There are three major steps in the extraction process: (a) extraction of the screen border; (b) extraction of the original dot centers; and (c) extraction of the extended dot centers.

(a) Extraction of screen border

In order to extract the screen border, we establish a geometric model for the border shape and use an edge matching technique to fit the model, thus obtaining a *polygon* for use as the shape of the screen border. The main purpose is to have a more precise extraction result of the screen border shape by which the subsequent calibration work can

be performed more accurately. More specifically, considering the shape distortion of the rectangular screen in an acquired image caused by imperfect optical geometry of the camera lens, which is mostly barrel or pincushion distortion, we model the screen region in the image as an *eight-sided polygon* with eight vertexes and eight line segments, denoted by P_i and L_i , respectively, where $i = 0, 1, \dots, 7$, as shown in Fig. 5.5.

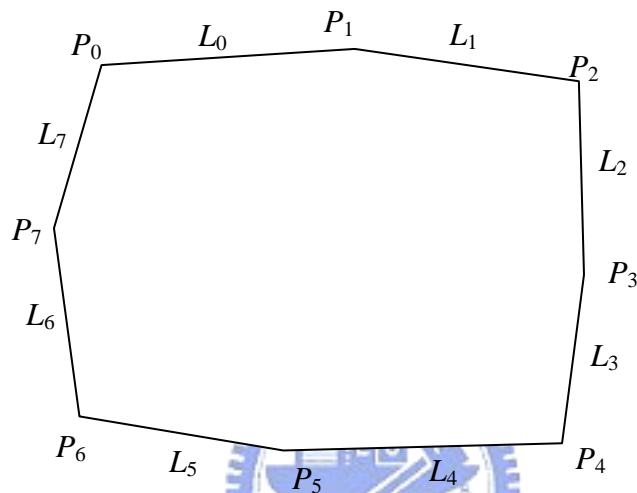


Fig. 5.5 Polygonal model of a screen region.

We then design a procedure to extract the vertexes from the TDM image precisely. First, two mutually-perpendicular lines are set at the center of the TDM image, as shown in Fig. 5.6. A 2×6 scan window as shown in Fig. 5.6(b) is moved from the TDM image center along the vertical line to its two ends to extract the vertexes P_1 and P_5 on the upper and lower border lines, respectively. P_3 and P_7 on the right and left border lines, respectively, are extracted similarly. Extraction of the vertexes is based on the concept of *edge detection* using the values of the 12 pixels in each scan window.

Specifically, referring to Fig. 5.6(b), let the average grayscale value of the group of the upper six pixels in the window be denoted as G_1 , and let that of the lower six pixels as G_2 . Then, either of the vertexes of P_1 and P_5 is detected by searching the maximum of the edge values $|G_1 - G_2|$ along the previously-mentioned vertical line. A pixel with the maximum

edge value is taken to be P_1 or P_5 . Search of P_3 and P_7 is conducted in a similar way.

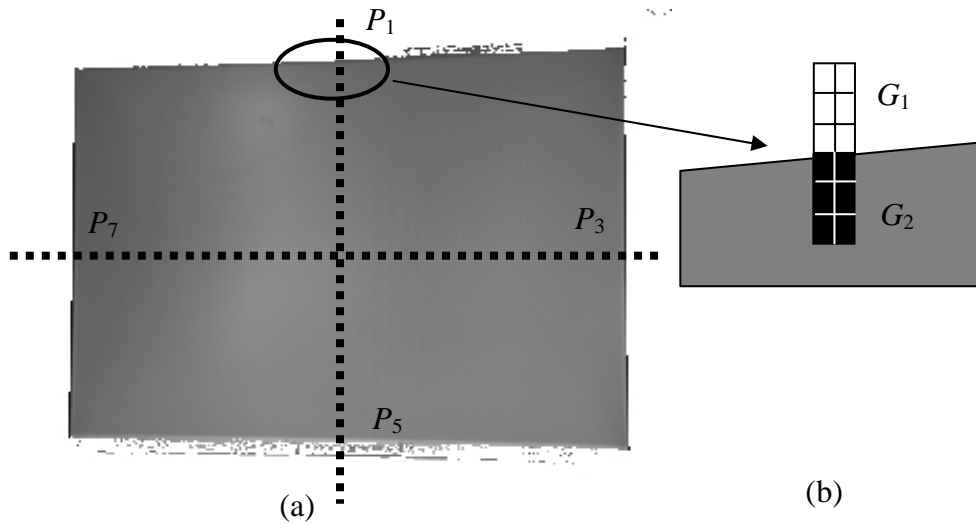


Fig. 5.6 Extracting the vertex P_1 from the TDM.

With P_1 , P_3 , P_5 , and P_7 extracted, we can extract the other vertexes and then the polygon sides accordingly. We only describe how we extract P_2 and the polygon sides L_1 and L_2 here. The others are extracted in similar ways. Referring to Fig. 5.7, first we define a *rectangular search window* using the coordinates of vertexes P_1 and P_3 , using the coordinates (x_1, y_1) of P_1 and the coordinates (x_3, y_3) of P_3 in the following way. The center P_a of the rectangular search window is taken to be (x_3, y_1) , and the width and the height of the window are taken to be $|x_3 - x_1| \times 2$ and $|y_3 - y_1| \times 2$, respectively. An illustration of the window formation is shown in Fig. 5.7(a).

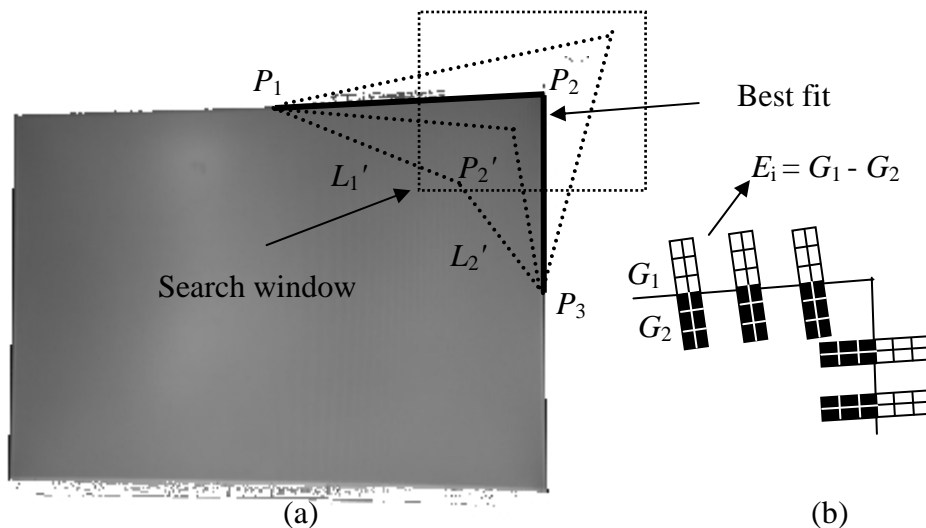


Fig. 5.7 Extracting P_2 , L_1 , and L_2 from TDM image.

Now, to find L_1 and L_2 , we conduct an *exhaustive search* of all possible line segments for L_1 and L_2 in the search window. For each pixel P_2' in the window, we construct first two candidate line segments L_1' and L_2' by connecting P_2' to P_1 and P_3 , respectively. We then place a fixed number of 6×2 or 2×6 scan windows (like those mentioned previously for detecting P_1 and P_3) on the candidate lines L_1' and L_2' and compute the edge value $|G_1 - G_2|$ for each of the scan windows. See Fig. 5.7(b) for an illustration. All such edge values are summed up to get a border fitting measure $E(P_2')$ for P_2' , which we call the *edge energy* of P_2' . Then the pixel $P_2'_{\max}$ with the maximum edge energy in the rectangular search window is chosen as the desired vertex P_2 . And the desired polygon sides L_1 and L_2 are just the line segments connecting $P_2'_{\max}$ to P_1 and to P_3 , respectively.

(b) Dot center extraction



The TDM and the dot-matrix pattern images are used next to extract the dot centers in the dot-matrix pattern image. The details are described in the following.

(b.1) Labeling connected components in binary dot-matrix pattern image

First, the grayscale image G_g of the dot-matrix pattern (for example, see Fig.5.4(c)) is thresholded into a binary dot-matrix pattern image G_b by Eq. (5.2). Fig. 5.4(e) illustrates an example of the result. Then, a connected-component analysis algorithm is applied to G_b to extract the connected components in G_b . Some constraints on the component area, the ratio of the component width to the height, etc. are used to filter out unwanted noise. The resulting connected components are white dots, denoted as $CC_i, i = 0, 1, \dots, n - 1$, in the sequel.

(b.2) Finding a cross-shaped group of five white dots near image center

Then we try to find a group of five white dots which form a *five-dot cross shape* near the center of G_b . The white dot nearest to the center G_b is first selected and denoted as C_0 .

The nearest dot obtained by searching upward from C_0 is selected next and denoted as C_3 . In the same way, the nearest dot obtained by searching downward from C_0 is denoted as C_4 . We also do leftward and rightward searches similarly and the results are denoted as C_1 and C_2 , respectively. See Fig. 5.8 for an illustration.

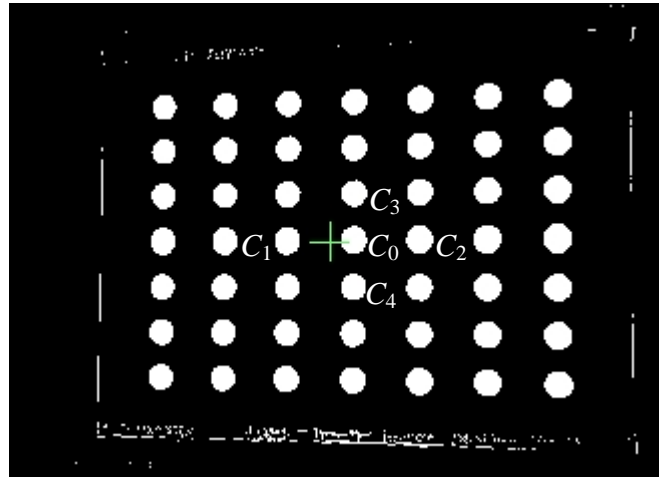


Fig. 5.8 A 5-dot cross shape near the center of binary dot-matrix pattern image.

Next, we want to construct a *cross-matrix pattern* (CMP) from the binary dot-matrix pattern image G_b . The CMP has $N \times N$ crosses, with each cross “+” being located at a dot center. For this purpose, we first constructed a *rough* CMP with their crosses *evenly* distributed horizontally and vertically with equal distances. An example constructed from Fig. 5.8 can be seen in Fig. 5.9. The horizontal distance W_g and the vertical one H_g between every two crosses are computed respectively as the averages of the halves of the width and the height of the five-dot cross shape obtained previously, i.e., W_g and H_g are computed as

$$W_g = (|C_1 - C_0| + |C_2 - C_0|)/2; H_g = (|C_3 - C_0| + |C_4 - C_0|)/2, \quad (5.3)$$

where $|\cdot|$ represents the distance between two white dots.

(b.3) *Fine tuning of positions of crosses in rough CMP to obtain desired CMP*

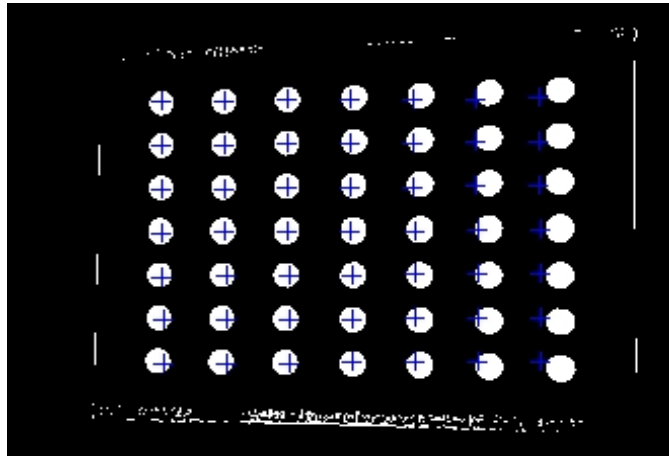


Fig. 5.9 Rough CPM superimposed on the binary dot matrix pattern image.

The crosses in the rough CMP are not located at the real centers of the white dots. See Fig. 5.9 for this phenomenon. But they can be used as the start positions for finding more accurate dot center locations. For this purpose, the rough CMP is first superimposed on the dot-matrix pattern image G_g with the center of the five-dot cross shape overlapping on the center of G_g , or equivalently, on the center of CC_0 . Then, we use the deformable template matching (DTM) method [43][44] to do the fine tuning work of finding more accurate dot centers.

A deformable template DT for circular region detection as shown in Fig. 5.10(b) with the parameters of its center (x_i, y_i) and radius r_i is used to detect each white dot in the dot-matrix image G_g . The detection is started from the position of the cross of a white dot CC_j in the rough CMP. And a search range, denoted as SR, is defined for the search of a best-match of the template DT with a desired dot. The width and the height of SR are defined to be the values of W_g and H_g defined in Eq. (5.3), respectively. SR is centered at a position in G_g corresponding to the center of a cross in G_b . See Fig. 5.10(a) for an illustration. At every *search point* (k, l) defined by a position (x_k, y_k) , denoted as SP_k , in the search range SR for a radius r_l of the template DT, the edge values of the eight search windows around the circle of the DT are computed and summed up as the edge energy of

the search point (k, l) . The best-match search point with the maximum edge energy is found after all search points are tried, and the corresponding SP_{\max} at location (x_{\max}, y_{\max}) is taken to be the desired center position for the white dot CC_j . A cross “+” is then drawn at (x_{\max}, y_{\max}) . We do the same process for all the other dots, and this completes the construction of the desired CMP. Fig. 5.4(f) shows an example of the final results of applying this process.

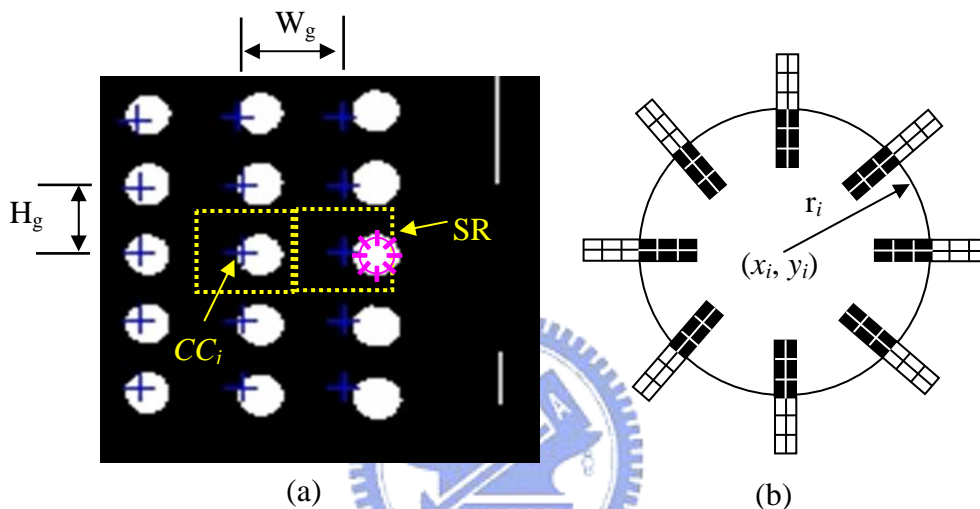


Fig. 5.10 Deformable template matching. (a) SR and CC_j . (b) Circular template with 8 scan windows.

(c) Extraction of extended dot centers on screen border

So far, we have obtained a CMP which includes accurate position information of the original dot centers, and an approximating polygon of the screen border. We need further the positions of the extended dot centers on the screen border. These positions can be calculated from the geometric information mentioned above. We describe the details below.

Referring to Fig. 5.11, an extended dot center point P_i on the left-hand side of the screen border can be extracted by utilizing the position information of the first and the second dots CC_{i1} and CC_{i2} on the i th row of the dot matrix. First, we obtain the line L_i which connects the centers of CC_{i1} and CC_{i2} and extend it to the left-hand side. If the line L_i is above vertex P_7 of the polygon, then we compute the intersection point of L_i with the

polygon side L_7 ; otherwise, we do so with the polygon side L_6 . The result is an extended dot center point P_i . We do similar works to obtain all the other extended dot centers. The details are omitted. In Fig. 5.4(f), the points marked with crosses “+” on the screen border are the extraction results.

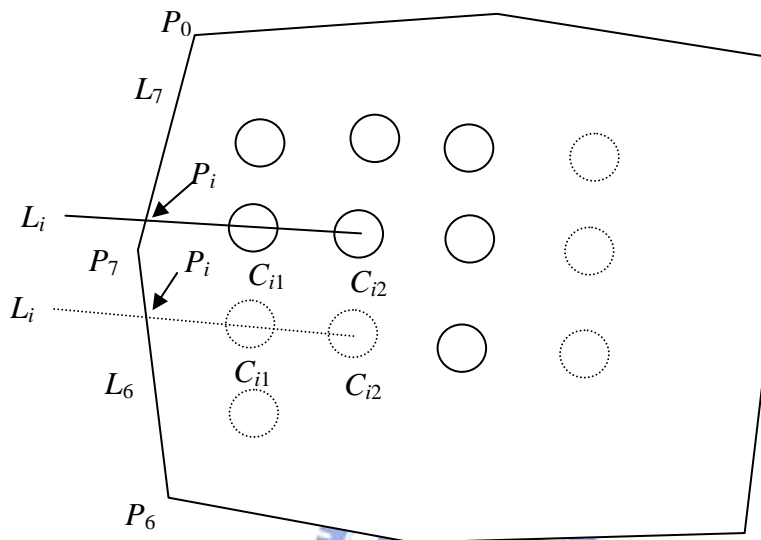


Fig. 5.11 Extracting Landmark point P_i at left side.

5.2.3 Coordinate transformation from image coordinates to screen coordinates

After all the original and the extended dot centers are extracted, they form an $(N+2) \times (N+2)$ dot center matrix. For each of the dot center points, denoted by DP_i , let its screen coordinates be denoted by (X_i, Y_i) and its corresponding image coordinates by (x_i, y_i) where $i = 0, 1, \dots, (N+2) \times (N+2) - 1$. The $(N+2) \times (N+2)$ points divide the field of view (FOV) (i.e., the full image) into $(N+3) \times (N+3)$ basic regions BR_i , which consists of two portions: (1) the $(N+1) \times (N+1)$ inner basic regions inside the display screen, together forming a rectangular region called *display area*, denoted by R_{display} , and (2) the *other* basic regions between the border of the display screen and the outer border of the FOV, together forming a round region called *border area*, denoted by R_{border} .

The last step of the desired calibration is a process of coordinate transformation from the image coordinates to the screen coordinates. The detail of the transformation, which is

based on the concept of inverse distance weighting interpolation (IDWI), is described in the following.

Referring to Fig. 5.12, let image point P_i with coordinates (x_i, y_i) be located within one basic region R_j of the display area $R_{display}$ and let the image and screen coordinates of the four corner points (i.e., four dot center points) $P_{j0}, P_{j1}, P_{j2},$ and P_{j3} of R_j be (x_{jk}, y_{jk}) and (X_{jk}, Y_{jk}) , respectively, where $k = 0, 1, 2, 3$. Let the distance between image points P_i and P_{jk} be denoted as d_k . We use the IDWI method to calculate the corresponding screen coordinates (X_i, Y_i) of P_i as follows, where w_k denotes $1/d_k$ and W denotes the sum of w_0 through w_3 (i.e., $W = w_0 + w_1 + w_2 + w_3$):

$$X_j = (X_{j0} \times w_0 + X_{j1} \times w_1 + X_{j2} \times w_2 + X_{j3} \times w_3) / W, \quad (5.4)$$

$$Y_j = (Y_{j0} \times w_0 + Y_{j1} \times w_1 + Y_{j2} \times w_2 + Y_{j3} \times w_3) / W. \quad (5.5)$$

It is noted that the screen coordinates (X_{jk}, Y_{jk}) are assumed to be known in advance.

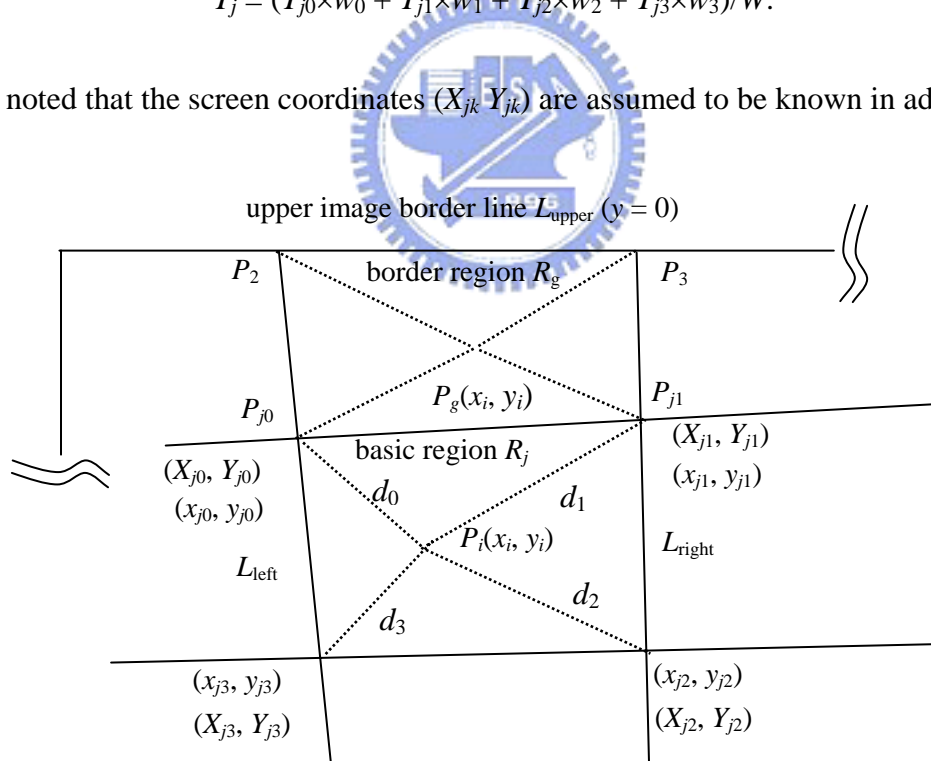


Fig. 5.12 Calculating screen coordinates (X_j, Y_j) in a basic region R_j and in border area R_g .

When a given image point P_i is outside $R_{display}$ and in R_{border} , referring to R_g of Fig. 5.12 for example, we do not have a confining basic region like R_j above to enclose P_i . We have to create such an enclosing region for the above IDWI method to be applicable. For

this, using the case illustrated by Fig. 5.12 as an example, let the basic region below P_g be R_j as mentioned above with L_{left} and L_{right} as its two vertical side lines and P_{j0} and P_{j1} as its two upper corner points. Then we compute the intersection points of L_{left} and L_{right} with the upper image border line L_{upper} (with equation $y = 0$) and let the results be denoted as P_2 and P_3 , respectively. Then, with P_{j0} , P_{j1} , P_2 , and P_3 as the corner points of a new basic region R_g , we can apply an IDWI process similar to the above-mentioned one to compute the corresponding screen coordinates for the image point P_g .

5.3 Experimental results

In this section, we show some experimental results of coordinate transformations from image coordinates to display screen coordinates by two different calibration methods. One is the proposed method and the other a method proposed by Tsai [13]. The landmark points used in these two methods were extracted from an identical image of calibration patterns. The only difference is the way adopted to do the coordinate transformation. First, we briefly describe Tsai's plane-to-plane calibration method which was implemented in this experiment. Then, we show the results of the two methods using the identical set of image points.

5.3.1 Plane-to-plane calibration by Tsai [13]

Assume that the landmark points in the world space (for our case here, on the planar display screen) are (X_i, Y_i, Z_i) , $i = 0, 1, \dots, N$, with $Z_i = 0$, and their corresponding image feature points are (u_i, v_i) . The relationship between (u_i, v_i) and (X_i, Y_i, Z_i) can be described by the following equations:

$$S_x = \frac{w_i}{w_s} (\text{pixel} / \text{mm}), \quad S_y = \frac{h_i}{h_s} (\text{pixel} / \text{mm}) \quad (5.6)$$

$$C_x = \frac{w_i}{2}, \quad C_y = \frac{h_i}{2} \quad (5.7)$$

$$u_d = \frac{x_i - C_x}{S_x}, \quad v_d = \frac{y_i - C_y}{S_x}, \quad r^2 = u_d^2 + v_d^2 \quad (5.8)$$

$$u_i = u_d(1 + \kappa_1 r^2), \quad v_i = v_d(1 + \kappa_1 r^2) \quad (5.9)$$

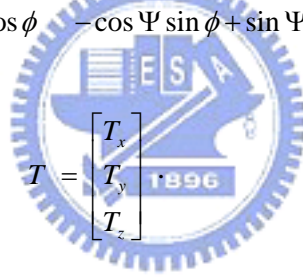
$$u_i = f \frac{x}{z}, \quad v_i = f \frac{y}{z} \quad (\text{perspective projection}) \quad (5.10)$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = R \begin{bmatrix} X_i \\ Y_i \\ Z_i \end{bmatrix} + T \quad (\text{rigid body transformation}) \quad (5.11)$$

where

$$R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \quad (5.12)$$

$$= \begin{bmatrix} \cos \Psi \cos \theta & \sin \Psi \cos \theta & -\sin \theta \\ -\sin \Psi \cos \phi + \cos \Psi \sin \theta \sin \phi & \cos \Psi \cos \phi + \sin \Psi \sin \theta \sin \phi & \cos \theta \sin \phi \\ \sin \Psi \sin \phi + \cos \Psi \sin \theta \cos \phi & -\cos \Psi \sin \phi + \sin \Psi \sin \theta \cos \phi & \cos \theta \cos \phi \end{bmatrix}$$



$$T = \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (5.13)$$

In Eq. (5.6), (w_i, h_i) is the dimension of the image in the computer memory (640×480 pixel² in this experiment), and (w_s, h_s) is the physical dimension of the used CCD sensor (3.2×2.4 mm² in this experiment). In Eq. (5.7), (C_x, C_y) is the image center. In Eq. (5.8), (u_d, v_d) specifies the distorted image point of (u_i, v_i) caused by optical imperfection. In Eq. (5.9), κ_1 is the radial distortion parameter of the camera lens. Here we assume that only the first order distortion is considered. In Eq. (5.10), (u_i, v_i) is the perspective projection of (x, y, z) onto the image plane with the camera lens center of the camera as the origin of the camera coordinate system. In Eq. (5.11), (x, y, z) is the result of the coordinate transformation of (X_i, Y_i, Z_i) by rigid body rotation and translation specified by the rotation matrix R and the translation vector T . In Eqs. (5.12) and (5.13), R and T are represented by

detailed parameters.

With sufficient known landmark point pairs (u_i, v_i) and (X_i, Y_i, Z_i) , we can use Tsai's method [13] to calculate all the parameters R , T and κ_1 . Then, we can use these parameters to calculate the world coordinates (X_i, Y_i, Z_i) of any image point (u_i, v_i) to complete the coordinate transformation, as was done in our experiment.

5.3.2 Comparison of results of proposed method and Tsai's method

Referring to Fig. 5.4 again, we used calibration patterns Figs. 5.4(a) through (c) to extract landmark points which are marked in Fig. 5.4(f) with "+." These landmark points were used to conduct calibration both by the proposed method and by Tsai's method as described previously. After the calibration process, the dot centers of the dot-matrix image were used to perform the coordinate transformation to get the coordinates of the dot centers in the display screen. Both results by using a 5×5 dot-matrix pattern for calibration are showed in Figs. 5.13 and 5.14 for comparison. Fig. 5.13 shows the results using the set of the extracted image points which are identical to those used for calibration. Fig. 5.14 shows the results using a set of image points which were extracted from a new dot matrix pattern image. In the figures, the listed data $r:\langle xxx, yyy \rangle$ specify the real coordinates of the dot centers in the dot-matrix pattern, $c:\langle xxx, yyy \rangle$ specify the coordinate transformation results using the proposed method, and $t:\langle xxx, yyy \rangle$ specify the coordinate transformation results using Tsai's method.

Observing Figs. 5.13 and 5.14, we can see that the coordinate transformation errors of Tsai's method are larger than those of the proposed method, especially near the image border. In Fig. 5.13, we see that no coordinate transformation error was created by the proposed method. This is owing to the fact that the input image points used in the transformation are the same as those used in calibration. In Fig. 5.14, we can see some small coordinate transformation errors created by the proposed method. They were caused

by small image variations, leading to extraction of the dot centers at slightly different positions. But these errors are still smaller than those yielded by Tsai's method.



Fig. 5.13 Comparison using calibration image points. The precise landmark position, the computed position by proposed method, and that by Tsai's method are listed under each dot.

5.4 Summary

A robust and accurate method for calibration and coordinate transformations from image coordinates to display screen positions has been proposed. By using three sequentially displayed calibration patterns, including a white-rectangle shape, a black-rectangle shape, and a dot matrix, relevant landmark points can be extracted accurately and robustly for calibration. Deformable pattern matching is used for creating more accurate geometric models for the display screen shapes in acquired images. The transformation of the coordinates of a point in the image to the coordinates of the display

screen is done by using an inverse distance weighting interpolation method. Comparing the results with a conventional calibration method, we showed that the proposed method has the merits of more accuracy, robustness, and simplicity.



Fig. 5.14 Comparison using re-captured image points. The precise landmark position, the computed position by proposed method, and that by Tsai's method are listed under each dot.

Chapter 6

A camera mouse for computer cursor control

6.1 Introduction

It is a common practice to control the cursor of a computer with a mouse on a flat pad at a close distance to the computer monitor. However, in cases of playing shooting games on computers or making presentations on large screens, it is required to maintain a certain distance from the user to the monitor. Also, the user tends to stand up in such cases, especially in the latter case of making presentations. It is so inconvenient to hold the mouse to control the cursor. It is desired to have a certain type of hand-held device, which can be operated in the air, for these applications.

From the technical point of view, we may adopt three ways to design such a kind of device: (1) using a conventional wireless remote controller with capabilities of paging and cursor movement control; (2) using a controller with a capability of detecting the device movement, which may be achieved by the use of inertial sensing devices like gyroscopes, accelerometers, etc. [46][47][48]; and (3) using a visual device like a video camera which has a self-locating capability provided by computer vision techniques [49][50].

Some drawbacks can be found in the first and second approaches. The main drawback inherent in the first approach is that the operator uses his/her fingers to push buttons. This results in slow responses, and is thus inappropriate for fast-speed game control. The main

drawback of the second approach is that inertial sensing techniques are not mature yet and the costs of the devices are high.

On the contrary, digital cameras and related devices of CCD sensors are becoming cheaper and popular. Computer vision methods based on the use of such devices also can be implemented more stably and reliably nowadays. Therefore, in this study we try to adopt the third approach and use a web camera as the hand-held device for cursor control.

Some related studies can be found in [49][50][51][52]. Nesi and Bimbo [49] proposed a kind of vision-based 3D mouse which used stereo vision for hand tracking and gesture recognition in the 3D space. The mouse position was represented by the 3D position of the hand that was estimated by computing the center of gravity at each time instant. Two independent hand postures were defined in order to emulate the buttons on traditional 2D mouse for switching off the transmission of hand movements. Dementhon and Davis [50] developed another kind of vision-based 3D mouse which was based on an algorithm of 2D-to-3D correspondences. The mouse was a small object held in one hand of the user, on which there were four non-coplanar infrared sources. One camera was fixed next to the computer monitor and adjusted to face the user. The centroids of the four spots were computed by the micro-controller integrated with the camera, and then transmitted to the computer to calculate the pose of the 3D mouse sixty times per second.

Yang and Tsai [51] proposed an inside-out vision-based 3D mouse, which is a camera held by hand to view a square mark in front of the mouse. The orientation and the position of the mouse are computed via monocular images of the mark. Resolution adjustments and some speed constraints were proposed to reduce computation errors. Simulations and real image sequences were both conducted. Li, Hsu and Pung [52] proposed a 3D mouse which

uses a mirror and a single camera to restore the 3D position of a finger tip. The camera is positioned in such a way that it captures both the hand as well as its mirror image. The captured images are then processed to extract the contour of the hand for locating the position of the finger tip. A prototype system was implemented and the performance of the 3D mouse before different backgrounds was analyzed.

All the above-mentioned methods are vision-based 3D mice which compute 3D information of the mouse position. For many applications, 3D information is not necessary, and a “vision-based 2D mouse” is sufficient. In this study, we concentrate on the design of such a kind of mouse. More specifically, we hold a web camera in hand as the mouse and let it look at a computer monitor screen. After taking an image of the display screen, we use image processing techniques to detect and track appropriate features of certain artificially-attached landmarks attached on the monitor and the monitor screen corners, thus achieving the function of locating the cursor which is controlled by the in-air movement of the hand-held camera. The experimental results show the feasibility of the proposed method.

Some merits of the proposed method are: (1) the method requires no complicated camera calibration; (2) the method is reliable because of the combined use of display feature detection and tracking; (3) the method is robust against loss of one or two landmark points in the tracking, which allows the user to have high freedom and space for hand movement; (4) the method allows unintended device rotation by affine transformation to correct the rotation error.

In the remainder of this chapter, we describe in detail the proposed method in Section 6.2. In Section 6.3, we show some experimental results. Finally, we make a summary of

this chapter in Section 6.4.

6.2 Proposed Method

A system setup for the proposed method is illustrated in Fig. 6.1. In the sequel, we call the web camera we use as the mouse a *camera mouse*. The camera mouse is a video camera. The video taken by the mouse is transmitted through a USB or wireless connection to the computer for image processing and cursor position determination. It is desired to allow the mouse to have a larger movement area in the air, widening the application domain of the mouse. For this purpose, we make the following two assumptions.

- (1) The camera mouse is operated at a sufficient distance from the computer monitor screen so that the area of the field of view (FOV) of the camera is at least four times of the area of the screen (i.e., both the width and the height of the FOV are two times of those of the screen, respectively).
- (2) The four corners of the outer frame of the computer monitor are attached respectively with four landmarks, each being a black circle at the center of a white rectangular shape.

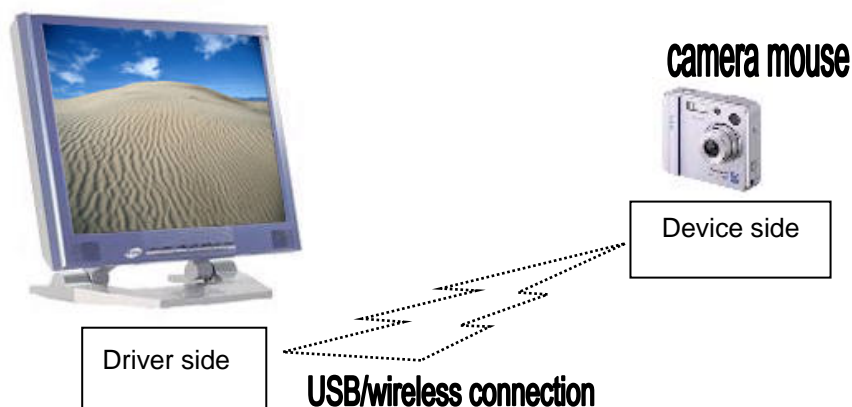


Fig. 6.1 Illustration of proposed camera mouse system.

The reason why we attach landmarks for corner detection is that the colors of the image displayed in the computer monitor sometimes will be identical to that of the outer frame of the monitor such that detection of the monitor corners becomes difficult by image processing. On the contrary, with the black circle against the white background on the landmark, detection of the landmarks becomes easier. With the landmarks being detected, we can then locate the corners of the monitor screen easily because the relative location of each corner with respect to the corresponding landmark center is fixed. Fig. 6.2 shows an example of images of a computer monitor in which an image of a boy is displayed.



Fig. 6.2 An example of computer monitor images.

In the proposed method, the camera mouse system initially, in the *detection stage*, detects the four landmarks and utilizes them to locate the four corners of the computer monitor screen. Subsequently, the system initializes a *tracking stage*, in which it tries to track the four landmarks in each image frame in the video sequence taken with the camera mouse. Sometimes the hand movement might be too large such that one or two of the four corners disappear in the taken image sequence. The proposed system will, in such cases, use the position information of the existing corners in the current image frame as well as

that of the corners in the last frame to predict the positions of the missing corners. The prediction is based on the use of the movement vectors of the corners. This type of prediction is conducted until the missing corners appear again in the FOV of the camera mouse.

With the four corners of the monitor screen in an image being detected, we can find next the relative location of the image center with respect to a corner of the monitor screen, which supposedly is the desired computer cursor position pointed to by the camera mouse. More details of the proposed method are described in the following.

6.2.1 Locating landmarks and computer monitor screen border in the detection stage

We locate the landmarks and the border of the computer monitor screen in the following way.

1. Take an image frame from the video taken by the camera mouse, and threshold it into a binary image I with a pre-determined threshold value.
2. Find black regions in I by a connected component labeling algorithm [24], aiming to detecting the black circular regions in the landmarks.
3. Filter out non-circular regions by checking the appropriateness of the area and the width-to-height ratio of each black region.
4. In an exhaustive manner, check in the following way the respective centers C_1, C_2, C_3, C_4 of every four remaining circular regions CR_1, CR_2, CR_3, CR_4 to see if the quadrilateral shape S formed by C_1 through C_4 meets the condition of being similar to the rectangular shape of the computer monitor screen:
 - a. check if the opposite sides of S are roughly equal in length, i.e., check if $|C_1 - C_2| \approx |C_3 - C_4|$ and if $|C_1 - C_4| \approx |C_2 - C_3|$, where the notation $|\cdot|$ means the

- distance between two points;
- b. check if the four corners formed by C_1 through C_4 are all of roughly right angles;
 - c. check if the width-to-height ratio R of S is roughly equal to that of the computer monitor which is measured manually in advance, where R is computed as the ratio of $(|C_1 - C_2| + |C_3 - C_4|)/2$ over $(|C_1 - C_4| + |C_2 - C_3|)/2$.

Fig. 6.3 shows an illustration of checking the above three conditions.

5. If there exists at least a group of four black regions which meet the above conditions, then collect all of them into a candidate set D_L of landmark regions and continue; otherwise, go to Step 1 to process the next image frame.
6. Find out the real landmark circles in D_L in the following way according to a concept of *edge strength* of the shape formed by the black region centers. Refer to Fig. 6.4 for an illustration of the following steps.
 - a. For each group G_j of four black regions in D_L with region centers C_1 through C_4 , take a side of the corresponding quadrilateral shape S_j , say C_iC_{i+1} , and find the corresponding parallel side $C_i'C_{i+1}'$ of the computer monitor screen border by applying the Hough transform within a certain rectangular search window W to the left or right of line C_iC_{i+1} or below or above it.
 - b. Take the corresponding peak value in the Hough space as the *edge strength* E_i of the detected line $C_i'C_{i+1}'$.
 - c. Perform the last two steps for all the four sides of S_j and sum up the four edge strengths E_1 through E_4 to obtain a *total energy strength* E_j for G_j .

- d. Find the maximum total energy strength E_m in the D_L , and take the corresponding group S_m of four black regions as the desired set of landmark circles.

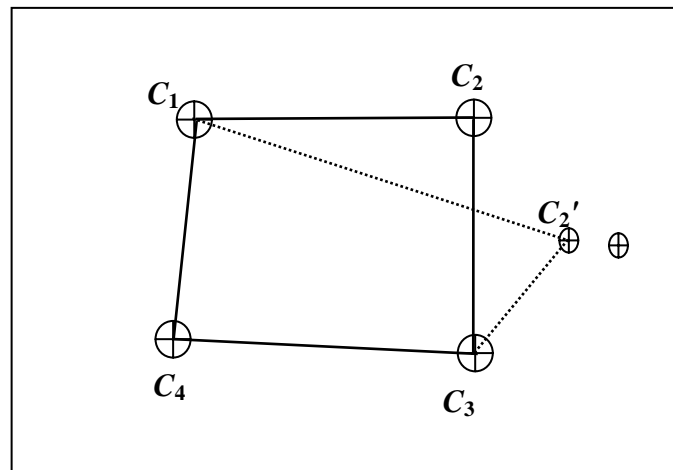


Fig. 6.3 Illustration of detecting candidate landmark circles.

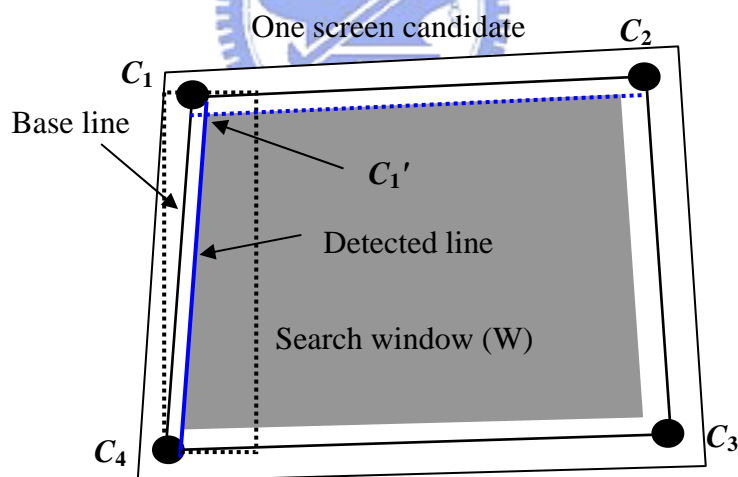


Fig. 6.4 Detecting of border lines.

6.2.2 Computing cursor location in the tracking stage

In the above process of detecting landmark circles in an image frame of the taken video of the computer monitor, we also obtain the border of the computer monitor screen,

i.e., the four sides $C_1'C_2'$, $C_2'C_3'$, $C_3'C_4'$, $C_4'C_1'$ of the screen. We may now use such border information to compute the location of the cursor in the computer monitor screen. Note that the position of the cursor is just the position of the image center *with respect to* the upper-left corner of the computer monitor screen, which we assume to be C_1' in the sequel.

Assume that the points C_1' through C_4' have image coordinates (x_1', y_1') through (x_4', y_4') , respectively, and that the image center, denoted as C_0' , have image coordinates (x_0', y_0') . Without perspective transformation created by camera movement and rotation, it is not difficult to figure out that the relative cursor position with respect to C_1' may be computed simply to have coordinates (u_0, v_0) with $u_0 = k(x_0' - x_1')$ and $v_0 = k(y_0' - y_1')$ in the monitor screen coordinate system, where k is a scaling factor.

However, perspective transformation does exist, and so the above simple cursor coordinate computation must be modified. For this purpose, we use an affine transform which is defined as a mapping of the four corners C_1' through C_4' of the perspectively-transformed monitor screen shape S' to the four corners C_1'' through C_4'' of a *corrected* rectangular screen shape S'' . Let C_1'' through C_4'' have coordinates (x_1'', y_1'') through (x_4'', y_4'') with C_1'' as the origin. How to define these coordinates will be described later in this section.

On the other hand, an affine transform may be described as

$$x'' = ax' + by' + c, \tag{6.1-1}$$

$$y'' = dx' + ey' + f, \tag{6.1-2}$$

where (x', y') and (x'', y'') are the coordinates of corresponding corners of S' and S'' , respectively, and a through f are six unknown coefficients to be determined. Substituting the coordinate data of three of the four corresponding corner pairs of S' and S'' into the

above equations, we can solve the six coefficients a through f . Then, the new coordinates (x_0'', y_0'') of the image center C_0' may be computed accordingly, and so may the new coordinates (x_1'', y_1'') of the upper-left corner C_1' of the monitor screen. In turn, the desired position (u_0, v_0) of the cursor may finally be computed as

$$u_0 = k(x_0'' - x_1'') = k \times [a(x_0' - x_1') + b(y_0' - y_1') + c]; \quad (6.2-1)$$

$$v_0 = k(y_0'' - y_1'') = k \times [d(x_0' - x_1') + e(y_0' - y_1') + f]. \quad (6.2-2)$$

Fig. 6.5 shows an illustration of the above process.

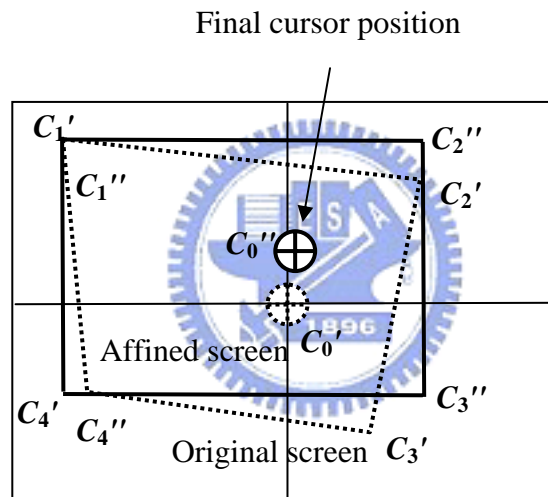


Fig. 6.5 Affine transformation and cursor position computation.

6.2.3 Computing corner coordinates of corrected rectangular monitor screen shape

We now describe how we define the coordinates (x_1'', y_1'') through (x_4'', y_4'') of the four corners C_1'' through C_4'' , respectively, of the corrected rectangular monitor screen shape S'' mentioned previously. Let the origin C_1'' be given the coordinates $(0, 0)$. Next, we compute the side lengths L and H for use as those of S'' from the corners C_1' through C_4' of the perspective-transformed monitor screen shape S' in the following way:

$$L = (|C_1' - C_2'| + |C_3' - C_4'|)/2, \quad (6.3)$$

$$H = (|C_1' - C_4'| + |C_2' - C_3'|)/2, \quad (6.4)$$

where $|\cdot|$ means the distance between two points. Then, the coordinates of corner C_2'' of S'' are set to be $(L, 0)$, those of C_3'' to be (L, H) , and those of C_4'' to be $(0, H)$.

6.2.4 Dynamic tracking of landmarks for continuous cursor position computation

So far we have described how we find the locations of the landmarks specified by their centers C_1 through C_4 , as well as the corners of the monitor screen C_1' through C_4' . Because the relative position between each pair of C_i and C_i' is fixed, we only have to track the landmark positions for continuous cursor position computation in the subsequent cycles, and so save time in the entire process.

Many methods have been proposed for target tracking [53][54][55]. In this study, we adopt the method of deformable template matching [56], which is effective for tracking objects with fixed geometric shapes. More specifically, we use each detected circular landmark center in the previous cycle to generate a tracking window for the current cycle. And within each tracking window, we try to detect exhaustively all possible circular shapes and select the best-matching circular shape as the detected circular landmark for the current cycle. The details of the proposed landmark tracking process are described in the following.

1. Use the four circular landmarks detected in the previous cycle as centers to generate four corresponding tracking windows, each with its side length being five times the diameter of a circular landmark.
2. Find a circular landmark within each tracking window W in an optimal way as described in the following.

- a. Assume that the circle to be detected has the parameters (r_i, x_i, y_i) where r_i is the radius, (x_i, y_i) specifies the coordinates of the circle center, $i = 1, 2, \dots, m$ and m is a pre-selected number of possible candidate circles within the tracking window W .
- b. For the candidate circular shape S_i corresponding to the parameter set (r_i, x_i, y_i) , compute a *fitting measure* F_i as illustrated in Fig. 6.6, where (1) F_i is computed as the sum of the *edge values* of eight “sampling bars” located evenly around S_i ; (2) each sampling bar is composed of n “sampling pixels” with $n/2$ ones within the circle (called *inner sampling pixels*) and the other $n/2$ ones outside the circle (called *outer sampling pixels*) where n is a pre-selected number; and (3) the *edge value* E_i of a sampling bar is computed as the difference between the sum of the gray vales of the inner sampling pixels and the sum of the gray values of the outer sampling pixels.
- c. If the largest fitting measure F_{\max} of all of the m candidate shapes is larger than a pre-selected threshold value, then take the corresponding candidate shape S_{\max} to be the detected circular landmark within the tracking window W ; otherwise, decide that no landmark exists in W .

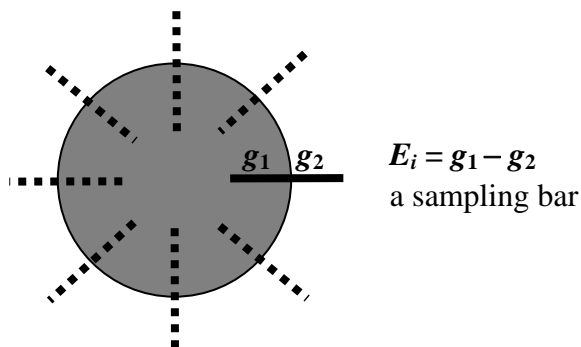
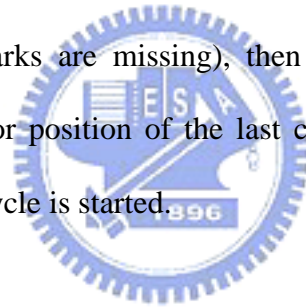


Fig. 6.6 Eight sampling bars at a candidate landmark.

6.2.5 Tracking with missing landmarks

Due to possible large movements of the camera mouse, screen corners might be out of the field of view of the camera, causing disappearance of one or two landmarks. In this case, we compute the displacement vector of a found landmark in the current cycle with respect to its version in the last cycle, and utilize the vector to compute a *predicted position* for each missing landmark, based on the position of the missing landmark in the last cycle. That is, we obtain the position of the missing landmark in the current cycle by adding the vector to the position of the landmark in the last cycle. We then use the predicted positions of the missing landmarks as well as the detected positions of the existing landmarks to compute the cursor position as described previously. Finally, if only one landmark is detected (i.e., if three landmarks are missing), then the detection of the landmarks is regarded to fail, and the cursor position of the last cycle is used as a substitute for the current cycle before the next cycle is started.



6.3 Experimental Results

In this section, we show the results of two of the experiments we have conducted. In the experiments, with a camera mouse we took continuously images of the screens of a notebook PC and desktop PC as two videos, simulating using the mouse for cursor control. The program implementing the proposed method was written in C⁺⁺ on a PC with a Pentium-4 CPU with the speed of 3.0 GHz and the memory of a 1-GB RAM. The sizes of the images are 320×240 pixels. The target objects in the images, the two PC's, have screen resolutions of 1024×768. The average time to process an image with the target object of the notebook PC is 11 mini-seconds, while that to process an image with the target object

of the desktop PC is 14 mini-seconds.

Fig. 6.7 shows six of the image frames of the first video, and Fig. 6.8 shows the results of superimposing the computed cursor positions over the image frames. In the figures, each coordinate pair (xxx , yyy) specifies the relative position of the image center with respect to the upper-left corner of the PC screen. It can be seen from the coordinate data that the computed cursor positions are precise enough for general applications, which means that the proposed approach is basically feasible.



Fig. 6.7 Some snapshots extracted from a video with 412 frames of images.

Fig. 6.9 shows the computed cursor positions of 12 image frames of the second video, from which the robustness of the proposed method in the aspect of tolerating missing corners in the landmark tracking process can be seen. The right side of the PC screen disappeared due the left movement of the mouse camera. But the proposed method still can compute the cursor positions by landmark position prediction until the mouse camera moved back to its normal positions in the frame of no: <745>.

Fig. 6.10 shows the layout of four square-shaped targets for testing the access ability of the proposed camera mouse. Fig. 6.11 shows the images of access results of these targets.

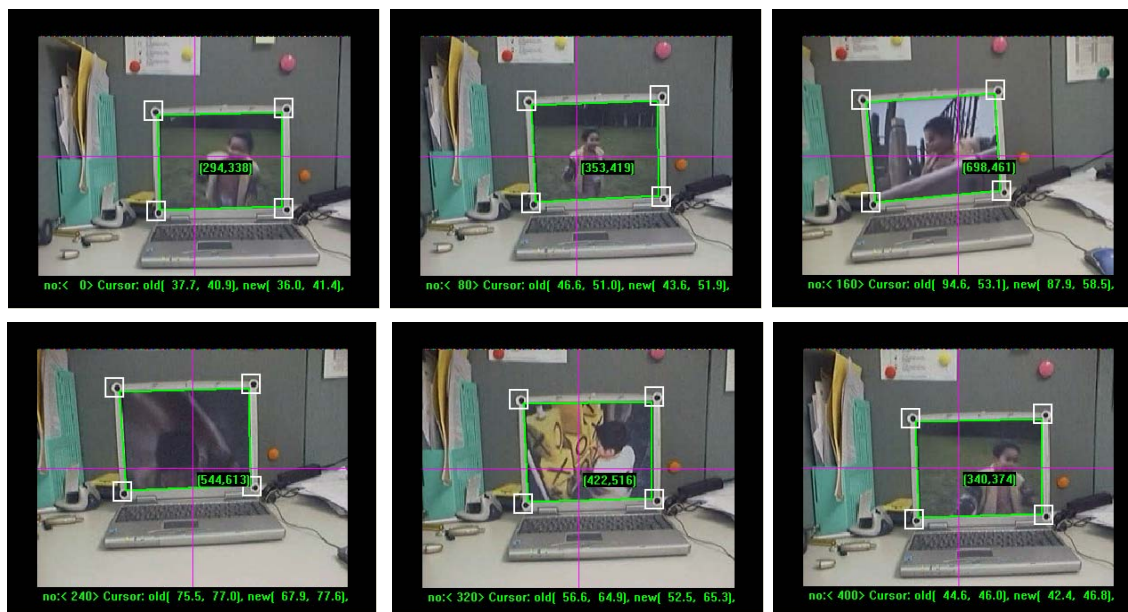


Fig. 6.8 Some snapshots of cursor locations on a notebook PC extracted from a video with 412 frames of images.

6.4 Summary



In this study, we have proposed a vision-based camera mouse for cursor control on computer monitor screens. A hand-held camera, which takes consecutive images of a computer screen, is used as a mouse. The cursor of such a mouse is regarded to be the center of the taken image, whose position with respect to the upper-left corner of the computer monitor screen is computed for each image frame. We attach four landmarks on the corners of the outer frame of the computer monitor and detect the landmarks in each image frame by a deformable template matching technique. We then use them to compute the corresponding monitor screen corner positions for the purpose of computing the cursor position. Perspective transformation of the monitor screen shapes is solved by affine transformation. And missing of landmarks in image frames due to large camera mouse

movement is solved by landmark position prediction using landmark information found in previous cycles. The proposed method does not need tedious camera calibration works. Good experimental results show feasibility of the proposed method. A good application of the proposed mouse camera is to replace the gun used in shooting games on large LCD display screens in order to increase the fun of game playing and to reduce the hardware cost.



Fig. 6.9 Some snapshots of cursor locations on a desktop PC extracted from a video with 880 frames of images.

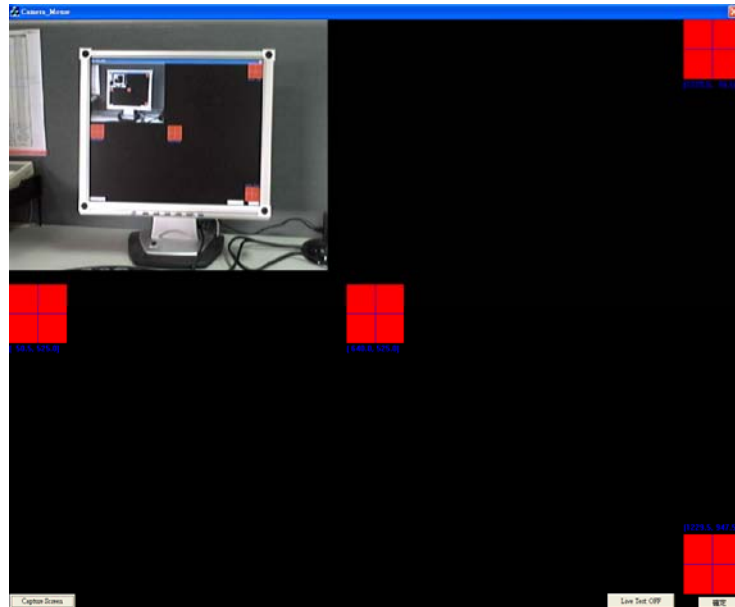


Fig. 6.10 Layout of four square targets.



Fig. 6.11 Images of access results.

Chapter 7

Conclusions and Suggestions for Future Research

7.1 Conclusions

In this study, three new methods are proposed for relative topics of image transformation of the omni-camera and two smart methods are proposed for applications of traditional camera calibration. First, a systematic method has been proposed to derive a set of new and general analytic equations for unwarping images taken from an omni-directional camera with a hypercatadioptric camera. The derivation is made possible by careful investigation on the system configuration and precise calibration of involved system parameters. Second, a new method called “edge-preserving 8-directional two-layered weighting interpolation” has been proposed for interpolating unfilled pixels in a perspective-view or panoramic image resulting from unwarping an omni-image taken by a non-SVP hypercatadioptric camera. Third, a unified approach for unwarping of omni-images into panoramic or perspective-view images has been proposed. The approach does not adopt the conventional technique of calibrating the related parameters of an omni-camera. Instead, it is based on a new concept of pano-mapping table, which is created once forever by a simple learning process for an omni-camera of any kind as a summary of the information conveyed by all the camera parameters. The learning process

takes as input a set of landmark point pairs in the world space and in a given image. With the help of the pano-mapping table, any panoramic or perspective-view image can be created from an input omni-image taken by the omni-camera according to an analytic computation process proposed in this study.

For applications of traditional camera calibration, first, a robust and accurate calibration method for coordinate transformation between display screens and their images has been proposed. The images of three specially designed calibration patterns, which are a white rectangle, a black rectangle, and a dot matrix picture, are captured and processed sequentially to assist achieving robust extraction of relevant features from the calibration pattern images, especially the border features in the image of the display screen. Good performance of the proposed method was demonstrated by comparing the experimental results of the method with those of a well-known calibration method. Second, a camera mouse with vision-based method for computer cursor control using a video camera held in hand in the air has been proposed. The method does not need any calibration procedure. Actually, it is an on-line calibration concept. The proposed method computes the cursor position with the help of four landmarks attached on the corners of the outer frame of the computer monitor. The landmarks and borders of the monitor screen are detected in the first image frame of a captured image sequence by a deformable template matching technique, and then a landmark tracking procedure is conducted to keep track of the landmarks in the subsequent image frames. The cursor position is computed as the coordinates of the image center relative to the upper-left corner of the monitor screen. Perspective distortion of the monitor screen shape is corrected by an affine transformation, and missing of screen corners in the taken image due to large camera movements is solved

by prediction of the corner position using corner information of the last image frame.

In the following, conclusions for each topic in this dissertation study are given individually.

(a) Analytic image unwarping for omni-directional cameras with hyperbolic-shaped mirrors

An approach to systematic calibration and analytic image unwarping for omni-directional non-SVP hypercatadioptric cameras with hyperbolic-shaped mirrors has been proposed. We used the calibrated parameters of the camera to derive precise unwarping equations. The derived equations have been validated to yield the same unwarping results as those yielded by a perfectly designed SVP camera by adjusting the calibrated parameters to fit the SVP constraint. Furthermore, we have shown the advantages of our method over the SVP-constrained method for real cameras by some simulation and experimental results.

(b) Improving quality of unwarped omni-images by a new edge-preserving interpolation technique

We have proposed in this study a method of “edge-preserving 8-directional two-layered weighting interpolation” for processing an unwarped image with irregularly distributed unfilled pixels which is the result of unwarping an omni-image taken from a non-SVP hypercatadioptric camera. In addition to possessing the edge-preserving capability, the method takes into consideration three concepts of weighting for the involved interpolation scheme: (1) inverse-distance weighting; (2) pixel-count weighting; and (3) region-wise weighting. This method has reasonable processing speed and produces good image quality, compared with other conventional methods, as seen from the

experimental results.

(c) Using pano-mapping tables for unwarping of omni-images into panoramic and perspective-view images

A new approach to unwarping of omni-images taken by all kinds of omni-cameras has been proposed. The approach is based on the use of pano-mapping tables proposed in this study, which may be regarded as a summary of the information conveyed by all the parameters of an omni-camera. The pano-mapping table is created once forever for each omni-camera, and can be used to construct panoramic or perspective-view images of any viewpoint in the world space. This is a simple and efficient way to unwarping an omni-image taken by any omni-camera, when some of the physical parameters of the omni-camera cannot be obtained.

(d) A robust and accurate calibration method for coordinate transformation between display screens and their images

A robust and accurate method for calibration and coordinate transformations from image coordinates to display screen positions has been proposed. By using three sequentially displayed calibration patterns, including a white-rectangle shape, a black-rectangle shape, and a dot matrix, relevant landmark points can be extracted accurately and robustly for calibration. Deformable pattern matching is used for creating more accurate geometric models for the display screen shapes in acquired images. The transformation of the coordinates of a point in the image to the coordinates of the display screen is done by using an inverse distance weighting interpolation method. Comparing the results with a conventional calibration method, we showed that the proposed method has the merits of more accuracy, robustness, and simplicity.

(e) A Camera Mouse for Computer Cursor Control

A vision-based camera mouse for cursor control on computer monitor screen has been proposed. A hand-held camera, which takes consecutive images of a computer screen, is used as a mouse. The cursor of such a mouse is regarded to be the center of the taken image, whose position with respect to the upper-left corner of the computer monitor screen is computed for each image frame. We attach four landmarks on the corners of the outer frame of the computer monitor and detect the landmarks in each image frame by a deformable template matching technique. Perspective transformation of the monitor screen shapes is solved by affine transformation. And missing of landmarks in image frames due to large camera mouse movement is solved by landmark position prediction using landmark information found in previous cycles. The proposed method does not need tedious camera calibration works. Good experimental results show feasibility of the proposed method. A good application of the proposed mouse camera is to replace the gun used in shooting games on large LCD display screens in order to increase the fun of game playing and to reduce the hardware cost.

From the application view of image unwarping for omni-cameras, if the omni-camera is used in the indoor environment with sufficient easily-identified corresponding point pairs between omni-images and the world space, we can totally use Method (c) to replace (a) with extra benefits and no need to do image interpolation. But if the non-SVP hypercataoptric camera is used in the outdoor environment, Method (c) is not easy to use because of difficulty to identify corresponding point pairs; instead, Methods (a) and (b) should be used to handle these situations.

7.2 Suggestions for Future Research

The research of image unwarping for omni-cameras is rising and flourishing in recent years. Many approaches and theories have been proposed in this field. Most of the previous works focus on the study of the omni-camera itself. In the future, studies about cooperation of multiple omni-cameras or hetero-type cameras in visual surveillance applications will play important roles. This study is just a beginner in the field. We hope to have chances to do furthermore studies. Some attractive and interesting topics that are related to this study are worth further researches. They are collected and described in the following.

- (1) The study of improving geometry accuracy of unwarping image with pano-mapping table method.

In Chapter 4, we assume that the mirror and lens in the omni-camera are perfect and rotation-invariant. So, a set of calibrated parameters for the radial stretching function is enough to describe the mapping between images and the world space at every azimuth angle. If the geometry accuracy is a serious concern, we can do calibration at each azimuth angle to get different parameter sets under different angles. This idea may be validated in the future work.

- (2) The study of the hybrid visual surveillance system, which includes multiple omni-cameras, PTZ (pan, tilt, zoom) cameras, and perspective cameras.

A visual surveillance system with a single omni-camera may be sufficient for low-end applications to detect abnormal intruding objects. But for high-end applications, for example, identification of what kind of intruding object, we need clearly captured perspective images, and the unwarping one from omni-images is too

coarse for this case. If a PTZ camera is incorporated into the surveillance system, a fine perspective image can be captured by this PTZ camera under the help of object locating capability of an omni-camera, which can satisfy this requirement.

(3) The study of multi-pointers/cameras pointing system for entertainment game.

We can extend current one-pointer and one-camera pointing system to multi-pointer/one-camera or one-pointer/multi-camera or even multi-pointer/multi-camera system to increase the enjoyment and versatility for applications in entertainment.



References

- [1] J. Salvi, X. Armangué, and J. Batle, “A Comparative Review of Camera Calibrating Methods with Accuracy Evaluation,” *Pattern Recognition*, Vol. 35, No. 7, pp. 1617-1635, July 2002.
- [2] S. K. Nayar, “Omnidirectional Vision,” *Proceedings of Eighth International Symposium on Robotics Research (ISRR)*, October 1997, Shonan, Japan.
- [3] J. Kannala and S. Brandt, “A Generic Camera Calibration Method for Fish-Eye Lenses,” *Proceedings of the 17th International Conference on Pattern Recognition*, Vol. 1, pp. 10-13, August 2004; Cambridge, U.K.
- [4] S. Shah and J. K. Aggarwal, “Intrinsic Parameter Calibration Procedure for A (High-Distortion) Fish-Eye Lens Camera with Distortion Model and Accuracy Estimation,” *Pattern Recognition*, Vol. 29, No. 11, pp. 1775-1788, November 1996
- [5] S. K. Nayar, “Catadioptric Omni-directional Camera,” *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 482-488, June 1997, San-Juan, Puerto Rico.
- [6] S. Baker and S. K. Nayar, “A Theory of Single-Viewpoint Catadioptric Image Formation,” *International Journal of Computer Vision*, Vol. 35, No. 2, pp. 175-196, November 1999.
- [7] K. Yamazawa, Y. Yasushi, and M. Yachida, “Omnidirectional Imaging with Hyperboloidal Projection,” *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1029-1034, July 1993, Yokohama, Japan.

- [8] K. Yamazawa, Y. Yagi, and M. Yachida, "New real-time omnidirectional image sensor with hyperboloidal mirror," *Proceedings of 8th Scandinavian Conference on Image Analysis*, Vol. 2, pp. 1381-1387, May 1993.
- [9] R. Swaminathan, M. D. Grossberg, and S. K. Nayar, "Caustics of Catadioptric Cameras," *Proceedings of 8th IEEE International Conference on Computer Vision*, Vol. 2, pp. 2-9, July 2001, Vancouver, BC, Canada.
- [10] X. H. Ying and Z. Y. Hu, "Catadioptric Camera Calibration Using Geometric Invariants," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 26, No. 10, pp. 1260-1271, October 2004.
- [11] T. Mashita, Y. Iwai, and M. Yachida, "Calibration Method for Misaligned Catadioptric Camera," *IEICE Trans. on Information and Systems*, Vol. E89-D, No. 7, pp. 1984-1993, July 2006.
- [12] S. W. Jeng, and W. H. Tsai, "Precise Image Unwarping of Omnidirectional Cameras with Hyperbolic-Shaped Mirrors," *Proceedings of 16th IPPR Conference on Computer Vision, Graphics and Image Processing*, pp. 414-422, August 2003, Kinmen, R. O. C ..
- [13] R. Y. Tsai, "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 364-374, June 1986, Miami Beach, Florida.
- [14] D. R. Olsen and T. Nielsen, "Laser Pointer Interaction," *Proceedings Of 2001 Conference on Human Factors in Computing Systems*, pp.17-22, April 2001, New York, USA.
- [15] C. Kirstein and H. Müller, "Interaction with A Projection Screen Using A Ccamera-Tracked Laser Pointer," *Proceedings of International Conference on Multimedia Modeling*, pp. 191-192, October 1998.

- [16] C. Geyer and K. Daniilidis, "Paracatadioptric Camera Calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24, No. 5, pp. 687-695, May 2000.
- [17] C. Geyer and K. Daniilidis, "Catadioptric Camera Calibration," *Proceedings of 7th IEEE International Conference on Computer Vision*, Vol. 1, pp. 398-404, September 1999, Corfu, Greece.
- [18] S. B. Kang, "Catadioptric Self-calibration," *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 1, pp. 201-207, June 2000, Hilton Head, South Carolina.
- [19] D. G. Aliaga, "Accurate Catadioptric Calibration for Real-time Pose Estimation in Room-size Environments," *Proceedings of 8th IEEE International Conference on Computer Vision*, Vol. 1, pp. 127-134, July 2001, Vancouver, BC, Canada.
- [20] J. Fabrizio, J. P. Tarel, and R. Benosman, "Calibration of Panoramic Catadioptric Sensors Made Easier," *Proceedings of 3rd Workshop on Omni-directional Vision*, pp. 45-52, June 2002, Copenhagen, Denmark.
- [21] Y. Onoe, N. Yokoya, K. Yamazawa, and H. Takemura, "Visual Surveillance and Monitoring System Using an Omni-directional Video Camera," *Proceedings of 14th International Conference on Pattern Recognition*, Vol. 1, pp. 588-592, August 1998, Brisbane, Australia.
- [22] S. W. Jeng and W. H. Tsai, "Construction of Perspective and Panoramic Images from Omni-images taken from Hypercatadioptric Cameras for Visual Surveillance," *Proceedings of 2004 IEEE International Conference on Networking, Sensing, and Control*, pp. 204-209, March 2004, Taipei, Taiwan.
- [23] V. N. Peri and S. K. Nayar, "Generation of Perspective and Panoramic Video from Omnidirectional Video," *Proceedings of DARPA Image Understanding*

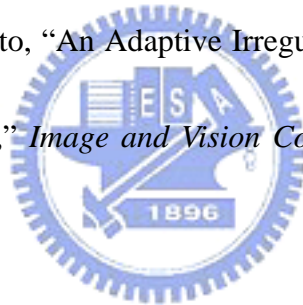
Workshop, pp. 243-246, May 1997, New Orleans, LA, USA.

[24] A. K. Jain, *Fundamentals of Digital Image Processing*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, U. S. A., 1989.

[25] G. H. Atwood and W. A. Davis, "Image Expansion Using Interpolation & Heuristic Edge Following," *Proceedings of 3rd International Conference on Image Processing and its Applications*, pp. 664-668, July 1989, London, U. K.

[26] X. Li and M. T. Orchard, "New Edge-Directed Interpolation," *IEEE Transactions on Image Processing*, Vol. 10, No. 10, pp. 1521-1527, October 2001.

[27] G. Ramponi and S. Carrato, "An Adaptive Irregular Sampling Algorithm and Its Application to Image Coding," *Image and Vision Computing*, Vol. 19, pp. 451-460, May 2001.



[28] R. Benosman, and S. B. Kang, editors. *Panoramic Vision: Sensor, Theory and Applications*. Monographs in Computer Science. Springer-Verlag, New York (2001).

[29] S. Morita, K. Yamazawa and N. Yokoya, "Networked Video Surveillance Using Multiple Omnidirectional Cameras," *Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 1245-1250, July 2003, Kobe, Japan.

[30] L. Tang and S. Yuta, "Indoor Navigation for Mobile Robots Using Memorized Omni-directional Images and Robot's Motion," *Proceedings of IEEE/RSJ*

International Conference on Intelligent Robots and System, Vol. 1, pp.269-274, September 2002.

[31] N. Winter, J. Gaspar, G. Lacey, and J. Santos-Victor, "Omnidirectional Vision for Robot Navigation," *Proceedings of IEEE Workshop on Omnidirectional Vision, OMNIVIS*, pp.21-28, June 2000, South Carolina, USA.

[32] G. Scotti, L. Marcenaro, C. Coelho, F. Selvaggi and C. S. Regazzoni, "Dual Camera Intelligent Sensor for High Definition 360 Degrees Surveillance," *IEE Proceeding on Vision Image Signal Processing*, Vol. 152, No. 2, pp. 250-257, April 2005.

[33] T. N. Mundhenk, M. J. Rivett, X. Liao, and E. L. Hall, "Techniques for Fisheye Lens Calibration using a Minimal Number of Measurements," *Proceedings of the SPIE Intelligent Robotics and Computer Vision Conference XIX*, November 2000, Boston, MA.

[34] D. Strelow, J. Mishler, D. Koes, and S. Singh, "Precise Omnidirectional Camera Calibration," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2001)* , Vol. 1, pp. 689-694, December 2001, Kauai, HI.

[35] D. Scaramuzza, A. Martinelli, and R. Siegwart, "A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion," *Proceedings of the Fourth IEEE International Conference on Computer Vision System*,

January 2006.

[36] Burden, L. Richard, and J. D. Faires, *Numerical Analysis*, 7th ed. Belmont, CA:

Brooks Cole, 2000. ISBN: 0534382169.

[37] Zelniker, Emanuel and Clarkson, “Maximum-Likelihood Estimation of Circle

Parameters via Convolution,” *IEEE Transactions on Image Processing*, Vol. 15, No. 4,

pp.865-876, April 2006.

[38] D. J. Kerbyson and T. J. Atherton, “Circle Detection Using Hough Transform

Filters,” *Image Processing Application*, pp.370-374, July1995, London, U.K.

[39] X. Chen and J. Tang, “Towards Monitoring Human Activities Using an

Omnidirectional Camera,” *International Conference on Multimodal Interfaces*, pp.

423–428, October 2002, Pittsburgh, PA., USA.



[40] K. Hansen and C. Stillman, “Computer Presentation System and Method with

Optical Tracking of Wireless Pointer,” US Patent, No. US6275214.

[41] S. H. Lin, “Interactive Display Presentation System,” US Patent, No.

US6346933.

[42] L. S. Smoot, “Light-Pen System for Projected Images,” US Patent, No.

US5115230.

[43] K. L. Tam, Lau, R. W. H. Lau, and C. W. Ngo, “Deformable Geometry Model

Matching by Topological and Geometric Signatures,” *Proceedings of 17th*

International Conference on Pattern Recognition, Vol. 3, pp. 910-913, August 2004.

[44] A. L. Yuille, P. W. Hallinan, and D. S. Cohen, "Feature Extraction from Faces Using Deformable Templates," *International Journal of Computer Vision*, Vol. 8, No. 2, pp. 99-111, August 1992.

[45] R. Sukthankar, R. G. Stockton, and M. D. Mullin, "Smarter Presentations: Exploiting Homography in Camera-Projector Systems," *Proceedings of International Conference on Computer Vision*, Vol. 1, pp. 247-253, July 2001, Vancouver, Canada.

[46] MacKenzie, I.S. and Jusoh, S., "An Evaluation of Two Input Devices for Remote Pointing," *Proceedings of the Eighth IFIP Working Conference on Engineering for Human Computer Interaction*, pp. 235-249, May 2001, Toronto, Canada.

[47] Gyration, Inc., Saratoga, California, <http://www.gyration.com/en-US>.

[48] J. Yang, E.S. Choi, W. Chang, W.C. Bang, S.J. Cho, J.K. Oh, J.K. Cho, and D.Y. Kim, "A Novel Hand Gesture Input Devices Based on Inertial Sensing Technique," *Proceedings of the 30th Annual Conference of the IEEE Industrial Electronics Society*, pp. 2786-2791, November 2004, Busan, Korea.

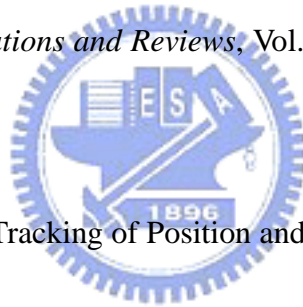
[49] P. Nesi and A. D. Bimbo, "A Vision-Based 3-D Mouse," *International Journal of Human-Computer Studies*, Vol. 44, No. 1, pp. 73-91, January 1996.

[50] D. F. Dementhon and L. S. Davis, "Model-Based Object Pose in 25 Lines of Code," *International Journal of Computer Vision*, Vol. 15, pp. 123-141, June 1995.

[51] Z. F. Yang and W. H. Tsai, "An Inside-Out Vision-Based 3-D Mouse," *Proceedings of 1998 Conference on Computer Vision, Graphics, and Image Processing*, August 1998, Taipei, Taiwan.

[52] S. Li, W. Hsu, and H. Pung, "A Real-time Monocular Vision-Based 3D Mouse System," *Proceedings of International Conference on Computer Analysis of Images and Patterns*, pp. 448-455, September 1997, Kiel, Germany.

[53] W. Hu, T. Tan, L. Wang, and S. Maybank, "A Survey on Visual Surveillance of Object Motion and Behaviors," *IEEE Transactions on System, Man, and Cybernetics – Part C: Applications and Reviews*, Vol. 34, No. 3, pp. 334-352, August 2004.



[54] N. Peterfreund, "Robust Tracking of Position and Velocity with Kalman Snakes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 21, No. 6, pp. 564-569, June 1999.

[55] M. Isard and A. Blake, "CONDENSATION – Conditional Density Propagation for Visual Tracking," *International Journal of Computer Vision*, Vol. 29, No. 1, pp. 5-28, August 1998.

[56] A. K. Jain, Y. Zhong, and S. Lakshmanan, "Object Matching Using Deformable Templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 3, pp. 267-278, March 1996.

[57] M. D. Grossberg, and S. K. Nayar, “The Raxel Imaging Model and Ray-Based Calibration,” *International Journal of Computer Vision*, Vol. 61, No. 2, pp.119-137, February 2005.

[58] M. Betke, J. Gips, and P. Fleming, “The camera mouse: Visual tracking of body features to provide computer access for people with severe disabilities,” *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, Vol. 10, No.1, pp. 1-10, March 2002.



Publication List

Journal Papers

1. S. W. Jeng and W. H. Tsai , “Using Pano-Mapping Tables for Unwarping of Omni-Images into Panoramic and Perspective-View Images,” *IET Image Processing*, Vol.1, No. 2 , pp. 149-155, June 2007.
2. S. W. Jeng and W. H. Tsai , “Improving Quality of Unwarped Omni-Images with Irregularly-Distributed Unfilled Pixels by A New Edge-Preserving Interpolation Technique,” accepted and to appear in *Pattern Recognition Letters*.
3. S. W. Jeng and W. H. Tsai , “A Robust and Accurate Calibration Method by Computer Vision Techniques for Coordinate Transformation Between Display Screens and Their Images,” accepted as short paper and to appear in *Journal of Information Science and Engineering*.
4. S. W. Jeng and W. H. Tsai , “Analytic Image Unwarping by A Systematic Calibration Method for Omni-Directional Cameras with Hyperbolic-Shaped Mirrors,” submitted to *Image and Vision Computing*. (revised)
5. S. W. Jeng and W. H. Tsai ,” A Camera Mouse for Computer Cursor Control in the Air by Computer Vision Techniques,” submitted to *The Journal of the Chinese Institute of Engineers*.

Conference Papers

1. S. W. Jeng and W. H. Tsai, “A Unified Approach to Unwarping of Omni-Images into Panoramic and Perspective-View Images Using Pano-Mapping Tables,” *Proceedings of 2006 IEEE International Conference on System, Man and Cybernetics*, pp. 2401-2406, October 2006, Taipei, Taiwan.
2. S. W. Jeng and W. H. Tsai, “Construction of Perspective and Panoramic Images from Omni-images taken from Hypercatadioptric Cameras for Visual Surveillance,” *Proceedings of 2004 IEEE International Conference on*

Networking, Sensing, and Control, pp. 204-209, March 2004, Taipei, Taiwan.

3. S. W. Jeng and W. H. Tsai, "Precise Image Unwarping of Omnidirectional Cameras with Hyperbolic-Shaped Mirrors", *CVGIP 2003 conference proceedings*, pp. 414-422, August 2003, Kinmen, R. O. C.
4. S. W. Jeng and W. H. Tsai, "Creating a Personalized 3D Face Model from a Frontal Face Color Image Using an Existing 3D Face Model", *CVGIP 2001 conference proceedings*, August 2001, Howard beach resort Kenting, Kenting, Taiwan.



Vita

Sheng-Wen Jeng was born in Tainan, Taiwan, R.O.C., in 1961. He received the B.S. degree in electronics engineering from National Chiao-Tung University in 1983, the M. S. degree in Electro-optical Engineering from National Chiao-Tung University in 1985.

Mr. Jeng worked as a researcher in the Industrial Technology Research Institute since October 1985 till now. He is also a Ph. D. student in the Department of Computer Science at National Chiao Tung University since 1997. His current research interests include computer vision, pattern recognition, MEMS, and their applications.

