Estimation and Application of Vision-based Camera Tracking Using Planar Structures in the Real World

March 2009

Yuko Uematsu

Preface

Recent development of digital technologies has significantly changed the environment for dealing with digital media contents. People can see such virtual objects not only in a normal screen but also in 3D display styles. Augmented Reality is one of the new presentation styles of digital media contents. 3D virtual objects generated with CG are superimposed onto the video images. People watch the scene through handheld displays or computerized glasses such as a camera phone, tablet PC, HMD and so on. Even if the camera which is attached to the display device is moving, the virtual objects are registered according to the camera motion to be seen as a part of the real objects. Therefore camera tracking is very important issue for AR.

This thesis presents a camera tracking method for AR registration and real-time AR applications. A single moving camera captures the real world where multiple planar structures exist. By using the appearance of the multiple planes in the images, camera position and pose are computed frame by frame. This registration algorithm is also extended to an on-line AR system which utilizes planar markers to be performed in real-time. Only by distributing the markers in the real world, AR can be performed not only on the tabletop but also in a large space; a human-sized object can be overlaid on the room as a virtual pet.

For using multiple planes for the camera tracking, 3D arrangement of the planes must be known. In the large space or complicated planar structures, however, it is almost impossible to measure the 3D arrangement of the planes manually. In contrast, this algorithm can automatically estimate the geometrical relationship of the planes by 3D Projective Space which is defined by projective reconstruction of two reference images. Since just corresponding points between the reference images are required for projective reconstruction, our algorithm

can use multiple planes without the geometrical relationship information or manual measurement task of the planes.

The on-line AR system is applied to the field of entertainment and tabletop AR applications are proposed. “AR Baseball Presentation System” is an observation system of a virtual baseball game, which replays the game on a tabletop field model by an input baseball game history. User can watch the game from favorite viewpoint with moving around the real world. In “Interactive AR Bowling System”, user rolls a real ball to knock down virtual pins on the lane model. Touching and rolling the real ball provide a sort of tangible feeling in this system. AR applications allow easy change of user’s viewpoint by just moving around the real world. Real-time interaction between the real and virtual world is also available. Therefore designing applications as AR style is effective to enhance the entertainment quality. It is evaluated by the user study in this thesis.

In addition, pattern-based approach with Particle Filter is proposed to improve the accuracy of tracking based on planar markers. Feature tracking of points or lines is necessary for the camera tracking, however, it is not so easy to track the same points or lines in real-time, continuously and accurately. Instead of feature tracking, our approach keeps some hypotheses of the camera parameters and compares the actual captured image with generated patterns by hypotheses. By searching the best matching, the best camera parameters can be obtained. This approach is also applied to the outdoor scene to implement a prototype system of AR + GIS. By combining top-view image such as a satellite image or an aerial image and an overview video image captured by a user’s camera phone, the map information in GIS database is overlaid onto the user’s display. It will be utilized for the navigation for tourists, geographical simulation, and so on.

Therefore, this thesis presents the new algorithm of basic theory of camera tracking and the application possibilities of AR. Its effectiveness is also demonstrated. Since AR is very future-oriented technique, I expect my work of this thesis will contribute to the field of AR.

Acknowledgments

I would like to sincerely thank my supervisor Professor Hideo Saito for his support throughout my graduate studies. He has always encouraged me since my undergraduate year and provided me with a lot of great experiences. His advice, suggestion and leading have been precious resources for me. Most of all, his genial, positive, friendly personality has greatly supported me. I am very happy to have been in his laboratory and studied with him for a long time. I would like to greatly appreciate his wonderful supervision.

I am grateful to Professor Myriam Servieres and Professor Guillaume Moreau at the "Institut de Recherche Sciences et Techniques de la Ville" (CERMA), Ecole Centrale de Nantes (ECN). Fortunately, I have got the opportunity to work with them and achieved the collaboration research about GIS. I gained a lot of valuable advice for my study and this thesis from them. I hope we can continue our excellent relationship and collaboration researches.

I would like to thank the members of my thesis committee: Professor Ken-ichi Okada, Professor Masafumi Hagiwara, and Professor Masaaki Ikehara for their comments and suggestions on this thesis.

My work for this thesis is supported in part by a Grant-in-Aid for the 21st century Center of Excellence (COE) for Optical and Electronic Device Technology for Access Network from the Ministry of Education, Culture, Sports, Science, and Technology in Japan, and a Grant-in-Aid for the Global Center of Excellence (GCOE) for high-Level Global Cooperation for Leading-Edge Platform on Access Spaces from the Ministry of Education, Culture, Sport, Science, and Technology in Japan. Japan Society for the Promotion of Science Research Fellowship for Young Scientists also supports my research.

Additionally, I would like to express my gratitude to all members of Hideo Saito laboratory

and Shinji Ozawa laboratory at Keio University. Especially, I have been able to advance my research with pleasure because of the 62nd members. They have been great colleagues and friends for me.

Finally, my grateful appreciation is extended to my family and dearest friends of long standing for supporting my study and everyday life all this time.

March 2009

Yuko Uematsu

Table of Contents

Chapter 1 : Introduction 1

1.1 Background and Motivation 2

1.2 Thesis Outline 5

Chapter 2 : Related Work 7

2.1 Geometrical Registration for Augmented Reality 8

2.1.1 Sensor-based Camera Tracking 9

2.1.2 Vision-based Camera Tracking 9

2.2 Plane-based Camera Tracking 11

2.3 AR Applications for Entertainment 14

Chapter 3 : Camera Tracking Method Using Multiple Planes 15

3.1 Overview of Registration Method 16

3.1.1 Definition of Coordinate Systems 17

3.1.2 Flow of Our Method 19

3.2 Relationship of Coordinate Systems 21

3.3 Assigning 3D Coordinate Systems on Multiple Planes 22

3.4 Definition of Projective Space 23

3.5 Computation of $\boldsymbol{T_i^{WP}}$ (Planes→Projective Space) 25

3.5.1 How to get corresponding points 25

3.5.2 How to compute the matrix 25

3.6 Computation of $\boldsymbol{P_i^{WI}}$ (plane → image) 27

3.6.1 Computation of $\boldsymbol{H_i}$ (X-Y plane $\leftrightarrow$ x-y image plane) 27
3.6.2 Relationship between $\boldsymbol{P_i^{WI}}$ and $\boldsymbol{H_i}$ 28
3.6.3 Estimation of intrinsic parameters 28
3.6.4 Estimation of extrinsic parameters 29
3.7 Computation and integration of $\boldsymbol{P_i^{PI}}$
(Projective Space $\rightarrow$ image) 30
3.8 Rendering of Virtual Objects 32
3.9 Experimental Results 33
3.10 Discussion 37
3.10.1 Accuracy comparison by number of planes used for computation . . . 37
3.10.2 Accuracy comparison with related work 39

Chapter 4 : On-line AR System with Planar Pattern Markers 41
4.1 Outline of On-line AR System 42
4.2 User's Operation and Flow of System 43
4.2.1 First Stage: Constructing Projective Space 44
4.2.2 Second Stage: Generating Augmented Reality View 45
4.3 Automatic Selection of Reference Images 46
4.4 Effective Marker Selection for Integration 49
4.5 Experimental Results 51
4.6 Effectiveness of Distributing Markers 56

Chapter 5 : AR Applications based on On-line AR System 58
5.1 Overview of Applications 59
5.2 AR Baseball Presentation System 60
5.2.1 Processing Outline 61
5.2.2 Input Scorebook Datafile 62
5.2.3 Actions of Offensive Players 63
5.2.4 Actions of Fielders and Ball 64
5.2.5 Demonstration 64
5.3 Interactive AR Bowling System 70
5.3.1 Processing Outline 72

5.3.2 Marker Tracking . . . 73
5.3.3 Lane Tracking . . . 74
5.3.4 Ball Tracking . . . 74
5.3.5 Transformation to Top-View Image . . . 75
5.3.6 Collision Detection between Ball and Pins . . . 76
5.3.7 Knocking Down and Overlay Virtual Pins . . . 77
5.3.8 Demonstrations . . . 78
5.4 User Study . . . 81
5.4.1 Baseball System . . . 81
5.4.2 Bowling System . . . 84

Chapter 6 : Pattern-based Tracking with Particle Filter for improving plane-based tracking 87
6.1 Motivation . . . 88
6.1.1 Related work using Particle Filter . . . 89
6.1.2 Camera tracking with Particle Filter . . . 90
6.2 Overview of Pattern-based Tracking Method . . . 92
6.3 Initialization Step . . . 94
6.4 Tracking Step . . . 95
6.5 Computing Weight Values . . . 96
6.5.1 Weight Value based on Edge . . . 96
6.5.2 Weight Value based on Pattern . . . 98
6.6 Experimental Results . . . 100
6.7 AR Mapping System of GIS Information . . . 105
6.7.1 Computation of Weight Value . . . 106
6.7.2 Experimental Results . . . 108

Chapter 7 : Conclusions 110
7.1 Summary . . . 111
7.2 Future Work . . . 113

Chapter 8 : Appendix 114
Stereo Algorithm in Assigning 3D Coordinate Systems 115

List of Figures

1.1 Relationship of chapters in this thesis. 5

2.1 Overlaying virtual object onto the real world by AR: The object scene is captured by a moving camera. According to the camera motion, a 3D virtual object is overlaid onto the captured images. A user can see that the virtual object really exists in the real world. 8
2.2 Natural features in images: Stronger geometrical constraints can be imposed on the feature points by lines and planes. 9
2.3 Example of failure using only 1 plane: If a part of the marker is not visible in the image, registration can not be continued. 11
2.4 Multiple marker-based registration of a virtual object: By switching a marker used for computing camera parameters, registration can be continued. For switching markers, the geometrical relationship of the markers (distance or direction) is required. 12

3.1 Projection matrix from the real world to input images: Projection matrix represents camera position and pose with respect to the plane. When using multiple planes, a projection matrix is computed with respect to each plane. . . . 16
3.2 Coordinate systems defined in our method: (X_i-Y_i-Z_i) is defined on each plane by clicking 4 points on each plane. (x-y) is corresponding to the camera coordinate system. (P-Q-R) is defined by two captured images. 18
3.3 Overview of our registration method. 19

3.4 Assigning a 3D coordinate system on each plane: (left column) 4 points are manually clicked on each plane in the reference images. (right 3 columns) 3D coordinate system is defined on each plane so that each plane defines X-Y plane and Z axis becomes perpendicular to the plane. The wire cubes on the planes represent the directions of 3D axes. 22
3.5 3D Projective Space: (P-Q-R) is computed by projective reconstruction of two reference images. 24
3.6 Homography between 3D plane and current frame: By using homography between adjacent frames, homography between 3D plane and current frame is computed. 27
3.7 Rendering of virtual object: By using the transformation matrix of the base plane $\boldsymbol{T}_{\boldsymbol{base}}^{\boldsymbol{WP}}$ and the integrated projection matrix $\boldsymbol{P}^{\boldsymbol{PI}}$, the virtual object is transformed to the image coordinate system (x-y). 32
3.8 Experiment 1 33
3.9 Experiment 2 34
3.10 Overlaid image sequence of a virtual object.
Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z). 35
3.11 Overlaid image sequence of a virtual object.
Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z). 36
3.12 Input images generated by OpenGL: 8 planes are distributed at arbitrary positions and poses. Orange plane is utilized as used single plane. 37
3.13 Comparison of x-y coordinates accuracy with theoretical value. Ground truth, related work, and proposed method. 37
3.14 Result images with changing the number of planes: Using 3 planes achieves best results than using a single plane and two planes. 38
3.15 Input images generated by OpenGL: 3 planes are perpendicular to each other. 39
3.16 Comparison of x-y coordinates accuracy with theoretical value. Ground truth, related work, and proposed method. 39

4.1 On-line AR System: The system consists of a PC and a web-camera. Multiple markers are freely distributed in the real world. 42

4.2 Flow of 1st stage. 44
4.3 Flow of 2nd stage. 45
4.4 Flow of automatic selection of reference images. 46
4.5 Selection of reference images. By comparing difference of the projected cubes $\boldsymbol{x}$ and $\boldsymbol{x'}$, the pair whose difference is smaller than threshold th is selected. (th=3 pixel). 48
4.6 Example scene of failure in definition of Z axis. These 3 frames are captured from similar viewpoints. Even though the marker is visible in frame 2, Z axis of lower-right marker is defined with different direction from previous and next frames. 49
4.7 Object scene: 4 markers are distributed in the real world. 51
4.8 Result images of overlaying static virtual object 53
4.9 Result images of overlaying moving virtual object 54
4.10 Result images of overlaying moving human-sized virtual object 55
4.11 Examples of failure in marker detection. Some of the markers can not be detected even though the markers are visible in the images. On the other hand, the other markers which are different directions with respect to the camera are successfully detected. 56

5.1 Handheld monitor and web-camera 59
5.2 AR Baseball Presentation System 60
5.3 Overview of AR Baseball Presentation System. 61
5.4 Scorebook Data File (SDF). 62
5.5 Actions of the offensive players. 63
5.6 Scorebook Data File of the fielders and the ball. 64
5.7 Experimental environment of AR Baseball Presentation System. 65
5.8 Example of play: 4th batter of team RED sends a hit to left with the bases loaded and scores a goal. 66
5.9 Closeup views of the same scenes as Fig. **??** seen from different viewpoints. . 67
5.10 Most of the markers which face to the same directions as the tabletop cannot be detected, even though they are captured in the image. The marker which faces to different direction is detected successfully. 68
5.11 Interactive AR Bowling System 70

5.12 Overview of Interactive AR Bowling System. 72
5.13 3D coordinate system defined on the real lane model. Virtual pins are overlaid in this coordinate system. 73
5.14 Lane detection by Hough Transform and markers. 74
5.15 Ball detection. 75
5.16 Transform the ball's position to top-view image. 75
5.17 Collision detection by trajectory of the ball. 76
5.18 Interpolated trajectory of the ball. 76
5.19 Direction of knocking down. 77
5.20 Direction of chain knocking down. 77
5.21 Experimental environment of Interactive AR Bowling System 78
5.22 Trajectory of ball and detected lane and ball. 79
5.23 Resulting images on which virtual pins are overlaid. 80
5.24 Prepared baseball system 81
5.25 Average time the examinees took to change their viewpoints to specified viewpoints. 82
5.26 Answer of questionnaire about changing viewpoints. 83
5.27 Prepared bowling system 84
5.28 Answer of questionnaire about the reality of playing bowling. 85

6.1 Example images on tracking 3D coordinate system: Depending on the camera angle, 3D coordinate system on the marker can not be tracked stably. . . . 88
6.2 Overview of our tracking method. 92
6.3 Points for computing weight values 94
6.4 Computing weight value based on edge. 97
6.5 Distance between projected points and edges. 97
6.6 Compting weight value based on pattern. 98
6.7 Regions around projected points and feature points. 99
6.8 Resulting images of computing weight values for all particles. i: index of particles $(0 \leq i \leq 299)$; π: weight value $(0 \leq \pi \leq 1)$. 100
6.9 Estimation result of parameters from the images whose viewpoint is perpendicular to the marker plane. 101

6.10 Resulting images of overlaying cubes by ARToolkit and our method. Green cube: by ARToolkit; Red cube: by our method. 102
6.11 Result of pattern-based weight value 102
6.12 Estimation result of rotation parameters from the images captured from a slanted viewpoint. 103
6.13 Resulting images of overlaying cubes by ARToolkit and our method. Green cube: by ARToolkit; Red cube: by our method. 104
6.14 GIS database 105
6.15 Images utilized in this system 106
6.16 Computation of pattern-based weight value. 107
6.17 Input images and overlaid map information. 108
6.18 Overlaid image sequence of map information. 109
6.19 Resulting frames that fail in registration of map information. 109

Chapter 1 :

Introduction

1.1 Background and Motivation

In the past years, it has become easy to generate digital media contents with Computer Graphics (CG). Such generated contents have been utilized for various aims such as computer games, television broadcasting, motion pictures, and so on. The next challenge is how to present the generated digital media contents.

Augmented Reality (AR) / Mixed Reality (MR) is one of the new presentation styles of digital media contents [1, 2, 3]. Digital media contents are superimposed onto images or video sequences which are captured with a moving camera. In contrast with typical presentation styles using Virtual Reality (VR), viewers can see both the digital media contents and the real scene. AR can create physically impossible scenes and enhance the visual information in the real scene by inserting texts or graphics.

Recently, we can see a lot of examples of augmentation in television broadcasting. Especially in sport coverage, real-time augmentation has been used for enhancing the event [4, 5, 6, 7]. As popular examples, virtual objects such as virtual first-down line in a football game and virtual offside line representing the last defender line in soccer are overlaid onto the ground [8]. Viewers can see the lines through the display of television. Such visual information can help viewers to understand the game easily and make them excited. Virtual record in track and field races or swimming and putting distance from the cup in golf are also inserted in the scene to add excitement to the race.

Some other applications have been also developed for sport broadcasting. "Virtual Confrontation System" has been introduced for Nagano Olympic held in 1998 [5, 9]. In speed skating, an image of a skater who has already skated at the different race is superimposed onto the next lane to the skater who is now skating. Therefore viewers can see that two skaters are racing at the same time, although they are actually skating in different times. "Simul Cam" also superimposes one athletic movement over another for precise comparisons by applying to car racing, ski jump, bobsledding, etc. [10]. These systems can contrast and compare the position, style and trajectory of competitors. In "Stro Motion", an athletic movement is unfold in time and space by compounding video images into a frame-by-frame sequence, because the concept of this system is based on stroboscoping [10]. Therefore it reveals the evolution of an athlete's movement, technique, execution and tactics over space and time in an immediate, intuitive fashion. It is applied to gymnastics, figure skating, snow boarding,

etc. “FoxTrax system” [6] highlights the location of a hard-to-see hockey puck as it moves rapidly across the ice. This system tracks the puck by lots of camera attached to the cell and then overlays colored texture onto the position of the puck in the captured images. Therefore viewers can easily track the game.

Alternatively, AR has been applied to various fields such as virtual advertising [4, 6], virtual studio [4], training for car repairing [11, 12], virtual trial fitting system of glasses [13], simulation of space design [14, 15], educations [16, 17, 18], medical issues [19, 20, 21, 22, 23], entertainment [24, 25, 26].

For superimposing virtual objects onto video sequence in these ways, the system should decide the position of the virtual objects to be inserted to the video frame by frame. This is because the camera is moving or the players whom the virtual players are compared are not usually static. In AR research field, this problem is called as “Geometrical Registration”. There are some kinds of approaches for geometrical registration; sensor-based, vision-based and hybrid-based. In most cases, the goal of registration is obtaining the position and pose of the camera frame by frame. According to the obtained parameters, the positions of virtual objects are computed and moved at every frame. Therefore geometrical registration is equal to “Camera Tracking”.

In this thesis, we propose a new camera tracking method for an uncalibrated handheld camera. Our method especially focuses on planar structures in the real world which are visible in the captured images. The position and pose of the camera are estimated frame by frame with natural feature points on multiple planes. By applying to the registration for Augmented Reality, then, augmented images are synthesized in which virtual objects are superimposed onto the real world. In the camera tracking based on planes, using multiple planes can obtain more accurate results than using a single plane. Therefore our method uses natural feature points on multiple planes. In contrast with conventional methods, our method does not require geometrical relationship information of the planes because of introducing “3D Projective Space” which is defined by two static images. 3D configuration of the planes can be reconstructed in the Projective Space.

Secondly, our camera tracking method is extended to an on-line system “On-line AR System” which is based on planar pattern markers. In AR, it is very important issue that the system can work in real-time and on-line. This system utilizes planar pattern markers in which a black square and patterns are drawn in stead of using natural feature points on the planes in the real world. Compared with natural features, planar markers can be easily de-

tected from the images. Therefore this system can compute the camera position and pose and then generate augmented images in real-time. By using this system, even human-sized virtual objects can be overlaid with users in the large room like a virtual pet.

We also extend "On-line AR System" to the field of entertainment applications using AR. Augmented Reality has been also utilized for the entertainment. "Aqua Gauntlet" is a shooter game developed with MR Platform SDK [27]. Players wear HMDs with CCD cameras and shoot virtual enemies in the room where magnetic sensors are installed. "AR2Hockey" is also an application based on the same platform [28]. Players can share a physical game field and a virtual puck to play an air-hockey game. In the EXPO 2005 held in Aichi, the Hitachi Group Pavilion has presented entertainment show using AR technology [29]. Rare animals selected from the World Conservation Union's "red list" of animals threatened with global extinction are realistically re-created and superimposed onto the diorama in front of visitors. The visitors can see the animals through a handheld display and can contact with them via a special hand glove which has acceleration sensors. Using positioning sensors, visitors' view positions and directions are tracked so that the images of animals are generated according to each visitor's viewpoint.

In this thesis, we propose two applications for entertainment. "AR Baseball Presentation System" is an observation system of a virtual baseball game. Users place a real baseball field model on the tabletop and input a baseball game history (scorebook) into the system. Then they can watch the game by replaying with virtual baseball players on the field model in front of them. In "Interactive AR Bowling System", users knock down virtual pins by rolling a real ball. On the lane model, there are virtual pins generated with CG. Touching and rolling the real ball provide a sort of tangible feeling in this system.

Finally, we introduce a method for improving the accuracy of the planar pattern marker-based camera tracking. When using planar pattern markers and the image plane of the camera is almost parallel to the patterns, tracking accuracy decreases. Our proposed method resolves this problem by using Particle Filter which is a kind of temporal sequence filtering. By using not only the square form but also the pattern in the marker, the position and pose of camera are estimated. Furthermore, this method is applied to the system for mapping Geographic Information System (GIS) information onto the user's images captured from the high place. By using satellite images, the user's camera is tracked and then GIS information is overlaid onto the user's image frame by frame.

1.2 Thesis Outline

The outline of this thesis is as follows. The relationship of chapters is shown in Fig. 1.1. In Chapter 2, related work is discussed. Augmented Reality (AR) in Computer Vision (CV) area is introduced and then some required techniques are described. Geometrical registration which is one of the most important issues for AR is explained with categorized base equipments.

Chapter 3 presents a camera tracking algorithm using multiple planes without the relationship information of the planes. It is a key algorithm of this thesis that our tracking method does not require a prior-knowledge and a manual measuring of the planes in contrast with conventional methods. In this chapter, we introduce "Projective Space" to estimate the planes' alignments and track the camera motion by the multiple planes. Then, the proposed camera tracking method is applied to the registration for AR. Experimental results show augmented views where virtual objects are superimposed onto captured images taken with an uncalibrated camera. The effectiveness of our method which can reduce registration errors by using multiple planes is demonstrated by changing the number of planes used for computing. The accuracy of our method without the relationship of the planes is demonstrated by comparing with the conventional method which uses the relationship information in advance.

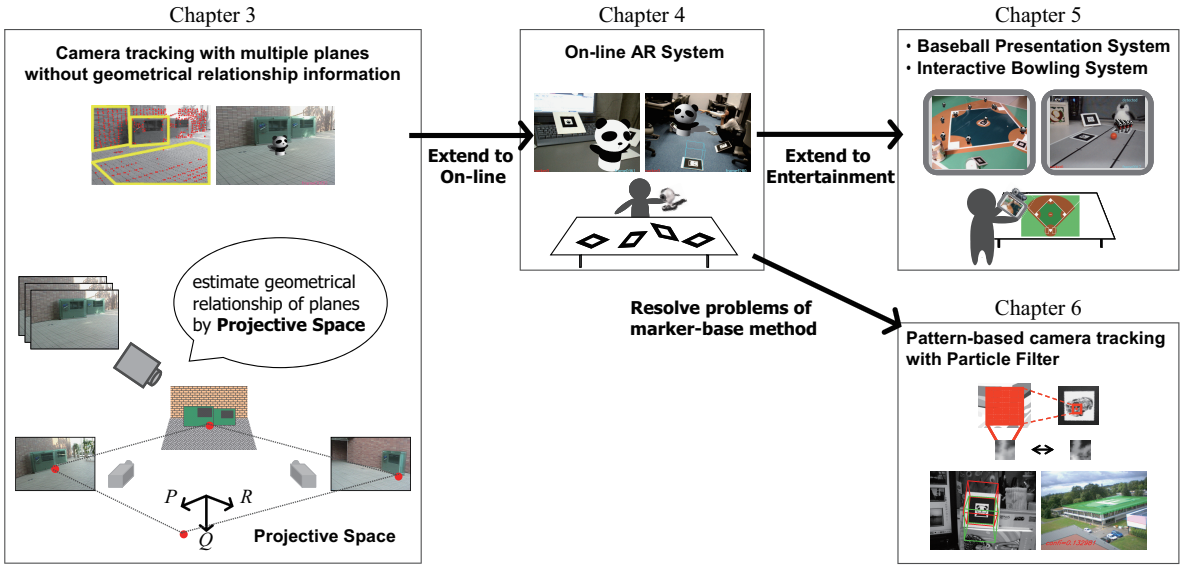


Fig. 1.1 Relationship of chapters in this thesis.

Chapter 4 describes an on-line system called as "On-line AR System" by extending the algorithm of Chapter 3. Multiple planar pattern markers are distributed in the real world and used for computing the camera position and pose. Since all processes in the system can be performed in real-time, the users can see that virtual objects exist in the real world with themselves via connecting a web-camera to a PC.

Chapter 5 presents two kinds of AR applications for entertainment by extending "On-line AR System" to specific objects. In both of "AR Baseball Presentation System" and "Interactive AR Bowling System", users can play them via a handheld display with a web camera by placing a baseball field model and a bowling lane model on a tabletop, respectively. The effectiveness of our applications, which has the form AR, is demonstrated by comparing with typical VR applications in which every object is generated with CG.

Chapter 6 introduces a method for improving the camera tracking based on planar pattern markers as described in Chapter 4. This method resolves the problem that the tracking accuracy becomes unstable when the image plane of the camera is almost parallel to the pattern by using Particle Filter. Experimental results show the effectiveness of our method by comparing our method with the conventional marker-based method. This method is also applied to an application for mapping GIS information.

Finally, Chapter 7 summarizes contributions of this work.

Chapter 2 :

Related Work

2.1 Geometrical Registration for Augmented Reality

Augmented Reality is a technique for mixing the real and virtual world. Viewers can see as if virtual objects really exist in the real world. In particular, images of virtual objects which are generated with CG are superimposed onto captured images as shown in Fig. 2.1.

In most cases, the camera is corresponding to the user's viewpoint, because the user sees both the real world and the virtual objects through some displays with cameras. When the camera captures the objective scene with moving, the position and pose of the camera are changed frame by frame. If the image sequence is captured from different viewpoints, the virtual objects also have to be overlaid onto the different positions and poses frame by frame so that the user can see that the virtual objects always stay at the same position in the real world as shown in Fig. 2.1.

Superimposing virtual objects onto video sequence according to the camera motion is called as "Geometrical Registration" in AR. The goal of the registration is obtaining the position and pose of the camera frame by frame. According to the obtained parameters, the positions and poses of virtual objects are computed and moved at every frame. Therefore geometrical registration is equal to "Camera Tracking". Some approaches for camera tracking which have been proposed are described as follows.

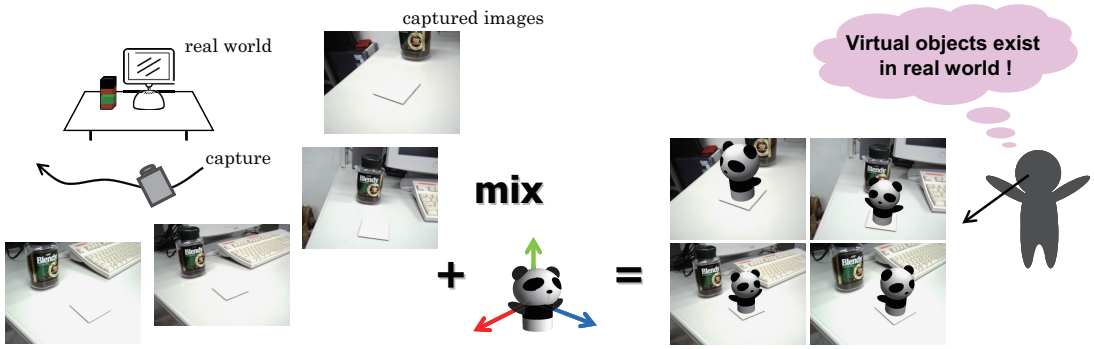


Fig. 2.1 Overlaying virtual object onto the real world by AR: The object scene is captured by a moving camera. According to the camera motion, a 3D virtual object is overlaid onto the captured images. A user can see that the virtual object really exists in the real world.

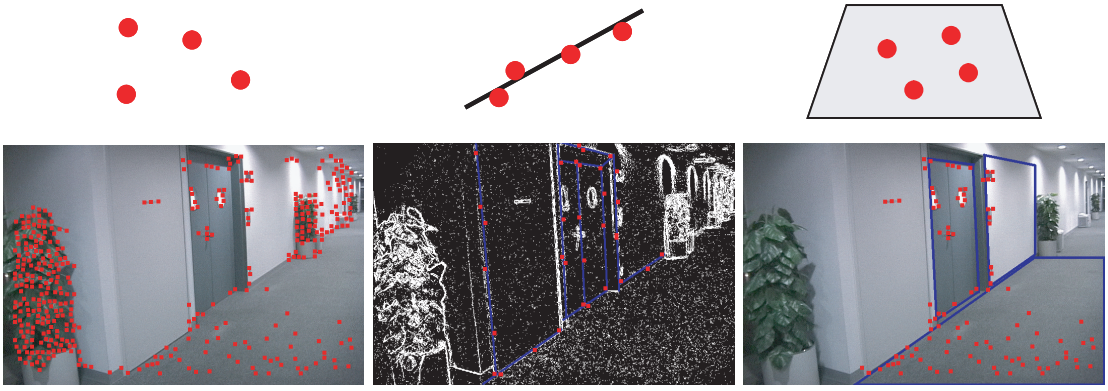
2.1.1 Sensor-based Camera Tracking

For tracking camera motion, some sensors are utilized especially for television broadcasting. In most cases, the camera platform has built-in positioning sensors such as gyroscopes, mechanical or optical sensors and obtains the parameters of pan, tilt, zoom, and so on [5, 4]. In MR Platform SDK [27], magnetic sensors are installed in the room and then the position and pose of the camera built in HMD are obtained in real-time.

Such sensors have advantages of a change in illuminations in contrast with vision-based approaches, because the color and appearance of the scene in captured image are changeable by illumination change or reflection. The sensor-based approaches are also robust against rapid moving of the camera, because motion blur is caused by such a rapid moving. Therefore hybrid approaches which uses both sensors and camera's images are also proposed [30, 31].

2.1.2 Vision-based Camera Tracking

In vision-based camera tracking, some features which are visible in captured images are utilized for computing the position and pose of camera. By detecting points, lines, or planes from the captured images as shown in Fig. 2.2 and analyzing the change in appearance of the features between frames, the camera motion with respect to the real scene is tracked frame by



(a) points (b) points on the line (c) points on the plane

Fig. 2.2 Natural features in images: Stronger geometrical constraints can be imposed on the feature points by lines and planes.

frame [32]. In contrast with sensor-based approaches, vision-based methods do not require any special or expensive devices. Therefore vision-based approaches are easily available for general users in the home.

These approaches can be categorized by type of features such as natural features, models and artificial markers. Using natural features is the most difficult and challenging approach, because it is not easy to detect and track natural feature points or lines in the real world in real-time and continuously. In natural feature based approaches, using points in the images is the most fundamental approach [33, 34, 35, 36, 37]. The method proposed by Neumann et al. [33] detects feature points, estimates Optical-Flow which represents the motion of the feature points in the video sequence and computes the camera motion. Chia et al. proposed the method which utilizes epipolar geometry between cameras [34]. Feature points are detected from reference images which are taken from two positions, and epipolar geometry (fundamental matrix) is estimated by the feature points. Corresponding points are then detected from the input images and the position of the virtual object is computed.

These feature points are generally detected or tracked by using the Harris operator [38], the KLT tracker [39], SIFT [40], and so on. However, there are not so many natural features in the real world which are available for camera tracking. This means that few natural feature points can be continuously tracked all over the video sequence. Moreover it causes registration jitters of virtual objects only by using feature points without any geometrical constraints. For keeping stable feature point tracking, some methods introduced a database in which 2D-3D correspondences of the feature points are stored [35, 41]. 2D and 3D position of good feature points which are correctly detected from the video sequence are stored in the database. Feature points are detected from the input images and then similar points are searched in the database. By using 2D-3D correspondences which are obtained from the database, camera pose is computed frame by frame.

Alternatively, edges (lines) in the images are also utilized for stabilizing the registration [41, 42, 43, 44, 45, 46]. The methods based on models or artificial markers often utilize edges. Klein et al. prepared a 3D model of labyrinth game and utilized edges of the model [47, 44]. By detecting edges in the input images and comparing the positions of the model's edges, camera position and pose are computed and a virtual object is overlaid onto the labyrinth. They also utilized LEDs and inertial sensors to simplify the estimation task.

2.2 Plane-based Camera Tracking

In order to impose more exact geometrical constraints on feature points, planar structures in the real world are available for camera tracking. Since there are a lot of planar structures in the real world such as indoor and outdoor walls and floors, some approaches have been proposed [48, 49, 50, 51]. Moreover a planar marker is also popular tool because it is easy to create [52, 53].

When using a single plane or a single planar marker, it fails in camera tracking if the plane is occluded or is not captured in the image as shown in Fig. 2.3. Therefore, using multiple planes is better than using a single plane, because the camera can move around a wide space by switching the planes based on the user's point of view. Moreover, using multiple planes simultaneously stabilizes tracking and improves its accuracy.

In order to use multiple planes, however, geometrical relationship of the planes has to be known in advance. This is because the camera position and pose are computed by integrating the parameters which are estimated based on each plane. For integrating the parameters obtained from each plane, geometrical relationship between the planes is required. Therefore conventional methods utilize the geometrical relationship information of the planes. In planar marker-based approaches [54, 55, 56, 57, 58], multiple markers are usually aligned at measured intervals as shown in Fig. 2.4(b).

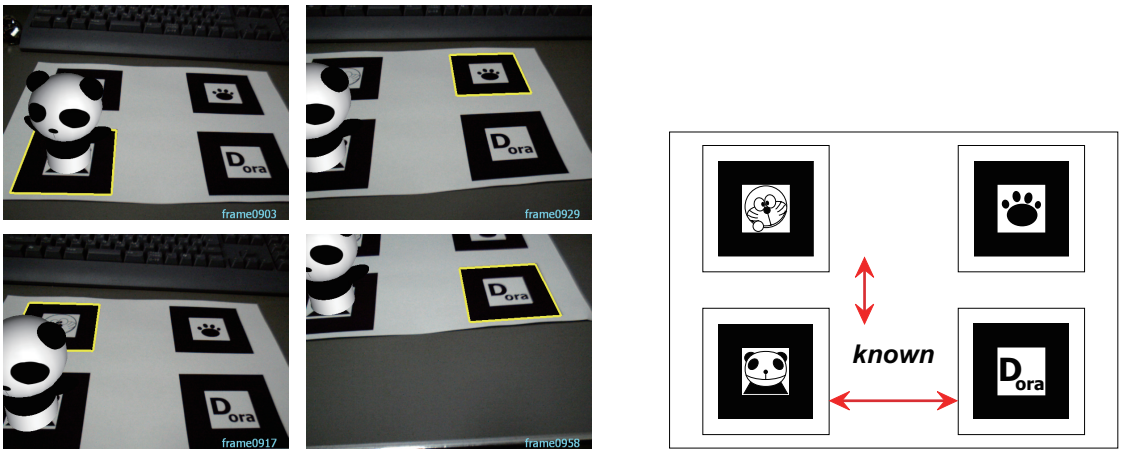
Simon et al. have proposed related approaches for AR, which use multiple planes in the real

(a) Success (b) Failure

Fig. 2.3 Example of failure using only 1 plane: If a part of the marker is not visible in the image, registration can not be continued.

world [48, 49, 50]. In one study [48], they tracked feature points on a plane in the real world and used them to estimate camera motion. Then, they overlaid virtual objects onto the plane based on the estimated camera motion. In another study [49], they extended this approach [48] in order to use multiple planes in the real world. A constraint they used was that the planes used to track the camera were perpendicular to a reference plane. In [50], they used multiple planes oriented in arbitrary positions and directions. The geometrical relationships between these planes and the camera motion were estimated by bundle adjustment, which is carried out over all frames.

In [54], they need the position and pose of a square marker and the position of a point marker in advance. In [55], they proposed marker-less registration method by setting up a learning process. In the learning process, however, the markers' geometrical information is required for learning the markers. In most cases, the task for the measurement of such information is implemented manually. However this task is very time-consuming and not sufficiently accurate. Kotake et al. [59] proposed a marker-calibration method combining multiple planar markers with bundle adjustment. Although they does not require a precise measurement of the markers, they need a priori knowledge of the markers such as qualitative information to compute markers' geometrical information from a set of images by bundle



(a) Overlaid virtual object

(b) Multiple markers

Fig. 2.4 Multiple marker-based registration of a virtual object: By switching a marker used for computing camera parameters, registration can be continued. For switching markers, the geometrical relationship of the markers (distance or direction) is required.

adjustment, *e.g.* multiple markers are coplanar.

In contrast, our registration method can use multiple planes at arbitrary position and pose. Multiple planar markers are also distributed freely. The geometrical relationship of the planes can be automatically estimated by constructing a "3D Projective Space" which is defined by projective reconstruction of two reference images. Through the Projective Space, then, the geometrical arrangement of the planes is recovered in 3D. Therefore we need not to manually measure the distance between the planes in advance. The planes used for registration do not have to be perpendicular each other. Thus our method can be applied to not only small space such as a tabletop but also large space like a whole room. In marker-based applications, we do not require that the specified marker is always captured in the images. Therefore this algorithm is quite suitable for AR when the camera moves around the real world and some markers may be occluded.

2.3 AR Applications for Entertainment

Because the real-time interaction is available and visual appearance is effective, AR is frequently applied to entertainment field such as collaboration systems and game applications. Billinghurst *et al.* have proposed a remote face-to face AR system [60]. The system allows users to share the same tabletop space when they exist in distant places. Arranging a lot of 2D markers around the tabletop, each user can see the other user's appearance via HMD. Therefore they can feel to share the same space each other. Similar approaches for collaboration system have also proposed so far [25, 26]. Recently, video conference connecting multiple places is not so special and frequently held. Therefore the concept of sharing the same space with multiple users is very important. AR can provide such environment by sharing the whole ambient appearance. For multiple users, multiple markers should be placed instead of using a single marker. However, all of the markers may be undetected at the same time, if they are placed on the tabletop and face to the same directions. In contrast, our AR applications presented in this thesis allow the markers to face various directions. Such distribution increases the chance of detecting at least one marker by every user's camera.

Tangibility is also a great advantage of entertainment AR. In the tangible AR, real objects are turned into input and output devices for computer interfaces. Since the physical objects have own properties and physical constraints, users can easily manipulate them in usual way to use them. For example, in [54], a user move a virtual furniture by scooping it with a real paddle on which a 2D marker is placed. Scooping something with a paddle is a usual way to use for us. Therefore such intuitive operation is quite friendly for every user. Many applications demonstrating the effectiveness of tangible AR have been proposed [61, 62].

In these days, a camera phone is utilized for AR because it is handy and has both display and camera. Henrysson *et al.* have proposed the AR tennis application [63]. By placing 2D markers on the tabletop tennis court, the users play the tennis with a virtual court and a virtual ball. Since a ball is a virtual object and user's position does not widely move, the 2D markers can be detected by the user's camera. If a real ball is employed instead of the virtual ball to be a tangible application, the real ball may occlude the marker. Since the ball is a real object in our bowling application, the ball occludes the marker on the lane. However, our system can place the markers to be slanted. When the ball goes over the markers on the lane, therefore, the other slanted markers can be detected to continue the augmented game.

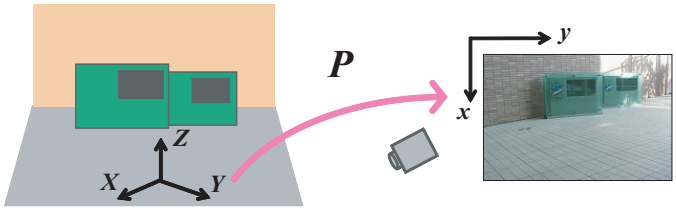
Chapter 3 :

Camera Tracking Method Using Multiple Planes

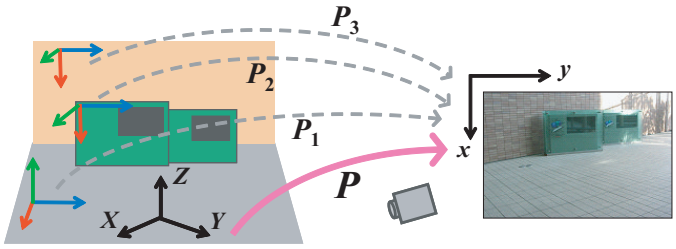
In this chapter, a camera tracking method using multiple planes is described [64]. This method does not require the geometrical relationship of the planes in contrast with conventional methods. In stead of a prior-knowledge of the planes or a manual measuring task, this method utilizes "Projective Space" defined by projective reconstruction of two reference images. In this thesis, this tracking method is applied to the registration of virtual objects for AR. In addition, this algorithm is extended to the planar pattern marker-based tracking and its applications described in following chapters.

3.1 Overview of Registration Method

The camera's position and pose with regard to the real world must be estimated frame-by-frame to superimpose virtual objects. This corresponds to computing a projection matrix



(a) Projection matrix between real world and input image is computed from one plane.



(b) Projection matrices computed from the multiple planes are integrated into one matrix.

Fig. 3.1 Projection matrix from the real world to input images: Projection matrix represents camera position and pose with respect to the plane. When using multiple planes, a projection matrix is computed with respect to each plane.

from the coordinate system of the real world (X-Y-Z) to the coordinate system of the input image (x-y), which is captured by a camera, as shown in Fig. 3.1(a).

Simon et al. have shown that the projection matrix can be computed from planar homography [48, 49]. Although the projection matrix can be computed using only a single plane [49], the accuracy of the projection matrix can be improved by using multiple planes.

In our proposed method, a projection matrix is computed from each plane independently, and then the multiple projection matrices are integrated into one matrix, as shown in Fig. 3.1(b). Use of the geometrical relationship among the multiple planes is necessary to integrate those projection matrices into one matrix. Our method can automatically estimate the geometrical relationship using "3D Projective Space", in which the position and pose of the planes are estimated. Therefore manual measurements of the planes and restrictions on their placement become unnecessary.

3.1.1 Definition of Coordinate Systems

In this section, we introduce some coordinate systems and transformation matrices among them used in our method. To track the camera's position and pose with regard to the real world and align the coordinates of the virtual objects with the input image, our method defines three different coordinate systems to represent the real world (X_i-Y_i-Z_i), the input image (x-y), and a virtual space (P-Q-R), respectively. We define such coordinate systems as shown in Fig. 3.2.

First, a 3D coordinate system (X_i-Y_i-Z_i) is independently assigned to each plane i. In particular, four points are assigned on each plane. The detail will be described in Sec. 3.3. The 2D coordinate system (x-y) represents the input image corresponding to the camera coordinate system. We select one plane as "base plane" whose 3D coordinate system is especially described as (X_{base}-Y_{base}-Z_{base}) in which the 3D coordinates of the virtual objects are described.

Since the multiple planes are placed at arbitrary positions and directions in the real world, the relationship among (X_i-Y_i-Z_i) is unknown. Therefore we introduce "Projective Space" (P-Q-R) to estimate the relationship. This space is a 3D non-Euclidean coordinate system and is defined by projective reconstruction of two images that are called "reference images" and are captured from two different viewpoints. This will be described in Sec. 3.4.

The relationship among each plane (X_i-Y_i-Z_i) is represented by a transformation matrix $\boldsymbol{T_i^{WP}}$. The camera position and pose are included in a projection matrix $\boldsymbol{P_i^{WI}}$. Therefore we can use multiple planes without the geometrical relationship information via the Projective Space.

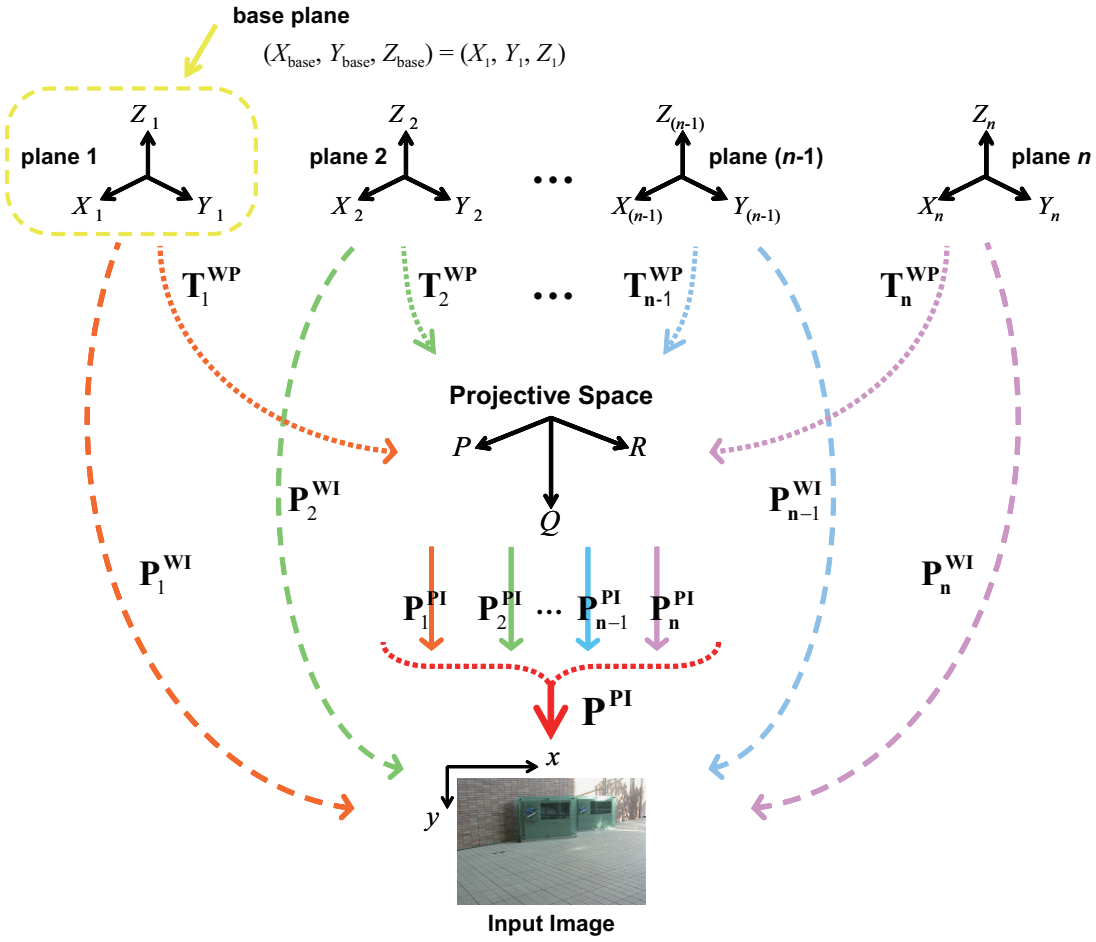


Fig. 3.2 Coordinate systems defined in our method: (X_i-Y_i-Z_i) is defined on each plane by clicking 4 points on each plane. (x-y) is corresponding to the camera coordinate system. (P-Q-R) is defined by two captured images.

3.1.2 Flow of Our Method

Our method can be divided into two phases as shown in Fig. 3.3. In the first phase, the object scene is captured from two different viewpoints. These images are reference images A and B, which are used to define a Projective Space. Here, we manually specify four points on each plane in each of the two reference images in order to define the (X-Y) axis of a 3D coordinate system for each plane, as shown in Fig. 3.4. The Z-axis is defined so that each plane becomes $Z=0$. The four points are characteristic feature points and selected by users.

Next, 3D Projective Space is defined by projective reconstruction of the two reference images. Then, $\boldsymbol{T_i^{WP}}$ is computed based on each plane, which is a transformation matrix from each 3D coordinate system of the plane (X_i-Y_i-Z_i) to the Projective Space (P-Q-R). $\boldsymbol{T_i^{WP}}$ is computed from five or more corresponding points between (X_i-Y_i-Z_i) and (P-Q-R) obtained by the reference images.

The 3D coordinates of the virtual objects are described in (X_{base}-Y_{base}-Z_{base}), which is the 3D coordinate system assigned to one of the multiple planes. After computing $\boldsymbol{T_i^{WP}}$ for all planes, then, the 3D coordinates of virtual objects are transformed into the coordinate

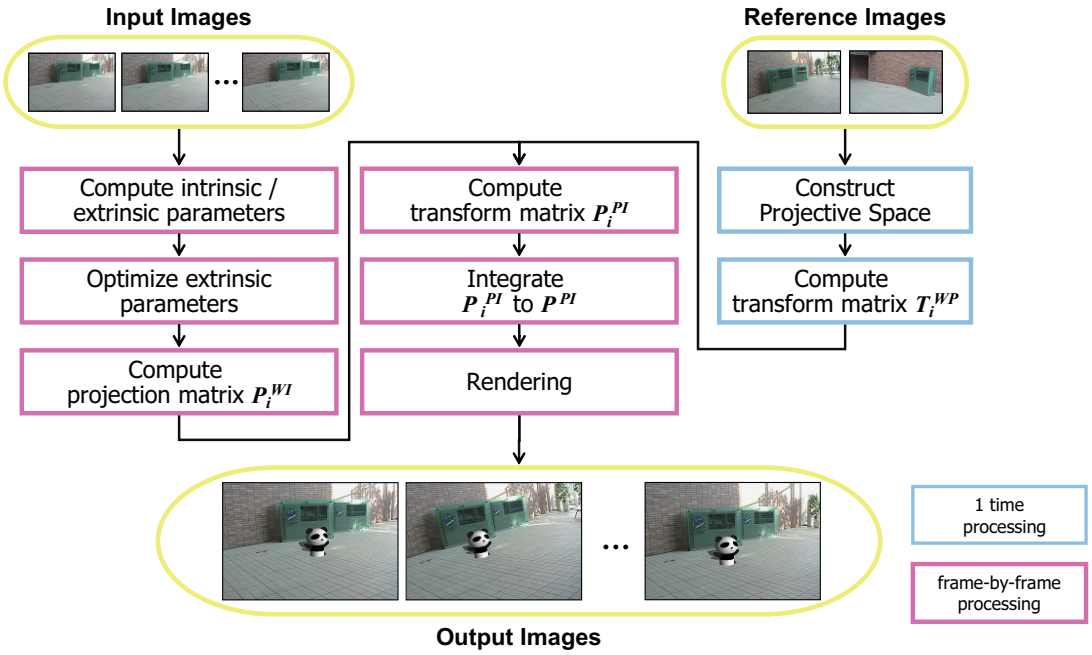


Fig. 3.3 Overview of our registration method.

system of the Projective Space by $\boldsymbol{T}_{\boldsymbol{base}}^{\boldsymbol{WP}}$.

In the second phase, a projection matrix $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{WI}}$ based on each plane i that is visible in the current frame of the input image sequence is computed from a homography, which is computed from four or more corresponding points between 2D points on the current frame and 3D points on (X-Y) plane in the real world ($Z = 0$).

After computing all $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{WI}}$ for the plane in the current frame, $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{PI}}$ is computed by eq. (3.1) from $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{WI}}$ and $\boldsymbol{T}_{\boldsymbol{i}}^{\boldsymbol{WP}}$, which have been computed in the first phase. Then all $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{PI}}$ are integrated into one projection matrix $\boldsymbol{P}^{\boldsymbol{PI}}$.

Finally, the virtual objects described in the Projective Space are projected onto the current frame by $\boldsymbol{P}^{\boldsymbol{PI}}$ as the output image. These processes of the second phase are repeated in every frame until the end of the input image sequence. Computing $\boldsymbol{T}_{\boldsymbol{i}}^{\boldsymbol{WP}}$ corresponds to indirect estimation of the geometrical relationship among the planes via the Projective Space. Computing $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{WI}}$ for every frame corresponds to the camera tracking in the current frame.

3.2 Relationship of Coordinate Systems

The geometrical relationship among the planes is estimated by computing each transformation matrix. This relates (X_i-Y_i-Z_i) to (P-Q-R). The camera's position and pose in every frame are tracked by computing each projection matrix from (X_i-Y_i-Z_i) to (x-y). These two kinds of matrices, which are based on the plane i, are represented as $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$, respectively, as shown in Fig. 3.2. The details of the computation will be described in Sec. 3.5 and Sec. 3.6.

When $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$ are obtained based on each plane i, a 3D point (P, Q, R) in the Projective Space is independently projected to a 2D point (x, y) in the image coordinate system by the following equation,

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{WI}} \left(\boldsymbol{T_i^{WP}}\right)^{-1} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{PI}} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} \tag{3.1}$$

where $\boldsymbol{P_i^{PI}}$ is a projection matrix computed based on plane i. Since each $\boldsymbol{P_i^{PI}}$ represents the projection from (P-Q-R) to (x-y), all $\boldsymbol{P_i^{PI}}$ should coincide with each other. Therefore, we can uniquely define the projection from the Projective Space to the input image by integrating these $\boldsymbol{P_i^{PI}}$ into one projection matrix. The integrated projection matrix is called $\boldsymbol{P^{PI}}$ as shown in Fig. 3.2. $\boldsymbol{P^{PI}}$ includes information on the geometrical relationship among the planes and camera's position and pose of the current frame.

As discussed in the previous section, the 3D coordinates of the virtual objects are described in (X_{base}-Y_{base}-Z_{base}), which is assigned to the base plane. If the virtual objects are transformed into the coordinate system of the Projective Space by $\boldsymbol{T_{base}^{WP}}$, the virtual objects can be projected onto the input image by $\boldsymbol{P^{PI}}$. Therefore, to overlay the virtual objects onto the input image based on the camera's motion, our method computes $\boldsymbol{P^{PI}}$ from $\boldsymbol{T_i^{WP}}$ and $\boldsymbol{P_i^{WI}}$ at every frame.

3.3 Assigning 3D Coordinate Systems on Multiple Planes

In our method, a 3D coordinate system is independently assigned to each plane that is used for registration. We specify the 3D coordinate system through the reference images used to define a Projective Space.

First, four points on each plane are clicked on the reference images A and B, as shown in Fig. 3.4. In particular, we specify the four points so that they become a square on the X-Y axes. The Z-axis becomes perpendicular to the plane. Next, one of the planes is selected as a base plane, and then the 3D coordinates of the four points on the base plane are arbitrarily defined as $(X, Y, 0)$. 2D coordinates of the points (x, y) are known in clicking the points in the reference images. By correspondence between (X, Y) and (x, y), therefore, two projection matrices which relates the base plane to reference image A and B are computed respectively. This computation of the projection matrix using plane is explained in Sec. 3.6. A similar approach is also taken in [49]. The 3D coordinates of the four points on the other planes are computed by 3D reconstruction using a stereo algorithm as described in the appendix. These four points are used for computing homography as described in Sec. 3.6.1.

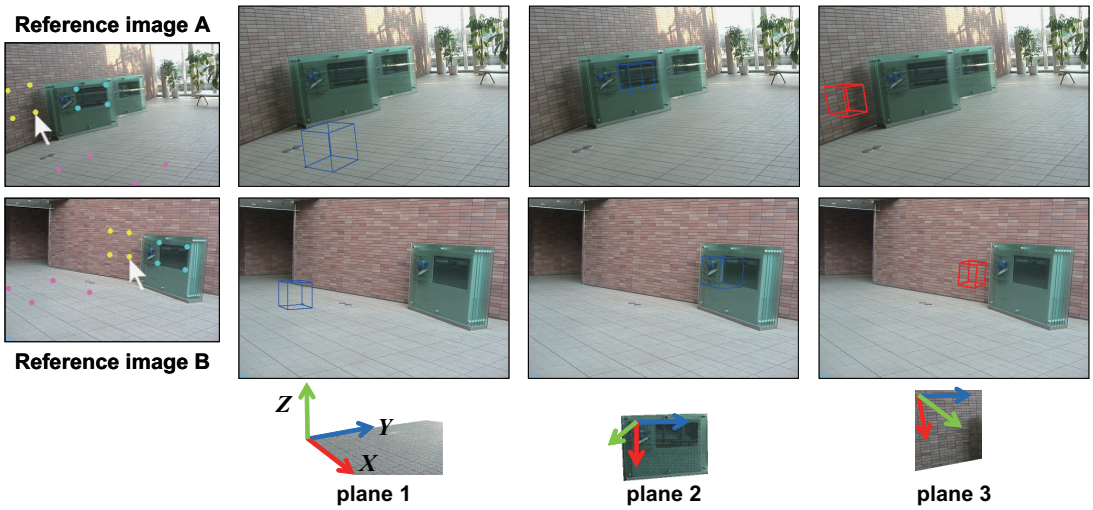


Fig. 3.4 Assigning a 3D coordinate system on each plane: (left column) 4 points are manually clicked on each plane in the reference images. (right 3 columns) 3D coordinate system is defined on each plane so that each plane defines X-Y plane and Z axis becomes perpendicular to the plane. The wire cubes on the planes represent the directions of 3D axes.

3.4 Definition of Projective Space

3D Projective Space is used to estimate the geometrical relationship among the multiple planes that occur in arbitrary positions and poses. The Projective Space is defined by projective reconstruction of two images that are captured from two different viewpoints and are called reference images.

As shown in Fig. 3.5, the object scene is captured from two viewpoints and the captured images become reference images A and B. Then, a 3D coordinate system (P-Q-R) is defined so that it has the following projective relationships with each of the reference images. This 3D coordinate system is called 3D Projective Space.

$$\begin{bmatrix} u_A \\ v_A \\ 1 \end{bmatrix} \simeq \boldsymbol{P_A} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix}, \qquad \begin{bmatrix} u_B \\ v_B \\ 1 \end{bmatrix} \simeq \boldsymbol{P_B} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} \tag{3.2}$$

$$\boldsymbol{P_A} = [\mathbf{I} \,|\, \mathbf{0}], \qquad \boldsymbol{P_B} = \left[-\frac{[\boldsymbol{e_B}]_\times \boldsymbol{F_{AB}}}{\|\boldsymbol{e_B}\|^2} \,|\, \boldsymbol{e_B} \right] \tag{3.3}$$

Here, $[u_A, v_A, 1]^\top$ and $[u_B, v_B, 1]^\top$ are homogeneous coordinates of 2D points in the reference images, and $[P, Q, R, 1]^\top$ is a homogeneous coordinate of a 3D point in the Projective Space. $\boldsymbol{F_{AB}}$ is a fundamental matrix from the image A to image B. $\boldsymbol{e_B}$ is an epipole on the image B, and $[\boldsymbol{e_B}]_\times$ is the skew-symmetric matrix of $\boldsymbol{e_B}$ [65].

Our method computes $\boldsymbol{F_{AB}}$ and $\boldsymbol{e_B}$ by eight or more corresponding points between the reference images, which are manually specified, and then $\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ are defined by eq. (3.3). 3D coordinates in (P-Q-R) space are computed by $\boldsymbol{P_A}$, $\boldsymbol{P_B}$ and 2D projected points onto the reference images. The detail is described in Sec. 3.5.1.

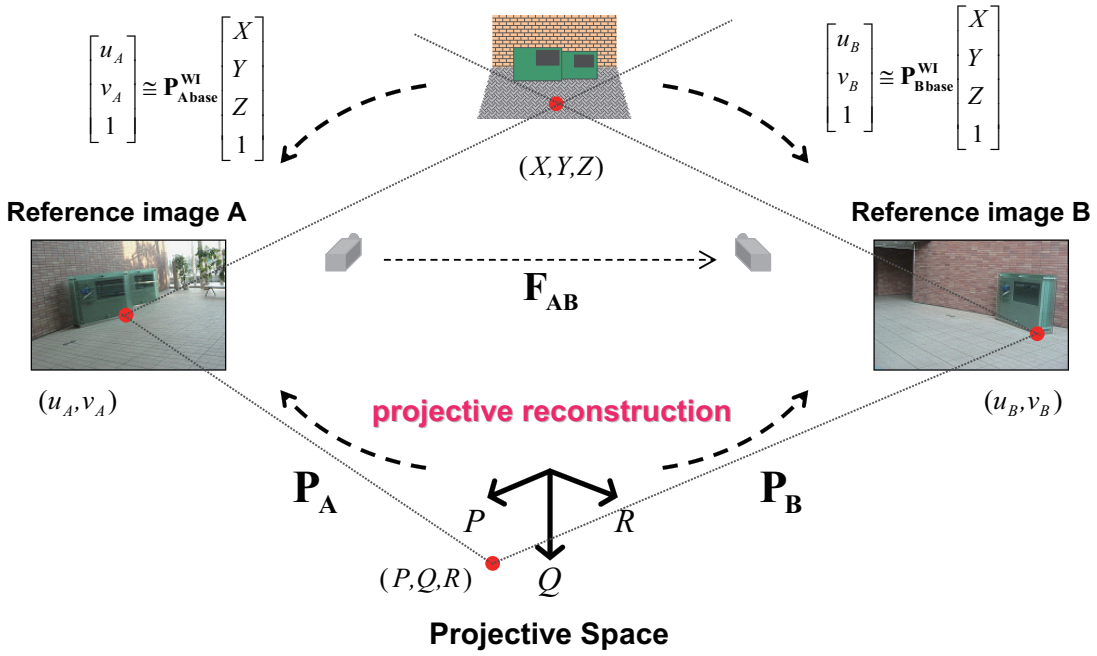


Fig. 3.5 3D Projective Space: (P-Q-R) is computed by projective reconstruction of two reference images.

3.5 Computation of $\boldsymbol{T_i^{WP}}$ (Planes→Projective Space)

Since the coordinate system associated with a plane i (X_i-Y_i-Z_i) and on the Projective Space (P-Q-R) are both 3D coordinate system, the transformation matrix $\boldsymbol{T_i^{WP}}$ is a 4×4 matrix and is computed from five or more corresponding points between (X_i-Y_i-Z_i) and (P-Q-R).

3.5.1 How to get corresponding points

We assume that a 3D point in (X_i-Y_i-Z_i) is $\boldsymbol{X_W} \simeq [X, Y, Z, 1]^\top$ and that the point is projected onto reference images A and B as 2D points (u_A, v_A) and (u_B, v_B), respectively. When these 2D points are projected into the Projective Space as a 3D point, $\boldsymbol{X_P} \simeq [P, Q, R, 1]^\top$, we can write

$$\begin{bmatrix} \boldsymbol{p_A^1} - u_A \boldsymbol{p_A^3} \\ \boldsymbol{p_A^2} - v_A \boldsymbol{p_A^3} \\ \boldsymbol{p_B^1} - u_B \boldsymbol{p_B^3} \\ \boldsymbol{p_B^2} - v_B \boldsymbol{p_B^3} \end{bmatrix} \begin{bmatrix} P \\ Q \\ R \\ 1 \end{bmatrix} = \boldsymbol{0}, \tag{3.4}$$

where, $\boldsymbol{p_A^j}$ and $\boldsymbol{p_B^j}$ are the jth row vectors of $\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ in eq. (6.3). Then $\boldsymbol{X_P}$ is computed by Singular Value Decomposition of the 4×4 matrix on the left-hand side of eq. (3.4). Therefore, the corresponding points $\boldsymbol{X_W}$ and $\boldsymbol{X_P}$ are obtained through the 2D points (u_A, v_A) and (u_B, v_B) on the reference images. Since five or more corresponding points are required to compute $\boldsymbol{T_i^{WP}}$, we draw a cube on each plane as shown in Fig. 3.4. The size of the cube is already known when defining 3D coordinate systems. Then, using eight vertices from each cube, eight corresponding points between (X_i-Y_i-Z_i) and (P-Q-R) can be obtained by eq. (3.4).

3.5.2 How to compute the matrix

As described in Sec. 3.2 and Fig. 3.2, the relationship between the coordinate system of the plane i and the Projective Space is expressed by the following equation.

$$\boldsymbol{X_P} \simeq \boldsymbol{T_i^{WP}} \boldsymbol{X_W} \tag{3.5}$$

$$\boldsymbol{T_i^{WP}} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & t_{14} \\ t_{21} & t_{22} & t_{23} & t_{24} \\ t_{31} & t_{32} & t_{33} & t_{34} \\ t_{41} & t_{42} & t_{43} & 1 \end{bmatrix} \tag{3.6}$$

In order to estimate the relationship of the planes, the elements of $\boldsymbol{T_i^{WP}}$ should be computed by eq. (3.5). To obtain $\boldsymbol{T_i^{WP}}$, therefore, we resolve the following equation by Least Square Method,

$$\min \left(\boldsymbol{X_P} - \boldsymbol{T_i^{WP}} \boldsymbol{X_W} \right)^2 \tag{3.7}$$

In particular, eq. (3.5) is transformed as

$$\boldsymbol{M}\boldsymbol{t} = \boldsymbol{b}, \tag{3.8}$$

where,

$$\boldsymbol{M} = \begin{bmatrix} \boldsymbol{X_W^\top} & \mathbf{0} & \mathbf{0} & -XP & -YP & -ZP \\ \mathbf{0} & \boldsymbol{X_W^\top} & \mathbf{0} & -XQ & -YQ & -ZQ \\ \mathbf{0} & \mathbf{0} & \boldsymbol{X_W^\top} & -XR & -YR & -ZR \end{bmatrix} \tag{3.9}$$

$$\boldsymbol{t} = \begin{bmatrix} t_{11} & t_{12} & t_{13} & \cdots & t_{41} & t_{42} & t_{43} \end{bmatrix}^\top \tag{3.10}$$

$$\boldsymbol{b} = \begin{bmatrix} P & Q & R \end{bmatrix}^\top \tag{3.11}$$

If there are j corresponding points of $\boldsymbol{X_W}$ and $\boldsymbol{X_P}$ $(j \geq 5)$, eq. (3.8) is

$$\begin{bmatrix} \boldsymbol{M_1} \\ \vdots \\ \boldsymbol{M_j} \end{bmatrix} \boldsymbol{t} = \begin{bmatrix} \boldsymbol{b_1} \\ \vdots \\ \boldsymbol{b_j} \end{bmatrix} \tag{3.12}$$

Then, the elements of $\boldsymbol{T_i^{WP}}$ are obtained by Least Square Method of eq. (3.12).

3.6 Computation of $\boldsymbol{P_i^{WI}}$ (plane → image)

In this section, we explain how to compute $\boldsymbol{P_i^{WI}}$ which projects a 3D point in the coordinate system of the plane i (X_i-Y_i-Z_i) to a 2D point on the coordinate system of the current frame of the input images (x-y). In our method, a homography between each plane i and the input image is computed first, then the $\boldsymbol{P_i^{WI}}$ is computed from the homography. Since a projection matrix consists of intrinsic and extrinsic parameters, we separately compute those parameters from the homography.

3.6.1 Computation of $\boldsymbol{H_i}$ (X-Y plane ↔ x-y image plane)

A homography represents the projective transformation between planes and is computed from four or more corresponding points on both planes.

In the first frame of the input images, we specify four points of each plane in the image. These four points corresponds to the four points which are assigned on the reference images when X and Y axes are assigned in the first phase, as shown in Fig. 3.4 and described in Sec. 3.3. $\boldsymbol{H_i}$ is computed based on each plane using these four corresponding points.

After the second frame, feature points on each plane in the input images are tracked by the KLT Feature Tracker [39] and used to compute the homography of each plane between the current frame and the previous frame. Then, a new $\boldsymbol{H_i}$ is obtained by multiplying the homography between the current frame and the previous frame by $\boldsymbol{H_i}$, which relates the

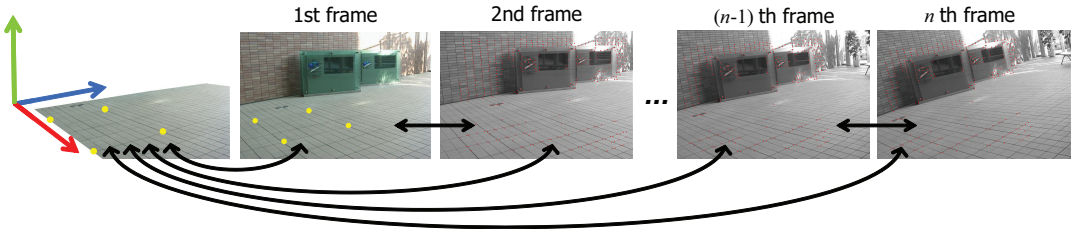


Fig. 3.6 Homography between 3D plane and current frame: By using homography between adjacent frames, homography between 3D plane and current frame is computed.

previous frame to the X-Y plane.

3.6.2 Relationship between $\boldsymbol{P_i^{WI}}$ and $\boldsymbol{H_i}$

Since a 3D coordinate system is projected onto a 2D coordinate system by a 3×4 projection matrix, each 3D coordinate system based on plane i is projected to the image coordinate system by each projection matrix $\boldsymbol{P_i^{WI}}$.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{WI}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}, \qquad \boldsymbol{P_i^{WI}} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & p_{34} \end{bmatrix} \tag{3.13}$$

As descried above, a 3D coordinate system is assigned to each plane so that each plane becomes $Z=0$. Therefore, eq. (3.13) can be written as the following equation,

$$\begin{aligned} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \simeq \boldsymbol{P_i^{WI}} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} &\simeq \begin{bmatrix} p_{11} & p_{12} & p_{14} \\ p_{21} & p_{22} & p_{24} \\ p_{31} & p_{32} & p_{34} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \\ &\simeq \hat{\boldsymbol{P}}_{\boldsymbol{i}} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \simeq \boldsymbol{H_i} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \\ \therefore \hat{\boldsymbol{P}}_{\boldsymbol{i}} &\simeq \boldsymbol{H_i} \end{aligned} \tag{3.14}$$

where, $\hat{\boldsymbol{P}}_{\boldsymbol{i}}$ is a 3×3 matrix that lacks the third column vector of $\boldsymbol{P_i^{WI}}$. $\boldsymbol{H_i}$ is a homography of plane i between the X-Y plane and the image plane (x-y plane). As described in eq. (3.14), $\hat{\boldsymbol{P}}_{\boldsymbol{i}}$ is equivalent to homography $\boldsymbol{H_i}$. Thus, we estimate the third column vector of $\hat{\boldsymbol{P}}_{\boldsymbol{i}}$ from $\boldsymbol{H_i}$. Actually, $\hat{\boldsymbol{P}}_{\boldsymbol{i}}$ is divided into intrinsic and extrinsic parameters for the estimation.

$$\boldsymbol{P_i^{WI}} = \boldsymbol{A}\,[\boldsymbol{R} \mid \boldsymbol{t}] = \boldsymbol{A}\,[\boldsymbol{r_1}\ \boldsymbol{r_2}\ \boldsymbol{r_3}\ \boldsymbol{t}] \tag{3.15}$$

$$\hat{\boldsymbol{P}}_{\boldsymbol{i}} = \boldsymbol{A}\,[\boldsymbol{r_1}\ \boldsymbol{r_2}\ \boldsymbol{t}] = \boldsymbol{H_i} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \tag{3.16}$$

3.6.3 Estimation of intrinsic parameters

If an uncalibrated camera is used to capture input images, intrinsic parameters of the camera are estimated by homography $\boldsymbol{H_i}$. In our method, the matrix of intrinsic parameters $\boldsymbol{A}$ is

defined as

$$\boldsymbol{A} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{3.17}$$

where, f is the focal length and (c_x, c_y) is the center point of the image. The skew and aspect ratios are fixed to 0 and 1, respectively. Therefore, we need to estimate f. Eq. (3.16) can be written as follows.

$$\boldsymbol{A}^{-1}\boldsymbol{H_i} = [\boldsymbol{r_1}\ \boldsymbol{r_2}\ \boldsymbol{t}] \tag{3.18}$$

The first and second vectors of $\boldsymbol{A}^{-1}\boldsymbol{H_i}$ corresponds to $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$. Based on the property of rotation matrix $\boldsymbol{R}$, which is that the inner product of $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$ is equal to 0, we can calculate the focal length f.

$$f^2 = \frac{(h_{11} - c_x h_{31})(h_{12} - c_x h_{32}) + (h_{21} - c_y h_{31})(h_{22} - c_y h_{32})}{-h_{31} h_{32}} \tag{3.19}$$

3.6.4 Estimation of extrinsic parameters

Extrinsic parameters of a camera consist of a rotation matrix $\boldsymbol{R}$ and a translation vector $\boldsymbol{t}$. Since $\boldsymbol{r_1}$, $\boldsymbol{r_2}$ (the first and second column vectors of $\boldsymbol{R}$), and $\boldsymbol{t}$ are already known, as shown in eq. (3.18), we should estimate only $\boldsymbol{r_3}$. Then, also according to the property of $\boldsymbol{R}$, which is that the cross product of $\boldsymbol{r_1}$ and $\boldsymbol{r_2}$ becomes $\boldsymbol{r_3}$, we compute $\boldsymbol{r_3}$. Therefore, $\boldsymbol{R}$ is

$$\boldsymbol{R} = [\boldsymbol{r_1}\ \boldsymbol{r_2}\ (\boldsymbol{r_1} \times \boldsymbol{r_2})] = [\boldsymbol{r_1}\ \boldsymbol{r_2}\ \boldsymbol{r_3}] \tag{3.20}$$

$\times$: crossproduct

3.7 Computation and integration of $\boldsymbol{P_i^{PI}}$ (Projective Space → image)

In this section, we explain how to compute $\boldsymbol{P_i^{PI}}$, which is a projection matrix from the Projective Space to the coordinate system of the input image, and how to integrate all $\boldsymbol{P_i^{PI}}$ into $\boldsymbol{P^{PI}}$.

After computing $\boldsymbol{P_i^{WI}}$ based on each plane, $\boldsymbol{P_i^{PI}}$ is obtained by using $\boldsymbol{P_i^{WI}}$ and $\boldsymbol{T_i^{WP}}$, which is computed in the first phase.

$$\boldsymbol{P_i^{PI}} = \boldsymbol{P_i^{WI}} \left(\boldsymbol{T_i^{WP}}\right)^{-1} \tag{3.21}$$

As described in Sec. 3.2, all $\boldsymbol{P_i^{PI}}$ should all coincide with each other because they represent a common geometrical projection between the Projective Space and the input image. Therefore, all $\boldsymbol{P_i^{PI}}$ are integrated into a single $\boldsymbol{P^{PI}}$. The details are as follows.

When $\boldsymbol{P_i^{PI}}$ is computed for each plane, a 2D point $\boldsymbol{x_j}$ in the input image, which corresponding to a 3D point $\boldsymbol{X}_{\boldsymbol{P_j}} \simeq [P_j, Q_j, R_j, 1]^\top$ in the Projective Space can be obtained by the following equation.

$$\boldsymbol{x_j} \simeq \boldsymbol{P_i^{PI}} \boldsymbol{X}_{\boldsymbol{P_j}} \tag{3.22}$$

If there are n planes in the real world, n equations are obtained as well as eq. (3.22) $(1 \leq i \leq n)$. Therefore, we integrate $\boldsymbol{P_1^{PI}}$ to $\boldsymbol{P_n^{PI}}$ into one $\boldsymbol{P^{PI}}$. By Least-Square-Method, the elements of $\boldsymbol{P^{PI}}$ are computed as follows.

$$\min \left(\boldsymbol{x} - \boldsymbol{P^{PI}} \boldsymbol{X}_{\boldsymbol{P}}\right)^2 \tag{3.23}$$

$$\begin{bmatrix} \boldsymbol{M_1} \\ \vdots \\ \boldsymbol{M_n} \end{bmatrix} \boldsymbol{p} = \begin{bmatrix} \boldsymbol{b_1} \\ \vdots \\ \boldsymbol{b_n} \end{bmatrix} \tag{3.24}$$

where, there are m corresponding points, and $\boldsymbol{M_i}$ is

$$\boldsymbol{M_i} = \begin{bmatrix} \boldsymbol{X}_{\boldsymbol{P_1}}^{\top} & \mathbf{0} & -P_1x_1 & -Q_1x_1 & -R_1x_1 \\ \mathbf{0} & \boldsymbol{X}_{\boldsymbol{P_1}}^{\top} & -P_1y_1 & -Q_1y_1 & -R_1y_1 \\ & & \vdots & & \\ \boldsymbol{X}_{\boldsymbol{P_m}}^{\top} & \mathbf{0} & -P_mx_m & -Q_mx_m & -R_mx_m \\ \mathbf{0} & \boldsymbol{X}_{\boldsymbol{P_m}}^{\top} & -P_my_m & -Q_my_m & -R_my_m \end{bmatrix} \tag{3.25}$$

$$\boldsymbol{b_i} = \begin{bmatrix} x_1 \\ y_1 \\ \vdots \\ x_m \\ y_m \end{bmatrix}, \qquad \boldsymbol{p} = \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ \vdots \\ p_{31} \\ p_{32} \\ p_{33} \end{bmatrix}, \qquad \boldsymbol{P^{PI}} = \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & 1 \end{bmatrix} \tag{3.26}$$

Therefore, the elements of $\boldsymbol{P^{PI}}$ are obtained by Least Square Method of eq. (3.24).

An alternative we have considered and rejected is determining the projection matrix by simply selecting the most accurate single projection matrix. The difficulty with this approach, and our reason for not adopting it, is that it introduces instability in moving from frame to frame, because the projection matrix is computed based on the different plane. Therefore we employ the integration process instead of switching the planes.

3.8 Rendering of Virtual Objects

As described in Sec. 3.1.2, the 3D coordinates of the virtual objects (X_V-Y_V-Z_V) are described in (X_{base}-Y_{base}-Z_{base}). After computing $\boldsymbol{P^{PI}}$, therefore, the projection matrix which is used for projecting the virtual objects onto the image is computed as following equation.

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cong \boldsymbol{P^{PI}} \boldsymbol{T_{base}^{WP}} \begin{bmatrix} X_V \\ Y_V \\ Z_V \\ 1 \end{bmatrix} \tag{3.27}$$

For rendering the virtual objects, we use the graphic library of OpenGL. The extrinsic parameters of the matrix $\boldsymbol{P^{PI}} \boldsymbol{T_{base}^{WP}}$ is corresponding to the modelview matrix of the OpenGL. Therefore, the 3D virtual objects are rendered on the image by using the modelview matrix frame by frame.

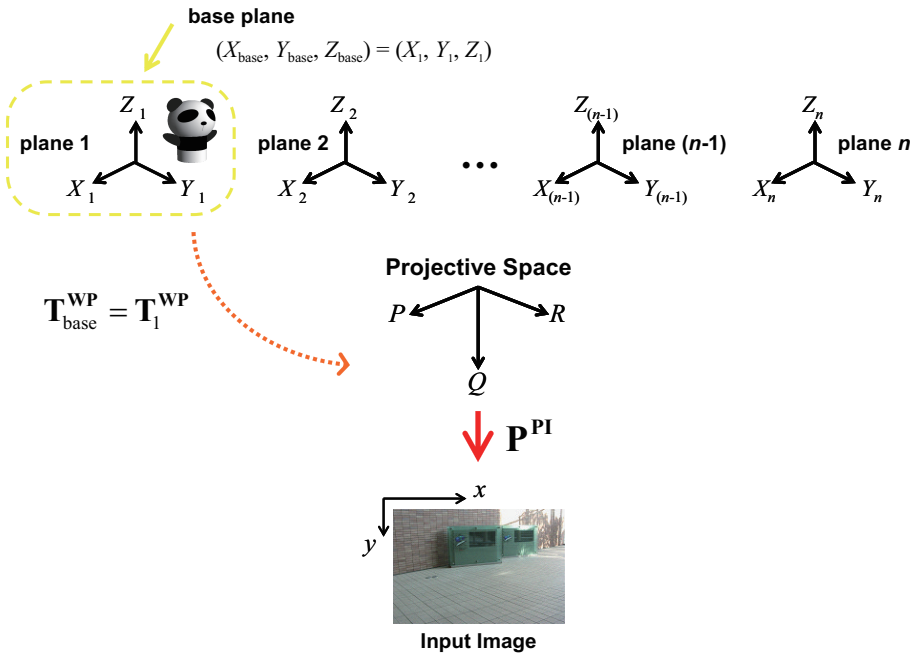


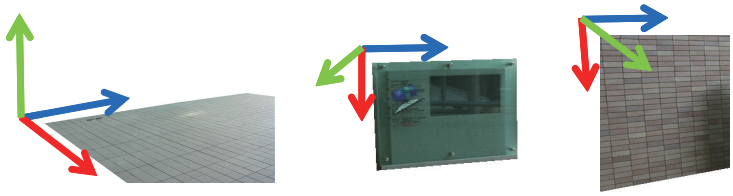
Fig. 3.7 Rendering of virtual object: By using the transformation matrix of the base plane $\boldsymbol{T_{base}^{WP}}$ and the integrated projection matrix $\boldsymbol{P^{PI}}$, the virtual object is transformed to the image coordinate system (x-y).

3.9 Experimental Results

In this section, two experiments are performed and the experimental results are shown to demonstrate the effectiveness of the proposed method. We implemented the AR system based on our method using only a PC (OS: Windows XP, CPU: Intel Pentium Ⅳ 3.20 GHz) and a CCD camera (SONY DCR-TRV900). The resolution of the input image is 720×480 pixels, and the graphical views of virtual objects are rendered using OpenGL.

In these experiments, we utilize the real scene where a lot of planar structures exist. The entrance of our campus and the tabletop in a room are captured by a handheld video camera in the experiment 1 and 2, respectively, as shown in Fig. 3.8(a) and Fig. 3.9(a). In both of the experiments, we use three planes in the object scene as shown in Fig. 3.8(b) and Fig. 3.9(b). Of course the geometrical relationship of the planes is unknown. Therefore, we select two frames from the captured images as the reference images shown in Fig. 3.8(c) and

(a) Input images: Entrance of our campus.



(b) Used planes

(c) Reference images

(d) Virtual object

Fig. 3.8 Experiment 1

Fig. 3.9(c) and construct the Projective Space to estimate the geometrical relationship of the planes. Then virtual panda objects shown in Fig. 3.8(d) and Fig. 3.9(d) are overlaid onto the same position in the real world.

The images produced by our augmentation method are shown in Fig. 3.10 and Fig. 3.11. In the experiment 1 and 2, the virtual objects are overlaid onto the floor plane and the mouse pad plane, respectively. The camera position and pose with respect to the first frame is described in the captions. As shown by these results, our registration method can successfully overlay the virtual object onto the input images based on the camera motion even though the geometrical relationship among the planes is unknown. This is the big advantage of our method comparing to the conventional method. In the next section, we'll discuss the advantage of using multiple planes and present the accuracy of our registration without geometrical relationship information.

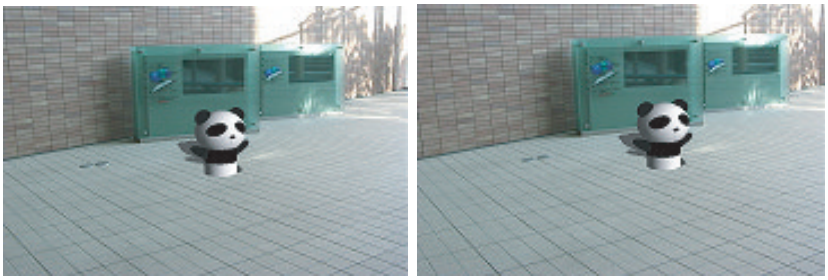
(a) Input images: Tabletop

(b) Used planes

(c) Reference images

(d) Virtual object

Fig. 3.9 Experiment 2



(a) Frame 0
Rot. ($\pm$ 0, $\pm$ 0, $\pm$ 0)
Trans. ($\pm$ 0, $\pm$ 0, $\pm$ 0)

(b) Frame 60
Rot. (-1.33, +0.84, -2.51)
Trans. (+5.29, -0.95, +1.58)

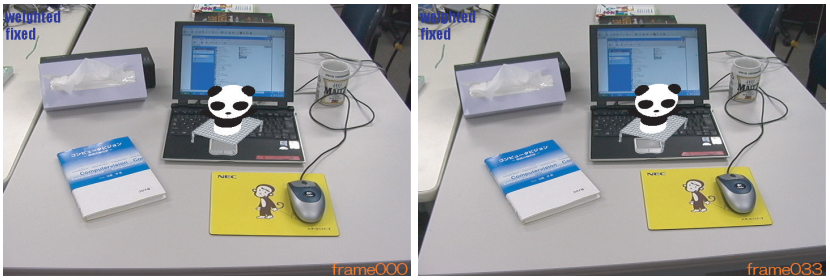
(c) Frame 120
Rot. (-0.18, -0.36, +4.56)
Trans. (-6.51, +1.54, +1.33)

(d) Frame 180
Rot. (-1.82, +8.58, +8.00)
Trans. (-8.11, +4.14, -0.79)

(e) Frame 240
Rot. (-4.07, +11.53, +8.25)
Trans. (-8.68, +4.36, -0.80)

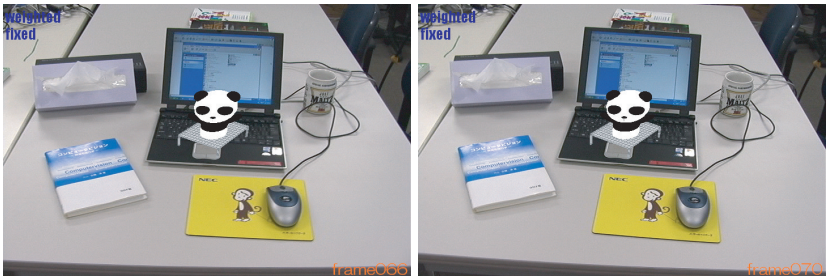
(f) Frame 300
Rot. (-1.90, +4.20, +5.07)
Trans. (-7.19, +4.78, +2.14)

Fig. 3.10 Overlaid image sequence of a virtual object.
Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z).



(a) Frame 0
Rot. ($\pm$ 0, $\pm$ 0, $\pm$ 0)
Trans. ($\pm$ 0, $\pm$ 0, $\pm$ 0)

(b) Frame 33
Rot. (+0.95, +1.14, -5.36)
Trans. (+2.81, -0.33, -0.02)



(c) Frame 66
Rot. (+2.17, +2.97, -0.16)
Trans. (-3.73, +0.29, -0.56)

(d) Frame 70
Rot. (+1.56, +5.59, -0.59)
Trans. (-4.52, -0.02, -1.70)

(e) Frame 74
Rot. (+0.11, +4.40, +2.93)
Trans. (-5.18, -0.31, +0.38)

(f) Frame 99
Rot. (+0.38, +3.12, +8.56)
Trans. (-9.75, -2.39, +3.81)

Fig. 3.11 Overlaid image sequence of a virtual object.
Camera's rotation and translation with respect to Frame 0 are represented as Rot. (X, Y, Z) and Trans. (X, Y, Z).

3.10 Discussion

3.10.1 Accuracy comparison by number of planes used for computation

We evaluate the advantage of using multiple planes comparing with using only a single plane by using synthesized images generated with OpenGL, as shown in Fig. 3.12. In these images, eight planes are distributed at arbitrary positions and poses. Then we compare the registration results of using a single plane and eight planes with the ground truth. In particular, we apply 120 frames to our method and compare the (x-y) position of the overlaid virtual object in the image. In Fig. 3.13, the results of using a single plane, eight plane and the ground truth.

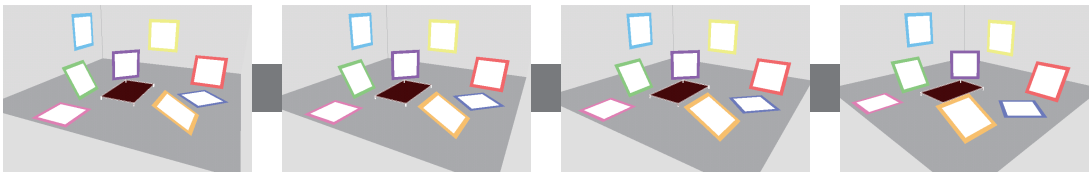


Fig. 3.12 Input images generated by OpenGL: 8 planes are distributed at arbitrary positions and poses. Orange plane is utilized as used single plane.

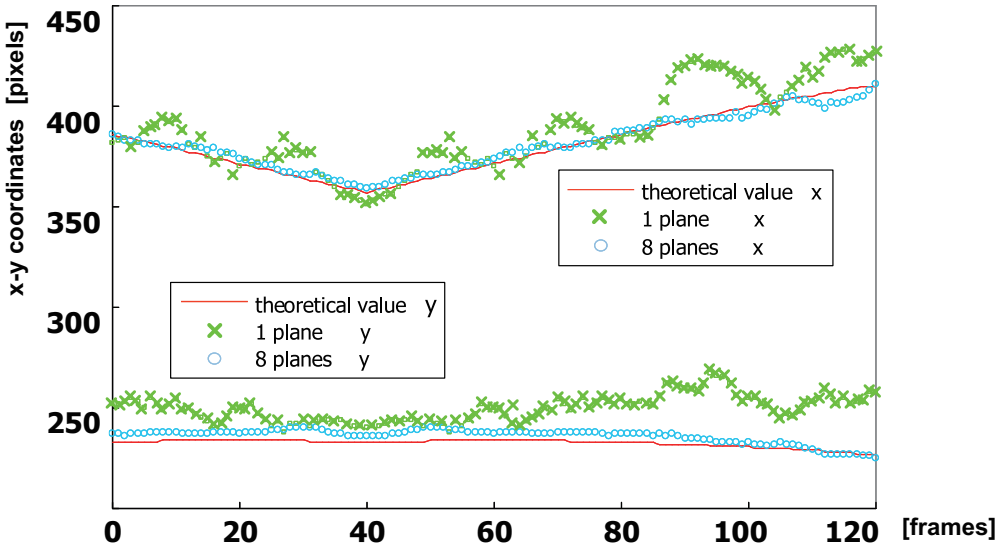


Fig. 3.13 Comparison of x-y coordinates accuracy with theoretical value. Ground truth, related work, and proposed method.

The result of using a single plane causes much registration errors compared to to the ground truth. In contrast, the result of using eight planes can follow the grouhd truth. Therefore we can say that registration accuracy can be improved by increasing the number of the planes.

In Fig. 3.14, we also compare the results of overlaying the virtual object by using a single plane, two planes and three planes. The input images are same as the experiment 2. In using a single plane and two planes, the overlaid position of the virtual object is misaligned according to the camera motion. In contrast, the virtual object is successfully overlaid onto the same position by using three planes. Moreover, the geometrical relationship among the planes can be successfully estimated by our method. Even though we do not require the geometrical relationship or constraints, we can use the multiple planes in the real world. This is big advantage for Augmented Reality.

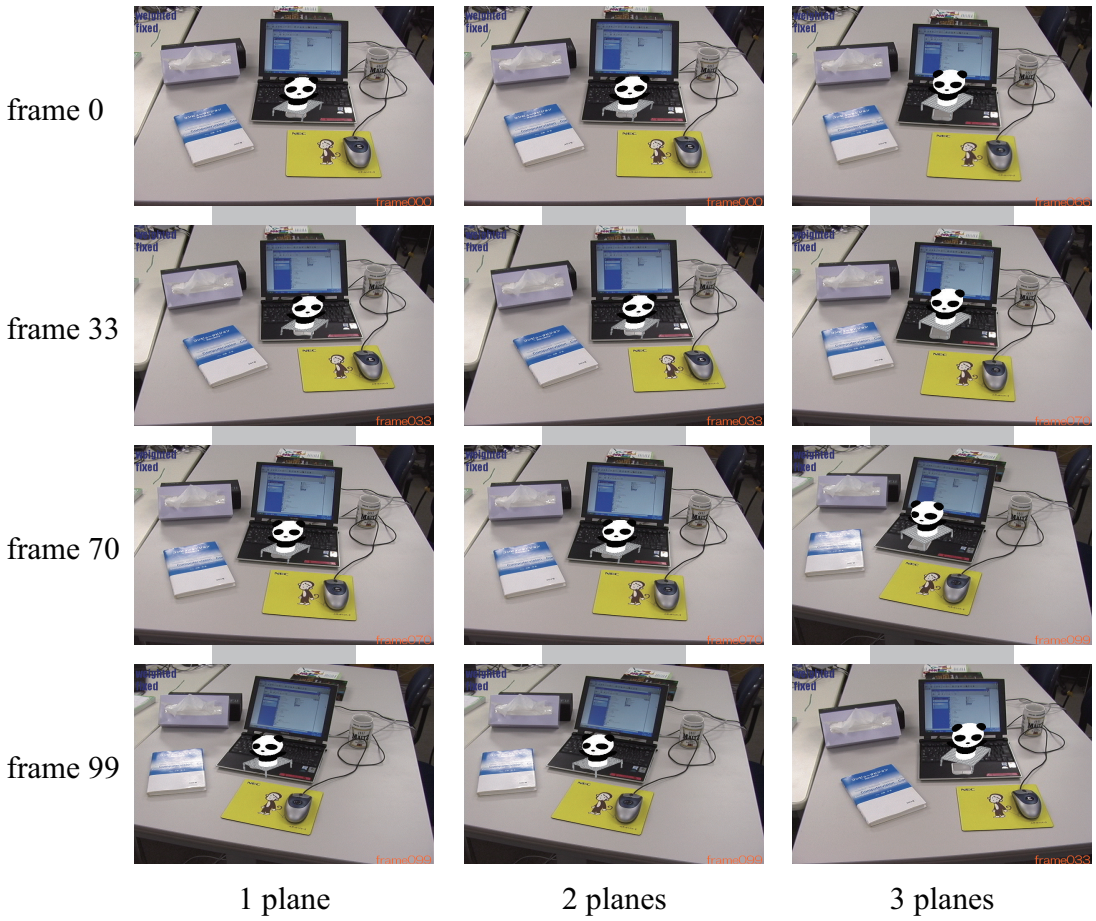


Fig. 3.14 Result images with changing the number of planes: Using 3 planes achieves best results than using a single plane and two planes.

3.10.2 Accuracy comparison with related work

We also evaluate that our method can correctly estimate the geometrical relationship of the planes. For the evaluation, we compare the results of our method with the results of the related work presented by Simon [49]. In Simon's method, multiple planes must be perpendicular to the reference plane (one of the multiple planes). Therefore, we use the image sequence as shown in Fig. 3.15 in which three perpendicular planes and feature points are on the planes and apply our method and Simon's method. In both methods, the camera motion is computed by tracking the feature points on the planes. Then, both registration results (estimated x-y coordinates which are projection of known 3D points) are compared with the ground truth for 40 frames. Simon's method uses the information that the planes are perpendicular each

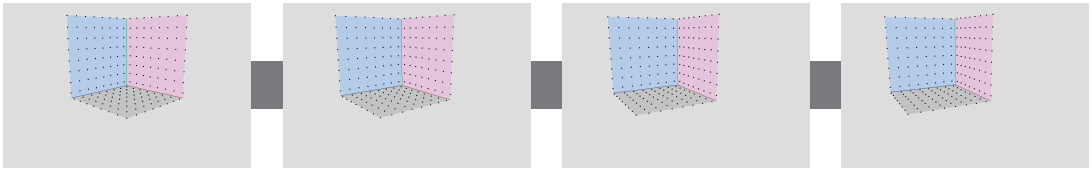


Fig. 3.15 Input images generated by OpenGL: 3 planes are perpendicular to each other.

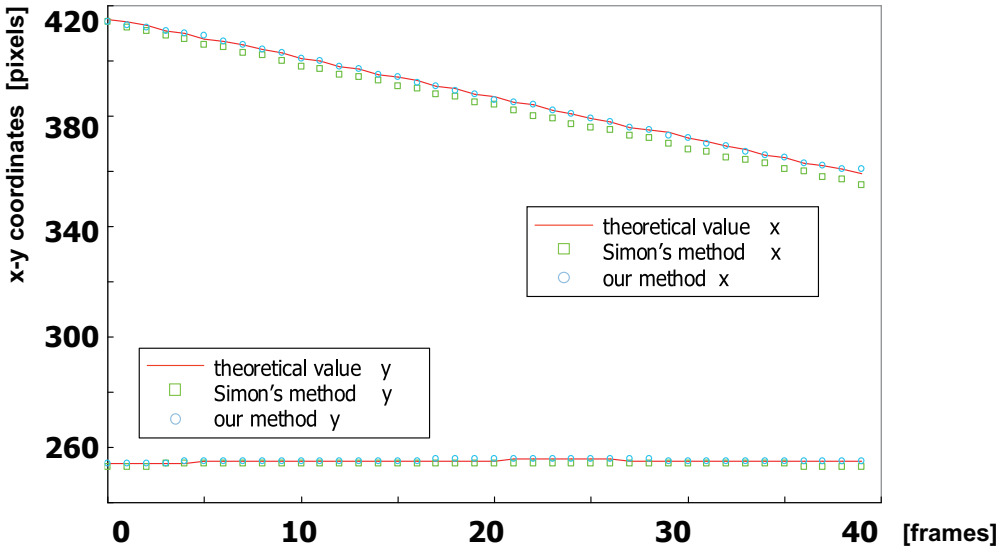


Fig. 3.16 Comparison of x-y coordinates accuracy with theoretical value. Ground truth, related work, and proposed method.

other. Our method does not require the geometrical relationship information.

The results are shown in Fig. 3.16. In Simon's method, the geometrical relationship which the planes are perpendicular is used for estimation. On the other hand, our method does not use the geometrical relationship because we estimate it by constructing the Projective Space. However, our results are very similar to the ground truth and Simon's method even though our method does not use the geometrical relationship information of the planes. Therefore we can say that our method can correctly estimate the geometrical relationship of the planes and estimate the camera motion.

Chapter 4 :

On-line AR System with Planar Pattern Markers

4.1 Outline of On-line AR System

We extend the proposed method to an on-line AR application that uses multiple planar markers [66]. Multiple markers can be distributed in the real scene without manual measurement or special devices (ex. a high resolution camera), which contrasts with related works [54, 55, 67, 68]. Therefore, a user can freely place the multiple markers in a wide area and move around inside it. The system can be implemented in a small space, such as a tabletop, and in a large space such, as a room.

User activities are listed as follows. First, the user places multiple markers in the real scene and captures the scene for a few seconds with a video camera. Then, the system automatically selects the two reference images, which are used to define a Projective Space from the captured images, as described in Sec. 3.4. After it selects the images, the system uses them to define the Projective Space.

This is all done automatically. When the Projective Space is ready, the system proceeds to the augmentation process. The user freely moves the camera to favorite viewpoints and watches through a display as virtual objects stay in place or move around the real scene.

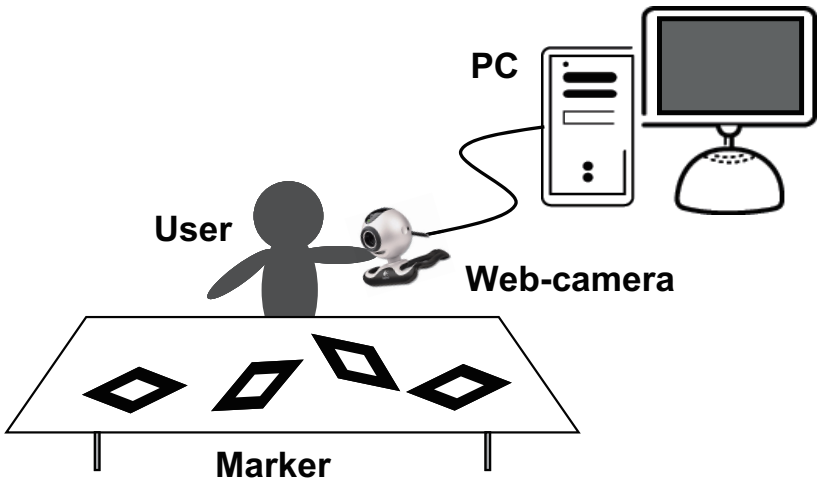


Fig. 4.1 On-line AR System: The system consists of a PC and a web-camera. Multiple markers are freely distributed in the real world.

4.2 User's Operation and Flow of System

In this section, the user's operation and the flow of processing in the system are explained. The operation of this system is divided two stages.

- 1st stage：Constructing Projective Space
- 2nd stage：Generating Augmented Reality View

At the 1st stage, two reference images which are used for defining the Projective Space are automatically selected, and then the geometrical relationship of the markers is estimated. The user only has to place the markers freely and capture the target scene for some frames. The captured images will be the candidates of the reference images.

At the 2nd stage, the input image sequence from the camera is applied to the proposed method, and then the result images where the virtual object is overlaid are output. User's operation is only moving the camera.

All the processes in both of 1st and 2nd stages can be performed on-line. The processing speed achieves video-rate except that the selection of the reference images takes about 1 minute.

4.2.1 First Stage: Constructing Projective Space

Firstly, some markers are arranged at arbitrary positions and poses in the object scene. Next, we start capturing the object scene and computing $\boldsymbol{P_i^{WI}}$ for each marker frame by frame. Then, we make some frames of the captured sequence candidates for two reference images. In this experiment, we use 100 frames as the candidates. In selection process, one reasonable combination of the reference images are selected from the candidate images automatically as shown in Sec. 4.3. Using the two reference images, a Projective Space is constructed and each $\boldsymbol{T_i^{WP}}$ is computed by the algorithm in Sec. 3.5. When the Projective Space is prepared, the system goes to 2nd stage.

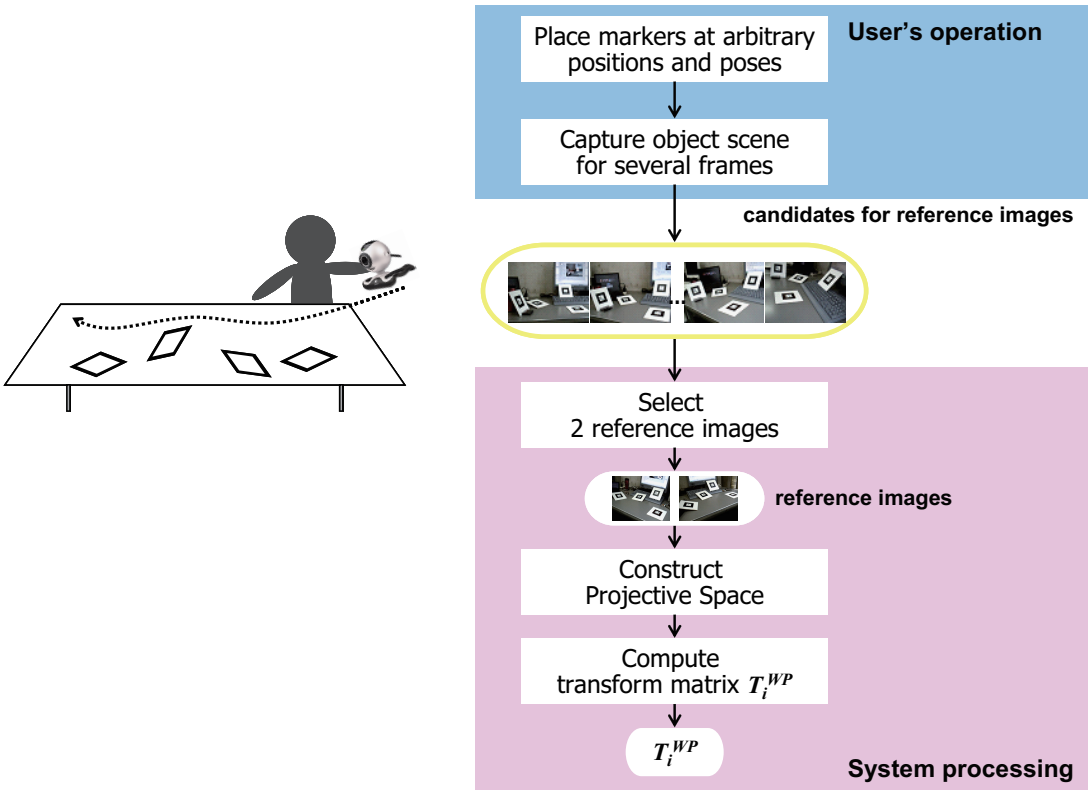


Fig. 4.2 Flow of 1st stage.

4.2.2 Second Stage: Generating Augmented Reality View

After constructing the Projective Space by selected the reference images and computing the transformation matrix $\boldsymbol{T_i^{WP}}$ which represents the relationship of the markers, the system goes to 2nd stage.

At 2nd stage, the projection matrix which relates each marker to the image is computed for the input image sequence from the video camera. Next $\boldsymbol{P_i^{PI}}$ is computed by using $\boldsymbol{T_i^{WP}}$ which is computed at 1st stage, and integrated into one matrix. Then, the virtual object is rendered onto the input image.

Since this system can estimate the relative relationship of the markers by the Projective Space, the virtual object can be overlaid onto the unique position and coordinate even if any marker is captured by the camera. Therefore the user can freely move the camera and see the augmented view from the favorite viewpoint. In addition, the registration errors are reduced when multiple markers are captured in the field of view, because all information of the markers is integrated by least square method.

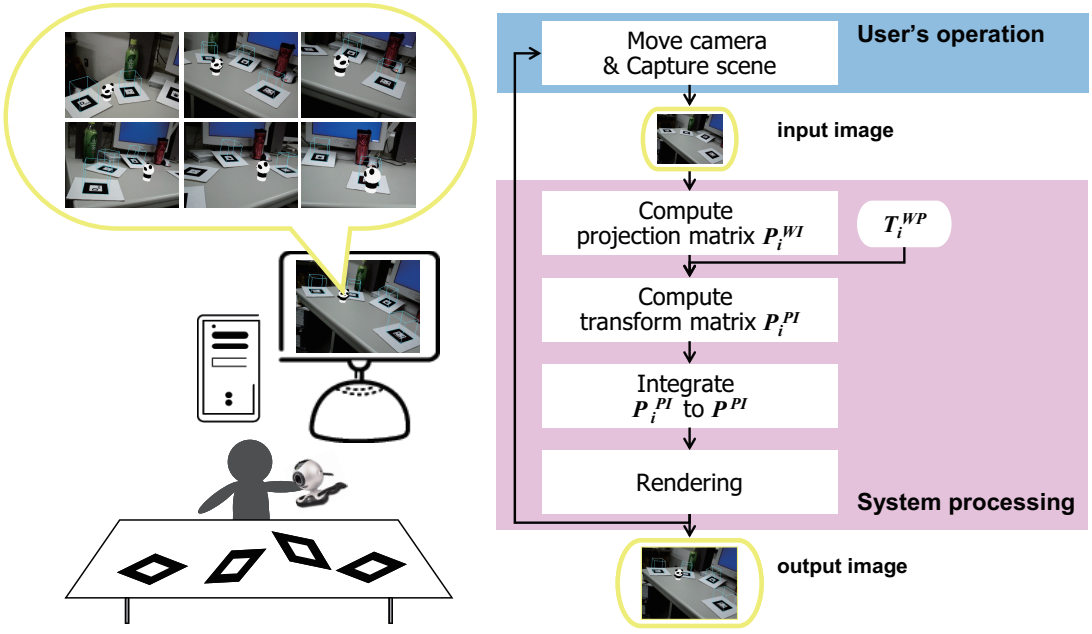


Fig. 4.3 Flow of 2nd stage.

4.3 Automatic Selection of Reference Images

As described in Sec. 3.4, the Projective Space is defined by projective reconstruction of two reference images. In projective reconstruction, a fundamental matrix between the two reference images is used for defining the Projective Space. This means that it is very important issue to compute an accurate fundamental matrix between the reference images. In this method, therefore, a reasonable pair of images as the reference images should be selected. This method introduces automatic selection algorithm of the reference images.

The overview is shown in Fig. 4.4. First, the object scene where multiple markers are distributed is captured for a few seconds by a moving camera. This image sequence in which all the markers should be included becomes the candidates of the reference image. In particular, the user captures the markers from left and right side of the scene. If the candidate images

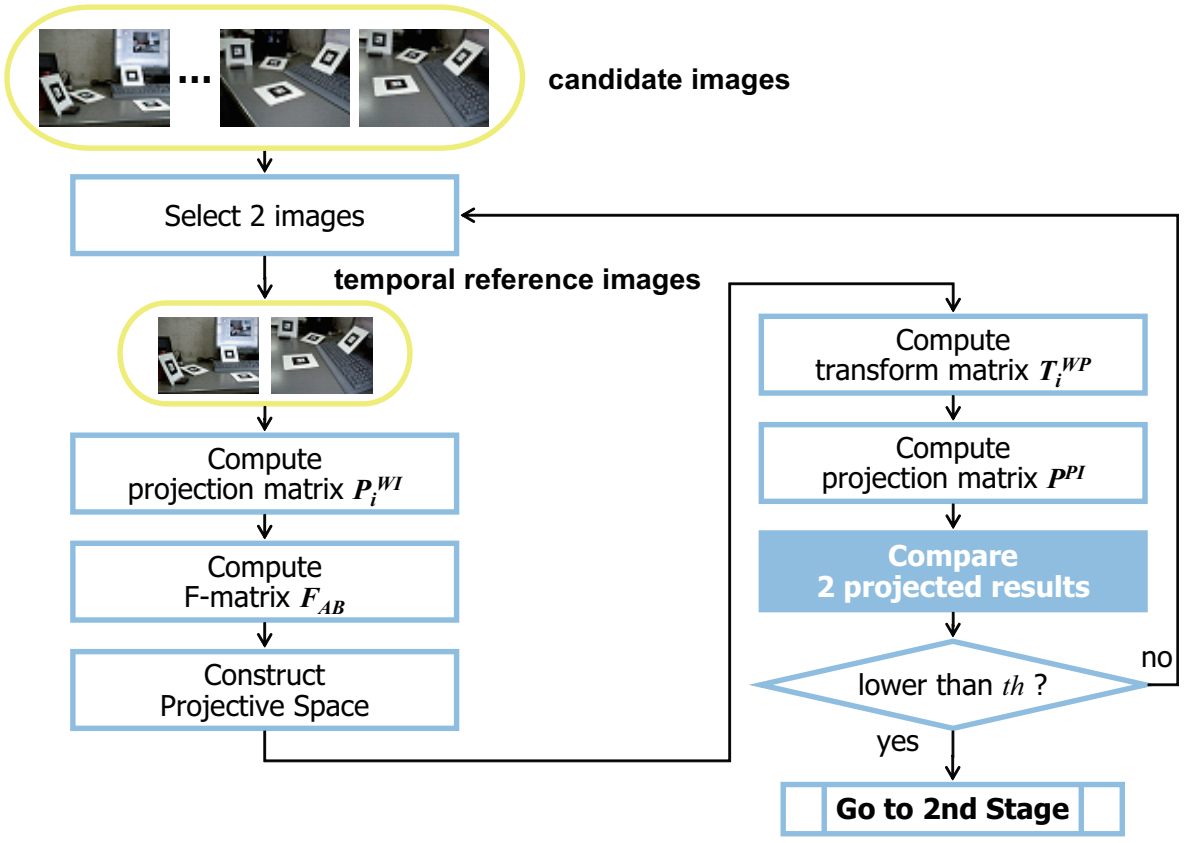


Fig. 4.4 Flow of automatic selection of reference images.

are captured for 100 frames, therefore, 50 frames are captured from left side of the scene, the other 50 frames are captured from right side. From the candidate images, a pair of images is sequentially selected as temporal reference image A and B. In particular, one of the pair is selected from the 50 frames of left side, the other one is selected from the 50 frames of right side. Therefore, the system tries 50×50 combination. By projective reconstruction of the pair of temporal reference images, a temporal Projective Space is defined. Then a good pair is selected by evaluating the accuracy of the Projective Space. The detail is shown in Fig. 4.5.

For evaluating the temporal reference images and Projective Space, first, a fundamental matrix is obtained by projection matrices computed from the markers in the temporal reference images using ARToolkit algorithm [52]. Where $\boldsymbol{P}_{\boldsymbol{A}_i}^{\boldsymbol{WI}}$ and $\boldsymbol{P}_{\boldsymbol{B}_i}^{\boldsymbol{WI}}$ are the projection matrix computed from marker i in the temporal reference image A and B, respectively. Then a fundamental matrix based on the marker i is computed as following equation.

$$\boldsymbol{F}_{\boldsymbol{AB}_i} = [\boldsymbol{e}_{\boldsymbol{B}_i}]_{\times} \boldsymbol{P}_{\boldsymbol{B}_i}^{\boldsymbol{WI}} \boldsymbol{P}_{\boldsymbol{A}_i}^{\boldsymbol{WI}-} \tag{4.1}$$

where, $[\boldsymbol{e}_{\boldsymbol{B}_i}]_{\times}$ is also the skew-symmetric matrix and $\boldsymbol{P}_{\boldsymbol{A}_i}^{\boldsymbol{WI}-}$ represents the pseudo inverse matrix of $\boldsymbol{P}_{\boldsymbol{A}_i}^{\boldsymbol{WI}}$ [65]. When n fundamental matrices are computed, we select one matrix which has the smallest error value $error$,

$$error = \boldsymbol{m}_{\boldsymbol{B}}{}^{\top} \boldsymbol{F}_{\boldsymbol{AB}_i} \boldsymbol{m}_{\boldsymbol{A}} \tag{4.2}$$

where, $\boldsymbol{m}_{\boldsymbol{A}}$ and $\boldsymbol{m}_{\boldsymbol{B}}$ are corresponding points in each reference image.

Decided one fundamental matrix, a temporal Projective Space is defined by eq. (3.3) and eq. (3.4). Then $\boldsymbol{T}_{\boldsymbol{i}}^{\boldsymbol{WP}}$ and $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{PI}}$ are computed for each maker via the temporal Projective Space. Additionally, all $\boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{PI}}$ are integrated into one $\boldsymbol{P}^{\boldsymbol{PI}}$ by eq. (3.24). Then two cubes are projected on each marker by using the projection matrix $\boldsymbol{P}_{\boldsymbol{B}_i}^{\boldsymbol{WI}}$ and the integrated projection matrix, respectively. The projected cubes' coordinates are respresented as $\boldsymbol{x}$ and $\boldsymbol{x'}$ as shown in eq. (4.3) and eq. (4.4).

$$\boldsymbol{x}_{\boldsymbol{i}} = \boldsymbol{P}_{\boldsymbol{i}}^{\boldsymbol{WI}} \boldsymbol{X}_{\boldsymbol{W}} \tag{4.3}$$

$$\boldsymbol{x}_{\boldsymbol{i}}' = \boldsymbol{P}^{\boldsymbol{PI}} \, \boldsymbol{T}_{\boldsymbol{i}}^{\boldsymbol{WP}} \, \boldsymbol{X}_{\boldsymbol{W}} \tag{4.4}$$

In Fig. 4.5, the cube projected by each marker is represented by red cube and the cube projected by the integrated projection matrix is represented by yellow cube. Although these two cubes should be overlapped, if the combination of the two reference images is not reasonable, they can not be overlapped. Therefore the system selects one pair in which the difference of cubes $\boldsymbol{x_i} - \boldsymbol{x'_i}$ is smaller than threshould th. In particular, the system iterates these selecting and comparing processes and selects the good pair. We let th 3 pixels.

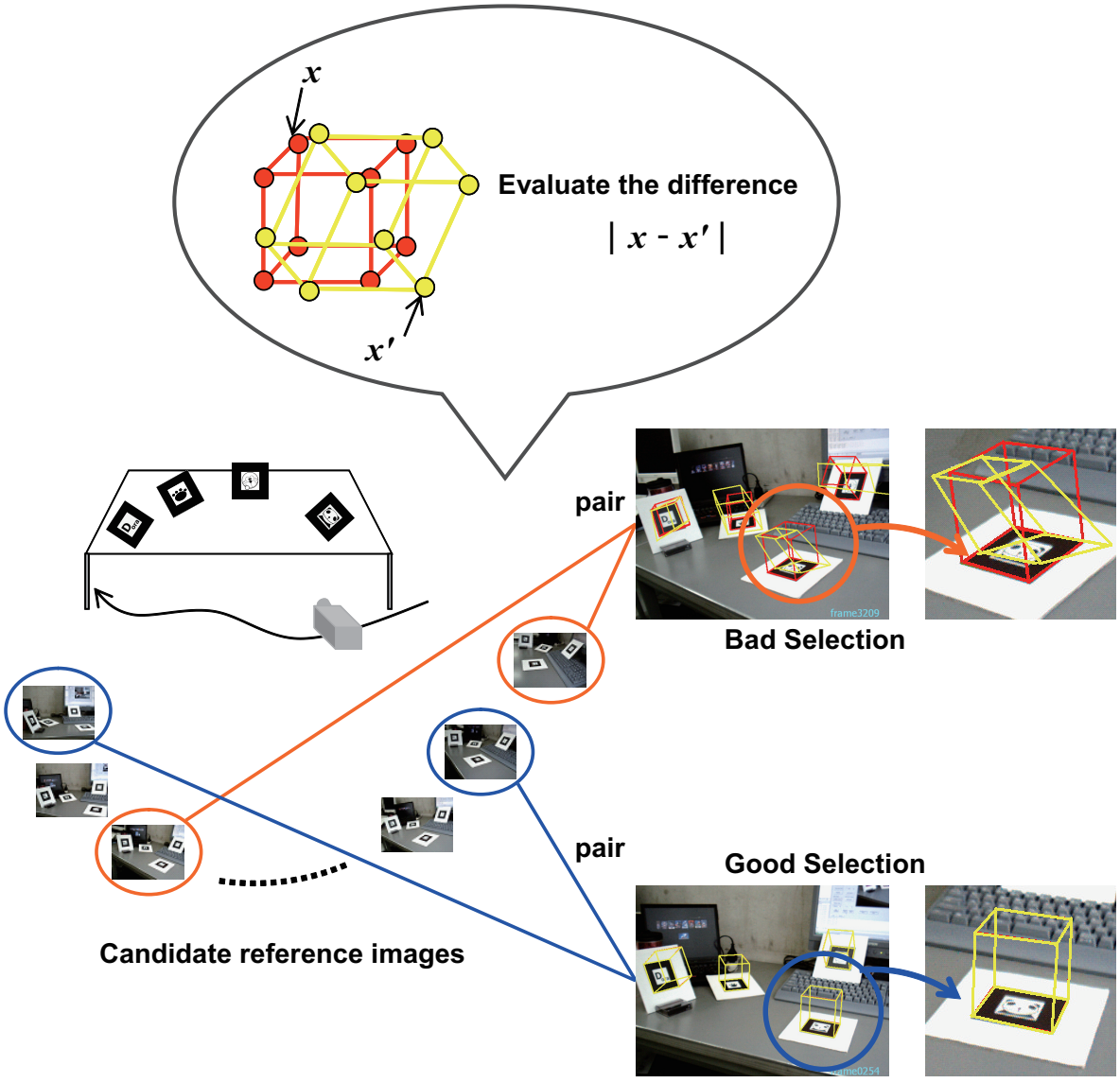


Fig. 4.5 Selection of reference images. By comparing difference of the projected cubes $\boldsymbol{x}$ and $\boldsymbol{x'}$, the pair whose difference is smaller than threshold th is selected. (th=3 pixel).

4.4 Effective Marker Selection for Integration

In our system, a 3D coordinate system is defined on the marker so that the marker plane consists of X-Y plane and Z axis is perpendicular to the plane. Therefore our system detects the marker's contour from the input images frame by frame and defines Z axis by the normal vector. However, the definition of the axis becomes unstable depending on the camera angle with respect to the marker.

Fig. 4.6 shows an image sequence (three frames) captured from similar viewpoints. Images in the upper row and lower row are results of defining 3D coordinate systems on the markers and overlaid images of the virtual objects, respectively. In frame 2, Z axis defined on the lower-right marker is not correct in contrast with previous and next frames, even though the marker is completely visible in the image. This means that the extrinsic parameters are not correctly computed from the marker. Therefore it fails in overlaying of the virtual object onto the correct position, if the parameters computed from all the markers are integrated. This problem is caused by the planar structure of the marker or reflection of illuminations.

In off-line approaches, it is possible to optimize the tracking error by iterative calculation, such as the steepest descent method. However our system should avoid such iterative calculation because it is too time-consuming to be applied to on-line system. Therefore our system

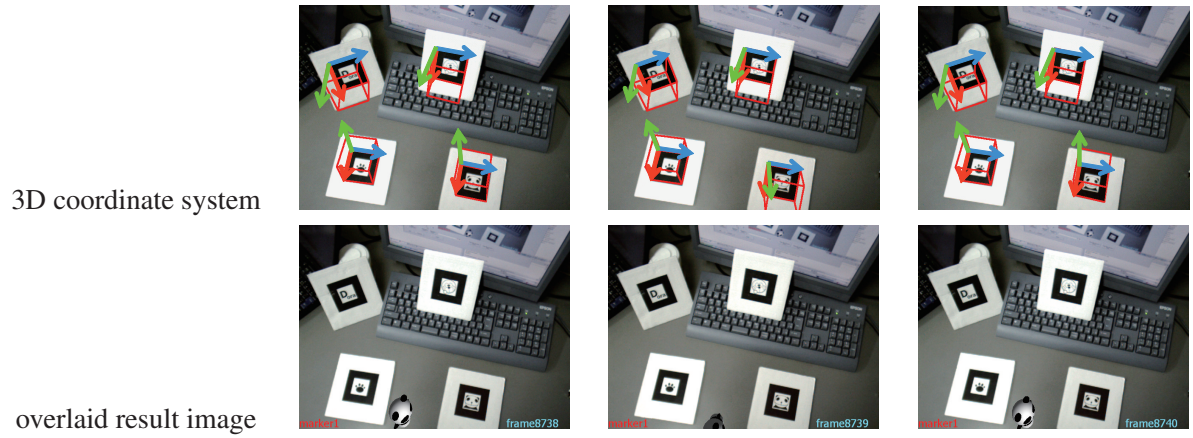


Fig. 4.6 Example scene of failure in definition of Z axis. These 3 frames are captured from similar viewpoints. Even though the marker is visible in frame 2, Z axis of lower-right marker is defined with different direction from previous and next frames.

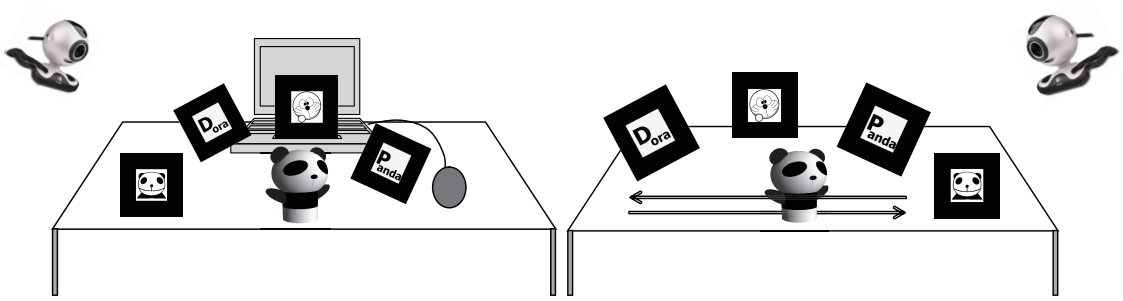
automatically rejects the bad marker. First, $\boldsymbol{P_i^{PI}}$ is computed from every marker which is detected in the image and integrated into $\boldsymbol{P^{PI}}$. If the projection results are not correct, one of the markers is rejected. Then $\boldsymbol{P_i^{PI}}$ is computed from all markers but the rejected marker and integrated into $\boldsymbol{P^{PI}}$ again. The criteria of projection results is the same as the algorithm described in Sec. 4.3 by comparing eq. (4.3) and eq. (4.4). In this way, our system avoids increase of computation time by iterative calculation.

4.5 Experimental Results

By using our system, three experiments are performed. In these experiments, four markers are distributed at arbitrary positions and poses in the real world as shown in Fig. 4.7.

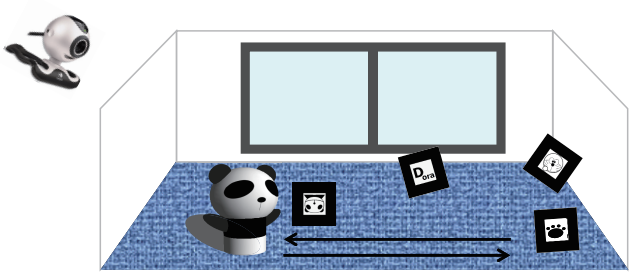
In the first experiment, a static virtual object is overlaid onto the same position on the tabletop. The camera captures the target scene from various viewpoints. In the second experiment, a virtual object moves on the tabletop and the camera follows the moving virtual object. In the third experiment, a human-sized virtual object is overlaid and moves around the room so that the users feel that they have a virtual pet in their house.

The result images of three experiments are shown in Fig. 4.8, Fig. 4.9 and Fig. 4.10, respectively. In Fig. 4.8, the virtual object can be stably aligned at accurate position and pose, although different markers are visible according to the camera motion. This means that the consistency of the geometrical relationship between the camera and the virtual object is kept



(a) Experiment 1: Static virtual object.

(b) Experiment 2: Moving virtual object.



(c) Experiment 3: Human-sized virtual object.

Fig. 4.7 Object scene: 4 markers are distributed in the real world.

properly, since the multiple markers' 3D coordinate systems can be integrated successfully even though the geometrical arrangement among them is unknown.

In Fig. 4.9, the virtual object is successfully overlaid onto the table with keeping the unique coordinate system even though the camera chases the moving virtual object and captures different markers. In the related works, the task of measuring all the markers is very time-consuming when the markers placed in the large space. However, our system can be easily applied to such a situation because of automatic estimation of the markers' relationship.

In Fig. 4.10, the human-sized virtual object can be overlaid in the room with a human. For applying multiple markers based registration in such a large space, it becomes more difficult to manually measure the arrangement of the markers. Therefore this method is useful because of estimating the arrangement only by capturing the object scene.

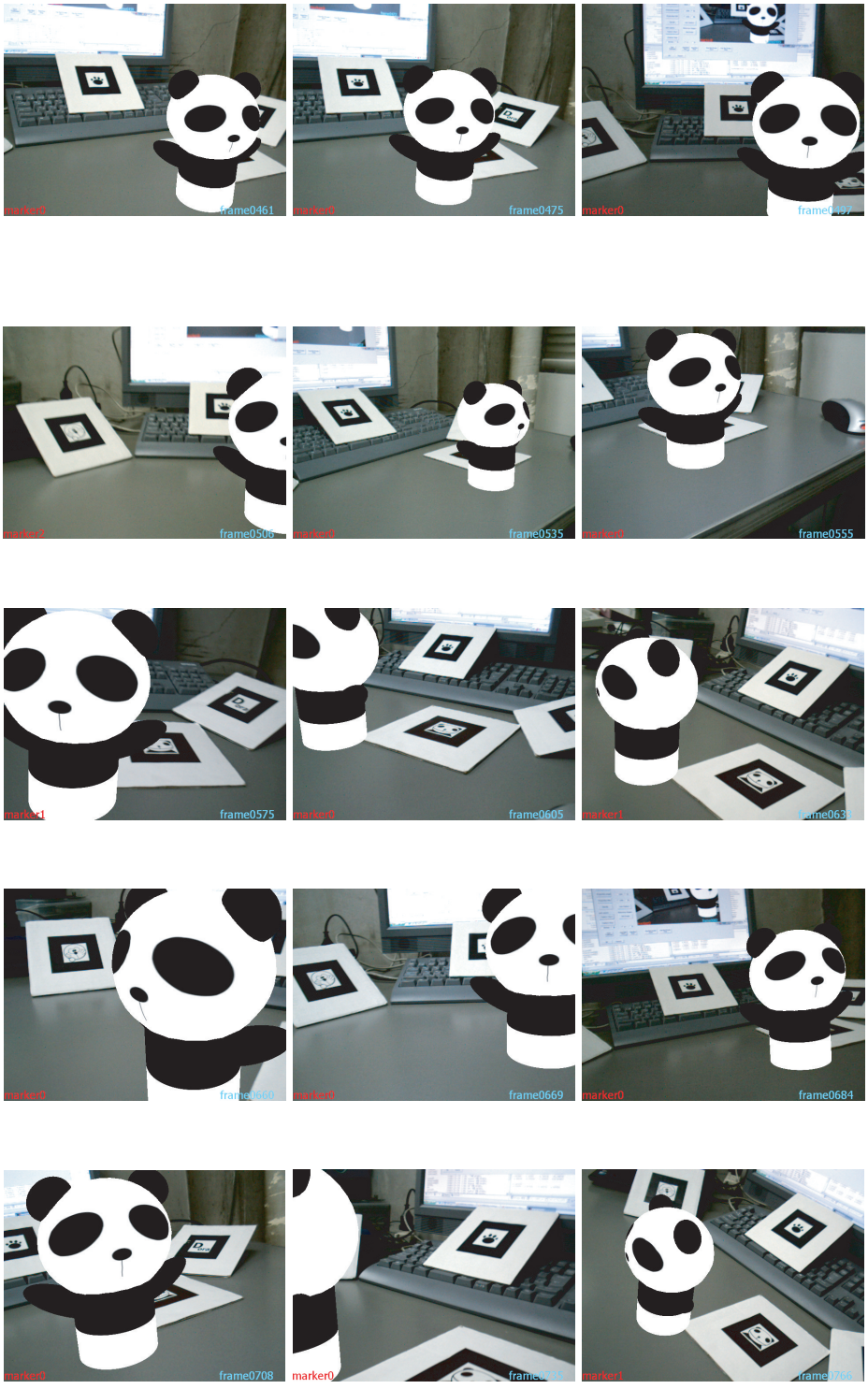


Fig. 4.8 Result images of overlaying static virtual object

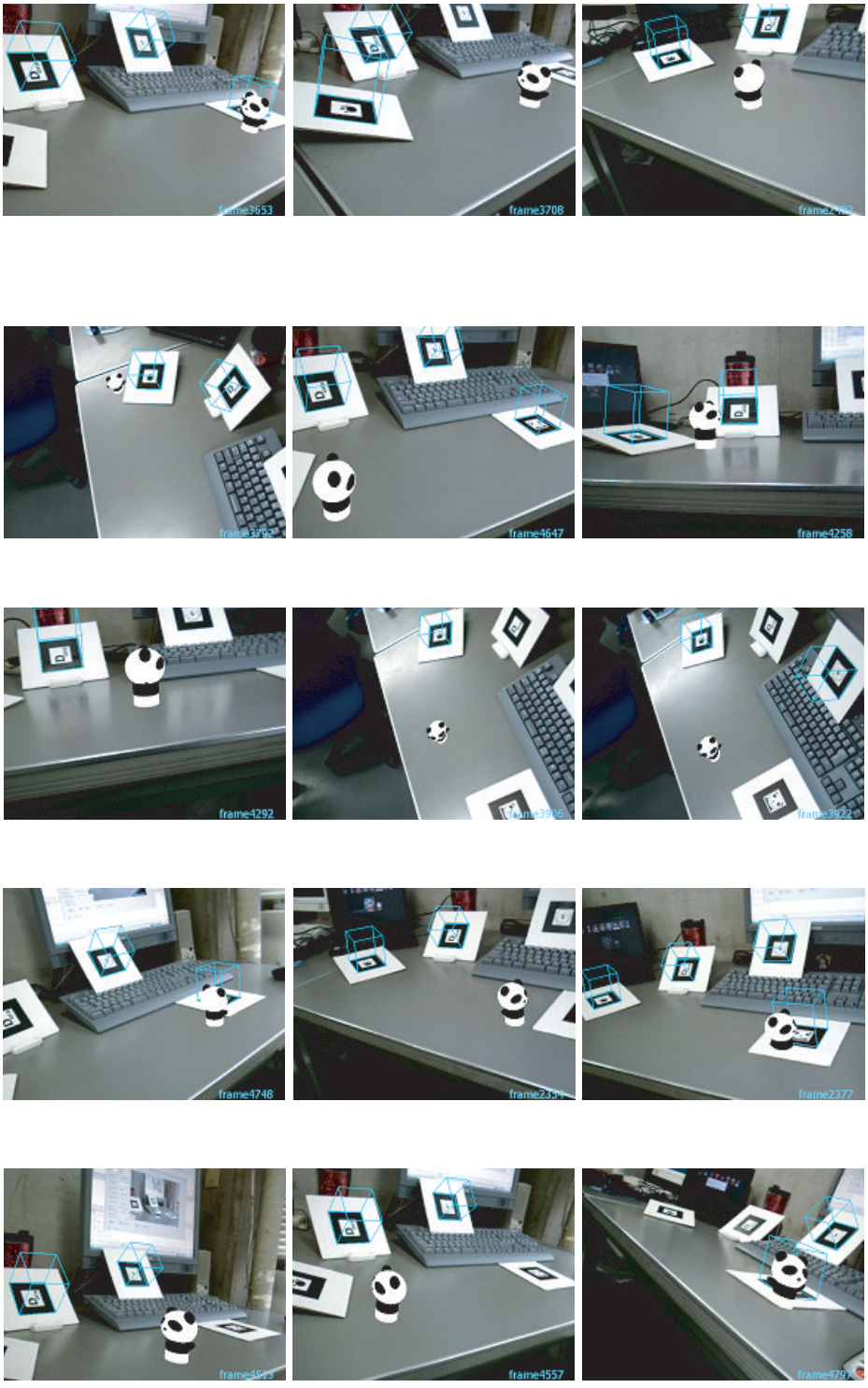


Fig. 4.9 Result images of overlaying moving virtual object

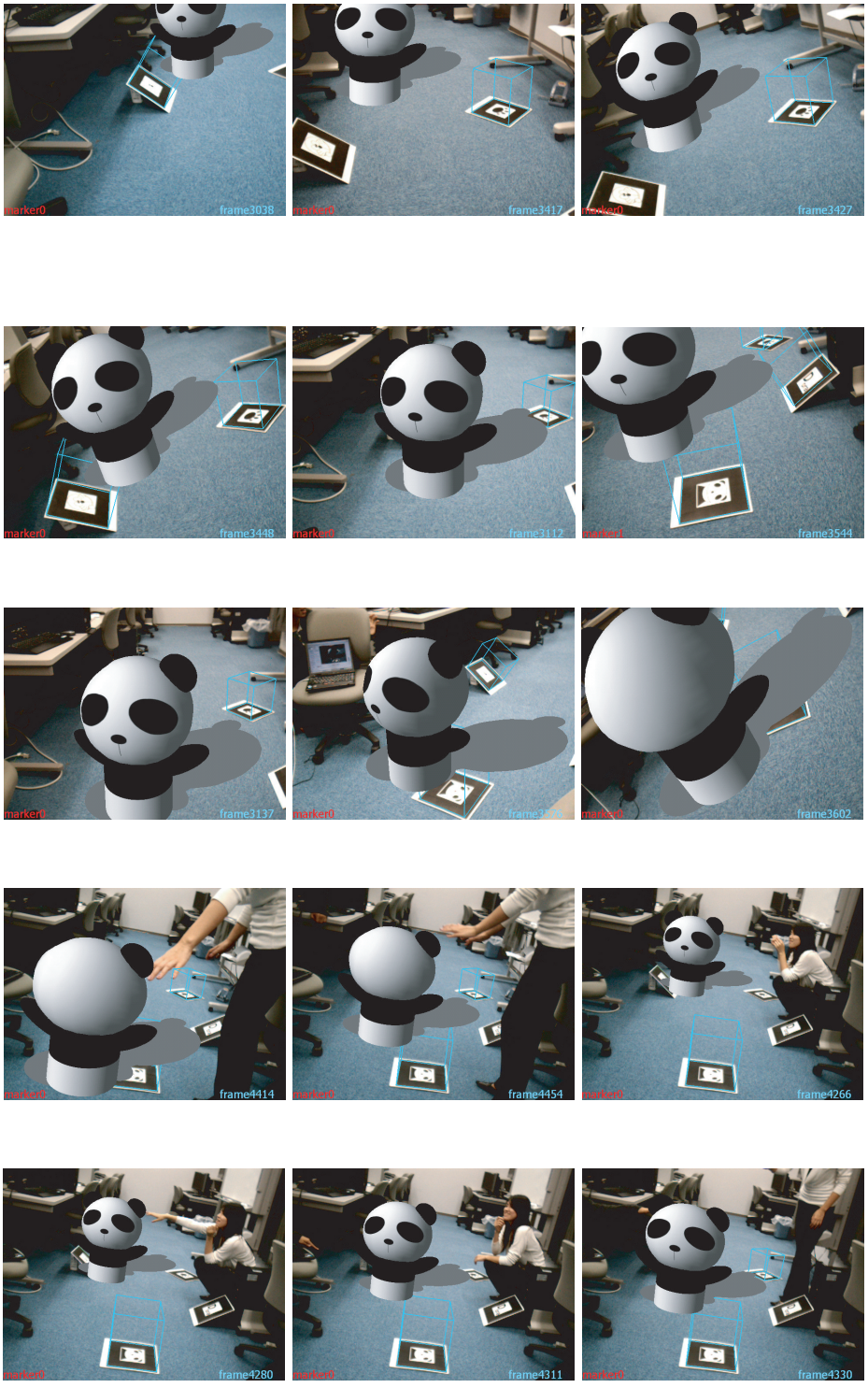


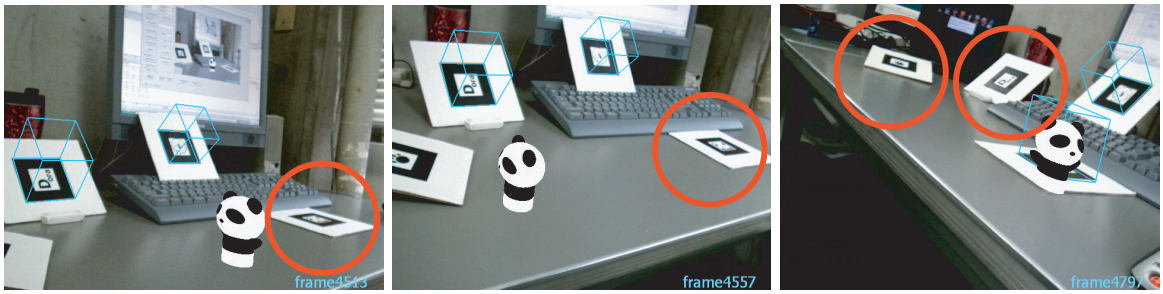
Fig. 4.10 Result images of overlaying moving human-sized virtual object

4.6 Effectiveness of Distributing Markers

Our registration method focuses on the planar structures in the real world for the registration. The planes can impose reasonable geometrical constraints on the feature points and frequently-used geometrical information in Computer Vision. However, appearance of the plane changes depending on the camera angle. In addition, as described in Sec. 4.4, tracking of Z axis (perpendicular to the marker plane) becomes unstable when the camera's optical axis is perpendicular to the marker plane. Therefore the accuracy of the plane detection depends on the camera angle.

For example, Fig. 4.11 shows the results of detecting the markers and overlaying the virtual object. The blue cube on the marker represents that the marker can be detected in the frame. In these frames, some markers are undetected even though the markers are visible. This is because the camera angle with respect to each marker plane is too small for the system to detect the marker in contrast with human's eye. Reflection of illuminations also causes such detection error. If all the markers are placed on the tabletop and face the same directions such as conventional applications which utilize regularly-located markers, all of them are undetected at the same time because of the small camera angle. In such a case, the registration can not be continued. Therefore the effectiveness of using multiple markers is decreased.

On the other hand, our system allows the markers to be distributed at arbitrary positions



(a) (b) (c)

Fig. 4.11 Examples of failure in marker detection. Some of the markers can not be detected even though the markers are visible in the images. On the other hand, the other markers which are different directions with respect to the camera are successfully detected.

and pose in the real world. This means that the markers can face various directions. Therefore the registration is successfully continued as shown in Fig. 4.11. This is a big contribution and advantage of our system for developing AR applications, because the markers become detectable even if they are captured from various viewpoints. In contrast with conventional applications which can not use such freely distributed markers, our system achieved to use multiple planes which are complicated structures without manual measuring task. Therefore we can say that our system is suitable for plane-based registration.

Chapter 5 :

AR Applications based on On-line AR System

5.1 Overview of Applications

In this section, we present two AR applications: AR Baseball Presentation System; Interactive AR Bowling System [69]. Users use these applications on the tabletop in the real world by using a web-camera attached to a handheld monitor.

AR Baseball Presentation System is an observation system of a virtual baseball game. Users place a real baseball field model on the tabletop and input a baseball game history (scorebook) that they want to watch into the system. Then they can watch the game by replaying with virtual baseball players on the field model in front of them. On the field model, 2D markers are placed for registration of the virtual players. Therefore the users can watch the game from their favorite viewpoints around the field.

With Interactive AR Bowling System, users can enjoy the bowling game by rolling a real ball down a real bowling lane model placed on a tabletop in the real world. On the lane model, there are virtual pins generated with CG. They knock down the virtual pins by rolling the real ball. Touching and rolling the real ball provide a sort of tangible feeling in this system. It is well-known that a tangible interface enhances the reality of communication [70, 71, 72, 73]. Because of placing some markers on the lane model, the users can watch the lane and pins from free viewpoints.

For registration of virtual objects such as the virtual players or the virtual pins, the motion of the user's camera is estimated by multiple 2D markers. In our applications, the multiple markers can be arranged at arbitrary positions and directions in the real world. Therefore the users can start our applications only with free-arranging the markers. These applications are based on the multiple marker-based On-line AR system [74].

Fig. 5.1 Handheld monitor and web-camera

5.2 AR Baseball Presentation System

This system visually replays the baseball game which was previously played in the other place. In contrast with the usual way to know the game history, such as watching the captured video or reading the recorded scorebook, this AR system can provide the users with much realistic sensation as an entertainment application.

Using this system, the user can watch the game from the favorite viewpoint by just moving around the field model, because multiple planar markers are placed on the field model. Such simple way for watching the CG-represented event using the AR system provides more immersive feeling than using usual CG viewers, in which a mouse or a keyboard is used for changing viewpoints.

This system focuses on visualizing the game by using a scorebook data, so that the user can understand the main game point. Thus the detail part of the game such as player's gesture is not replayed. In this system, the virtual players are generated as a cartoon character instead of

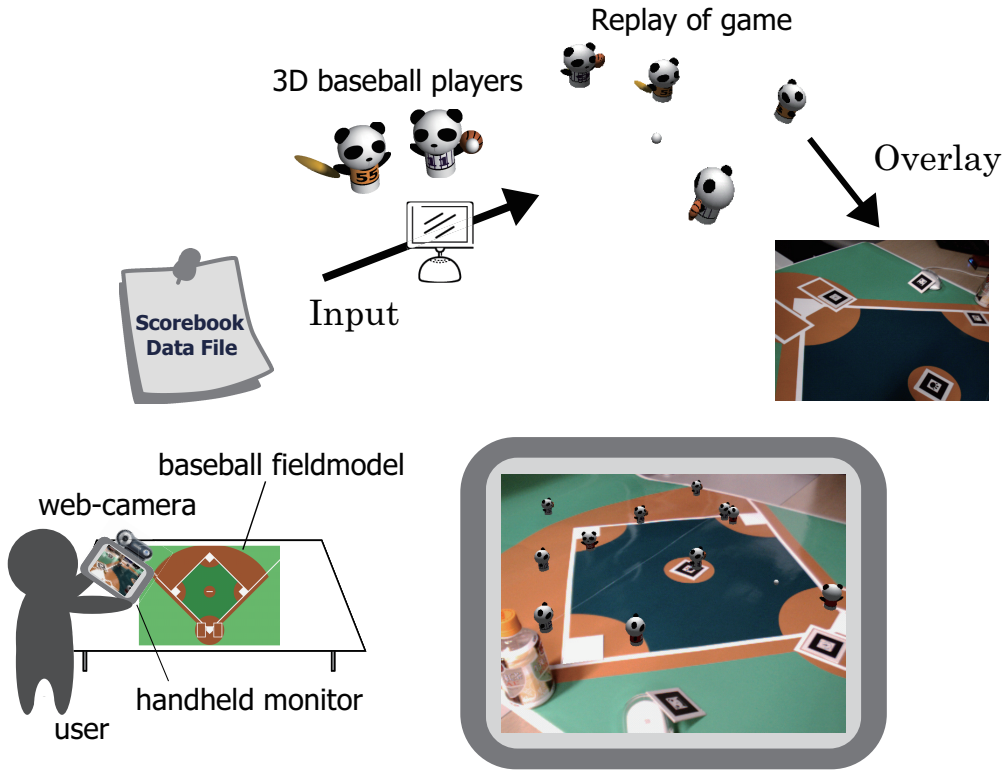


Fig. 5.2 AR Baseball Presentation System

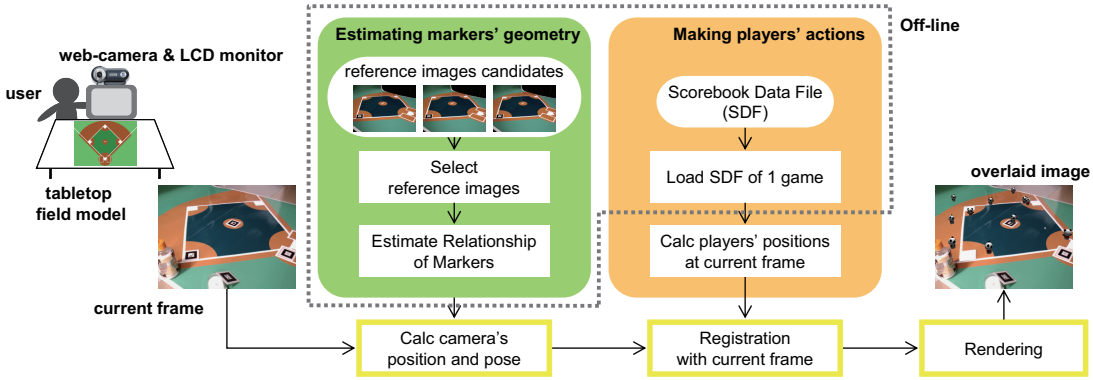


Fig. 5.3 Overview of AR Baseball Presentation System.

a human player. There is a limit even if a human player is made in great detail by CG. Increase in the number of polygons in CG is also big problem for real-time processing. Therefore we decide to use a funny and friendly character.

5.2.1 Processing Outline

Fig. 5.3 shows overview of the system. Multiple 2D markers are distributed inside and outside of the baseball field model which is placed on the tabletop in the real world. The markers can be placed at arbitrary positions and poses without measuring the arrangement of them. The image of the tabletop field model is captured by a web-camera attached and displayed on a handheld monitor.

This system can be divided into off-line and on-line process. At the off-line process, first, a game history data file of a baseball game is prepared and loaded. In this file, history of game results are described play-by-play. Next, the field model is captured by the moving web-camera for some seconds to automatically estimate the markers' arrangement. The detail of the algorithm is described in Chapter 3. These processes are executed once in advance.

At the on-line process, the three steps are repeated on-line: (1) synthesizing the baseball game scene while 1 play according the input data, (2) computing the camera's position and pose at the current frame, and (3) overlaying virtual players onto the field model. At the first step, when one line of the data file is read out, the positions of the players and the ball at every frame while 1 play are computed according to the data to render them on the field model. At

the second step, the camera's rotation and translation are estimated using the markers in the current frame. At the final step, the virtual baseball scene, such as the players and the ball synthesized with CG, is overlaid onto the tabletop field model.

5.2.2 Input Scorebook Datafile

The game played on the field model is the replayed game of the actual game which is synthesized according to input data file called "Scorebook Data File" (SDF). As shown in Fig. 5.4, the game history of the actual game is described play-by-play in the SDF. "1 play" means the actions of the players and the ball from the moment that the pitcher throws the ball to the moment that the ball returns to the pitcher again. It is about for 15 to 30 seconds. The actions of the players and the ball in 1 play are described on one line in the SDF. The former part of the line represents the actions of the fielders and the ball, while the latter part describes the actions of the offensive players. This file is loaded in starting the system and is sequentially read out line-by-line at every 1 play. In this way, the actions of the baseball scene are described in the SDF.

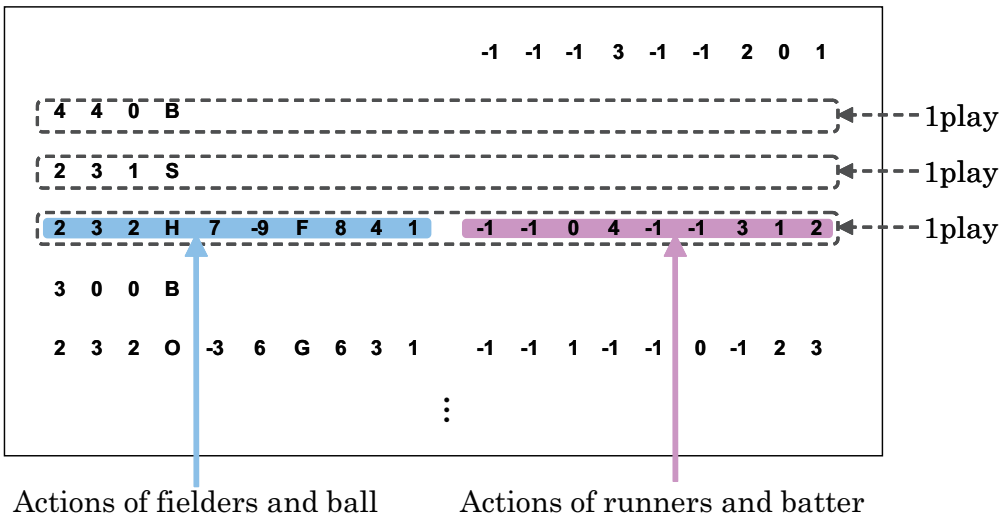
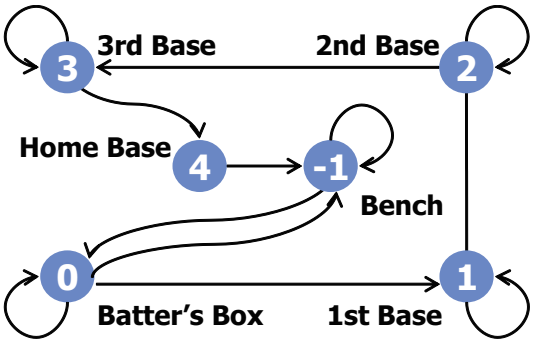


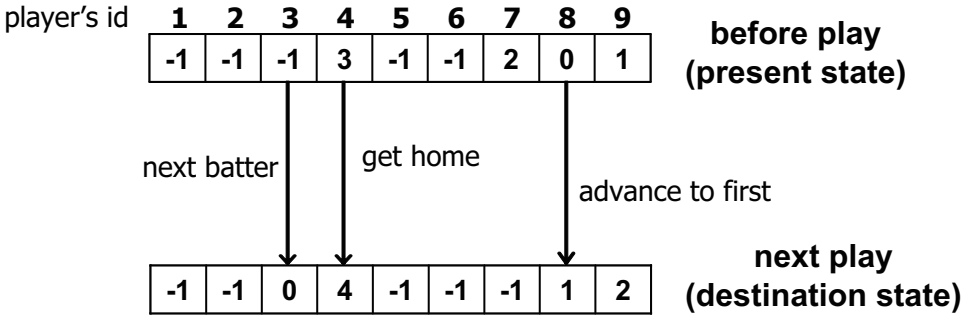
Fig. 5.4 Scorebook Data File (SDF).

5.2.3 Actions of Offensive Players

Offensive players indicate a batter, runners, and players who are waiting in the bench. Each player belongs to one of the six states as shown in Fig. 5.5(a). The batter is in the batter's box, so its state is "0", third runner is "3", and the waiting players are "-1". In SDF, the destination state to which every player changes in each play is sequentially recorded. When one line of the file is read out, the destination of each player is decided according to the data like Fig. 5.5(b). Then the game scene that 3D players are moving from the present state to the destination state while 1 play is created with CG.



(a) State transition of offensive players.



(b) Example of Scorebook Data File for offensive players.

Fig. 5.5 Actions of the offensive players.

5.2.4 Actions of Fielders and Ball

In contrast to the offensive players who are just moving from present state to destination while 1 play, the fielders are doing some actions while 1 play, such as moving around the field and throwing and catching the ball, etc. Therefore only the action of the ball is described in the SDF. Fielders move to catch the ball according to the action of the ball. The action of the ball while 1 play is described as shown in Fig. 5.6

Fielders basically stay own positions. First, the ball is thrown by the pitcher and hit to the position which is described in part D of Fig. 5.6. Then the player whose position number is described in the fist of part E moves to the position of part D to catch the ball. After catching the ball, the player throws the ball to the next player whose position number is described next. The next player moves to the nearest base and catches the ball. After the same iterations, finally, the ball is thrown to the pitcher.

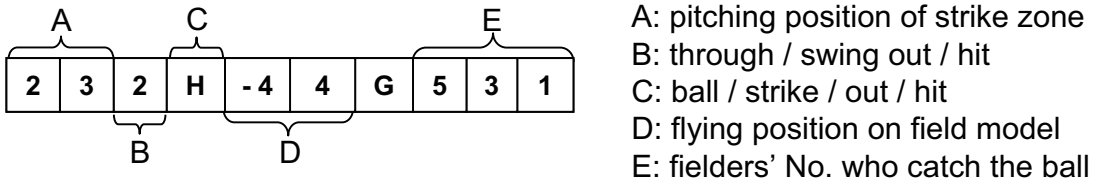


Fig. 5.6 Scorebook Data File of the fielders and the ball.

5.2.5 Demonstration

This section shows the demonstration results of this system. This demonstration is performed with a web-camera (ELECOM UCAM-E130HWH) attached to a handheld monitor connected a PC (OS: Windows XP, CPU: Intel Pentium IV 3.2GHz). The resolution of the captured image is 640×480 pixels. Multiple planar markers are distributed inside and outside the field model. In this system, one of the markers must be put on one of the bases in order to determine relationship between the field model and the markers. The other markers can be placed at arbitrary positions and poses. In this demonstration, 4 markers are utilized and one of them is placed on the third base. Fig. 5.7 shows the actual experimental environment of

this system. A Scorebook Data File of a baseball game is manually prepared in accordance with Sec. 5.2.2. 3D models of virtual objects, such as players and a ball, are rendered with OpenGL. The actual user's operations are described as follows.

User's Operations

1. **Arrangement**
   Place field model and multiple markers at arbitrary positions and poses;
2. **Capturing**
   Capture the object scene as candidates for two reference images;
3. **Input**
   Input Scorebook Data File;
4. **Observation**
   Start system and observe game with moving around;

First, the user places the baseball field model on the tabletop and distributes the markers. Next the object scene is captured with moving around the field model for 100 frames as candidates of the reference images. Inside of the system, the best pair of the reference images is automatically selected from the candidate images. Then the Projective Space is constructed by the selected reference images. The geometrical relationship of the markers is also estimated. These automatic processes take about 60 seconds. After the automatic preparations,

Fig. 5.7 Experimental environment of AR Baseball Presentation System.

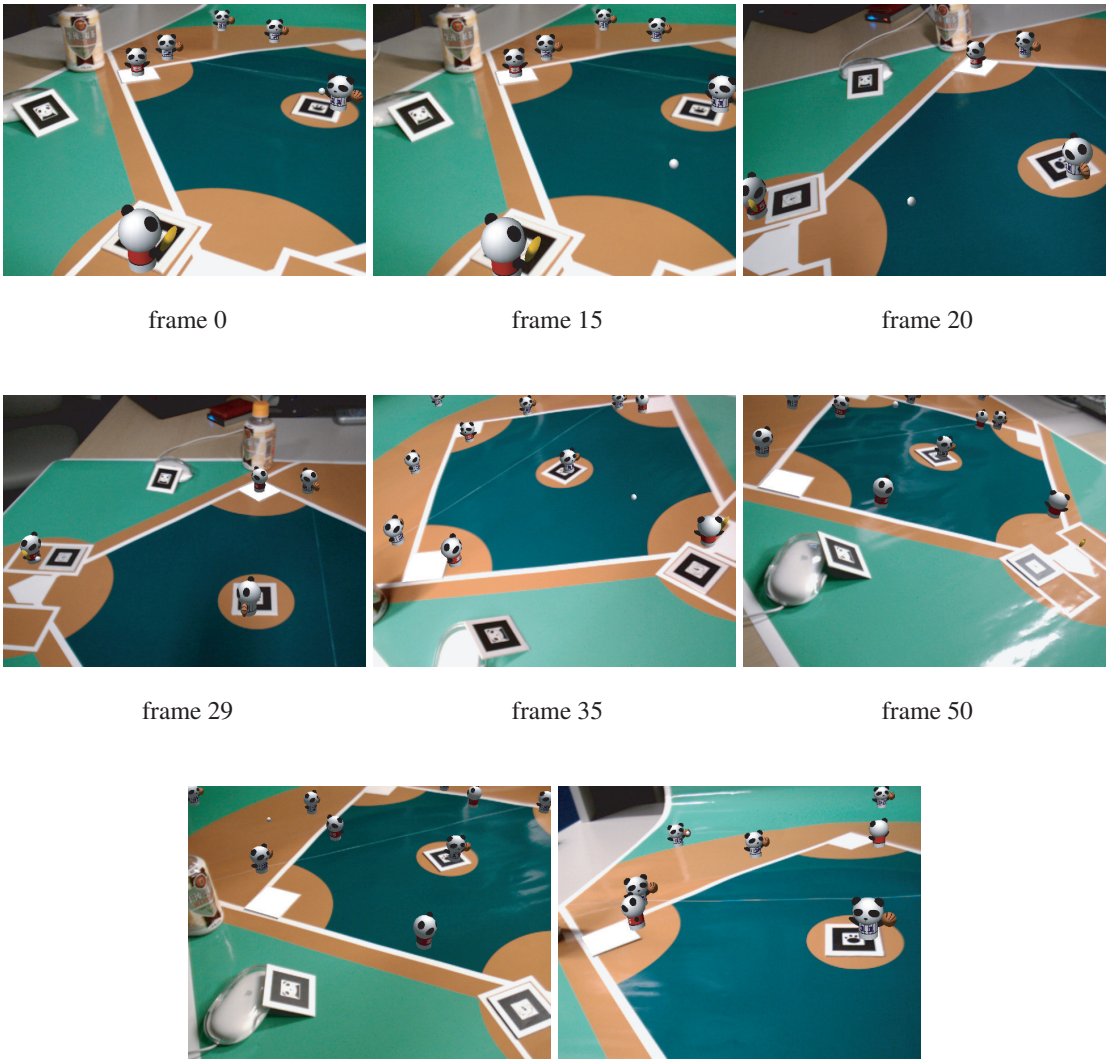


Fig. 5.8 Example of play: 4th batter of team RED sends a hit to left with the bases loaded and scores a goal.

the user inputs a Scorebook Data File and starts the system. The virtual baseball game begins on the field model and the user can watch the game from favorite viewpoint with moving around the real world.

Fig. 5.8 shows an example scene of a baseball game: team RED vs. team WHITE. In this situation, team WHITE is in the field and team RED is at bat. The bases are loaded and 4th batter of team RED is in the batter's box (frame 0–15). The pitcher throws the ball (frame

frame 0 frame 29

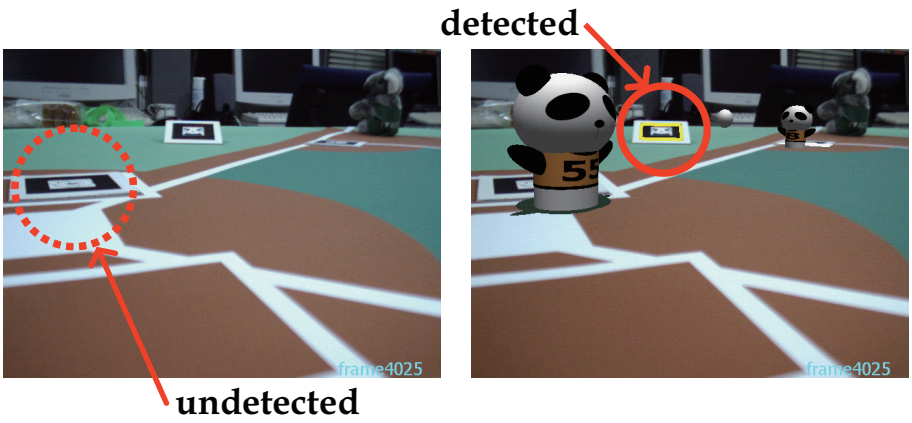
frame 35 frame 89

Fig. 5.9 Closeup views of the same scenes as Fig. 5.8 seen from different viewpoints.

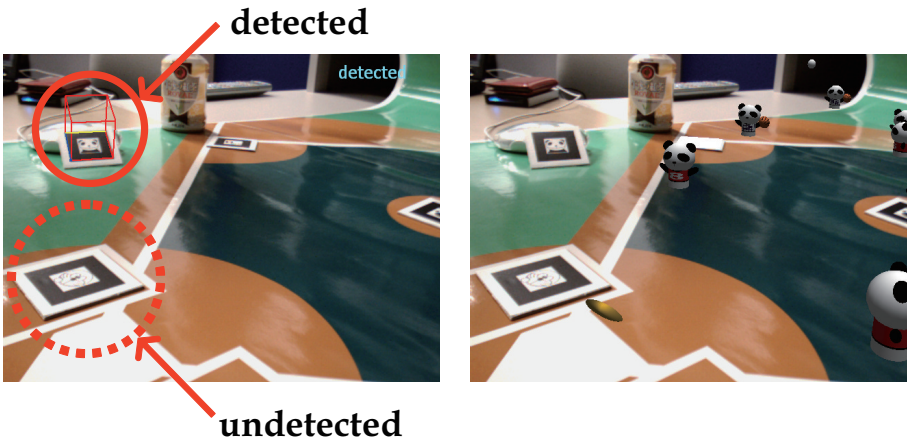
15–29). The batter hits safely to left (frame 29–35), and then all runners move up a base (frame 50–89). In the result, team RED gets a score. In this experiment, frame rate of AR presentation is about 30 fps. Thus user can see the baseball game at video-rate.

Fig. 5.9 shows some closeup views of the same scene. Since these images are captured from closeup viewpoints, only a few markers can be captured in the field of view, and the captured markers are different in every frame. Even though particular markers are not continuously captured over the frames, the virtual players and the ball can correctly registered onto the real tabletop field with the same world coordinate. This means that the consistency of the geometrical relationship between the camera and the virtual objects is kept properly, although the geometrical arrangement of the markers is unknown.

In Fig. 5.10, the angle of the camera with respect to the tabletop is too small to detect the



(a) case 1



(b) case 2

Fig. 5.10 Most of the markers which face to the same directions as the tabletop cannot be detected, even though they are captured in the image. The marker which faces to different direction is detected successfully.

markers lying on the tabletop plane. One marker is placed at different pose from the ground plane and the other markers are placed on the ground plane. In such a case, the markers which face to the same directions as the tabletop plane (placed near the home base) cannot be recognized because of the angle of the camera. Since another marker placed at different pose from the ground plane can be detected, the registration of the virtual players can be continued.

If all the markers have to be placed on the same plane, it even fails recognition for most of the markers. However, our system allows the markers to be placed at arbitrary positions and poses. Therefore, this is quite useful for observation system so that the user can watch the object scene from various viewpoints. It is a big advantage of the proposed system for applying to entertainment AR applications.

5.3 Interactive AR Bowling System

In the Interactive AR Bowling System, as shown in Fig. 5.11, a real bowling lane model is placed on the tabletop. The users roll a real ball down the bowling lane model to knock down virtual pins generated with CG. Of course they can move around the lane model and can see the virtual bowling game scene from favorite viewpoints by applying the algorithm presented in Chapter 3.

As for related work with bowling, Matysczok et al. have also proposed a bowling system using AR [75]. A user wears a HMD, in which a virtual ball, lane and pins are displayed, and interacts with the virtual ball by hand gesture. Since all objects, including the ball, are not real but virtual objects, this system just generates virtual bowling scenes with the input of hand gesture from the sensors. Thus, it is unnecessary to overlay the virtual scene onto the real scene like an AR system. Moreover, the user can hardly see the real world because the virtual lane is covering the real scene. Therefore the meaning of AR that is mixing the real world and the virtual world is lost.

In our system, in contrast, since the ball is a real object, the virtual scenes are generated

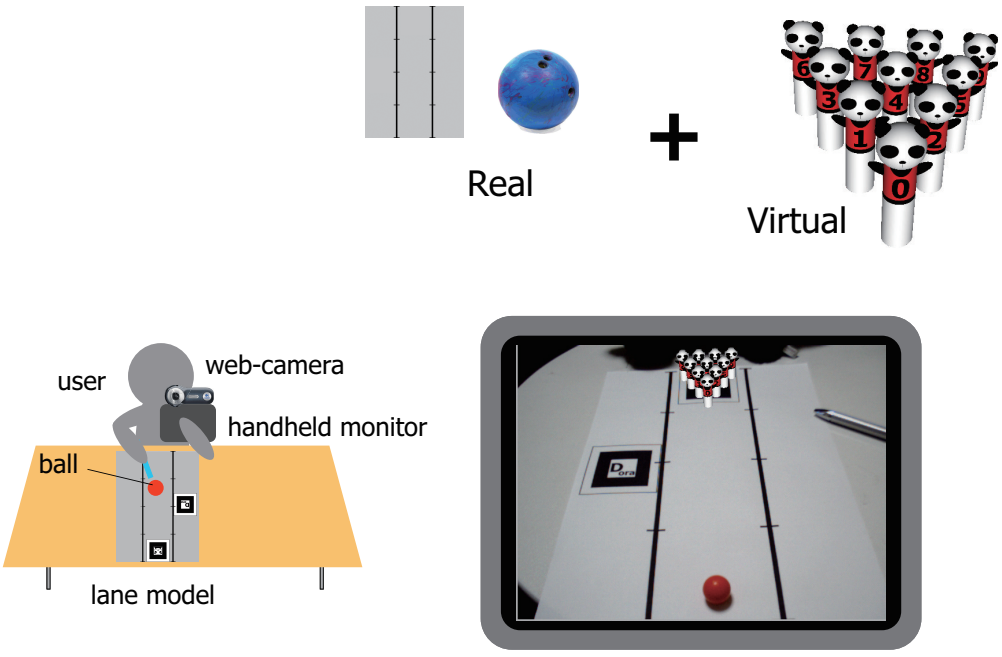


Fig. 5.11 Interactive AR Bowling System

according to the ball's motion in the real scene. Therefore it is effective to be an AR system with overlaying the virtual pins onto the real lane model. Moreover the user can touch the real ball, so our system achieves a real bowling style. In Matysczok 's system, special gloves with physical sensors are also required for user's interaction, however, our system needs only a camera and a PC and use a real ball and lane model.

For realize this kind of bowling system, we have to perform the following tasks. There are two lines and 2D markers on the bowling lane model. These two lines define the lane, which means that the ball should be rolling between the two lines. In case the ball goes out of the lane, it is considered as "gutter". Therefore, the lane and the ball have to be detected and tracked at every frame.

When the ball hits any virtual pins, the pins are knocked down. For generating such virtual pins according to the ball, the geometrical relationship between the real ball and the virtual pins has to be computed interactively. In our method, the ball's position on the input image is transformed onto a top-view image, which is the input image seen from top-view, to obtain the ball's position to the pins.

Finally, the virtual pins are overlaid onto the input image according to the camera's position and pose, which are corresponding to extrinsic parameters of the camera. The extrinsic parameters are estimated by multiple 2D markers.

5.3.1 Processing Outline

Fig. 5.12 shows a flowchart of our proposed system. First, the images captured by the web-camera are applied to three kinds of processing; marker tracking, lane tracking and ball tracking. During the marker tracking process, AR-Toolkit [52] detects 2D markers placed around the lane model. Since the relationship of the lane with respect to the marker is fixed by the lane model, one of the markers is arranged between two lines which consist of the lane. Then a 3D coordinate system where the virtual objects should be overlaid is defined on the lane model by the marker. During the lane tracking process, some lines are detected by Hough Transform [76] and two of the detected lines are selected as the lane. During the ball tracking process, a region of the ball is detected. We assume that the centroid of the region is the ball's position.

After the tracking of the markers, the lane and the ball, the ball's position is transformed to the top-view image to obtain the ball's trajectory. By the trajectory, the geometrical relationship between the real ball and virtual pins can be computed. Therefore collisions between the ball and pins are computed according to their relative positions. Finally, the pins are overlaid onto the input image by the extrinsic parameters computed at the marker tracking process.

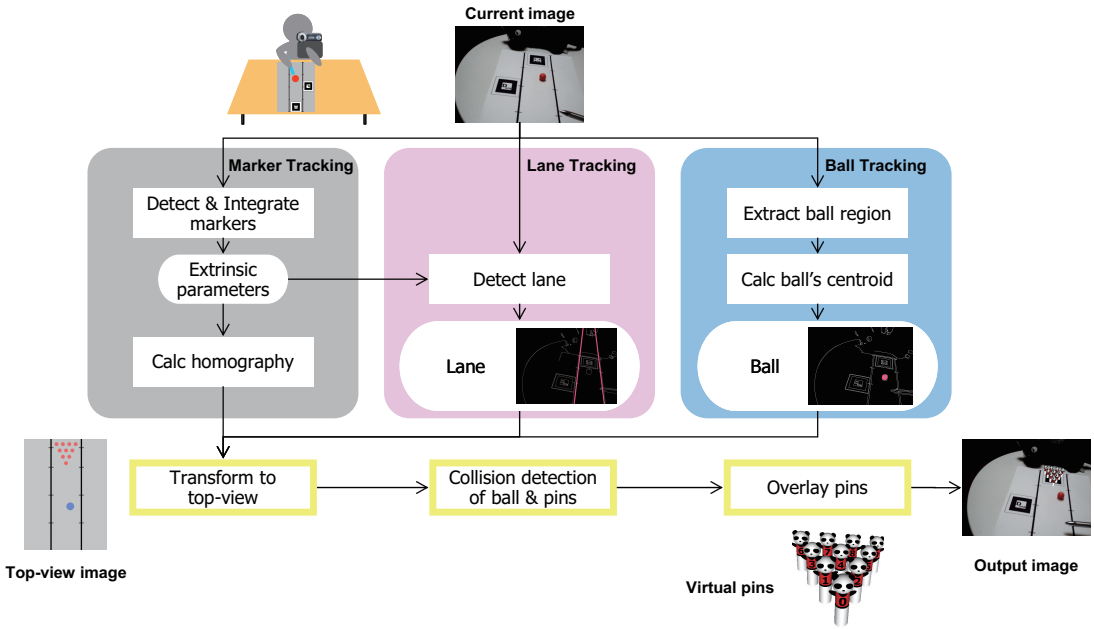


Fig. 5.12 Overview of Interactive AR Bowling System.

5.3.2 Marker Tracking

The multiple markers placed around the lane model are detected as well as our baseball system. The geometrical relationship of the markers are also estimated by the registration method described in Chapter 3. Then a 3D coordinate system where the virtual objects should be overlaid is defined by using one of the markers on the lane model as shown in Fig. 5.13.

For tracking the trajectory of the ball on the lane, we use a top-view image. The top-view image is a image which is the input image transformed to the top viewpoint. By this transformation, the ball's 2D motion becomes understandable. In order to transform the input image to the top-view image, we compute a homography $\boldsymbol{H}$ [76] which is the planar projection matrix and relates the real lane model and the lane model area in the input image. In this system, the $\boldsymbol{H}$ is computed by the corresponding points on the lane model in the real world and the input image. Actually, when the camera parameters with respect to the marker are computed frame by frame, the position of corresponding points in the input image can be computed. Therefore the $\boldsymbol{H}$ is easily computed and used in the transformation whose detail will be described in Sec. 5.3.5.

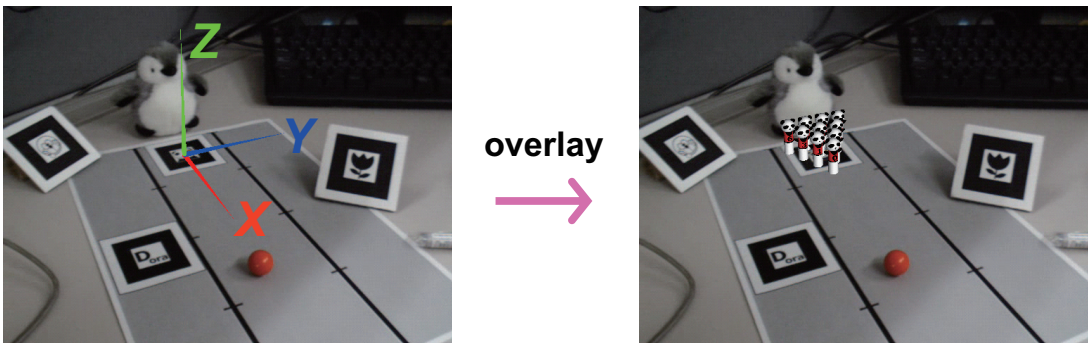
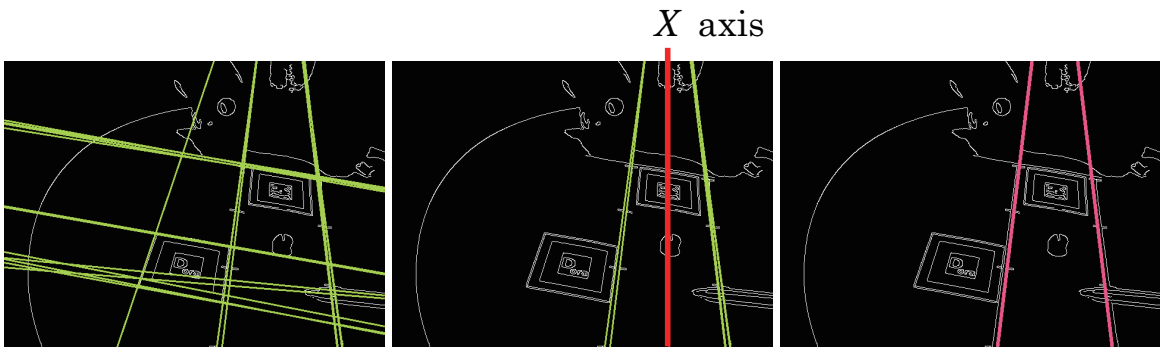


Fig. 5.13 3D coordinate system defined on the real lane model. Virtual pins are overlaid in this coordinate system.



(a) Detected lines. (b) Similar slopes. (c) Detected lane.

Fig. 5.14 Lane detection by Hough Transform and markers.

5.3.3 Lane Tracking

The lane consists of two parallel lines on the lane model. There is one marker between the two lines to determine the relationship between the real lane model and the virtual pins. To detect and track the lane, first, some lines existing in the input image are detected by Hough Transform like Fig. 5.14(a). From the group of detected lines, lines which have similar slope to X axis are selected as candidate lines of the lane as shown in Fig. 5.14(b). Then two lines closest to X axis are selected on the both sides of X axis, respectively, as shown in Fig. 5.14(c). In case that any markers cannot be detected in the current frame, X axis which was defined in the previous frame is used instead.

5.3.4 Ball Tracking

In this system, we assume that the color of the ball should be quite different from the lane model. In this work, we use a red ball on a gray lane model as shown in Fig. 5.15(a). For detection of the ball, first, red regions are extracted from the input image by dividing it into R,G,B channel images. Fig. 5.15(b) shows the image after dilation and erosion a few times. If multiple regions are extracted, the biggest region is selected as the ball's region. Finding the minimal circumscribed circle (contour) for the detected region, the center of the circle is considered as the 2D ball's position in the input image as shown in Fig. 5.15(c). Actually,

(a) Original image. (b) Ball region. (c) Detected ball.

Fig. 5.15 Ball detection.

the detected ball's position is the centrum of the ball and is not corresponding to the position in contact with the lane ground. In this system, however, we ignore the height of radius and assume the detected position (the center of the circle) is the ball's position.

5.3.5 Transformation to Top-View Image

Using homography $\boldsymbol{H}$ computed at the marker tracking process, the ball's position in the input image is transformed onto the top-view image frame by frame as shown in Fig. 5.16. The transformed ball's position becomes the ball's trajectory. The positions of virtual pins with respect to the lane are already known. Therefore the trajectory provides a geometrical

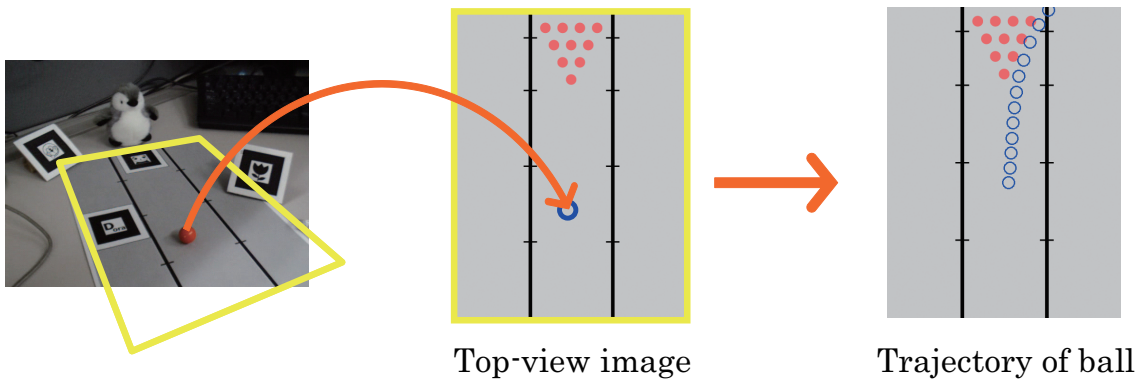


Fig. 5.16 Transform the ball's position to top-view image.

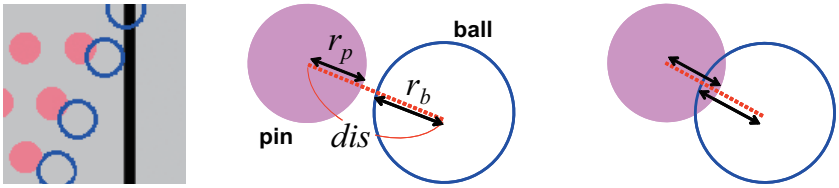
relationship between the ball and the pins on the lane model. By using this trajectory, the collision between the ball and the pins is detected. The direction of each pin's knock down is also computed by using the trajectory.

5.3.6 Collision Detection between Ball and Pins

We assume that the radius of the ball and each pin are r_b and r_p, respectively, and define the distance between the ball and each pin as dis. For detecting a collision between the ball and the pins, the distance dis is computed for each pin at every frame. The collision is detected by comparing the distance dis and radius r_b, r_p as following equations and Fig. 5.17(b)(c).

Because of the frame rate of the web-camera, few number of the ball's position can be detected when the ball is quickly rolled. In that case, the moment of the collision may be not captured in frame sequence. Therefore the ball's trajectory is interpolated by the captured ball's positions as shown in Fig. 5.18.

$$\begin{cases} (r_b + r_p) \leq dis \Rightarrow \text{Not hit} \\ (r_b + r_p) > dis \Rightarrow \text{Hit} \end{cases} \tag{5.1}$$



(a) Trajectory. (b) Not hit (c) Hit

Fig. 5.17 Collision detection by trajectory of the ball.

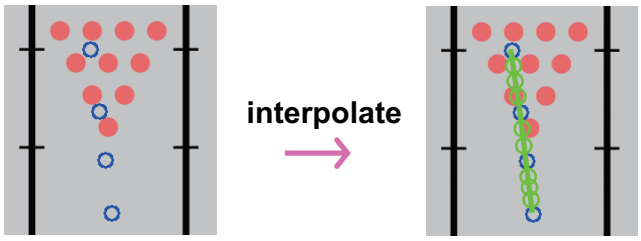


Fig. 5.18 Interpolated trajectory of the ball.

5.3.7 Knocking Down and Overlay Virtual Pins

After the collision detection, the pins which are generated with CG are overlaid onto the image in the style of standing or knocking down. The direction of knocking down is defined by trajectory of the ball. As shown in Fig. 5.19, the direction is computed by two vectors; from the ball's position in the previous frame to the current frame; from the ball in the current frame to each pin. If some pins are knocked down by collision with the ball, neighboring pins may be knocked down in a chain reaction depending on the ball's direction. In that case, the direction of knocking down is decided by the direction of the pin which is previously knocked down as shown in Fig. 5.20.

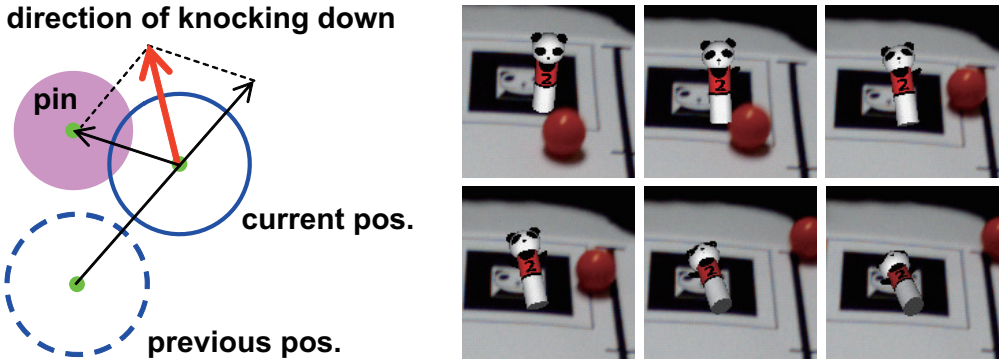


Fig. 5.19 Direction of knocking down.

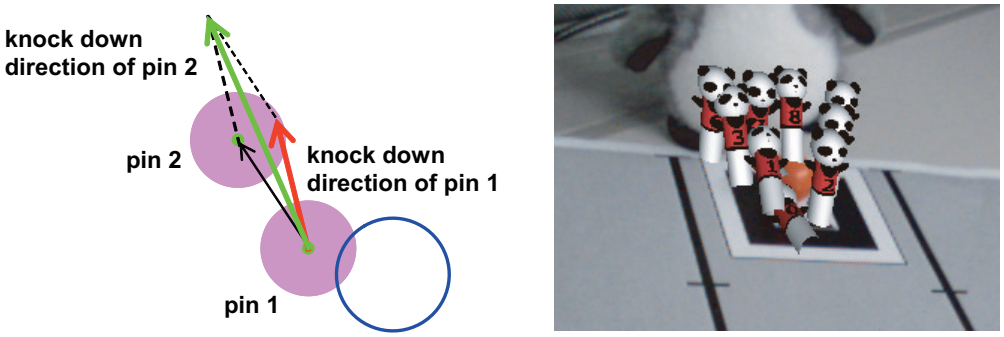


Fig. 5.20 Direction of chain knocking down.

5.3.8 Demonstrations

In this experiment, four 2D markers on the lane model are placed on the tabletop to estimate the camera motion (extrinsic parameters). Some of them are placed on the same plane as the tabletop; the others are aligned in various directions. The geometrical relationship between every marker is automatically computed by our method. One of the markers, which lies between the lines, is also used for defining the 3D coordinate system. The resolution of the captured image is 640×480 pixels. The virtual pins are rendered with OpenGL library.

Fig. 5.21 Experimental environment of Interactive AR Bowling System

Fig. 5.22 shows the detected lane and ball's trajectory. Both of the lane and the ball can be correctly detected and tracked over all frames by our tracking method, according to the camera motion. The ball's position is also successfully transformed onto the top-view image by the homography computed by 2D markers.

Fig. 6.18 shows example scenes where the virtual pins are overlaid according to the camera motion. If we use only one marker for overlaying the pins, the registration becomes impossible when the ball is rolling over the marker because the marker can not be detected. Therefore we have to use multiple markers. Even though particular markers are not continuously captured over the frames, the virtual pins can be correctly registered on the lane model because

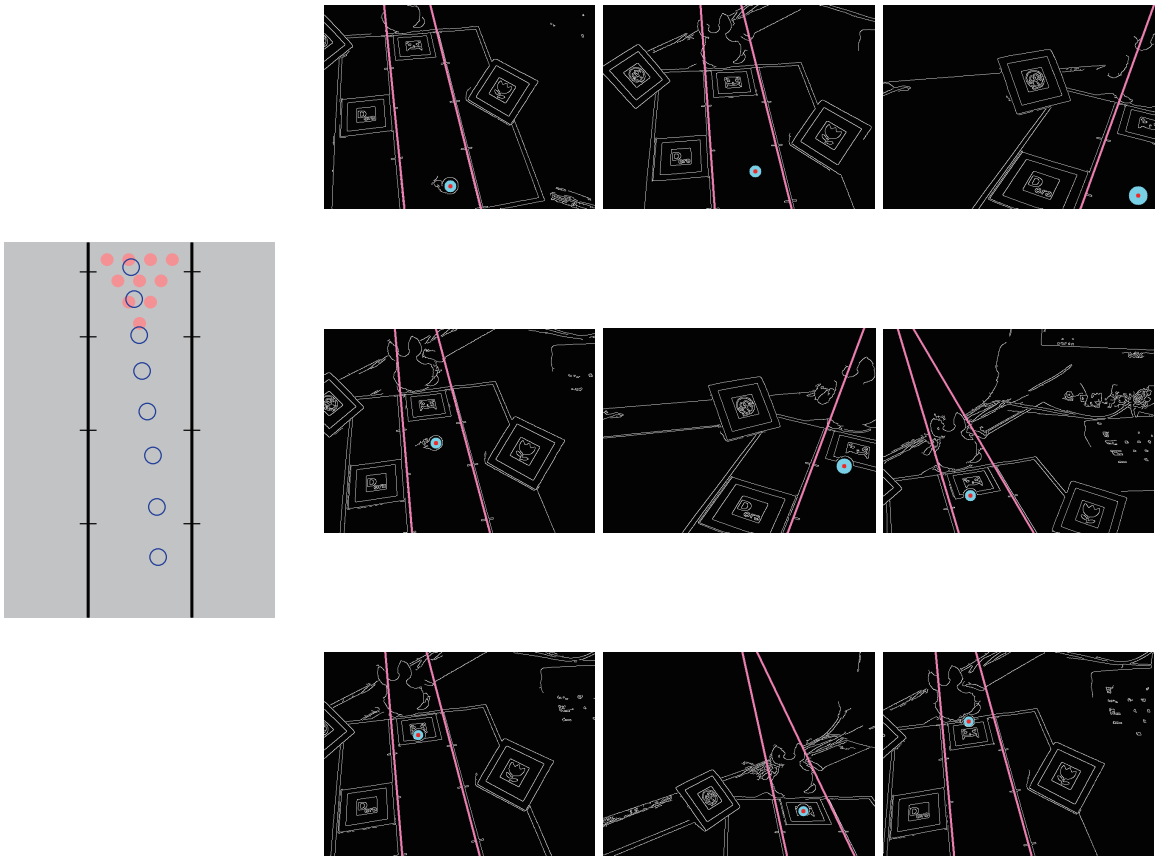


Fig. 5.22 Trajectory of ball and detected lane and ball.

of our registration algorithm which can estimate the geometrical relationship of the markers.

Moreover, since the collision of the real ball and the virtual pins are successfully detected, some pins are knocking down by hitting of the ball. The pins existing behind the hit pins are also knocked down as a chain reaction of the front pins by computing the direction of knocking down from the trajectory. This system runs 30 fps, so the user can enjoy the bowling game at video rate.

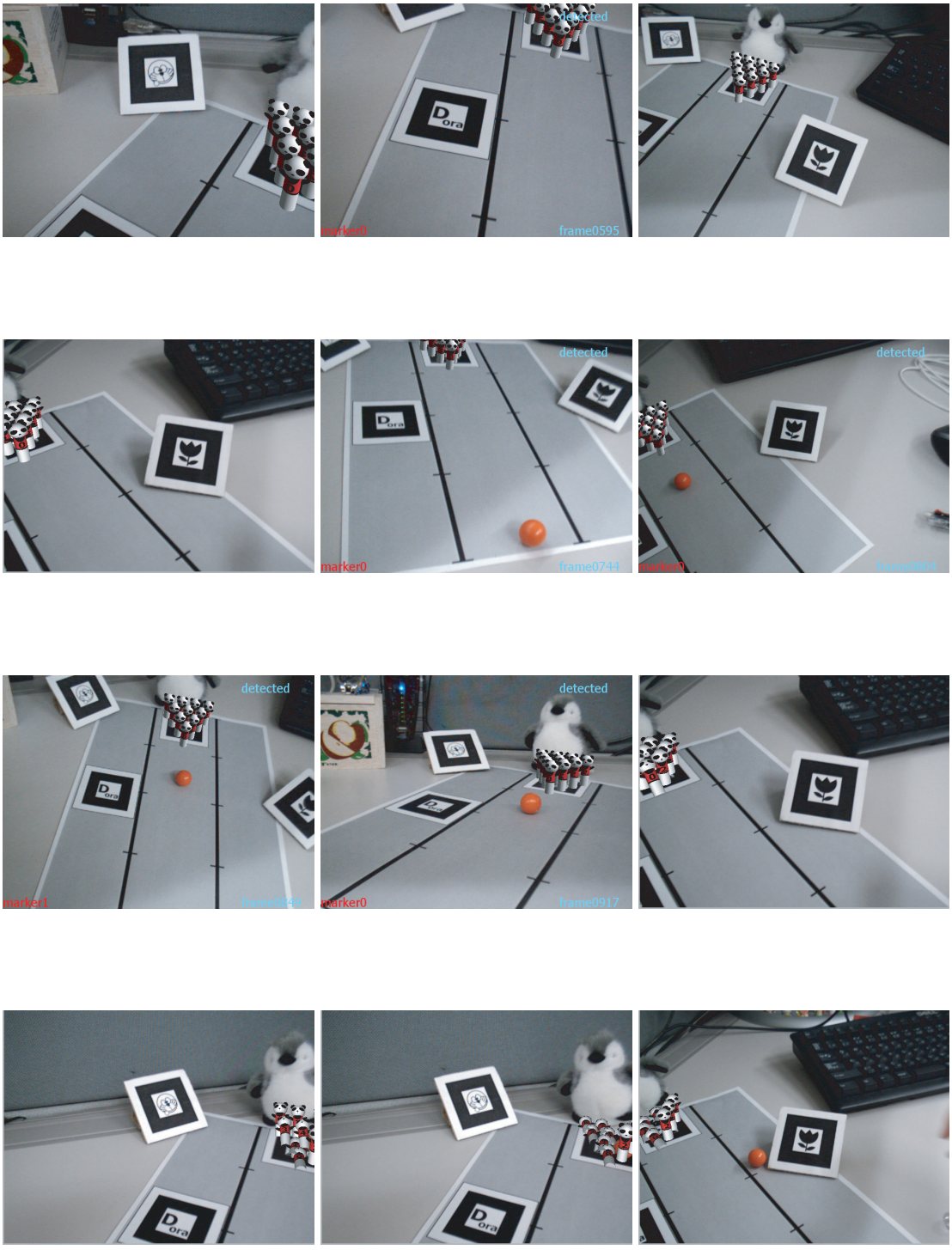


Fig. 5.23 Resulting images on which virtual pins are overlaid.

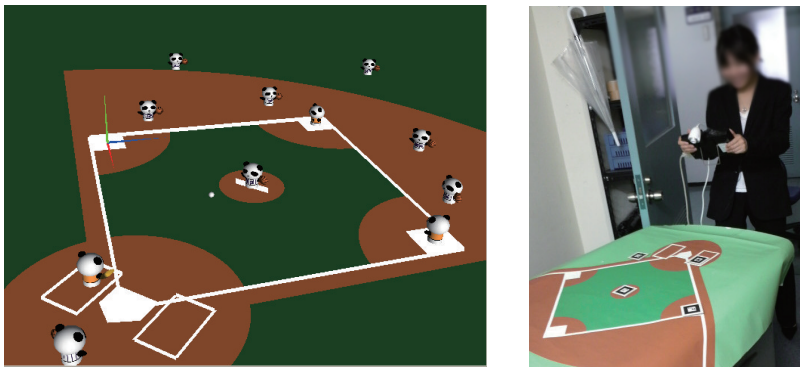
5.4 User Study

5.4.1 Baseball System

AR Baseball Presentation System visually replays the baseball game which was previously played in the other place. In contrast with the usual way to know the game history such as watching the captured video or reading the recorded scorebook, our AR system can provide the users with immersive feelings as an entertainment application.

We design our system as AR system which can overlay the CG scene on the real field model in front of the user as well as visualize the recorded baseball game by 3D CG. Using this system, the user can watch the game from the favorite viewpoint by just moving around the field model. Such simple way for watching the CG-represented event using the AR system provides more immersive feeling than using usual CG viewers, in which a mouse or a keyboard is used for changing viewpoints [77].

In this user study, we intend to evaluate how the AR system is effective to enhance the quality of entertainment. There are a lot of factors that may enhance the quality of entertainment such as usability, interactivity, visual effects, and so on. Those factors are evaluated by studying “how quickly”, “how easily” and “how intuitively” the user can change their viewpoint. The same factors are also evaluated for the usual CG viewer. Then both of results are

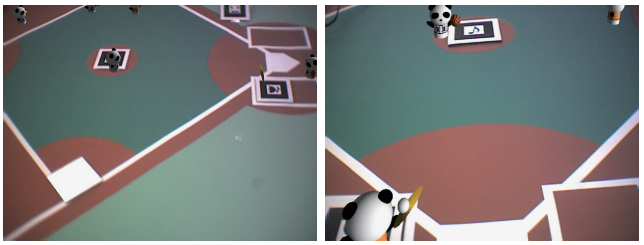


(a) CG (b) AR

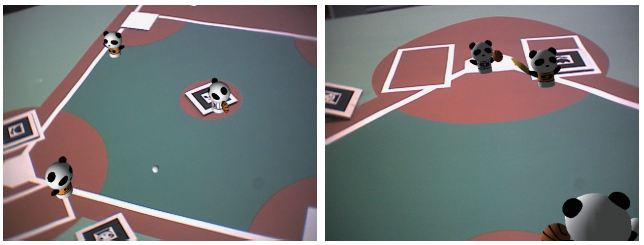
Fig. 5.24 Prepared baseball system

compared to evaluate the effectiveness of designing our system as AR system.

In this evaluation experiment, we prepared two kinds of baseball observation systems as shown in Fig. 5.24, one is our AR Baseball Presentation System, another one is created by only CG. In this CG system, the user watches the baseball game with changing the viewpoint by using a keyboard of PC. The rotation and translation about X-Y-Z axis are assigned to



(a) from 3rd stand (b) from catcher



(c) from 1st stand (d) from pitcher

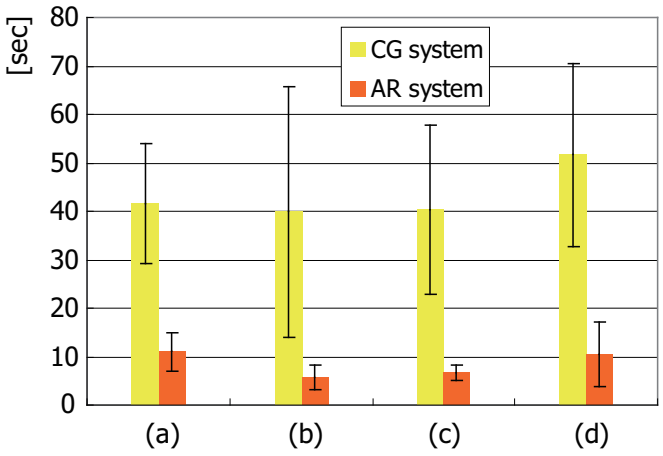


Fig. 5.25 Average time the examinees took to change their viewpoints to specified viewpoints.

each key. In the AR system, they just move around the field model with a handheld monitor. Then, we asked 15 examinees to use these systems, and measured the time each examinee spent on moving the viewpoint to the specified viewpoints as shown in Fig. 5.25 (a) $\sim$ (d).

Fig. 5.25 shows the average time which the examinees spent to change their viewpoints. We can find that the CG system took much longer time than the AR system. In this experiment, every examinee spent triple to ten times longer times to change the viewpoint in CG system than AR system. Because the users only have to move around the field model with a handheld monitor to their favorite positions, our AR system can quickly change the viewpoints.

This result can also be found in Fig. 5.26, which shows answers of questionnaire about changing viewpoints. We asked four questions (a) $\sim$ (d), and then the examinees rated on a scale of 1 to 5. In the same way as the actual measurement time in Fig. 5.25, most of the examinees felt that our AR system was easier than the CG system to change their viewpoints to their desired positions quickly and intuitively. The questionnaire also asked whether they could change the viewpoint with watching the game. As a result, most of examinees felt that the AR system was easier to change the viewpoint with watching the game. This is because the viewpoint is corresponding to the user's own viewpoint, while the viewpoint of the CG system is the virtual camera position. Therefore designing our system as the AR system is quite helpful for any users to handle this kind of digital contents because the operation is very easy and intuitive.

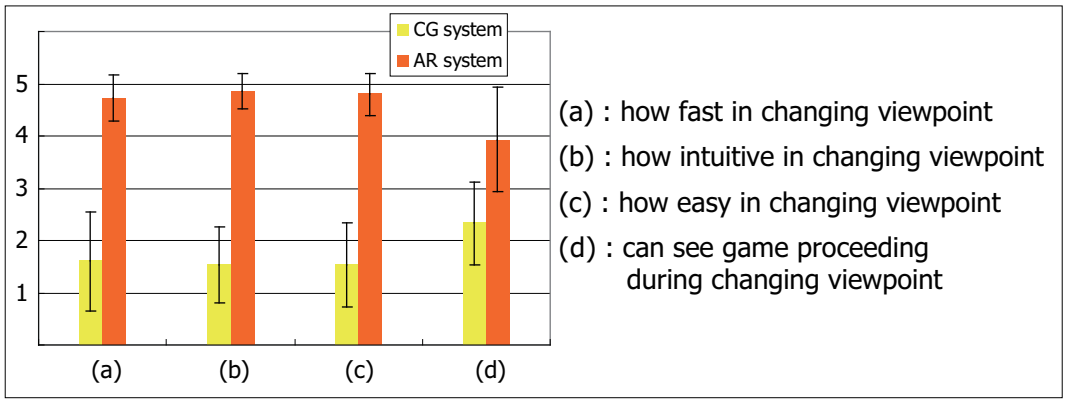


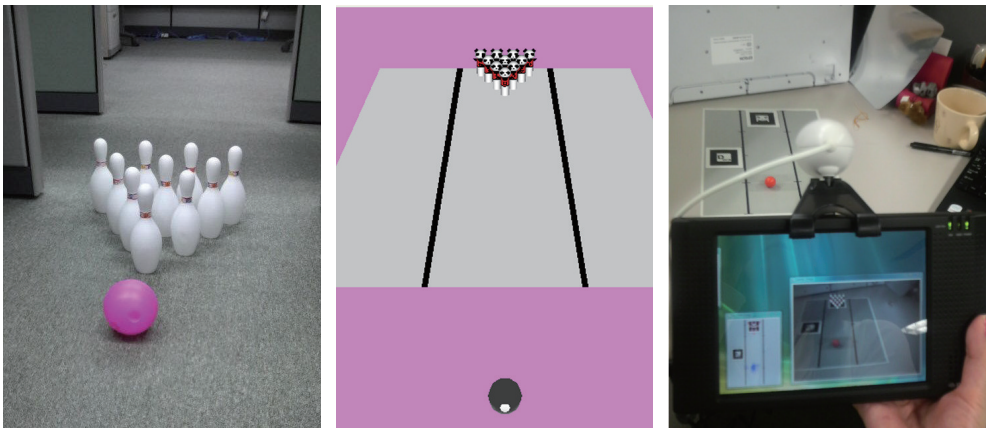
Fig. 5.26 Answer of questionnaire about changing viewpoints.

5.4.2 Bowling System

AR Bowling System consists of real and virtual objects, such as the real ball, the real lane model and the virtual pins. In this system, the users physically touch and roll the real ball on the real lane. Such physical communication provides the users with much reality as a tangible interface [70, 71, 72, 73].

Therefore we focus on the effectiveness of such tangibility of our AR system as the evaluation point of the AR Bowling System. The AR system can interact with the virtual world by rolling the real ball with a hand unlike a CG system which entirely consists of virtual objects. This means that we should evaluate how effective the direct touch to the ball for the bowling game as an entertainment application. For evaluating this point, we evaluate "how naturally the users can control the ball", "whether they actually feel that the ball is controlled by themselves" and "whether the game is challenging".

In the same way as the baseball system, we prepared two kinds of bowling system, our AR Bowling System, a CG bowling system and a real toy bowling game as shown in Fig. 5.27. We also asked the examinees to play the real toy bowling game before using the CG and AR bowling systems. In the CG system, when the user drags the virtual ball on the display by a mouse, the ball starts rolling. The direction of the ball is defined according to user's dragging. The speed of the ball is also defined as the length of dragging of the mouse. In the AR system,



(a) Real (b) CG (c) AR

Fig. 5.27 Prepared bowling system

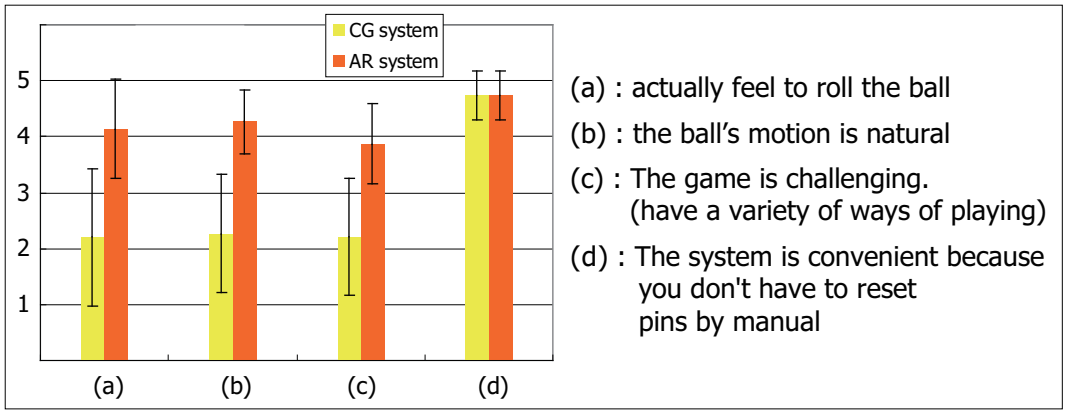


Fig. 5.28 Answer of questionnaire about the reality of playing bowling.

the user rolls the real ball on the lane model with a hand and watches the proceeding through the handheld monitor. After playing all the systems, they answered questionnaire by rating on a scale of 1 to 5. The questionnaire items and the results are shown in Fig. 5.28.

Although they could only decide the direction and speed of the ball in the CG system with a mouse, the users could freely control the ball in the AR system. And also they could really touch the ball. As a result, they actually felt rolling the ball by themselves in the AR system. Since the ball is virtual object in the CG bowling system, the users can roll the ball only linearly. Therefore some users said that they wanted to roll the curve-ball. In order to achieve such a curve-ball in CG system, some random elements have to be included. Since such randomness cannot be handled by the users, however, such system is unacceptable as computer games. On the other hand, the users can roll any kind of ball depending on their skills because the ball of the AR system is real object. Therefore most of the users felt that the ball's motion of the AR system was more natural like the real bowling game than the CG system. Because there are various ways to roll the ball in the AR system, the game is not too simple to complete. For example, some users inclined the lane; other users used a pen to roll the ball instead of their hands. Therefore they felt that the AR system was more challenging than the CG system.

By the way, when playing the real bowling game, we asked the examinees to raise and reset the fallen pins by themselves. As a result, they felt that it's troublesome to reset the pins every time. As described before, physical communication is very effective for the young users, however, we can afraid that little children cannot arrange the pins very well. On the other

hand, virtual bowling game (both of AR system and CG system) do not require such a task because the users only have to press 1 button to reset the fallen pins. Therefore, the concept of the AR Bowling is very helpful for any users by allowing the physical communication without troublesome task.

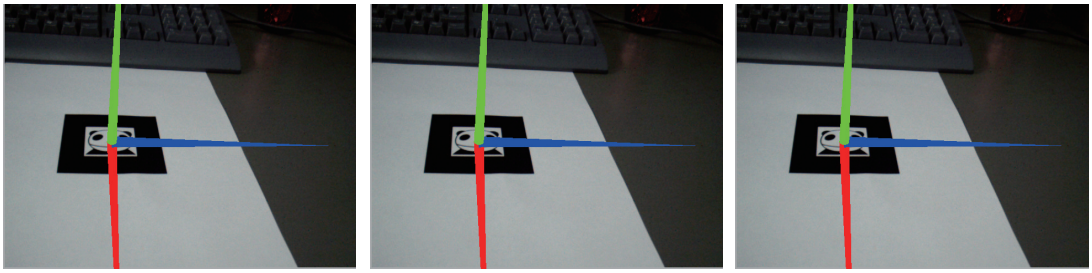
Chapter 6 :

Pattern-based Tracking with Particle Filter for improving plane-based tracking

6.1 Motivation

A 2D (planar) rectangular marker like ARToolkit [54] is very popular tool for the camera tracking in AR. Marker-based method can estimate rotation and translation of the camera in real-time by extracting the contour of the rectangular from the input images and estimating position and pose of the marker.

When estimating the rotation and translation parameters of the camera by using such markers, it is important to detect the change of 2D appearance of the marker in the input images. However detection errors of 2D features like edges or corners of the marker may affect camera tracking accuracy depending on the camera position and pose with respect to 3D coordinate system fixed on the marker plane. Especially when tracking the Z axis which is perpendicular to the marker plane (X-Y plane), the angle of the camera relative to the marker plane



(a) Images captured from an angle: 3D coordinate system is easily tracked from these images.

(b) Images captured in the front of the plane: Tracking 3D corodinate system is difficult and unstable, especially for Z axis.

Fig. 6.1 Example images on tracking 3D coordinate system: Depending on the camera angle, 3D coordinate system on the marker can not be tracked stably.

extremely affects the tracking accuracy.

If the camera captures the marker from an angle as shown in Fig. 6.1(a), for example, the 3D coordinate axes fixed on the marker can be correctly tracked over frames. On the other hand, when the image plane of the camera is almost parallel to the marker as shown in Fig. 6.1(b), directions of axes are not stable even though the camera hardly moves between frames. This is because slight difference of detection of 2D features such as edges or corners between the frames significantly affects computation of 3D position and pose of the camera.

Therefore, we introduce a pattern-based tracking method to improve the estimation accuracy of the rotation and translation parameters in 2D marker-based tracking. In contrast with the algorithm of [54], our method utilizes the patterns inside of the marker as well as the edges. In this case, patterns (a part of the input image) are generated by using predicted parameters and compared to the actual input images. Then best parameters which can generate the pattern which is very good match with the actual input image are decided as the camera parameters in the current frame.

The best way in terms of accuracy is full search of all parameter candidates. This means that all the possible parameters are kept as candidates and tried to compare with the actual input image. However, it is quite unreasonable for an on-line system from a computational perspective. Therefore we employ an algorithm of Particle Filtering [78]. Particle Filter is a kind of temporal sequence filtering methods and can effectively keep candidate parameters according to the probability density function. Therefore, our method reasonably keeps a lot of rotation and translation parameters and compares the patterns in the input image and in the generated image by each candidate parameter. Since we do not estimate the parameters not only by detecting 2D feature points but search the best parameters by comparing generated images with actual images, estimation errors of the rotation and translation parameters such as jitters between frames can be reduced, especially for Z axis.

6.1.1 Related work using Particle Filter

The tracking method using filtering technique is often used for object tracking. Since it is performed by probabilistic estimation based on the previous status, it can be robust against the random noise in the input images. In recent computer vision area, a lot of tracking method such as human position tracking, head pose tracking, face tracking, etc. are studied by using

filtering methods. Especially, Particle Filter that is one of the filtering methods is highly used for the tracking [79, 80, 81, 82, 83]. Oka et al. use the state vector which is composed of 6 parameters representing 3D human head position and pose. Then they achieved accurate tracking of human head by adaptively controlling diffusion of particles [80, 81].

Particle Filter is also used for camera tracking [84, 85]. Pupilli et al. proposed a camera tracking method using some feature points in the real scene. They applied Particle Filter to track the feature points, however, the tracking is based on the template matching without template renewal, so the tracking is not robust against the change in appearance of the feature points. Although the initial feature points are natural feature points, their 3D positions and poses have to be known as well as using artificial markers.

As a close method to our method, Marimon et al. applied Particle Filter to marker-based tracking [85]. They focus on a problem that the marker is occluded when the whole marker is not completely captured in the image frame. They employ Particle Filter to combine feature point-based tracking with marker-based tracking only for tracking when the marker can not be completely seen, however their method cannot improve the accuracy when the marker can be seen. Therefore their method is essentially different from our method because the purpose of our method is improving the tracking accuracy of marker-based tracking itself.

6.1.2 Camera tracking with Particle Filter

Kalman Filter is a similar filtering technique to Particle Filter. Since Kalman Filter approximates the probability density function (PDF) as a Gaussian function, the range in application is limited. Moreover, Kalman Filter decides the solution only by prior probability. On the other hand, Particle Filter represents the PDF by discrete hypotheses which are called particles. Therefore, Particle Filter is not limited to Gaussian distribution. In addition, Particle Filter decides the solution by not only prior probability but also posterior probability. This means that Particle Filter can estimate the best parameters by comparing the estimation results with the actual image.

For computing accurate parameters, there is another approach that is an optimization approach such as the Steepest Descent Method, Least Square Method, and so on. For applying such optimization approaches, in most cases, feature point (or edge) tracking is necessary. This is because the optimization approaches minimize the difference between the estimated

position of the feature points and the actual feature points in the image. Therefore the correct answer that is correctly-detected feature points is necessary to move the estimated results closer there. However, it is well known that feature point tracking in real-time is not so easy. On the other hand, our pattern-based tracking with Particle Filter estimates the parameters by keeping candidate parameters and selecting the best one from them. Therefore such feature point tracking is not required. Moreover, the optimization approaches need iterative calculations for minimizing errors. However, it is not suitable for real-time tracking. In contrast, pattern-based tracking does not perform iterations. Therefore it is reasonable for real-time tracking.

6.2 Overview of Pattern-based Tracking Method

In this section, we introduce our proposed method that uses Particle Filter and estimates camera rotation and translation by tracking a 3D coordinate system fixed on a marker plane from input images captured by a moving camera.

The overview of our method is shown in Fig. 6.2. The proposed method is divided into two steps; Initialization step, Tracking step. In the initialization step, we use the algorithm of ARToolkit to extract a marker from input images and estimate the initial parameters of the camera. Initial hypotheses (particles) are generated around the initial parameters. The set of discrete hypotheses configures the probability density function (PDF) of Particle Filter.

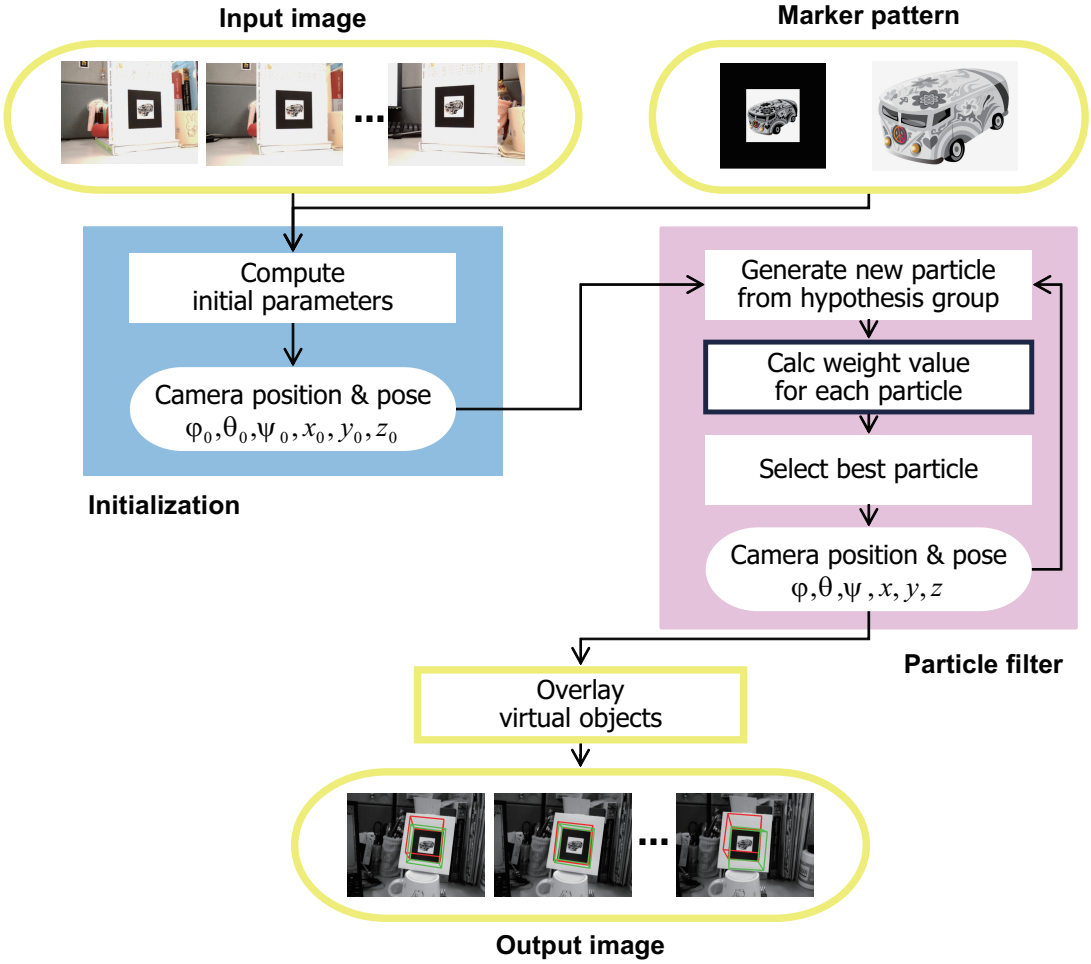


Fig. 6.2 Overview of our tracking method.

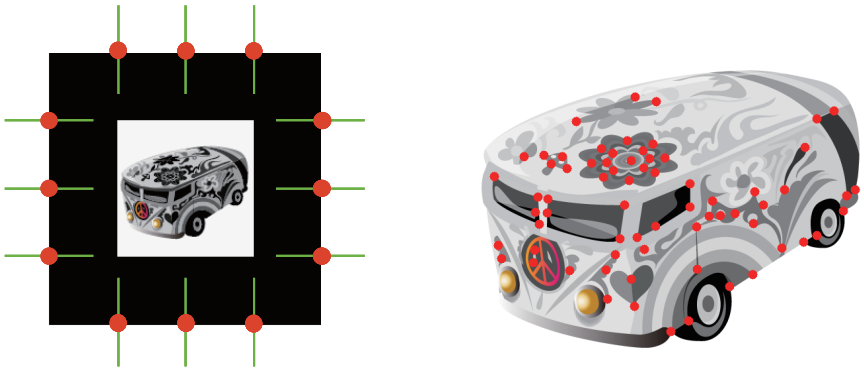
In the tracking step, the rotation and translation parameters of camera are estimated frame by frame. In particular, the best particle is selected from the set of hypotheses which are candidates of the camera parameters. The particle is selected according to corresponding weight values. Our method computes the weight value based on two approaches; edge-based, pattern-based. In the edge based approach, the contour of the marker is projected on the input image by each particle. Then the total distance from sampled points on the projected contour to the nearest edge in the input image becomes the weight value for the particle. In pattern-based approach, the input image is transformed to the front-view image by using each particle. Then regions around feature points in the pattern are compared between the front-view image and the original model image. The similarity becomes the weight value for the particle. Finally the combination of the edge-based value and pattern-based value becomes the weight value for the particle. According to the computed weight values for all particles, the best particle is selected. New hypotheses are also generated around the best particle based on the weight values.

6.3 Initialization Step

In the initialization step, the initial parameters of the camera are estimated by using the algorithm of ARToolkit [60]. Initial hypotheses (particles) are generated around the initial parameters.

For computing weight values based on edges, K points are sampled on a contour of the marker's rectangular. For computing weight values based on patterns, M feature points are detected from the model image by using OpenCV library function. In the experiment described later, we let the number of feature points K=12, where three points are sampled from every side of the contour as shown in Fig. 6.3(a), and M=60 as shown in Fig. 6.3(b).

Each feature point has a 3D position in a 3D coordinate system fixed on the marker plane ($Z=0$). In order to estimate the position and pose of a camera with respect to the 3D coordinate system at every frame, 2D-3D correspondences have to be known. Therefore, 2D corresponding points of K sampled points and M feature points in the initial image frame are computed by the initial parameters.



(a) Sampled points on edge. ($K = 12$) (b) Feature points in pattern. ($M = 60$)

Fig. 6.3 Points for computing weight values

6.4 Tracking Step

In the tracking step, particle filter estimates the camera position and pose which represent rotation and translation parameters in the t th image frame. The parameters are represented as a 6D vector $\boldsymbol{p}_t$=$(\phi_t, \theta_t, \psi_t, x_t, y_t, z_t)^\top$ in a 6D state space $\boldsymbol{S}$. t represents the frame number. As described before, these rotation and translation parameters represent 3D position and pose of the camera with respect to the 3D coordinate system which is fixed on the marker plane.

Particle filtering represents the probability density function (PDF) as a set of N discrete hypotheses (particles) $\{\boldsymbol{s}_t^i\}$ in the 3D state space $\boldsymbol{S}$ and the corresponding weight values π_t^i $(i = 1 \cdots N)$. This sample set can approximate an arbitrary PDF.

At the beginning of the tracking, we generate N new samples as $\boldsymbol{s}_t^0$ in neighborhood of the initial values which are rotation and translation parameters obtained at the initialization step. Then a constant value $\pi_0^{(i)} = 1/N$ is given to every particle. After the initial frame, the tracking is performed based on the previous particles and the current image. The particle set in t th frame $(\boldsymbol{s}_t^i; \pi_t^i)$ is estimated based on the previous assumption set $(\boldsymbol{s}_{t-1}^i; \pi_{t-1}^i)$ and a motion model as following equation.

$$\boldsymbol{s}_t^i = \boldsymbol{s}_{t-1}^i + v_{t-1} + \mu \tag{6.1}$$

where, v_{t-1} is velocity of the camera and represents the distance from t–2 th frame to t–1 th frame. μ is random noise. In our method, each particle is moved from previous sample based on the concept that the camera moves with uniform motion v_{t-1} and is diffused by adding random noise μ to become the particle in t th frame.

After obtaining N new particles $\{\boldsymbol{s}_t^i\}$, we compute the corresponding weight values π_t^i for $\boldsymbol{s}_t^i$ by evaluation based on the current image frame. The weight values mean the level of confidence for the corresponding particles. Therefore we give larger value to the particle which is closer to the truth. In our method we use the contour of the marker and evaluate how far the contour in the input image from the projected contour by each particle. In particular, we compute distances from the sampled K points on the contour to the edge in the input image and similarities between the regions around M feature points in the front-view image and the original model image. The detail will be described in the next section.

Finally we consider the particle $\boldsymbol{s}_t^i$ which has the maximum weight value as the camera pose $\boldsymbol{p}_t = (\phi_t, \theta_t, \psi_t, x_t, y_t, z_t)^\top$ in t th image.

6.5 Computing Weight Values

As described in the previous section, the weight values are the level of confidence for the corresponding particles. Therefore we give big value to the particle which seems to be close to the truth; give smaller value to the particle which seems to be far from the truth. After computing the weight values for all particles based on edges and patterns as $\pi e_t^{(i)}$ and $\pi p_t^{(i)}$, respectively, the average value of edge-based value and pattern-based value becomes the final weight value of the particle.

$$\pi_t^{(i)} = \frac{\pi e_t^{(i)} + \pi p_t^{(i)}}{2} \tag{6.2}$$

The particle which has the largest weight value is selected as the rotation and translation parameters in the current frame. Because of evaluating parameters by checking how fit each particle is to the actual input image, we can always search the best parameters at every frame.

6.5.1 Weight Value based on Edge

The flow of processing is shown in Fig. 6.4. First, edges are detected from the input image by Canny operator. K feature points shown in Fig. 6.3(a) are projected onto the edge image by using the parameters of each particle. If the parameters of a particle are completely correct, the projected points should be on the edge lines in the image. Therefore the distance between each projected point j and the nearest edge in the input image is computed as d_j $(1 \leq j \leq K)$. The nearest edge is searched along the perpendicular line to the contour as shown in Fig. 6.5. In particular, the searching is started from the projected point to both sides of the line.

The sum of the distance d_j for every feature points obtained by the parameters of the particle i is normalized between -1 and 1. The score value is considered as $c_t^{(i)}$.

$$c_t^{(i)} = 1 - \frac{2\sum_{j=1}^{K} d_j}{max} \qquad (-1 \leq c_t^{(i)} \leq 1) \tag{6.3}$$

where, max is a distance from the projected point to the end of the searching line. Therefore the score value $c_t^{(i)}$ of the particle which has closer parameters to the true parameters is

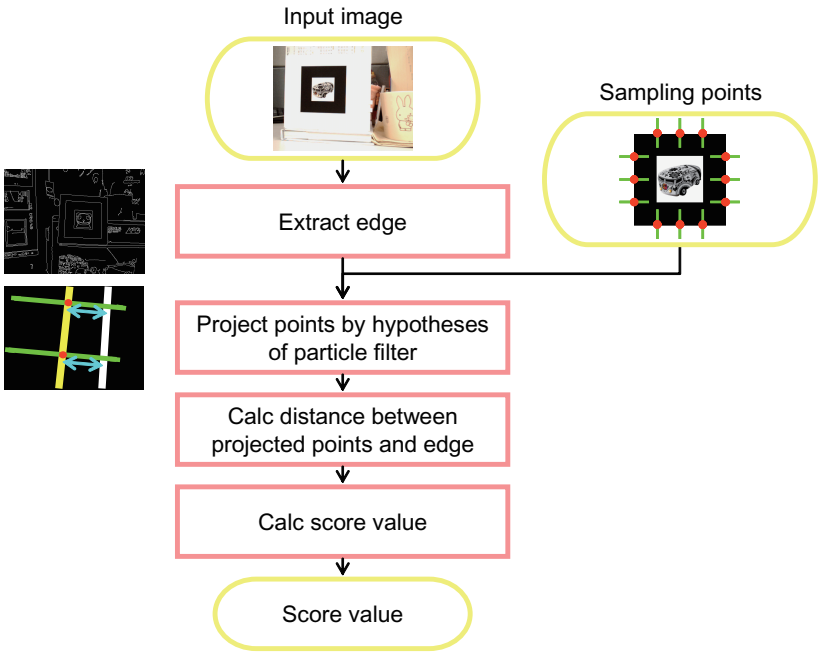


Fig. 6.4 Computing weight value based on edge.

closer to 1, conversely, the score value of the particle which is far away from the truth is close to -1.

After computing the score values $c_t^{(i)}$ for N particles, weight value $\pi e_t^{(i)}$ is computed by gaussian function as following equation.

$$\pi e_t^{(i)} \propto e^{-\frac{\left(1-c_t^{(i)}\right)^2}{2\sigma^2}} \tag{6.4}$$

Where, σ is standard deviation of Gaussian function. In our experiment, we let $\sigma = 3.0$.

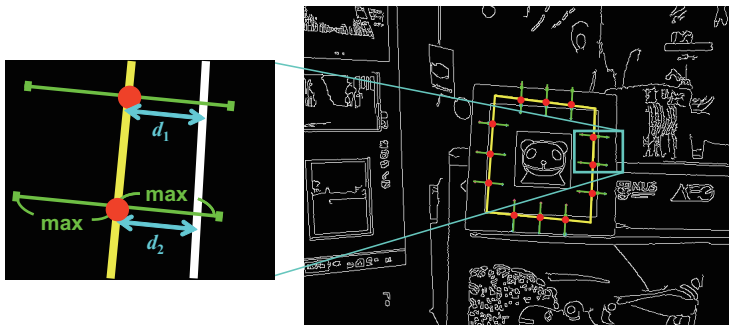


Fig. 6.5 Distance between projected points and edges.

Each $\pi e_t^{(i)}$ is normalized so that the sum of all the $\pi e_t^{(i)}$ becomes 1.0. Therefore the particle which has closer to the true will obtain bigger weight value $\pi e_t^{(i)}$.

6.5.2 Weight Value based on Pattern

The flow of processing is shown in Fig. 6.6. First, M feature points in the model image shown in Fig. 6.3(b) are projected onto the input image by using the parameters of each particle. The regions around the projected points j are transformed to front-view image. This transformation is done by homography computed from the particle's parameters. Then the regions corresponding to the front-view regions are extracted from the original model image as shown in Fig. 6.7. By comparing both of regions, then, the similarity is computed as f_j

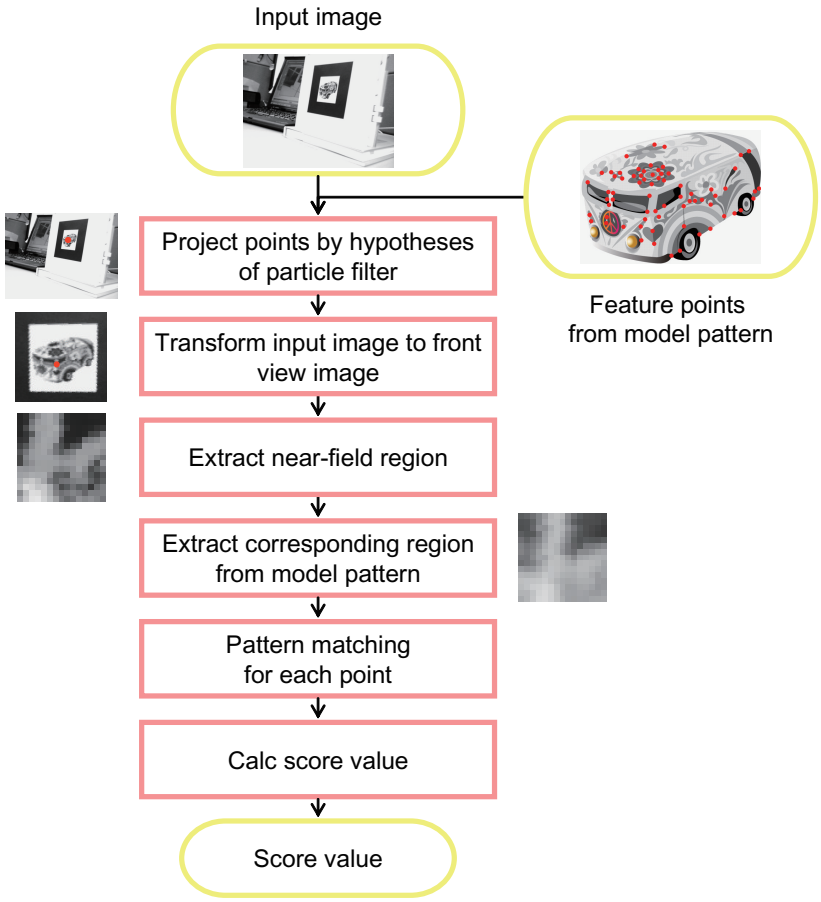


Fig. 6.6 Compting weight value based on pattern.

$(1 \leq j \leq M)$. In this method, they are compared based on NCC. Since each f_j has the value between -1 and 1, the sum of f_j for every feature point obtained by using the particle i is normalized between -1 and 1 as follows.

$$c_t^{(i)} = \frac{\sum_{j=1}^{M} f_j}{M} \qquad (-1 \leq c_t^{(i)} \leq 1) \tag{6.5}$$

After computing the score values $c_t^{(i)}$ for N particles, weight value $\pi_t^{(i)}$ is computed by gaussian function as following equation.

$$\pi p_t^{(i)} \propto e^{-\frac{\left(1-c_t^{(i)}\right)^2}{2\sigma^2}} \tag{6.6}$$

As same as the edge-based approach, we let $\sigma = 3.0$. Each $\pi p_t^{(i)}$ is normalized so that the sum of all the $\pi p_t^{(i)}$ becomes 1.0. Therefore the particle which has closer to the true will obtain bigger weight value $\pi p_t^{(i)}$.

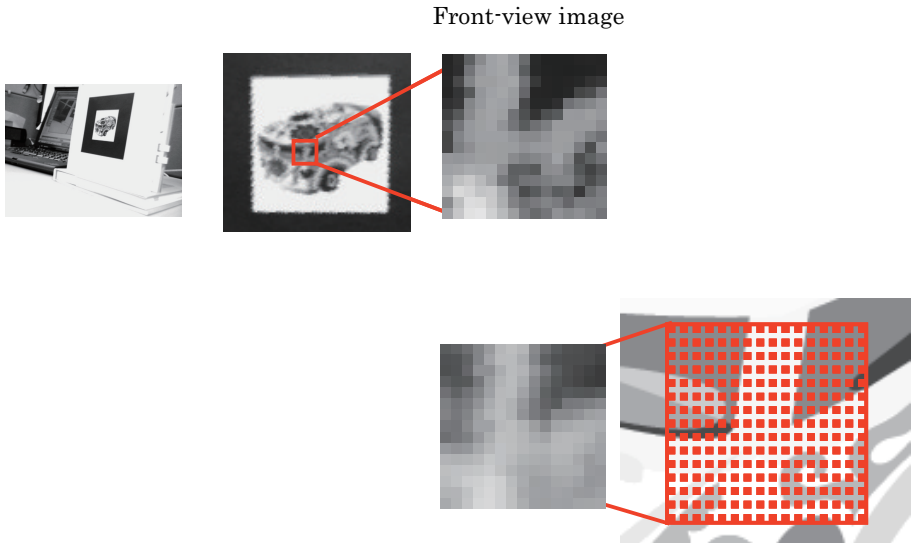
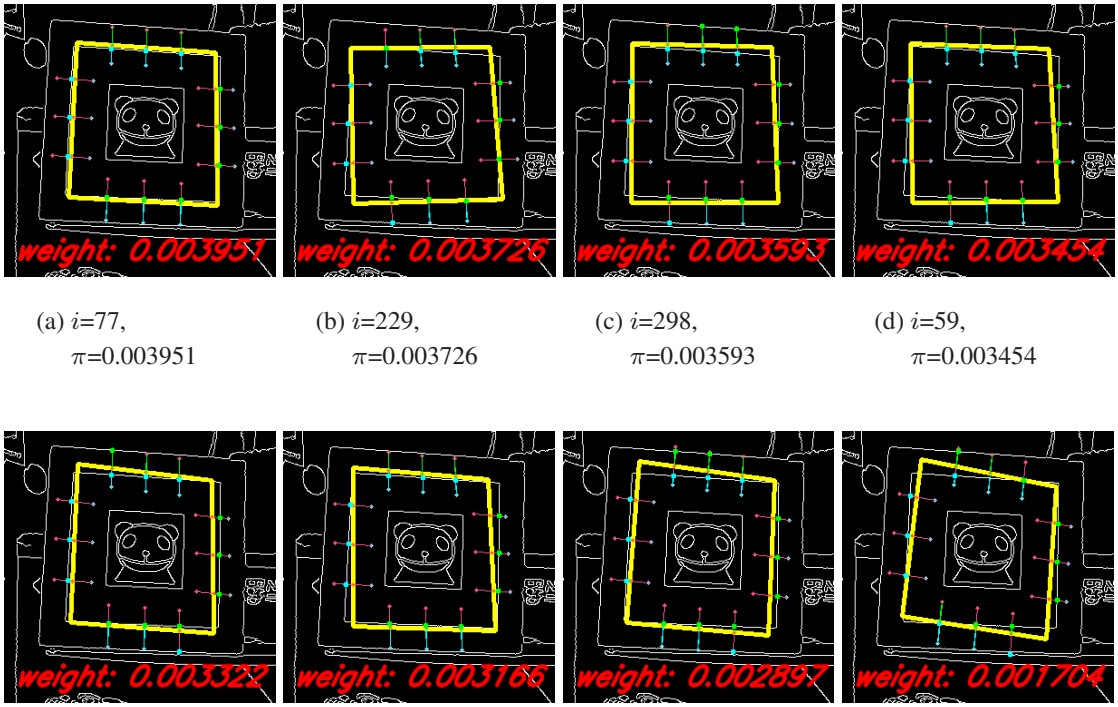


Fig. 6.7 Regions around projected points and feature points.

6.6 Experimental Results

We have implemented some experiments to evaluate our estimation method. Our system consists of a PC (OS: Windows XP, CPU: Intel Pentium 4.3 GHz) and a USB camera whose resolution is 640×480 pixel. The size of a 2D marker rectangular marker is 80×80 mm. The number of particles is set to 300. Under this condition the frame rate is 15 fps.

First, the resulting images of computing weight values in particle filter are shown in Fig. 6.8. Yellow rectangular in the image is the marker's contour projected by the parameters of each particle. The nearest edge is searched along each perpendicular line to the contour. Fig. 6.8(a) is the result image with maximum weight value. You can see that the projected yellow contour is extremely fit to the actual contour. In contrast, in Fig. 6.8(h) which is the



(a) i=77, π=0.003951
(b) i=229, π=0.003726
(c) i=298, π=0.003593
(d) i=59, π=0.003454
(e) i=164, π=0.003322
(f) i=100, π=0.003166
(g) i=232, π=0.002897
(h) i=175, π=0.001704

Fig. 6.8 Resulting images of computing weight values for all particles. i: index of particles $(0 \leq i \leq 299)$; π: weight value $(0 \leq \pi \leq 1)$.

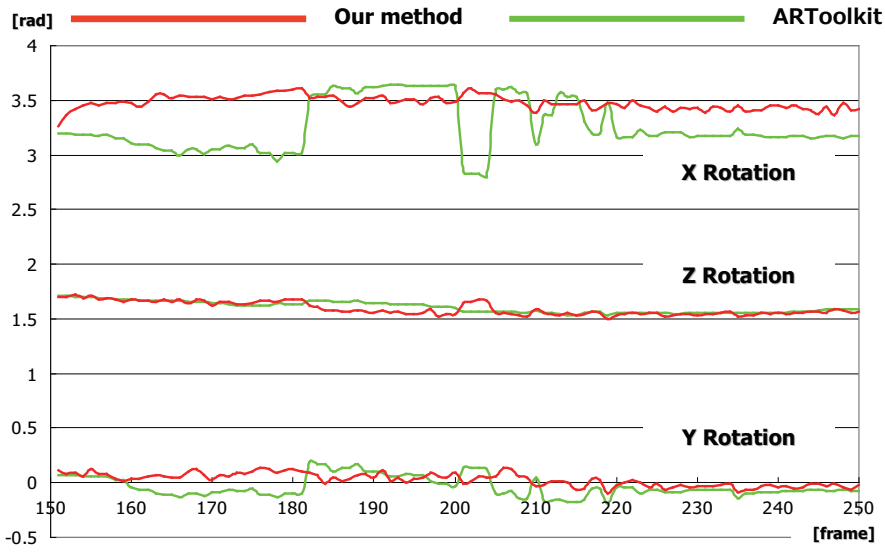


Fig. 6.9 Estimation result of parameters from the images whose viewpoint is perpendicular to the marker plane.

result image with minimum weight value, since the projected yellow contour is deformed, the contour is not fit to the actual contour at all. From these resulting images, therefore, you can find that our method can correctly assign the weight values according to the difference of appearance between the actual input image and the projected contour from the particles.

Next, we apply our method to the image sequence in which the marker is captured from the moving camera whose view direction is almost perpendicular to the marker plane as shown in Fig. 6.1(b). The camera is moved as smooth as possible. Fig. 6.9 shows a tracking result of rotation parameters about X, Y and Z axes. The direction of Z axis is decided by rotations of X and Y axes. Fig. 6.18 shows resulting images in which cubes are projected on the marker by using the parameters computed from ARToolkit (green cube) and our method (red cube). The resulting images of the cubes while representative three frames are shown in below of Fig. 6.9. These frames notably indicate the difference of results between our method and ARToolkit.

Comparing the rotation parameters by our method and ARToolkit, each parameter from our method smoothly changes according to the camera motion. In contrast, the rotation parameters of X and Y from ARToolkit are not stable and rapidly changing, for example in the three frames shown in Fig. 6.9, in which the green cube projected by ARToolkit is signifi-

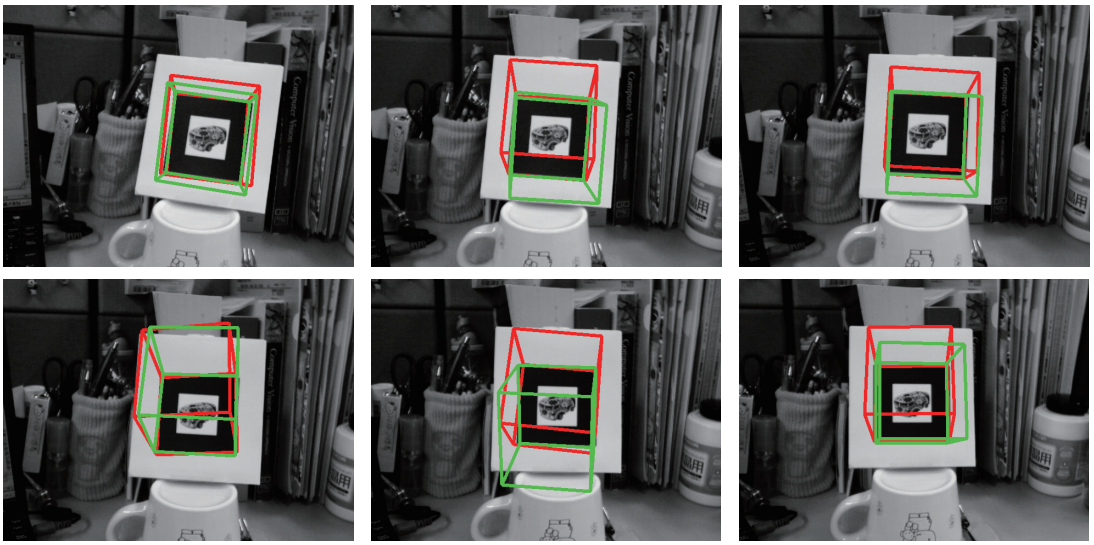


Fig. 6.10 Resulting images of overlaying cubes by ARToolkit and our method. Green cube: by ARToolkit; Red cube: by our method.

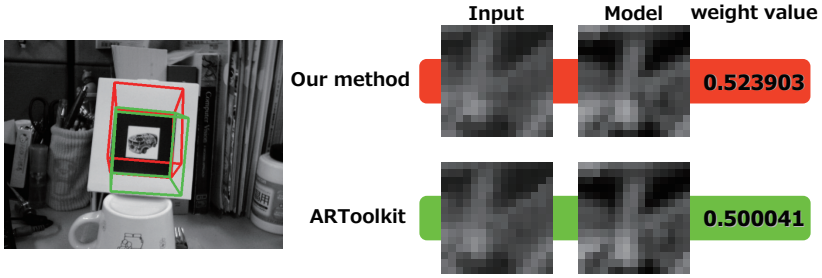


Fig. 6.11 Result of pattern-based weight value

cantly inclined compared to the previous and after frame. As described before, the rotation parameters of X and Y axes decide the direction of Z axis. Therefore accurate estimation of the rotations of X and Y axes is very important to improve tracking accuracy when the view direction of the camera is perpendicular to the marker plane. Our method achieves accurate estimation by using particle filtering. The rapid changes in the rotation parameters also cause some jitters between frames. You can also see the difference between our method and ARToolkit in Fig. 6.18. The (green) cube projected by ARToolkit is unstable with jitters, however the (red) cube projected by our method is stably aligned with the same position and pose. This is due to accurate estimation of the direction of Z axis.

We also apply the same comparison experiment to another image sequence, in which the

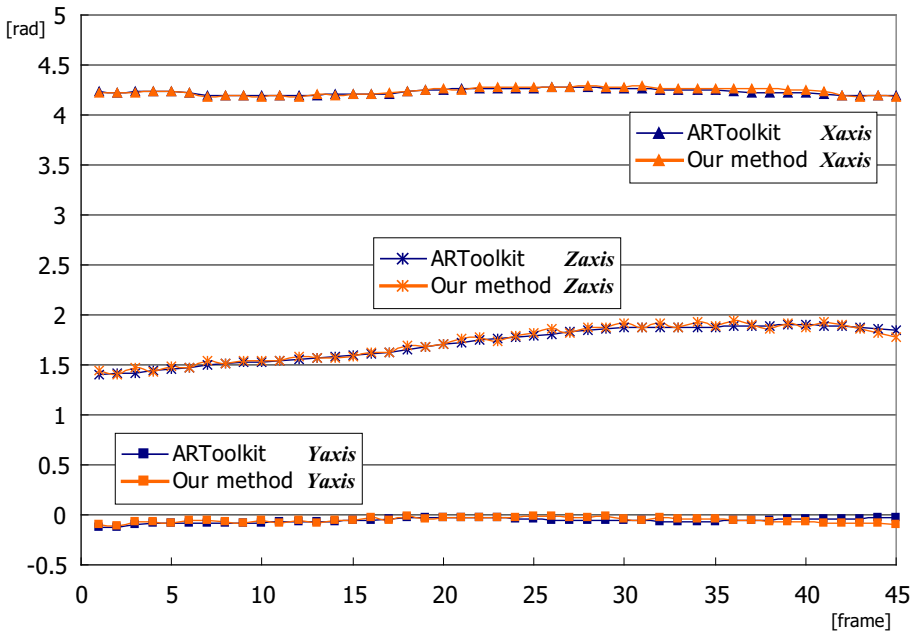


Fig. 6.12 Estimation result of rotation parameters from the images captured from a slanted viewpoint.

marker is captured from a slanted viewpoint as shown in Fig. 6.1(a). It is known that ARToolkit can correctly estimate each parameter from this kind of image sequence. Here, therefore, we demonstrate that our method can also correctly estimate the rotation parameters without changing the algorithm. Fig. 6.12 also shows a tracking result of rotation parameters about X, Y and Z axes, and Fig. 6.13 shows the resulting images of projected cubes: by ARToolkit (green cube) and our method (red cube).

Comparing the results in Fig. 6.12, both of the results of our method and ARToolkit are quite similar transitions. In the same way, the both cubes in Fig. 6.13 are projected on almost the same position and pose. Therefore the rotation parameters can be correctly estimated by our method as well as ARToolkit. This result indicates that our method using the particle filtering can also be applied to this kind of image sequence without changing the algorithm. It is effective to apply our method to a lot of marker-based AR approaches.

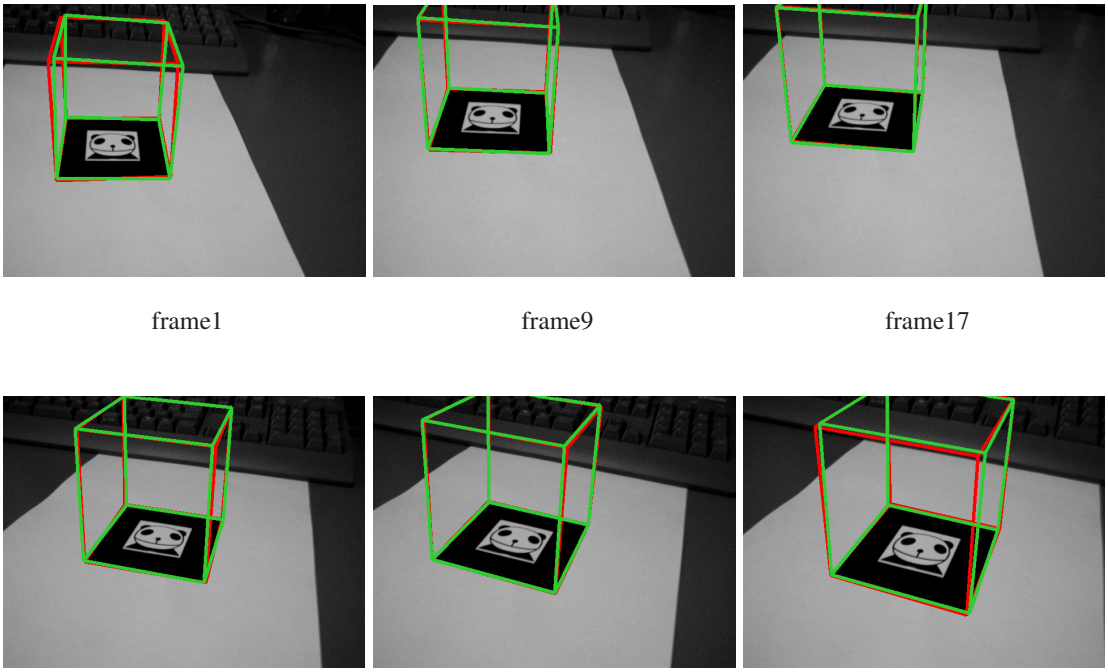


Fig. 6.13 Resulting images of overlaying cubes by ARToolkit and our method. Green cube: by ARToolkit; Red cube: by our method.

6.7 AR Mapping System of GIS Information

Geographical Information Systems (GIS) are databases that include a spatial component. They have become unavoidable tools for the development and management of territories. They are used for diagnosis, public debate and as decision making tools. The goal of GIS is to provide information about a given space. They include many types of geographical objects organized in themes frequently represented as layers (tiers architecture) containing objects of the same type (roads, buildings). GIS data can be vectorial as shown in Fig. 6.14 (a) or raster data as shown in Fig. 6.14 (b).

We introduce AR mapping system of GIS data on the captured images. When a user watches the scene of the town from a high place through a camera phone, the map information about town, such as the name of the buildings, roads, rivers etc., is overlaid onto the image in the display. Therefore this system is available for the navigation or the sightseeing guide.

For overlaying such GIS map information which is generated with CG onto the captured images, the position and pose of the camera has to be estimated at every frame. Such geometrical registration problem is one of the most important issues for implementing AR applications. Therefore, we utilize the framework of particle filtering which is previously proposed in Sec. 6.2.

As described before, the particle filter has some hypotheses (particles) which are sampled candidates. It is assumed that the correct value is probably included in the candidates. The confidence of each particle is evaluated at every frame and is propagated to the next frame. Therefore how to evaluate the confidence is the key of this method.

For evaluating the hypotheses (particles) of the parameters, this method uses the top-view

(a) Vectorial data (b) Raster data

Fig. 6.14 GIS database

image which is captured from an airplane as shown in Fig. 6.15(a) and is included in the GIS database. By comparing the top-view image with the input image captured by a user as shown in Fig. 6.15(b), the position and pose of camera are computed. In particular, we focus on the planar area of a building, such as a roof, and transform the top-view image to the input image by planar transformation (homography). Then the transformed top-view image is compared with the input image.

(a) Top-view image captured by airplane

(b) Captured image by user from high place

Fig. 6.15 Images utilized in this system

6.7.1 Computation of Weight Value

Fig. 6.16 shows the overview of computing pattern-based weight value. From the top-view image, feature points are detected by using OpenCV library function. The feature points are projected onto the input image by using the parameters of each particle. Since these projected points become corresponding points at every frame, feature tracking is not necessary. In the pattern-based computation described in Sec. 6.5.2, the input images captured from various viewpoints are transformed to the front-view image. On the other hand, this system transforms the top-view image to the input images. The information of a part of the scene is lost in the input image because the input images are captured from an angle. Therefore the input image cannot be transformed to the top-view image enough to be compared. Then the regions around the feature points in the transformed top-view image are extracted. The regions around the projected points in the input image are also extracted to be compared as

shown in Fig. 6.16.

If the parameters of the selected particle are completely correct, the regions around the projected points on the input image are consistent with the region around the feature points on the top-view image. Therefore, the difference between the regions becomes the weight value for the particle. The difference value for each feature point j between the top-view image's region and the input image's region is computed based on NCC as f_j $(1 \leq j \leq M)$. Since each f_j has the value between -1 and 1, the sum of f_j for every feature point obtained by using the particle i is normalized between -1 and 1 as follows.

$$w_{patt}^{(i)} = \frac{\sum_{j=1}^{M} f_j}{M} \qquad (-1 \leq w_{patt}^{(i)} \leq 1) \tag{6.7}$$

The score value $w_{patt}^{(i)}$ of the particle which has closer to the true parameters is closer to 1, conversely, the score value of the particle which is far away from the truth is close to -1.

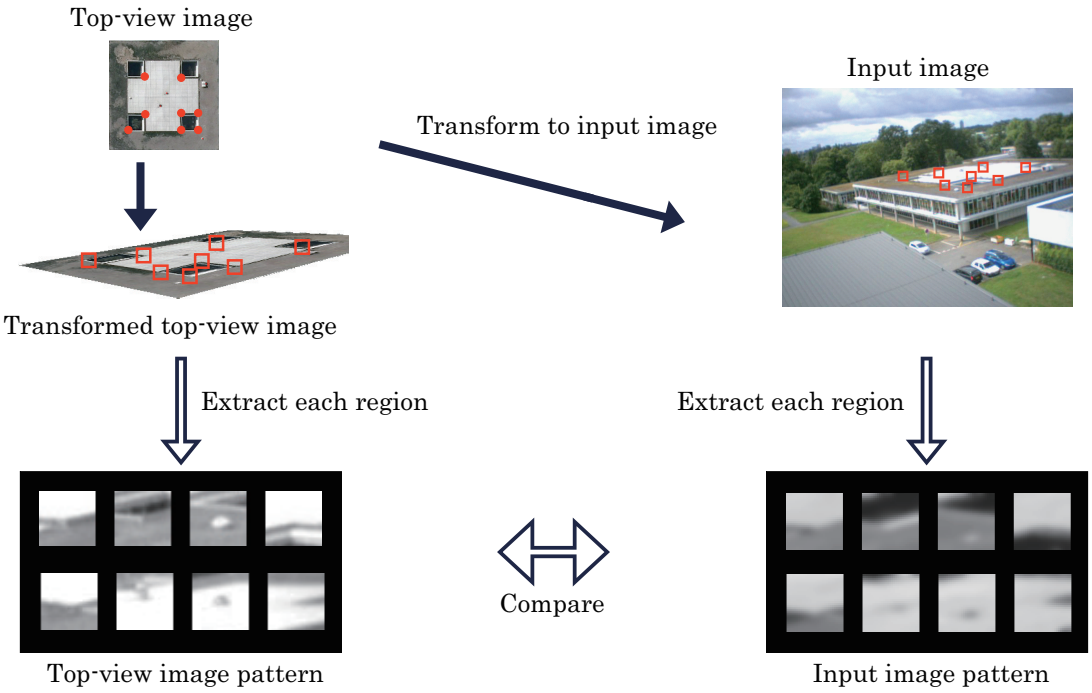


Fig. 6.16 Computation of pattern-based weight value.

6.7.2 Experimental Results

In this section, the experimental results are shown. We apply our method to the image sequence of the campus of Ecole Centrale de Nantes which is captured from a high building. Then map information of the campus is overlaid onto the image sequence as shown in Fig. 6.17.

(a) Input image sequence (b) Map of ECN

Fig. 6.17 Input images and overlaid map information.

First, 8 feature points are detected from the top-view image, and then the corresponding points are specified on the first frame of the input images. By using the feature points, the initial homography and the initial parameters of the camera are computed. By generating the hypotheses from the initial parameters, the particle filter starts to estimate the best parameters from the hypotheses.

Fig. 6.18 shows the resulting image of overlaying the map information. The map information is almost overlaid onto the correct positions of the buildings' roofs. Because this method transforms the map information by using planar-transformation (homography), the information of the buildings which are not the same height as the center building can not be overlaid onto the correct position in the present algorithm. If we can use the information about the buildings' height, we can solve the problem. Fig. 6.19 shows another resulting images which fail in the registration. This is method evaluates the particles using only pattern matching (NCC). However, such a local matching does not work when the object pattern does not have distinctive texture or features. Therefore, we will also utilize histogram of the pattern to evaluate the particles in the future work.

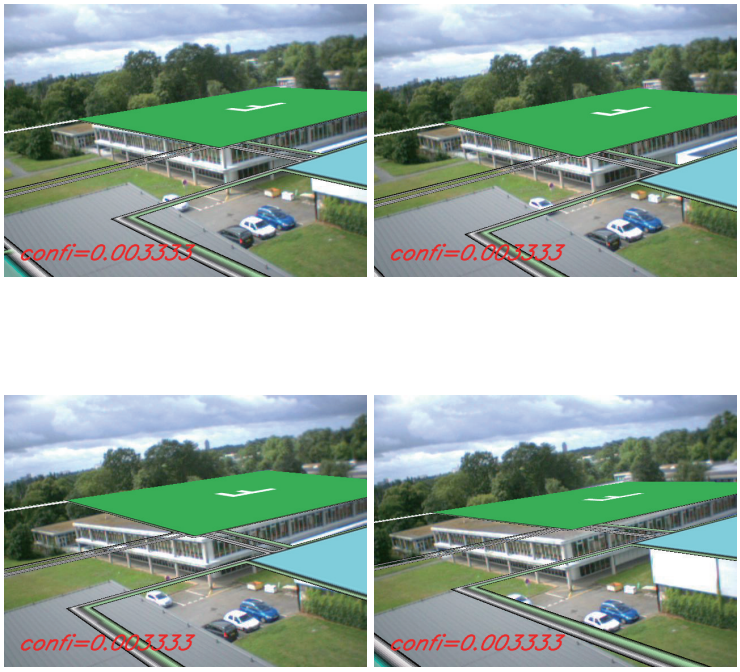


Fig. 6.18 Overlaid image sequence of map information.

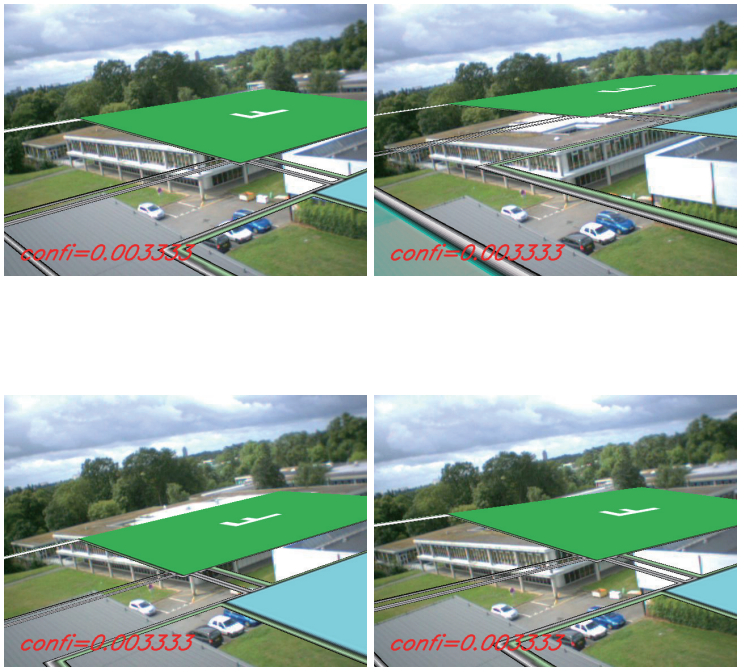


Fig. 6.19 Resulting frames that fail in registration of map information.

Chapter 7 :

Conclusions

7.1 Summary

We have proposed a camera tracking method using multiple planes in arbitrary positions and directions in the real world. The planar structures in the real world, such as a floor, wall and roof etc., are often utilized for such camera tracking, however, related works require the geometrical relationship information of the planes. Therefore there are some limitations on the arrangement of the planes that is the planes should be perpendicular each other or manual measurement is necessary in advance. On the other hand, our tracking method does not require such limitation on the arrangement of the planes or manual measuring task. By constructing Projective Space via two reference images, the geometrical relationship of the planes can be estimated even though we do not have a prior-knowledge. Therefore our method can be applied not only to a small space like a tabletop but also to a real scene which consists of complicated structures of planes or a large space which is too large to measure the arrangements by manual. Furthermore, camera motion can be tracked frame-by-frame without using all frames of the input image sequence. This means the algorithm can be applied to on-line AR systems.

We have extended our tracking method to present an on-line AR system based on multiple-marker-based registration. The multiple markers used in our system can be placed at arbitrary positions and poses, so that we do not need to measure their geometrical arrangement in advance because of the merging process via a Projective Space. Therefore this system can be constructed very easily, unlike most marker-based approaches which require a priori knowledge about the multiple markers. Furthermore we can move a camera freely according to the widely moving virtual object, and the registration can be continued independently of the camera's viewpoint in real-time. Placing the markers at any positions enables to achieve the registration in the wide area. Making the markers face to various directions also allows keeping the accurate registration stably. Thus our system can highly contribute as an effective AR system.

We have also demonstrated the performance of the proposed method by applying it to two AR applications. Both of the baseball system and bowling system can be enjoyed on the tabletop in the real world only with a web-camera and a handheld monitor connected a general PC. It is a big advantage for home users that these applications do not require any special device. Users can interactively change their viewpoints by moving around the tabletop

because of this registration method with multiple markers. In contrast with usual CG viewers in which a mouse or a keyboard is used for changing viewpoints, it is very intuitive and easy to changing viewpoints by moving of the users. By visualizing 3D objects in front of the users, these applications will be future-oriented 3D game.

Finally we have proposed a marker-based camera tracking method which improves the tracking accuracy of 3D camera pose, especially when the camera's image plane is almost parallel to the marker plane. Our method estimates the rotation parameters of the camera by comparing the actual input image and the generated pattern by a lot of hypotheses of the parameters by employing the particle filtering technique. The 3D axes of the coordinate system fixed on the marker plane, especially Z axis, are stably tracked all over the frames, while some jitters are caused by using only ARToolkit. Moreover, our method can be applied not only the images in which marker plane is almost perpendicular to optical axis of a camera, but also the images captured from an inclined view point without changing the algorithm.

7.2 Future Work

If the proposed algorithm is applied to practical use, the main concern is feature point tracking on the planes. In the natural scene, there are not so many feature points to be continuously detected and tracked in real-time. Although a lot of feature tracking methods have been proposed so far, there is trade-off between the processing speed and tracking accuracy. Our method described in Chapter 3 puts priority to the tracking accuracy. Therefore feature points are tracked in off-line processing to reject bad correspondences. Since the estimation of the planar structures using the Projective Space is just pre-processing and camera parameters are computed frame by frame, our algorithm can be implemented in real-time. If a high-speed and accurate feature tracking is proposed or computational time of PC is improved, our method can be utilized in the space consists of multiple planar structures without 3D configuration information.

As for the proposed tabletop AR applications, the system environment should be considered. In our applications, we assume a handheld display and web-camera to provide the augmentation views. This is one style of AR usage. Recently, a digital camera is installed on commercial cellular phones and people can easily use a device which has both display and camera. Such a camera phone is quite suitable for AR applications. Some applications have already been proposed so far. In order to practically use the camera phone for AR, a server-client style might be employed to reduce the computation task of client side such as computational speed, graphical environment, amount of memory and so on. The application which uses the information on the web or database is suitable for this kind of style. The baseball system described in Sec. 5.2 can download the score data of today's game from the web site and replay the game in front of the users. The GIS system described in Sec. 6.7 can use GIS information from the web database. The latest data is available because of easy update.

For using user's both hands, other solution is necessary instead of the handheld display including the camera phone. A kind of computerized grasses such as HMD is one possibility. In this case, we have to consider not only generating stereoscopic view but also gaze direction to decide the focus area of virtual objects. Since our algorithm can compute the extrinsic parameters, the disparity and focal length may be easily obtained. This kind of applications will be available not only in the amusement park but also at usual home.

Chapter 8 :

Appendix

Stereo Algorithm in Assigning 3D Coordinate Systems

$\boldsymbol{P_A}$ and $\boldsymbol{P_B}$ are projection matrices which are computed from four clicked points on the base plane and relates the base plane to the reference image A and B, respectively.

$$\boldsymbol{P_A} = \begin{bmatrix} p_{11}^A & p_{12}^A & p_{13}^A & p_{14}^A \\ p_{21}^A & p_{22}^A & p_{23}^A & p_{24}^A \\ p_{31}^A & p_{32}^A & p_{33}^A & 1 \end{bmatrix}, \qquad \boldsymbol{P_B} = \begin{bmatrix} p_{11}^B & p_{12}^B & p_{13}^B & p_{14}^B \\ p_{21}^B & p_{22}^B & p_{23}^B & p_{24}^B \\ p_{31}^B & p_{32}^B & p_{33}^B & 1 \end{bmatrix} \tag{8.1}$$

We assume that a 2D coordinate of a point which is clicked on other plane in the reference image A and B are (x_A, y_A) and (x_B, y_B), respectively. The 3D coordinate of the point (X, Y, Z) can be computed by following equation.

$$\begin{bmatrix} (p_{31}^A x_A - p_{11}^A) & (p_{32}^A x_A - p_{12}^A) & (p_{33}^A x_A - p_{13}^A) \\ (p_{31}^A y_A - p_{11}^A) & (p_{32}^A y_A - p_{12}^A) & (p_{33}^A y_A - p_{13}^A) \\ (p_{31}^B x_B - p_{11}^B) & (p_{32}^B x_B - p_{12}^B) & (p_{33}^B x_B - p_{13}^B) \\ (p_{31}^B y_B - p_{11}^B) & (p_{32}^B y_B - p_{12}^B) & (p_{33}^B y_B - p_{13}^B) \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} p_{14}^A - p_{34}^A x_A \\ p_{14}^A - p_{34}^A y_A \\ p_{14}^B - p_{34}^B x_B \\ p_{14}^B - p_{34}^B y_B \end{bmatrix} \tag{8.2}$$

In the same way, 3D coordinates of the other clicked points on the plane can be computed. Then we define a 3D coordinate system on the plane so that one of the four points becomes the origin of the coordinate system. Therefore, we can define the 3D coordinates of the four points which are used for computing Homography in the same scale as the base plane's coordinate system.

Reference

[1] R. T. Azuma, “A Survey of Augmented Reality”, Presence, pp. 355–385, 1997.

[2] H. Tamura, and Y. Ohta, “Mixed Reality”, Journal of the Institute of Image Information and Television Engineers, vol. 52, no. 3, pp. 266–272, 1998 (in Japanese).

[3] R. T. Azuma, “Recent Advances in Augmented Reality”, IEEE Computer Graphics and Applications, vol. 21, no. 6, pp. 34–47, 2001.

[4] “Orad”, http://www.orad.co.il/.

[5] “NHK”, http://www.nhk.or.jp/strl/index.html.

[6] “Sport Vision”, http://www.sportvision.com/.

[7] “Symah Vision”, http://www.epsis.com/.

[8] T. Misu, M. Takahashi, S. Gohshi, M. Tadenuma, Y. Fujita, and N. Yagi, “Visualization of Offside Lines Based on Realtime Video Processing”, Transactions of the Institute of Electronics, Information and Communication Engineers D-II, vol. 88, no. 8, pp. 1681–1692, 2005 (in Japanese).

[9] H. Habu, and D. Kato, “Virtual Confrontation System (New Technology for Nagano Olympic Winter Games)”, Journal of the Institute of Image Information and Television Engineers, vol. 52, no. 6, pp. 797–798, 1998 (in Japanese).

[10] “Simul Cam, Stro Motion”, http://www.dartfish.com/.

[11] “BMW Augmented Reality”, http://www.bmw.com/com/en/owners/service/augmented_reality_introduction_1.html.

[12] J. Platonov, H. Heibel, P. Meier, and B. Grollmann, “A mobile markerless AR system for maintenance and repair”, Proc. of the 5th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’06), pp. 105–108, 2006.

[13] “Virtual Mirror”, http://www.ray-ban.com/.

[14] “Augmented Reality Team”, http://virtual.vtt.fi/virtual/proj2/multimedia/index.html.

[15] P. Georgeli, P. Schroederi, S. Benhimanei, S. Hinterstoisseri, M. Appel, and N. Navab, "An Industrial Augmented Reality Solution For Discrepancy Check", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 111–115, 2007.

[16] M. Billinghurst, "Augmented reality in education", new horizons for learning, http://it.civil.aau.dk/it/education/reports/ar_edu.pdf, 2002.

[17] S. Prince, et al., "3D Live: Real Time Captured Content for Mixed Reality", Proc. of the IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 7–13, 2002.

[18] E. Klopfer, Augmented Learning: Research and Design of Mobile Educational Games, MIT Press2008.

[19] F. Morgan, "Developing a New Medical Augmented Reality System", tech. report CMU-RI-TR-96-19, Robotics Institute, Carnegie Mellon University, 1996.

[20] R. Bernhard, P. Werlberger, B. Alexander, B. Reinhard, and S. Dieter, "Spatial Measurements for Medical Augmented Reality", Proc. of the 4th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'05), pp. 208–209, 2005.

[21] "Augmented-Reality-Medical-Visualization-Research", http://www.cs.unc.edu/Research/us/.

[22] C. Bichlmeier, S. Heining, M. Rustaee, and N. Navab, "Laparoscopic Virtual Mirror for Understanding Vessel Structure Evaluation Study by Twelve Surgeons", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 125–128, 2007.

[23] C. Bichlmeier, F. Wimme, S. Heining, and N. Navab, "Contextual Anatomic Mimesis: Hybrid In-Situ Visualization Method for Improving Multi-Sensory Depth Perception in Medical Augmented Reality", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 129–138, 2007.

[24] A. D. Cheok, et al., "Human Pacman: A Mobile Entertainment System with Ubiquitous Computingand Tangible Interaction over a Wide Outdoor Area", Personal and Ubiquitous Computing, vol. 8, no. 2, pp. 71–81, 2004.

[25] M. Haller, et al., "Coeno-Enhancing face-to-face collaboration", Proc. of the ICAT, pp. 40–47, 2005.

[26] S. Minatani, I. Kitahara, Y. Kameda, and Y. Ohta, "Face-to-Face Tabletop Remote Collaboration in Mixed Reality", Proc. of the 6th IEEE and ACM Symposium on Mixed

and Augmented Reality (ISMAR'07), pp. 43–46, 2007.

[27] S. Uchiyama, K. Takemoto, H. Yamamoto, and H. Tamura, "MR Platform: A Basic Body on Which Mixed Reality Applications Are Built", Proc. of the IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 246–253, 2002.

[28] T. Ohshima, K. Satoh, H. Yamamoto, and H. Tamura, "AR2Hockey: A Case Study of Collaborative Augmented Reality", Proc. of IEEE Virtual Reality Annual International Symposium, pp. 268–275, 1998.

[29] "Nature Contact", http://www.expo2005.or.jp/.

[30] A. State, G. Hirota, D. Chen, W. Garett, and M. Livingston, "Superior Augmented Reality Registration by Integrating Landmark Tracking and Magnetic Tracking", Proc. of SIGGRAGH'96, pp. 429–438, 1996.

[31] R. Miguel, P. Lang, H. Ganster, M. Brandner, C. Stock, and A. Pinz, "Hybrid Tracking for Outdoor Augmented Reality Applications", IEEE Computer Graphics and Applications, pp. 54–63, 2002.

[32] Y. Seo, and K. S. Hong, "Calibration-free augmented reality in perspective", IEEE Trans. on Visualization and Computer Graphics, vol. 6, no. 4, pp. 346–359, 2000.

[33] U. Neumann, and S. You, "Natural Feature Tracking for Augmented Reality", IEEE Trans. on Multimadia, vol. 1, no. 1, pp. 53–64, 1999.

[34] K. W. Chia, A. Cheok, and S. J. D. Prince, "Online 6 DOF Augmented Reality Registration from Natural Features", Proc. of IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 305–313, 2002.

[35] I. Skrypnyk, and D. G. Lowe, "Scene modeling, recognition and tracking with invariant image features", Proc. of the 3rd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'04), pp. 110–119, 2004.

[36] G. Reitmayr, and T. W. Drummond, "Going out: robust mode-based tracking for outdoor augmented reality", Proc. of the 5th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'06), pp. 109–118, 2006.

[37] G. Klein, and D. Murray, "Parallel tracking and mapping for small ar workspaces", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 225–234, 2007.

[38] C. Harris, and M. Stephens, "A combined corner and edge detector", Proc. of Alvey Vision Conference, pp. 147–152, 1988.

[39] J. Shi, and C. Tomasi, "Good Features to track", IEEE Conf. on CVPR, pp. 593–600,

1994.

[40] D. G. Lowe, “Object recognition from local scaleinvariant features”, Proc. of IEEE International Conference on Computer Vision (ICCV), pp. 1150–1157, 1999.

[41] V. Lepetit, L. Vacchetti, D. Thalmann, and P. Fua, “Fully Automated and Stable Registration for Augmented Reality Applications”, Proc. of the 2nd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’03), pp. 93–102, 2003.

[42] T. Drummond, and R. Cipolla, “Real-time tracking of complex structures with on-line camera calibration”, Proc. of the BMVC, pp. 574–583, 1999.

[43] A. I. Comport, and F. Marchand, E. Chaumette, “A real-time tracker for markerless augmented reality”, Proc. of the 2nd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’03), pp. 36–45, 2003.

[44] G. Klein, and T. Drummond, “Sensor Fusion and Occlusion Refinement for Tablet-based AR”, Proc. of the 3rd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’04), pp. 38–47, 2004.

[45] P. David, D. DeMenthon, R. Duraiswami, and H. Samet, “Simultaneous pose and correspondence determination using line features”, Proc. of Computer Vision and Pattern Recognition (CVPR2003), pp. II 424–431, 2003.

[46] D. Kotake, S. Kiyoshide, S. Uchiyama, and H. Yamamoto, “A Fast Initialization Method for Edge-based Registration Using an Inclination Constraint”, Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’07), pp. 239–248, 2007.

[47] D. Gennery, “Visual Tracking of Known Three Dimensional Objects”, International journal of Computer Vision, vol. 7, no. 3, pp. 243–270, 1992.

[48] G. Simon, A. Fitzgibbon, and A. Zisserman, “Markerless Tracking using Planar Structures in the Scene”, Proc. of the ISAR, pp. 120–128, 2000.

[49] G. Simon, and M. Berger, “Reconstructing while registering: a novel approach for markerless augmented reality”, Proc. of the IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR’02), pp. 285–294, 2002.

[50] G. Simon, and M. O. Berger, “Real time registration known or recovered multi-planar structures: application to AR”, Proc. of the BMVC, pp. 567–576, 2002.

[51] S.Sugimoto, and M. Okutomi, “Fast Plane Parameter Estimation From Stereo Images”, Proc. the IAPR Conference on Machine Vision Applications (MVA2007), pp. 567–570, 2007.

[52] M. Billinghurst, et al., “Magic Book: Exploring Transitions in Collaborative AR Inter-

faces", Proc. of SIGGRAPH, 87, 2000.

[53] "ARTag", http://www.artag.net/.

[54] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, and K. Tachibana, "Virtual object manipulation on a table-top AR environment", Proc. of the ISAR, pp. 111–119, 2000.

[55] Y. Genc, S. Riedel, F. Souvannavong, C. Akinlar, and N. navab, "Marker-less Tracking for AR: A Learning-Based Approach", Proc. of the IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'02), pp. 295–304, 2002.

[56] E. Foxlin, and L. Naimark, "Miniaturization, Calibration & Accuracy Evaluation of a Hybrid Self-Tracker", Proc. of the 2nd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'03), pp. 151–160, 2003.

[57] E. Foxlin, and L. Naimark, "VIS-Traker: A Wearable Vision-Inertial Self-Tracker", Proc. of Virtual Reality, pp. 199–206, 2003.

[58] E. Foxlin, et al., "FlightTracker: A Novel Optical/Inertial Tracker for Cockpit Enhanced Vision", Proc. of the 3rd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'04), pp. 212–221, 2004.

[59] D. Kotake, et al., "A Marker Calibration Method Utilizing A Priori Knowledge on Marker Arrangement", Proc. of the 3rd IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'04), pp. 89–98, 2004.

[60] M. Billinghurst, and H. Kato, "Collaborative Mixed Reality", Proc. of ISMR, pp. 261–284, 1999.

[61] S. White, L. Lister, and S. Feiner, "Visual Hints for Tangible Gestures in Augmented Reality", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 47–50, 2007.

[62] J. Looser, R. Grasset, and M. Billinghurst, "A 3D Flexible and Tangible Magic Lens in Augmented Reality", Proc. of the 6th IEEE and ACM Symposium on Mixed and Augmented Reality (ISMAR'07), pp. 51–54, 2007.

[63] A. Henrysson, et al., "AR Tennis", Proc. of SIGGRAPH, 2006.

[64] Y. Uematsu, and H. Saito, "Vision-based Registration for Augmented Reality with Integration of Arbitrary Multiple Planes", Proc. of ICIAP, LNCS 3617, pp. 155–162, 2005.

[65] R. Hartley, and A. Zisserman, Multiple View Geometry in computer vision, CAMBRIDGE UNIVERSITY PRESS2000.

[66] Y. Uematsu, and H. Saito, "On-line Augmented Reality System by Integration of Multiple Planar Markers Placed in Arbitrary Positions and Directions", Journal of the Institute

of Image Information and Television Engineers, vol. 60, no. 6, pp. 909–919, 2006 (in Japanese).

[67] A. Henrysson, M. Billinghurst, and M. Ollila, “Virtual Object Manipulation using a Mobile Phone”, Proc. of the ICAT, pp. 164–171, 2005.

[68] D. Wagner, M. Billinghurst, and D. Schmalstieg, “How real should virtual characters be?”, Proc. the ACM SIGCHI international conference on Advances in computer entertainment technology (ACE2006), Article No. 57, 2006.

[69] Y. Uematsu, and H. Saito, “Visual Enhancement for Sports Entertainment by Vision-Based Augmented Reality”, Journal of Advances in Human-Computer Interaction, vol. 2008, ID 145363, 2008.

[70] H. Ishii, C. Wisneski, J. Orbanes, B. Chun, and J. Paradiso, “curlybot: designing a new class of computational toys”, Proceedings of the SIGCHI conference on Human factors in computing systems, pp. 394–401, 1999.

[71] B. Piper, and H. Ishii, “PegBlocks: a learning aid for the elementary classroom”, Proceedings of CHI ’02 extended abstracts on Human factors in computing systems, pp. 686–687, 2002.

[72] K. Ryokai, S. Marti, and H. Ishii, “Designing the world as your palette”, Proceedings of CHI ’05 extended abstracts on Human factors in computing systems, pp. 1037–1049, 2005.

[73] J. Zigelbaum, A. Millner, B. Desai, and H. Ishii, “BodyBeats: whole-body, musical interfaces for children”, Proceedings of CHI ’06 extended abstracts on Human factors in computing systems, pp. 1595–1600, 2006.

[74] Y. Uematsu, and H. Saito, “AR Registration by Merging Multiple Planar Markers at Arbitrary Positions and Poses via Projective Space”, Proc. of ICAT, pp. 48–55, 2005.

[75] C. Matysczok, et al., “AR-Bowling: Immersive and Realistic Game Play in Real Environments Using Augmented Reality”, Proc. of ACE, pp. 269–274, 2004.

[76] D. Forsyth, and J. Ponce, Computer Vision: A Modern Approach, Prentice Hall2003.

[77] “GLT ZPR”, http://www.nigels.com/glt/gltzpr/.

[78] M. Isard, and A. Blake, “Condensation-conditional density propagation for visual tracking”, Int. J. of Computer Vision, vol. 29, no. 1, pp. 5–28, 1998.

[79] Y. Matsubara, and K. Shakunaga, “Sparse Template Matching and Its Application to Real-time Object Tracking”, Transactions of Information Processing Society of Japan, vol. 46, no. SIG9, pp. 60–71, 2005 (in Japanese).

[80] K. Oka, Y. Sato, Y. Nakanishi, and H. Koike, “Head Pose Estimation System Based on Particle Filtering with Adaptive Diffusion Control”, Transactions of the Institute of Electronics, Information and Communication Engineers D-II, no. 8, pp. 1601–1613, 2005 (in Japanese).

[81] K. Oka, Y. Sato, Y. Nakanishi, and H. Koike, “Head Pose Estimation System Based on Particle Filtering with Adaptive Diffusion Control”, Proc. of IAPR Conf. Machine Vision Applications (MVA 2005), pp. 586–589, 2005.

[82] Y. Kobayashi, D. Sugimura, and Y. Sato, “3D Head Tracking using the Particle Filter with Cascaded Classifiers”, Proc. of British Machine Vision Conference (BMVC2006), pp. 37–46, 2006.

[83] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade, “Tracking in Low Frame Rate Video: A Cascade Particle Filter with Discriminative Observers of Different Lifespans”, Proc. of Computer Vision and Pattern Recognition (CVPR2007), 2007.

[84] M. Pupilli, and A. Calway, “Real-Time Camera Tracking Using a Particle Filter”, Proc. of British Machine Vision Conference (BMVC), pp. 519–528, 2005.

[85] D. Marimon, Y. Maret, Yousri, Abdeljaoued, and T. Ebrahimi, “Particle filter-based camera tracker fusing marker and feature point cues”, Proc. of IS&T/SPIE Conf. on Visual Communications and Image Processing, 2007.