

NAIST-IS-DT0061212

Doctor's Thesis

**Stable Three Point based Registration
for Vision-Based Augmented Reality
Using Monocular and Binocular Vision**

STEVE VALLERAND

February 6, 2004

Department of Information Systems
Graduate School of Information Science
Nara Institute of Science and Technology

NAIST-IS-DT0061212

Doctor's Thesis

Stable Three Point based Registration for Vision-Based Augmented Reality Using Monocular and Binocular Vision

STEVE VALLERAND

February 6, 2004

Department of Information Systems
Graduate School of Information Science
Nara Institute of Science and Technology

© Steve Vallerand, 2004

Doctor's Thesis
submitted to Graduate School of Information Science,
Nara Institute of Science and Technology
in partial fulfilment of the requirements for the degree of
DOCTOR of ENGINEERING

STEVE VALLERAND

Thesis committee: Naokazu Yokoya, Professor
Kunihiro Chihara, Professor
Kazumasa Yamazawa, Professor

Stable Three Point based Registration for Vision-Based Augmented Reality Using Monocular and Binocular Vision¹

STEVE VALLERAND

Abstract

In order to perform the registration of virtual objects in vision-based AR systems, the estimation of the relation between the real and virtual worlds is needed. This thesis suggests a vision-based registration method for video see-through augmented reality systems using stereo-paired cameras. A registration method is presented as an alternative method to keep producing the registration when only three points are available. In order to perform the three point based registration, both monocular and stereoscopic vision-based techniques are used. As a result, the registration method succeeds to perform the registration depth independently of the distance. Also, a correction method that performs a correction of the 2D positions in the images of the feature points is proposed in order to optimize the registration.

An augmented reality system using both registration and correction methods have been developed. The developed system uses a color based detection strategy in order to be robust to rapid user's motion. In addition, a new way to evaluate the registration is proposed to quantify the produced registration. Using this evaluation method, the correction is proven to improve the stability and accuracy of the proposed registration method. Also, the proposed registration method combined to the correction method produces better results compared to other three point based registration methods. Consequently, our proposed methods succeed to produce stable three point based registration and can be considered as interesting alternatives to produce the registration in augmented reality systems.

Keywords:

augmented reality, vision-based registration, three point registration, monocular vision, stereo vision, quantitative evaluation

1. Doctor's Thesis, Department of Information Systems, Graduate School of Information Science, Nara Institute of Science and Technology, NAIST-IS-DT0061212, February 6, 2004.

Acknowledgements

First, I would like to thank my supervisor Dr. Naokazu Yokoya for having accepted me as a student member of his laboratory and for all his guidance and advice.

Also, I would like to thank Dr. Masayuki Kanbara who was an essential source of information concerning either my research or the general aspects of living in Japan.

My sincere gratitude also goes to all my Japanese friends who made me enjoy my daily life, in particular, Fukutomi, Ikeda, Keiko and “Ichise san tachi”, Morita, Naoko, Sangsoo, Tenmoku, Yasuyo, and all the other members of the laboratory. Furthermore, I have a thought for all my good international friends: Cyrus, Damien “et la famille”, Elaine, Fatia, Fernando, Kent, Julie, Jonathan, Philippe, Supriato, Yannick.

Finally, I especially want to thank my wife, Josee, for her love, patience, courage, understanding and support during this long stay in Japan.

This research was supported by a scholarship from the Ministry of Education, Culture, Sports, Science and Technology.

Contents

Abstract	page i
Acknowledgements	page iii
Contents	page v
List of figures	page ix
List of tables	page xi
List of symbols	page xiii
1. Introduction	page 1
1.1 Augmented Reality	page 1
1.1.1 Augmented reality problems	page 2
1.1.2 Overlay methods	page 4
1.2 Definitions	page 6
1.2.1 Point, feature and marker	page 6
1.2.2 Tracking strategy using searching windows	page 7
1.2.3 Monocular vision-based registration	page 8
1.2.4 Stereoscopic vision-based registration	page 10
1.2.5 Toed-in compensation	page 12
1.3 Related Work	page 12
1.4 Motivation	page 16
1.4.1 Increasing the registration depth of stereoscopic systems	page 19
1.4.2 Correction of point positions	page 19
1.4.3 Coherence between the left and right registrations	page 20
1.4.4 Robustness to fast user's motion	page 20
1.4.5 Minimal number of points	page 21
1.4.6 Low cost hardware requirements	page 21
1.5 Proposed methods	page 22
2. Robust registration method for vision-based augmented reality using monocular and binocular vision with feature position correction	page 24
2.1 Overview	page 24
2.2 Marker extraction	page 25
2.2.1 Design of the markers	page 26
2.2.2 Marker extraction	page 28
2.2.3 Feature point positions	page 34

2.2.4 Marker identification and orientation	page 35
2.3 Registration method	page 41
2.3.1 Overview	page 41
2.3.2 Three point space resection problem (monocular)	page 44
2.3.3 Evaluation using the best solution approach (binocular)	page 45
2.3.4 Evaluation using the paired solution approach (binocular).	page 47
2.3.5 Three point based registration	page 49
2.3.6 Simplification of the registration.	page 52
2.4 Correction method.	page 53
2.4.1 Correction by true paired progression.	page 57
2.4.2 Adjustement of the correction factors	page 58
2.5 Conclusion	page 60
3. Experiment	page 61
3.1 System description	page 61
3.2 Measures to evaluate the registration quality	page 63
3.3 Evaluation of the robustness to fast user's motion	page 66
3.4 Evaluation of the registration method.	page 69
3.4.1 Effect of the distance.	page 69
3.4.2 Effect of the angle	page 75
3.4.3 Registration coherence	page 77
3.4.4 Augmented image results	page 78
3.5 Evaluation of the correction method.	page 81
3.5.1 Effect on the registration stability	page 81
3.5.2 Effect on the registration coherence	page 88
3.5.3 Augmented image results	page 89
3.6 Evaluation of the processing time.	page 92
3.7 Comparison with the ARToolKit	page 94
3.8 Effect of the correction factors	page 98
3.9 Conclusion	page 101
4. Conclusion	page 104
References	page 107
Appendix A : Finsterwalder's manipulations and equations	page 115
Appendix B : Roots of a third degree polynomial equation.	page 118
Appendix C : Stereoscopic equations.	page 122
Appendix D : Straight line fitting using least square minimization.	page 125

Appendix E : The HSV color system..... page 127

List of figures

Figure 1.1 Augmented reality vs. augmented virtuality.....	page 2
Figure 1.2 Illustration of the occlusion problem.	page 3
Figure 1.3 Illustration of an optical see-through HMD.....	page 4
Figure 1.4 Illustration of a video see-through HMD.....	page 5
Figure 1.5 Example of widely used feature points.....	page 6
Figure 1.6 Tracking of feature positions using previous frame information.	page 7
Figure 1.7 Presentation of the camera and world coordinate systems.....	page 8
Figure 1.8 Geometry of the three point space resection problem.	page 9
Figure 1.9 Geometry of the stereoscopic vision.	page 11
Figure 1.10 Illustration of the toed-in and the parallel axis HMD setup...	page 12
Figure 1.11 The three tasks of the system with their requirements.....	page 22
Figure 2.1 Global flow char of the proposed system.....	page 25
Figure 2.2 The main steps of the detection.	page 28
Figure 2.3 Illustration of the marker extraction strategy.....	page 29
Figure 2.4 Illustration of the square neighborhood division pattern.	page 29
Figure 2.5 Illustration of the hexagonal neighborhood division pattern. ..	page 30
Figure 2.6 Example of extraction using the hexagonal pattern.	page 30
Figure 2.7 Relation between the corner positions and the estimations. ...	page 32
Figure 2.8 Localization of the six edge pixels.....	page 33
Figure 2.9 Illustration of the wanted orientation of corners.	page 37
Figure 2.10 Algorithm for merging incomplete marker structure.....	page 40
Figure 2.11 Relations between the coordinate systems.....	page 41
Figure 2.12 Decomposition of the camera-world relation.....	page 42
Figure 2.13 Influence of the distance on the 2D positions.....	page 43
Figure 2.14 Illustration of the projection.....	page 46
Figure 2.15 Illustration of the camera coordinate system.	page 49
Figure 2.16 Simplified relations between the coordinate systems.....	page 52
Figure 2.17 Theoretical case: the 3D positions are errorless.....	page 54
Figure 2.18 General case: the 3D positions differ.	page 54
Figure 2.19 Illustration of the correction method.	page 56
Figure 2.20 Illustration of the correction by true paired progression.	page 57
Figure 3.1 Schema of the Canon's HMD.....	page 61
Figure 3.2 Pictures of the Canon's HMD.	page 62
Figure 3.3 Configuration of our augmented reality system.....	page 62
Figure 3.4 Illustration of the virtual point position.	page 64
Figure 3.5 Distance between two virtual points.....	page 65
Figure 3.6 Average processing time.	page 67
Figure 3.7 Percentage of successful marker extractions (1).	page 68
Figure 3.8 Block diagram of the registration methods.....	page 69

Figure 3.9 Fluctuation in position without correction (marker S).	page 70
Figure 3.10 Fluctuation in position without correction (marker M).	page 70
Figure 3.11 Fluctuation in position without correction (marker L).	page 71
Figure 3.12 Fluctuation in position for the stereoscopic method (Snoc). . .	page 72
Figure 3.13 Fluctuation in position for the proposed method (Xnoc).	page 72
Figure 3.14 Fluctuation in orientation without correction (marker S).	page 73
Figure 3.15 Fluctuation in orientation without correction (marker M).	page 73
Figure 3.16 Fluctuation in orientation without correction (marker L).	page 74
Figure 3.17 Fluctuation in orientation for the stereoscopic method (Snoc). .	page 74
Figure 3.18 Fluctuation in orientation for the proposed method (Xnoc). . .	page 75
Figure 3.19 Evaluation of the angle effect on the stability.	page 75
Figure 3.20 Angle effect on the fluctuations in position.	page 76
Figure 3.21 Angle effect on the fluctuations in orientation.	page 77
Figure 3.22 Coherence in position without correction.	page 77
Figure 3.23 Coherence in orientation without correction.	page 78
Figure 3.24 Block diagram of the registration methods with correction. . .	page 81
Figure 3.25 Fluctuation in position for the proposed method (marker S). .	page 81
Figure 3.26 Fluctuation in position for the proposed method (marker M). .	page 82
Figure 3.27 Fluctuation in position for the proposed method (marker L). .	page 82
Figure 3.28 Angular fluctuations for the proposed method (marker S). . . .	page 83
Figure 3.29 Angular fluctuations for the proposed method (marker M). . .	page 83
Figure 3.30 Angular fluctuations for the proposed method (marker L). . . .	page 84
Figure 3.31 Position fluctuations for the stereoscopic method (marker S). .	page 84
Figure 3.32 Position fluctuations for the stereoscopic method (marker M). .	page 85
Figure 3.33 Position fluctuations for the stereoscopic method (marker L). .	page 85
Figure 3.34 Angular fluctuations for the stereoscopic method (marker S). .	page 86
Figure 3.35 Angular fluctuations for the stereoscopic method (marker M). .	page 86
Figure 3.36 Angular fluctuation for the stereoscopic method (marker L). .	page 87
Figure 3.37 Fluctuations in position with correction.	page 87
Figure 3.38 Fluctuations in orientation with correction.	page 88
Figure 3.39 Coherence in position with correction.	page 89
Figure 3.40 Coherence in orientation with correction.	page 89
Figure 3.41 Percentage of successful marker extractions (2).	page 95
Figure 3.42 Fluctuation in position with the ARToolKit (marker S).	page 97
Figure 3.43 Fluctuation in orientation with the ARToolKit (marker S). . .	page 97
Figure 3.44 Fluctuation in position using variable factors (marker S). . . .	page 99
Figure 3.45 Fluctuation in orientation using variable factors (marker S). .	page 99
Figure 3.46 Fluctuation in position with random errors (marker S).	page 100
Figure 3.47 Fluctuation in orientation with random errors (marker S). . .	page 101
Figure A.1 Geometry of the three point space resection problem.	page 115
Figure C.1 Geometry of the stereoscopic vision.	page 122

List of tables

Table 1.1: Registration characteristics	page 17
Table 1.2: Required characteristics of the proposed method	page 18
Table 2.1: Evolution of the marker design	page 27
Table 2.2: Algorithm to find the three farthest pixels , and	page 31
Table 2.3: Hue range for the four identification colors.	page 36
Table 2.4: Codification of the colors	page 37
Table 2.5: Valid global code for	page 38
Table 2.6: Marker structure.	page 39
Table 2.7: Description flags	page 39
Table 3.1: Dimension of the small, middle and large size markers	page 66
Table 3.2: Average frame rates of the marker extraction for one frame. . .	page 68
Table 3.3: Examples of augmented images using the “Snoc” method	page 79
Table 3.4: Examples of augmented images using the “Xnoc” method. . . .	page 80
Table 3.5: Examples of augmented images using the “Scor” method	page 90
Table 3.6: Examples of augmented images using the “Xcor” method	page 91
Table 3.7: 3D objects registered by the system.	page 92
Table 3.8: Time for merging different 3D objects on stereo images.	page 93
Table 3.9: Time for merging multiple 3D objects on stereo images.	page 94
Table 3.10: Timing of the systems during the registration of the monster. .	page 96
Table 3.11: Augmented images of a teapot.	page 98
Table 3.12: Augmented images produced with the developed system. . .	page 102

List of symbols

- \mathbb{N} : set of the integer number.
- i : feature counter ($i \in \{1, 2, 3\}$).
- j : feature counter ($j \in \{1, 2, 3\}$)
- k : undetermined counter.
- u : undetermined counter.
- v : undetermined counter.
- t : iteration or time counter.
- $Y(\dots)$: function
- L : left camera image plane.
- R : right camera image plane.
- \vec{f}_i : vector giving the 2D position of the feature i .
- f_i : 2D position of the feature i .
- $f_{l_i}(x_l, y_l)$: 2D position f_i of components (x_l, y_l) of a feature i in the left image.
- $f_{r_i}(x_r, y_r)$: 2D position f_i of components (x_r, y_r) of a feature i in the right image.
- $\vec{f}'_a(a_x, a_y)$: 2D position with pixel precision of the first marker corner.
- $\vec{f}'_b(b_x, b_y)$: 2D position with pixel precision of the second marker corner.
- $\vec{f}'_c(c_x, c_y)$: 2D position with pixel precision of the third marker corner.
- $\vec{f}_a(a_x, a_y)$: 2D position with sub-pixel precision of the first marker corner.
- $\vec{f}_b(b_x, b_y)$: 2D position with sub-pixel precision of the second marker corner.

- $\vec{f}_c(c_x, c_y)$: 2D position with sub-pixel precision of the third marker corner.
- $\vec{f}_1 \vec{f}_2 \vec{f}_3$: the three corners \vec{f}_a , \vec{f}_b and \vec{f}_c listed in the defined order.
- $p_a(x_a, y_a)$: one of the three farthest pixels of a segmented region u .
- $p_b(x_b, y_b)$: one of the three farthest pixels of a segmented region u .
- $p_c(x_c, y_c)$: one of the three farthest pixels of a segmented region u .
- e_k : estimation of the 2D position of an edge pixel k .
- E_k : 2D position of an edge pixel k .
- $\vec{r}_u(r_{ux}, r_{uy})$: 2D position of a colored region inserted in the triangular marker.
- \vec{h}_p : middle of the hypotenuse edge of the triangle marker.
- \vec{q}_u : the two corners of the hypotenuse edge of the triangle marker.
- p_i : projection of F_i in an image.
- p_{li} : projection of F_i in the left image.
- p_{ri} : projection of F_i in the right image.
- p_{lli} : projection of F_i computed from the left image ($F_i \in R_l$) in the left image.
- p_{lri} : projection of F_i computed from the right image ($F_i \in R_r$) in the left image.
- p_{rli} : projection of F_i computed from the left image ($F_i \in R_l$) in the right image.
- p_{rri} : projection of F_i computed from the right image ($F_i \in R_r$) in the right image.
- $p(p_x, p_y)$: position in an image plane (L or R) given by the projection of a point c in the camera coordinate system.
- n_{li} : corrected position of f_{li} .

- n_{ri} : corrected position of f_{ri} .
- $C_g(c_x c_y)$: center of gravity of a segmented region u .
- $C_f(C_x C_y)$: center of the triangle marker computed with the \vec{F}_u .
- C_p : center of perspective of a camera.
- P_n : 3D position $(x_n y_n z_n)$ of a point in a HMD with a toed-in axes coordinate system.
- P_c : 3D position $(x_c y_c z_c)$ of a point in a HMD with a parallel axes coordinate system.
- \vec{F}_i : vector giving the 3D position of the feature i .
- F_i : 3D position of the feature i .
- $F(x, y, z)$: 3D position P of components (x, y, z) of a feature where the components (x, y, z) also defined the vectors \vec{F} .
- \vec{F}_{li} : 3D feature positions computed from a left camera image.
- \vec{F}_{ri} : 3D feature positions computed from a right camera image.
- \vec{F}_S : true solution of the 3D feature positions computed with the best solution approach.
- \vec{F}_{li} : 3D position \vec{F}_i computed from the left image.
- \vec{F}_{ri} : 3D position \vec{F}_i computed from the right image.
- \vec{F}_L : left components \vec{F}_{L1} , \vec{F}_{L2} and \vec{F}_{L3} of the true paired solution computed with the paired solution approach.
- \vec{F}_R : right components \vec{F}_{R1} , \vec{F}_{R2} and \vec{F}_{R3} of the true paired solution computed with the paired solution approach.
- \vec{F}_{Ci} : the 3D position of a feature i in the global camera coordinate system C .
- \vec{F}_{Cli} : if \vec{F}_{Ci} has been computed with the left image.

- $\vec{F}_{C_{ri}}$: if \vec{F}_{C_i} has been computed with the right image.
- $\vec{F}'_{C_{li}}$: corrected position of $\vec{F}_{C_{li}}$.
- $\vec{F}'_{C_{ri}}$: corrected position of $\vec{F}_{C_{ri}}$.
- s_t : position of a virtual point in a coordinate system created from the elements of the orientation matrix T_{Rt} at the current frame t .
- s_l : position of a virtual point in a coordinate system created from the elements of the orientation matrix T_{Rl} .
- s_r : position of a virtual point in a coordinate system created from the elements of the orientation matrix T_{Rr} .
- M : marker coordinate system.
- M_x : \vec{X} axis of the marker coordinate system.
- M_y : \vec{Y} axis of the marker coordinate system.
- M_z : \vec{Z} axis of the marker coordinate system.
- m : 3D position of a point in the marker coordinate system.
- O : virtual object coordinate system.
- o : 3D position of a point in the virtual object coordinate system.
- C : global camera coordinate system.
- \vec{C}_x : \vec{X} axis of the global camera coordinate system C .
- \vec{C}_y : \vec{Y} axis of the global camera coordinate system C .
- \vec{C}_z : \vec{Z} axis of the global camera coordinate system C .
- C_l : left camera coordinate system.
- C_r : right camera coordinate system.
- c : 3D position of a point in the camera coordinate system.

- \mathcal{W} : world coordinate system.
- T_{CO} : the model-view matrix or the transformation matrix from an object coordinate system \mathcal{O} to a camera coordinate system \mathcal{C} .
- T_{RCO} : rotation matrix of the matrix T_{CO} .
- \vec{T}_{TCO} : translation vector of the matrix T_{CO} .
- T_{WO} : transformation matrix between a virtual object coordinate system \mathcal{O} and the world coordinate system \mathcal{W} .
- T_{RWO} : rotation matrix of the matrix T_{WO} .
- \vec{T}_{TWO} : translation vector of the matrix T_{WO} .
- T_{CW} : transformation matrix between the camera coordinate system \mathcal{C} and the world coordinate system \mathcal{W} .
- T_{wm} : transformation matrix between a marker coordinate system \mathcal{M} and the world coordinate system \mathcal{W} .
- T_{cm} : transformation matrix between a marker coordinate system \mathcal{M} and the camera coordinate system \mathcal{C} .
- T_{Rt} : orientation matrix of the matrix T_{cm} at the current frame t .
- T_{Rl} : orientation matrix of the model view matrix associated with the left image.
- T_{Rr} : orientation matrix of the model view matrix associated with the right image.
- \vec{T}_{Tt} : translation vector of the matrix T_{cm} at the current frame t .
- \vec{T}_{Tl} : translation vector of the model view matrix associated with the left image.
- \vec{T}_{Tr} : translation vector of the model view matrix associated with the right image.
- S_u : set including all the n_u processed pixels of the segmented region u .

- S : set of the l pixels used to identify the color of a region.
- S_l : set of solutions computed from the left image.
- S_r : set of solutions computed from the right image.
- R_l : set of remaining solutions of the left camera.
- R_r : set of remaining solutions of the right camera.
- P : set of possible paired solution.
- n_u : number of pixels included in the segmented region u .
- n : number of pixels used to compute sub-pixel precision needed for the edge equation.
- l : number of pixels used to identify the color of a region.
- n_p : number of possible paired solutions.
- l_{ab} : length of the edge running from $p_a(x_a, y_a)$ and $p_b(x_b, y_b)$.
- l_{al} : length of the edge running from $p_a(x_a, y_a)$ and a pixel $p_l(x_l, y_l)$.
- l_{bl} : length of the edge running from $p_b(x_b, y_b)$ and a pixel $p_l(x_l, y_l)$.
- L_N : length of the normal vector \vec{N}_{uv} .
- d_{ij} : length of the side linking the feature i and the feature j of a marker.
- s_i : distance from the center of perspective C_p to the 3D feature position of the feature i or the length of \vec{F}_i .
- h_u : distance between two adjacent neighborhood pixel of the hexagonal pattern.
- d : distance between two points.
- D : sum of euclidean distances between the left 3D feature positions \vec{F}_{li} and the right 3D feature positions \vec{F}_{ri} .

- T_D : threshold value applied on the sum of euclidean distances D .
- \vec{N}_{uv} : normal vector of the edge that runs from a pixel $p_u(x_u, y_u)$ and a pixel $p_v(x_v, y_v)$ where $u, v \in \{a, b, c\}$ and $u \neq v$.
- \vec{N}_m : normal vector of the plane create with the axis \vec{X} and the axis \vec{Y} of the marker coordinate system.
- L_{uv} : length of the line segment joining two corners $p_u(x_u, y_u)$ and $p_v(x_v, y_v)$.
- S_o : fluctuation in orientation.
- S_p : fluctuation in position.
- K_o : coherence in orientation.
- K_p : coherence in position.
- E_p : projection error for the 3D positions F_i .
- E_l : projection error in the left image.
- E_r : projection error in the right image.
- E_{pl} : projection error for the 3D positions \vec{F}_{li} .
- E_{pr} : projection error for the 3D positions \vec{F}_{ri} .
- E_{ll} : error of the projection of \vec{F}_{li} in the left image.
- E_{lr} : error of the projection of \vec{F}_{li} in the right image.
- E_{rl} : error of the projection of \vec{F}_{ri} in the left image.
- E_{rr} : error of the projection of \vec{F}_{ri} in the right image.
- μ : correction factor.
- τ : correction factor.

- j_i : vectors pointing from the center of perspective C_ρ to the observed point p_j .
- α : angles at the center of perspective C_ρ between j_2 and j_3 .
- β : angles at the center of perspective C_ρ between j_1 and j_3 .
- γ : angles at the center of perspective C_ρ between j_1 and j_2 .
- b : baseline between the two stereoscopic cameras.
- f : focal length of the cameras.
- θ_l : toed-in angle of the left camera.
- θ_r : toed-in angle of the right camera.
- θ_c : theoretical sum of θ_l and θ_r .
- θ : evaluation of θ_c .
- θ_j : evaluation of θ_c from the feature j .
- H : density parameter of a hexagonal pattern.
- A : area of the surface created from the pixels $p_a(x_a, y_a)$, $p_b(x_b, y_b)$ and $p_c(x_c, y_c)$.
- $D(u)$: one dimension edge detector that returns true only if the pixel u is white and the pixel $u-1$ is not.
- Δ_{ux} : variation in x between two pixels used to identify the color of a region.
- Δ_{uy} : variation in y between two pixels used to identify the color of a region.
- h : hue value.
- v_u : color code value for a region u .
- G : global color code value given by the sum of v_a , v_b and v_c .

- s : value of the polynomial used to identify \vec{f}_2 and \vec{f}_3 .
- \vec{N}_p : plane normal.
- φ : angle between two plane normals \vec{N}_{p_l} and \vec{N}_{p_r} .
- T_φ : threshold value applied on the angle φ .
- m_e : measure used to evaluate which feature needs the stronger correction.
- m_{ej} : measure m_e of the feature j .
- m_i : measure used to evaluate if a correction should be stronger in the left or the right image.
- m_{ij} : measure m_i of the feature j .
- Δ_j : measure of variation in intensity for a feature j in an image.
- Δ_l : measure of variation in intensity in the left image.
- Δ_r : measure of variation in intensity in the right image.

1. Introduction

This chapter introduces the field of augmented reality, surveys the augmented reality system developed so far, presents the objective of this research and suggests some improvements to increase the stability of vision-based registrations.

The first section of this chapter presents the augmented reality. The section 1.2 defines some important terms and techniques associated to the field of augmented reality. Then, the section 1.3 surveys augmented reality systems. Next, section 1.4 details the objectives which motivate the present research. Finally, the section 1.5 quickly presents the methods proposed to fulfil the objectives.

The second chapter of this thesis details the proposed methods. In the third chapter, the system developed in order to test the suggested methods is presented. Then, the experimental results are listed and commented.

The last chapter resumes the important aspects of this research. In addition, five appendices are attached at the end of the document as a reference to the solutions of important geometric and mathematical problems.

1.1 Augmented Reality

The two poles of the reality-virtuality continuum are the purely real and the purely virtual environments. In other words, virtual reality environments stand at the opposite of the real environments. Real environment data encompass any kind of unmodelled data sampled from the real environment such as videos or photographic images. Virtual environment data define an imaginary world where every object is completely modelled by human. With a virtual reality system, users enter and interact with a virtual world. A mixed reality environment lies between these two poles. Mixed reality environments are created using both real and virtual components. In mixed reality, we eventually begin to encounter the problem of deciding whether in fact what we are doing is augmenting a real world with virtual graphic objects, or whether we are modifying a virtual environment by augmenting it with real data[MC99]. Therefore, two types of mixed reality environments have been defined: augmented virtuality and augmented reality. The augmented virtuality is an environment where few real components are added to a virtual world and the augmented reality is an environment where few virtual components are added to a real environment. Figure 1.1 illustrates these concepts.

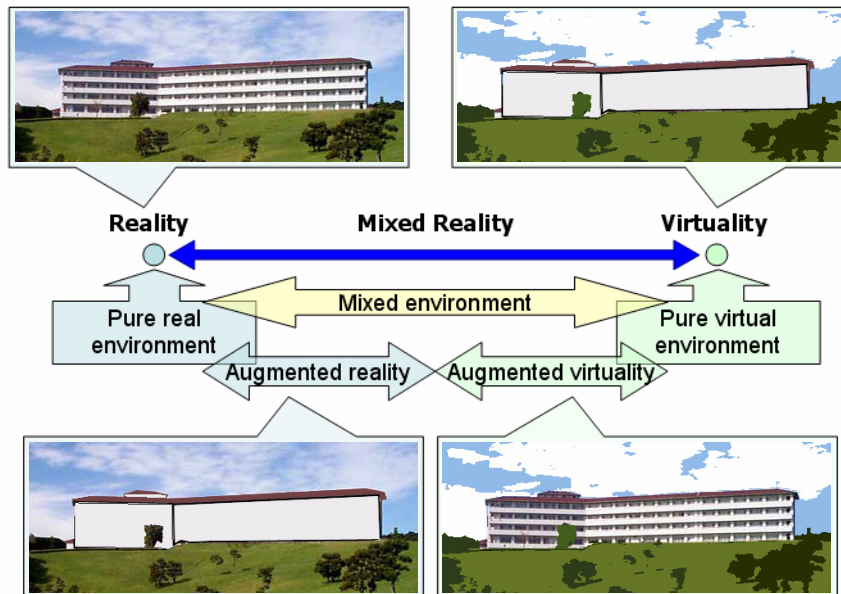


Figure 1.1 Augmented reality vs. augmented virtuality.

An augmented reality system enhances a user's perception and interaction with the real world since the virtual objects added display information that the user cannot directly detect with their own senses. This information conveyed by the virtual objects helps a user to perform real-world tasks. Augmented reality applications have been explored in many potential fields, such as medical visualization, maintenance and repair, annotation, robot path planning, entertainment, and military aircraft navigation and targeting[Azu97].

1.1.1 Augmented reality problems

This subsection lists three of the biggest problems associated to augmented reality systems. The problems are the geometric registration, the photometric registration and the occlusion.

Geometric registration

The registration problem is currently the problem limiting the most augmented reality applications. The objects in the real and virtual worlds must be properly aligned with respect to each other, or the illusion that the two worlds coexist will be compromised. Two types of geometric registration error exist: static and dynamic errors. Static registration errors are the registration errors that appear even when the user's viewpoint and the objects in the environment remain completely still. Examples of static registration error sources are optical distortion, errors in tracking system, mechanical misalignments or

incorrect estimation of the system parameters. Dynamic registration errors are the registration errors that have no effect until either the viewpoint or the objects start moving. Dynamic errors are caused by the system delays, or lag, between the moment when the tracking system measures the position and orientation of the viewpoint and the moment when the generated images associated to the viewpoint position and orientation appear on the displays[Azu97].

Photometric registration

To improve the perception quality of virtual objects in augmented reality applications, the ability to automatically capture the environmental illumination and reflectance information is required[ABB+01]. Illumination and reflectance information allow the computation of the shading and the shadows of the virtual objects. Another photometric aspect is to match the resolution of the virtual objects to the resolution of the captured images of the real scene.

Occlusion

To produce realistic augmented images, the augmented objects must be correctly occluded by foreground real objects. The Figure 1.2 shows an example of an augmented image without and with occlusion when a user's hand is located in front of a virtual object.

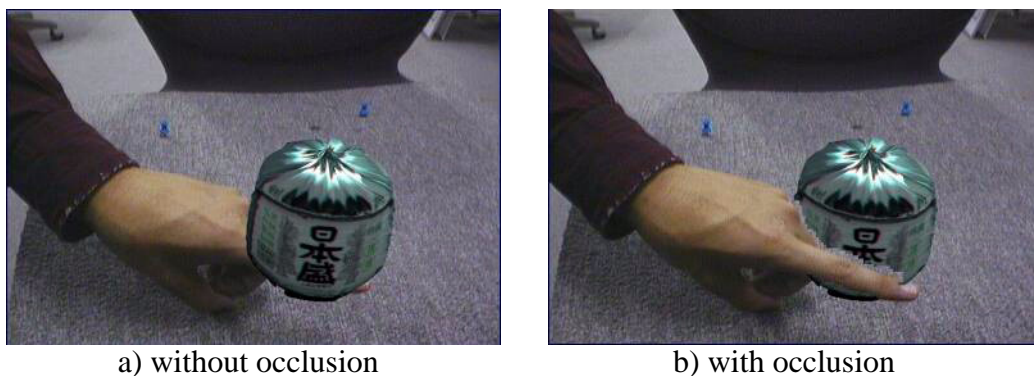


Figure 1.2 Illustration of the occlusion problem.

Theoretically, the occlusion problem can only be solved by comparing the depth of the virtual objects to the depth of the real scene objects[LB00]. Unfortunately, the depth of the real scene objects is rarely known and is hardly measurable by a system.

1.1.2 Overlay methods

The augmented world is generally communicated to the user using a head mounted device(HMD). Two basic technologies exist to combine real and virtual worlds: optical see-through and video see-through technologies. Each technology employs a specific head mounted device. Optical see-through head mounted devices use an optical combiner placed in front of the user's eyes. These combiners are partially transmissive. Therefore, the user can look directly through them to see the real world. Since these combiners are also partially reflective, the user also sees the virtual objects reflected from the head mounted monitors by the combiners[Azu97]. In contrast to optical see-through head mounted devices, video see-through head mounted devices work by replacing the user's eyes by one or two head mounted video cameras. The video cameras provide the user's view of the real world. Video images from these cameras are combined with generated computer graphics of the virtual objects. The augmented images are sent to the head mounted device monitors located in front of the user's eyes. The rest of this subsection notes particular advantages and disadvantages of each technology. Figure 1.3 and Figure 1.4 illustrate each head mounted device technology. In the figures, a virtual fish generated by a PC is placed inside a real fishbowl. The augmented images are respectively shown to the user using an optical see-through head mounted device or a video see-through head mounted device. The head mounted devices are highlighted by dotted lines in the figures.

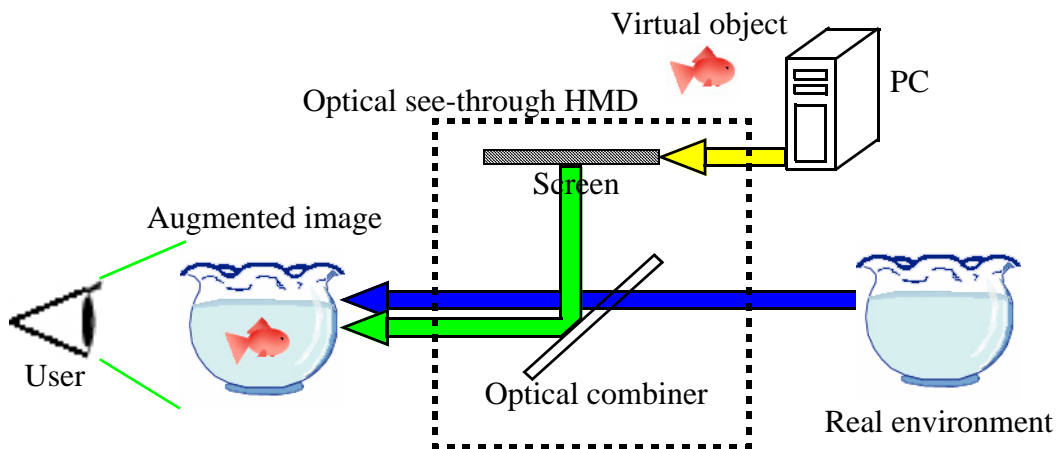


Figure 1.3 Illustration of an optical see-through HMD.

Optical see-through

The simplicity is the main advantage of the optical see-through technology. That is, since the real world is seen directly through the combiners, optical see-through systems have only the graphic images to worry about. Also, views of the real world are perceived without major distortion when using optical see-through head mounted devices. But, the

amount of light that the user receives from the world is usually reduced by the optical combiners. Another advantage of the optical see-through technology is that the real world is seen from the user's eye positions with the user's eye resolution. Only the virtual world view is limited by the resolution of the display device. On the other hand, optical see-through head mounted devices offer an instantaneous view of the real world but a delayed view of the virtual world. This temporal mismatch causes dynamic registration errors. Another problem with optical see-through head mounted devices is that the virtual objects do not completely obscure the real world objects. Furthermore, coherent occlusion between virtual and real objects is very difficult to perform using optical see-through head mounted devices[Azu97][KKO+00].

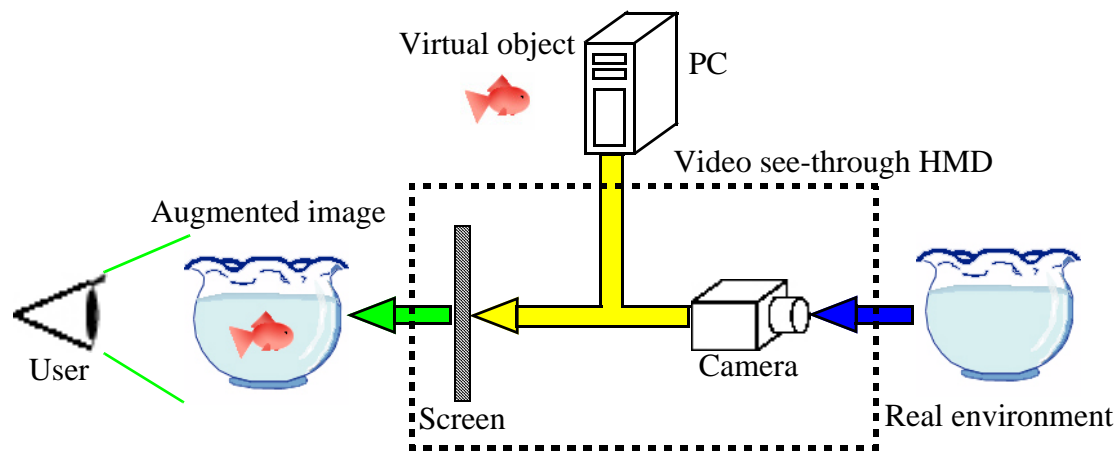


Figure 1.4 Illustration of a video see-through HMD.

Video see-through

Video see-through technology is more complicated since video see-through systems must merge the video streams with the virtual object graphics. Even with real-time video digitization, digitizing the video images delays the user's view of the real world scene. Consequently, a delay is created between a user's motion and the visual response. But, the delays of the real world view and the virtual world view are matched to eliminate dynamic registration errors caused by temporal mismatch between the views of both worlds. Also, the camera locations and the user's eye locations generally differ in video see-through head mounted devices. Therefore, a small contradiction between what the user sees and what the user expects to see is introduced since the camera locations and the eye locations differ. Furthermore, video see-through technologies limit the resolution of both real and virtual objects. The resolution is limited by the resolution of the cameras and the resolution of the head mounted device monitors. With the current technologies, this resolution is far less than resolution of the human eyes. Besides, video see-through technology has several advantages. In the first place, video see-through systems have the

possibility to use computer vision technologies. Therefore, static registration errors, such as the distortions caused by the optical devices, can be corrected by applying image processing techniques. Also, image processing can be used to match the brightness of real and virtual objects. Another advantage of video see-through systems is the possibility to use video-based registration techniques to perform the registration. In addition, video see-through systems are also able to perform correct occlusion of virtual and real objects[Azu97][KKO+00].

1.2 Definitions

The following sections present important terms and methods used in the augmented reality field. The first section differentiates between the terms “point”, “feature” and “marker”. The second section presents a tracking strategy widely used in augmented reality systems. The section 1.2.3 presents the registration approach used with monocular vision-based systems. Then, the section 1.2.4 explains the registration approach used in stereoscopic vision-based systems. Finally, the section 1.2.5 talks about the toed-in setup of some head mounted devices.

1.2.1 Point, feature and marker

The difference between a point, a feature and a marker must be understood to avoid any confusion. A point is the smallest indivisible geometric element. A feature is a simple regrouping of points that makes a special point of interest stand out. This point of interest is called a feature point. A marker is a geometrical figure created using shapes and colors. A marker may contain one feature or more according to the needs of a system. Also, a position may be associated to a marker. The marker position may be given by the center of the marker shape or by a specific feature of the marker.

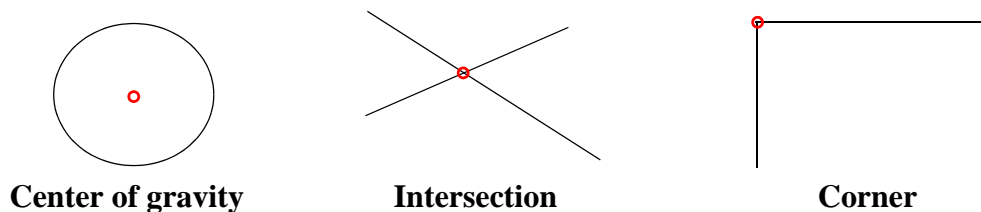


Figure 1.5 Example of widely used feature points.

To perform the registration, vision-based augmented reality systems need to track the positions of some points in the scene. Consequently, those systems must extract some feature points in the input images. The Figure 1.5 shows three types of feature points widely used in vision-based augmented reality systems. The feature points are marked by

a circle in the figure. When a system uses predefined markers, easily detectable feature points associated to the predefined markers simplify the registration task.

1.2.2 Tracking strategy using searching windows

Some augmented reality systems manage to process entirely every frame in real-time [KBP+00][PYN98]. Unfortunately, augmented reality systems are sometimes unable to process entirely every frame in real-time either because the system is slowed down by further process or because the system doesn't possess the hardware to process entirely each frame. Consequently, strategies to decrease the processing time have been developed. One popular approach in monocular and binocular systems is the extraction of the marker positions based on the positions of the markers in the previous frame. This approach is usually divided in two steps: the detection step and the tracking step.

When the system starts, the first frame is entirely processed and the positions of every marker are detected. Then, in subsequent frames, the system keeps tracking the marker positions based on the positions of the markers in the previous frame [KIT+00][OKT+98].

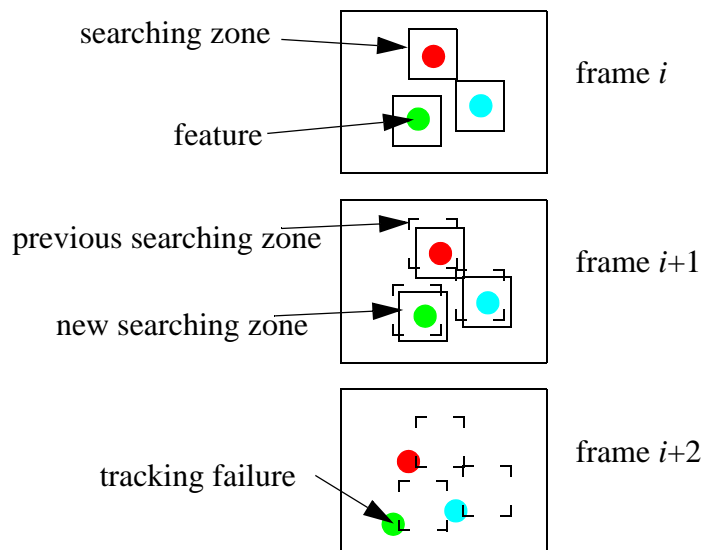


Figure 1.6 Tracking of feature positions using previous frame information.

During the tracking step, a searching square zone is centered at the previous position of a marker. All the pixels inside the searching zone are processed to extract the marker position. The size of the region is given either by the marker size in the previous frame or by a constant defined in the system. When the system fails to retrieve the marker positions, a tracking failure occurs and the system must restart with the detection step. A tracking failure occurs generally when the markers exit the searching zone after a user's

motion. The Figure 1.6 illustrates the tracking of the marker positions using information from previous frame.

1.2.3 Monocular vision-based registration

As mentioned before, the goal of the geometric registration is to align the real and virtual worlds properly with respect to each other. In order to perform the geometric registration, the camera pose is needed. In other words, the position and the orientation of the camera in the world coordinate system, which defines the coordinate reference for the entire scene, must be computed. If the pose of the camera and the poses of the virtual objects are known, the pose of the virtual objects in the field of view of the camera can be deduced. Therefore, the poses of the virtual objects in the world coordinate system are transformed into the correspondent poses in the camera coordinate system which is a coordinate system centered on the camera. Figure 1.7 illustrates the camera coordinate system C and the world coordinate system W .

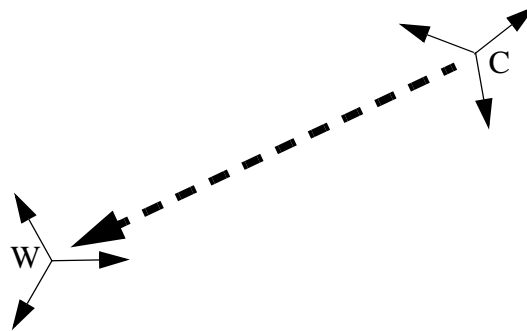


Figure 1.7 Presentation of the camera and world coordinate systems.

If six points with known positions in the world coordinate system are observed by a camera, the 3D position in the camera coordinate system of these points can be determined by solving the inverse perspective projection. In the special case that four points with known positions in the world coordinate system are on the same plane, the inverse perspective projection is considerably simplified. Therefore, most of the monocular vision-based augmented reality systems use four-sided polygon markers, usually square-shaped markers, to perform the registration[KBP+00][BKP01][ARTK].

The registration can also be performed with three points which define the vertices of a triangle in the world coordinate system. If the three points have known positions in the world coordinate system, the position of the vertices in the camera coordinate system can be determined from the perspective projection of those points in an image. Determining the vertex 3D positions from three points is often called the three point space resection problem. This problem is important in photogrammetry as well as in computer vision.

Many direct solutions to the three point resection problem have been proposed. Haralick et al., reviewed most of those direct solutions in their paper[HLO+91]. Once the 3D positions of the points in the camera coordinate system have been computed, the camera pose in the world coordinate system can be determined.

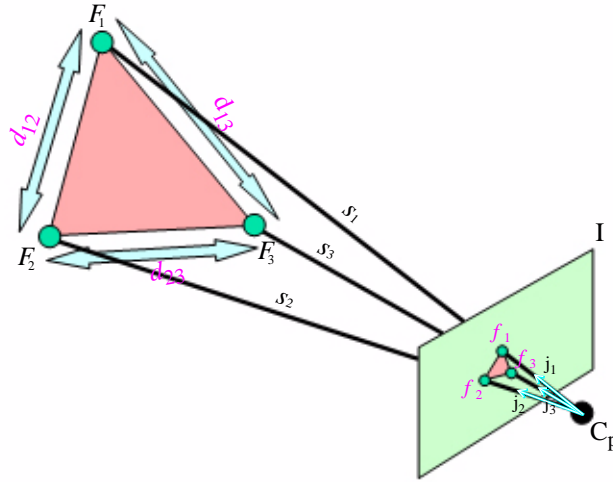


Figure 1.8 Geometry of the three point space resection problem.

The Figure 1.8 illustrates the geometry of the three point space resection problem. The three vertices of a known triangle F_1 , F_2 and F_3 define a triangle of side lengths d_{12} , d_{13} and d_{23} in the world coordinate system. The origin of the camera coordinate is defined as the camera center of perspective C_p . The perspective projection in the image of F_1 , F_2 and F_3 is respectively denoted as f_1 , f_2 and f_3 . The vectors j_1 , j_2 and j_3 are the unit vectors pointing from the center of perspective C_p to the observed point f_1 , f_2 and f_3 . Consequently, the angles at the center of perspective are given by $\alpha = \text{acos}(j_2 \cdot j_3)$, $\beta = \text{acos}(j_1 \cdot j_3)$ and $\gamma = \text{acos}(j_1 \cdot j_2)$. Finally, the distance from the center of perspective C_p to each of the three vertices are defined as s_1 , s_2 and s_3 . By the law of cosines, the three point space resection problem is written as follows:

$$s_2^2 + s_3^2 - 2s_2s_3 \cos \alpha = d_{23}^2 \quad (1.1)$$

$$s_1^2 + s_3^2 - 2s_1s_3 \cos \beta = d_{13}^2 \quad (1.2)$$

$$s_1^2 + s_2^2 - 2s_1s_2 \cos \gamma = d_{12}^2 \quad (1.3)$$

This system of equations can be expressed as a fourth order polynomial equation. Since the three point space resection problem requires solving a fourth order polynomial equation, up to four real roots can be computed. To solve the system of equations, Finsterwalder has proceeded in a way that requires only finding the roots of a cubic polynomial and the roots of two quadratic polynomials rather than finding all the roots of

a fourth order polynomial[HLO+91]. Finsterwalder's manipulations and equations are given in the Appendix A. Also, the Appendix B explains how to find all the roots of a third order polynomial equation.

Using Finsterwalder's method, up to 12 real solutions are generated. Therefore, an augmented reality system which wants to solve the three point space resection problem with Finsterwalder's method needs to select the correct solution among the multiple solutions computed. As an example, Finsterwalder's method has been used by Okuma et al. to perform the registration in their system when one of the four feature points normally tracked is missing[OKT+98]. The last unique solution computed with the four points is used by the system to select the best solution among the multiple solutions given by Finsterwalder's method.

The Finsterwalder's method gives the 3D position of the three points in the camera coordinate system. From the 3D position of the points, the geometric registration can be performed. First, a local coordinate system is defined from the 3D positions of the points in the camera coordinate system. The three axes of the local coordinate system are computed using the set of equations 1.4. Then, the position of the camera in the local coordinate system is easily retrieved by transformation of coordinate spaces. Finally, if the relation between the local and the world coordinate systems can be determined beforehand, the virtual object can correctly be registered in the camera coordinate system. In other words, the virtual object can be drawn with the correct position and the correct orientation in the output images.

$$\begin{aligned}
\vec{M}_x &= \frac{\vec{F}_2 - \vec{F}_1}{|\vec{F}_2 - \vec{F}_1|} \\
\vec{M}_y &= \left(\frac{\left((\vec{F}_3 - \vec{F}_1) - \frac{(\vec{M}_x \cdot (\vec{F}_3 - \vec{F}_1))}{|\vec{M}_x|} \vec{M}_x \right)}{\left| \left((\vec{F}_3 - \vec{F}_1) - \frac{(\vec{M}_x \cdot (\vec{F}_3 - \vec{F}_1))}{|\vec{M}_x|} \vec{M}_x \right) \right|} \right) \\
\vec{M}_z &= \vec{M}_x \times \vec{M}_y
\end{aligned} \tag{1.4}$$

1.2.4 Stereoscopic vision-based registration

In binocular systems, the stereoscopic vision can be used to compute the position of one point which appears in both camera views. Given that the position of the point in left camera image is $f_l(x_l, y_l)$ and the position of the same point in the right camera image is $f_r(x_r, y_r)$, the 3D position $F(x, y, z)$ of the point in the camera coordinate system is

given by the set of equations 1.5 where b is the baseline between the cameras and f is the focus of the cameras.

$$\begin{aligned}
 x &= \frac{b(x_l + x_r)}{2(x_l - x_r)} \\
 y &= \frac{b}{\frac{x_l}{y_l} - \frac{x_r}{y_r}} \\
 z &= \frac{fb}{x_l - x_r}
 \end{aligned}
 \tag{1.5}$$

The epipolar constraint which stipulates that theoretically $y_l = y_r$ is used to simplify the equation of the component y of P . The simplified equation is given by equation 1.6. The Figure 1.9 illustrates the stereoscopic vision geometry. More details about the stereoscopic vision equations are given in Appendix C. Then, like in the monocular registration, the pose of the camera in the world coordinate system can be retrieved from the 3D positions of three points.

$$y = \frac{b(y_l + y_r)}{2(x_l - x_r)}
 \tag{1.6}$$

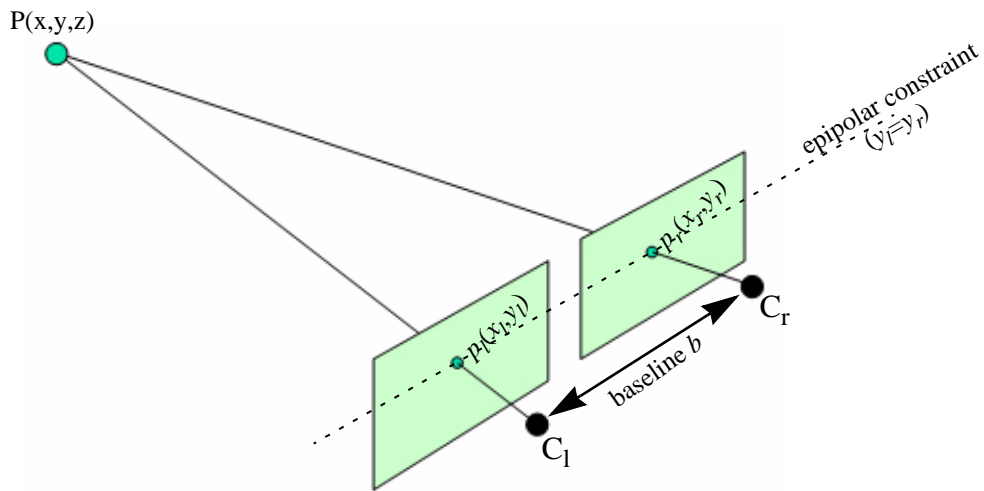


Figure 1.9 Geometry of the stereoscopic vision.

1.2.5 Toed-in compensation

The optical axes of the head mounted device cameras are often toed-in[TYS+00]. The toed-in setup of the head mounted device is illustrated in Figure 1.10a. The toed-in angles are noted θ_l and θ_r . In this configuration, stereoscopic estimations of the 3D point positions are mismatched. Since the axes of the stereoscopic cameras must be parallel to each other to perform standard stereoscopic algorithms, the toed-in effect must be compensated. Figure 1.10b illustrates the parallel axis head mounted device setup.

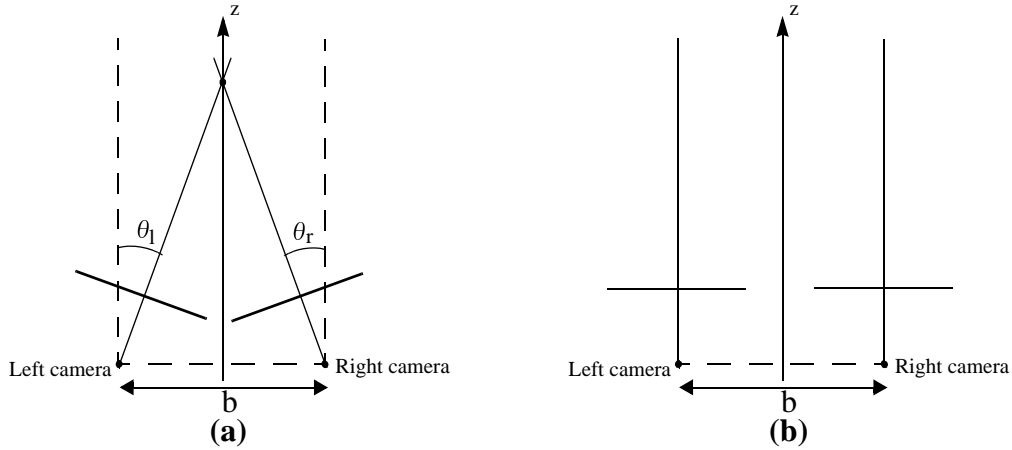


Figure 1.10 Illustration of the toed-in and the parallel axis HMD setup.

The compensation of the toed-in axes is performed with equation 1.7. The angle θ takes the value θ_l in the case of the left camera and the value θ_r in the case of the right camera. The equation 1.7 transforms the position of a pixel $P_n(x_n, y_n, z_n)$ from a HMD with a toed-in axes setup to the equivalent position of the pixel $P_c(x_c, y_c, z_c)$ in a HMD with the parallel axes setup. Each feature point position extracted by the system is corrected before any subsequent process starts. Consequently, each position referred in this chapter has been transformed beforehand to the position associated to the parallel axes setup.

$$P_c = \begin{bmatrix} \cos\theta & 0 & \cos(\theta + 90^\circ) \\ 0 & 1 & 0 \\ \cos(\theta - 90^\circ) & 0 & \cos\theta \end{bmatrix} P_n \quad (1.7)$$

1.3 Related Work

The registration of the real and the virtual worlds is one of the most important technical aspects of augmented reality systems. To align a virtual space with a physical space, the

position and orientation of the user's viewpoint must be known[SAY+01][Azu97][ABB+01]. The literature proposes various techniques to measure the position and orientation of a user's viewpoint. Those techniques are divided into three groups: 3D sensors based, vision-based and hybrid techniques.

When the user's position is limited to a certain area, fixed 3D sensors covering this limited area can be used. Sawada et al., propose a system which uses three fiber optic gyroscopes and three accelerometers[SON01]. In their system, Retmayr et al., also use a gyroscope sensor to measure the position and the orientation of the user's view[RS01]. Newman et al., perform the registration using an inertial sensor and wireless devices emitting ultrasonic pulses[NIH01]. A system using magnetic tracker has also been proposed by Bajura et al[BFO92].

Many augmented reality systems achieve the required registration using computer vision technologies. The approach in most cases is to detect visual features which have a known position. Then, a matching or pose recovery process computes the camera orientation and position[Beh99]. Two types of vision-based techniques exist: monocular and binocular techniques. Monocular techniques employ only one camera to perform the registration. Basically, monocular systems perform the registration by solving the perspective pose estimation problem from the position of three or more feature points in the camera images. Appendix A gives more detail about perspective pose estimation. Neumann and Park's system is a good example of simple monocular system[NP98]. Many improvements have been proposed to increase the registration accuracy of monocular vision-based systems. Kobayashi et al., have developed a system similar to Neumann and Park's system, but they added the use of two reference images to help solving the perspective pose estimation[KIO00]. On their side, Uenohara and Kanade's system uses template matching to track feature points of the object[UK95]. Retro-reflective features have also been used by Sauer et al., in their system[SKB+01]. Another system created by Sauer et al., uses a camera with a fisheye lens to increase the field of view of the tracking process[SWV+00]. Resolving the perspective pose estimation problem is generally arduous. Therefore, techniques that avoid solving the perspective pose estimation problem have been studied. Instead of solving the perspective pose estimation problem, Seo and Hong use a projective reconstruction based on two sequence views to perform the registration without metric calibration of the camera[SH00]. Binocular techniques have also been proposed to perform the registration without solving the perspective pose estimation problem. In binocular system, the registration is generally computed using stereoscopic equations which give the 3D positions of tracked features according to their positions in the two camera images. In their system, Kanbara et al., perform the registration from the stereo-paired images of the real scene captured by two cameras set on a head mounted device[KIT+00].

Recent registration systems employ hybrid-tracking techniques to exploit strengths and compensate weaknesses of individual 3D sensors based and vision-based techniques. So far, hybrid systems have produced a better registration performance[ABB+01][Azu97][TYK01]. Ohshima et al., have developed a hybrid augmented reality air hockey system which performs the registration using vision-based tracking of features and a magnetic sensor[OSY+98]. A system which performs the registration based on horizon silhouettes and the user's location given by a GPS has been proposed by Behringer[Beh99]. The system proposed by You and Neumann combines a mechanical gyroscope and vision-based tracking[YN01]. An inertial sensor which contains gyroscopes and accelerometers have been employed by You et al., in their hybrid system[YNA99]. The system developed by Yokokohji et al., is only based on accelerometers and vision processing[YSY00]. State et al., have developed a hybrid augmented reality system based on a mechanical tracker, a magnetic tracker and vision which simulates ultrasound-guided needle biopsies[SLG+96]. Hybrid binocular systems have also been proposed. State et al., have developed a hybrid system using magnetic trackers and two camera vision-based tracking[SHC+96]. The stereoscopic vision-based system proposed by Kanbara et al.[KIT+00], has also been extended to a hybrid system with the addition of an inertial sensor to the initial system[KFT+00a].

Researchers are now interested in developing mobile augmented reality systems. Mobile augmented reality system must be able to work in unprepared environments. After all, covering the environment with features or with tracker devices are not practical tasks. Global range tracking devices are needed for mobile augmented reality systems. GPS (Global Positioning System) can be used for outdoor applications, but GPS have significant limitations. Therefore, tracking user's position and orientation may rely heavily on tracking visible natural features using vision-based techniques[ABB+01]. Some systems have been developed to track natural features in real scenes. As already mentioned, a system using the horizon silhouettes and a GPS has been proposed by Behringer[Beh99]. Satoh et al., have created a hybrid system based on a gyroscope, but their system uses a template matching algorithm instead of standard fiducial features[SAY+01]. Based on the tracking of planar local image patches, Ferrari et al., have developed a featureless augmented reality system[FTV01]. A concept for featureless tracking with reference images has been introduced by Stricker and Kettenbach[SK01]. The system developed by Simon et al., operates in featureless environments which contain one or more planes[SFZ00]. Kanbara et al., also use natural features detected by Moravec's interest operator to increase the registration range in one version of their binocular system[Moh79][KFT+00b].

Another important aspect of augmented reality systems is to perform the occlusion between virtual and real objects in the scene. Only few systems perform the occlusion because actual depth sensors are not small and light enough to be attached to a HMD. A system which performs the fusion between stereo images and volumetric medical images

has been developed by Betting et al[BFA+95]. The system deals with occlusion, but the processing is too heavy to allow the system to run in real-time. Lepetit and Berger have developed a semi-automatic system which asks the user to point out two key-frames corresponding to views where an aspect of the occluding object changes[LB00]. The user is also asked to outline the occluding object in the key-frames. The 3D boundary of the occluding object is estimated using stereo-triangulation from the two silhouettes outlined by the user in the key-frames. If the translation of the camera between the two key-frames is not null, the system is able to perform occlusion in all the frames between the two key-frames using the 3D boundary of the occluding object. Some systems succeed to automatically perform the occlusion in real-time. Ohta et al., estimate the depth map of the user's view by converting 3D information supplied from a fixed position sensor and the user's pose[OSI+01]. The estimated depth map is then used to perform occlusion between objects. To perform occlusion in their system, State et al., acquire beforehand the patient's skin shape and location[SLG+96]. The system assumes that the patient will not move during and after the procedure. Conventional optical see-through systems cannot perform mutual occlusion since the synthetic objects always appear as semitransparent objects floating in front of the real scene, but Kiyokawa et al, have proposed an optical see-through system which performs the occlusion using a liquid crystal display (LCD) panel positioned in front of a modified optical see-through head mounted device in order to block any ray coming from outside[KKO+00]. A version of the stereoscopic vision-based system proposed by Kanbara et al.[KIT+00], has been developed in order to perform the mutual occlusion of real and virtual objects based on the depth information of the real scene acquired in real-time by stereo-paired cameras[YTO+99][KOT+00].

Some systems have the problem to select between more than one pose solutions. For instance, in vision-based or hybrid systems, solving the perspective pose estimation problem leads to more than one pose solutions when the detected number of feature points is not sufficient. A particular case is the three point perspective pose estimation[HLO+91]. Okuma et al., have developed a monocular system which uses four feature points to avoid multiple pose solutions[OKT+98]. When their system only detects three feature points, the system identifies, using previous temporal information, the best solution among the multiple pose solutions obtained from solving the three point perspective pose estimation problem.

The hybrid system proposed by State et al., also deals with the problem to select the best solution from the three point perspective pose estimation[SHC+96]. They made the hypothesis that additional feature points can be used to select the correct solution. For every solution, the additional feature points can be projected onto the image plane, and the system checks how closely the projections match the detected positions of the additional feature points. But, they actually used the additional feature points to perform an optimization of the registration. When no additional feature points have been detected by the system, the system resorts to the magnetic tracker.

When more information than needed is obtained, optimization methods are often used in order to improve the quality of a registration. An optimization method compute a registration which minimizes the error associated to all the extracted feature points. The optimization method used by State et al. is a typical least square minimization[SHC+96]. Bajura and Neumann have proposed a dynamic correction method to optimize the registration performance of augmented reality systems[BN95]. They performed a dynamic correction based on an evaluation of the registration error computed from the difference of a recognizable point positions in both the real and augmented images. The strongest argument in favor of dynamic compensation is that no matter how carefully vision-based and trackers measurement and calibration are performed, registration errors will certainly exist in the augmented images. Also, since a dynamic correction can compensate for tracking errors, systems which use dynamic correction method need less accurate and less expensive tracking systems.

1.4 Motivation

The most important characteristic of a registration method is if the method is independent of the distance. If the registration is dependent on the distance, the quality of the registration decreases when the distance between the observed points and the camera increases. So far, monocular systems are commonly chosen over standard stereoscopic system as proposed by Kanbara et al. since the monocular vision-based registration is independent of the distance and the stereoscopic vision-based registration is dependent on the distance. Therefore, only a few pure stereoscopic systems have been proposed.

Another aspect of a registration method is the computation cost. Augmented reality systems are usually asked to run in real-time. In term of speed, the number of frames doubles in binocular systems. Therefore, image processing methods used by monocular systems are often too costly in time for binocular system. In other words, since less time is allocated to the image processes in binocular system, the 2D positions of the extracted points in the images are often less refined.

Even if only three points are used, the stereoscopic vision-based registration produces a single pose solution for the cameras. With three points, the monocular vision-based registration produces up to 12 solutions for each camera. The problem is solved if four points are used instead of three at the condition that the four points lay on a single plane. Also, stereoscopic vision-based registration doesn't need information measured beforehand about the position of the points. But, in monocular vision-based registration, the distance between the feature points must have been measured beforehand.

With stereoscopic computation, the 3D position of any point observed simultaneously by the two stereoscopic cameras can be evaluated. This characteristic can have many

interesting uses in binocular systems. For example, stereoscopic systems are able to resolve the occlusion between real and virtual objects because depth information of a real scene can be estimated in real-time by stereo-paired cameras. Kanbara et al., have already proposed a stereoscopic prototype which resolves the occlusion between real and virtual objects[KOT+00]. Another example is that stereoscopic systems can also use natural points in the images to increase the quality of the registration since the 3D positions of these points can be estimated. Kanbara et al., have also proposed a stereoscopic prototype which uses natural points to improve the registration stability of their stereoscopic system[KFT+00b].

When a binocular system performs the registration for each camera with a monocular vision-based technique, the registration of the left and right images are not correlated. In other words, the positions and the orientations of a virtual object in the two augmented images may differ from each other. If the difference of positions or orientations is important, the user would be confused by the incoherence between the left and the right eye scenes. In stereoscopic vision-based systems, because the relation between the cameras is considered, the left and right registrations are always coherent.

Table 1.1: Registration characteristics

	Monocular 4 points	Monocular 3 points	Stereoscopic 3 points
Number of points	4	3	3
Independent of distance	Yes	Yes	No
Computation time	1x (real-time)	1x (real-time)	2x (real-time)
Estimation of the 3D position of any points in the scene	No	No	Yes
Single pose solution	Yes	No	Yes
Coherence between left and right registrations	No	No	Yes
Special conditions	<ul style="list-style-type: none"> • Four points on the same plane • Distances between the points are needed 	<ul style="list-style-type: none"> • Distances between the points are needed 	<ul style="list-style-type: none"> • Two cameras

All in all, the stereoscopic vision-based registration possesses a significant disadvantage compared to monocular vision-based registration in practical augmented reality applications. As mentioned earlier, stereoscopic registration is dependent on the

distance. The low resolution of the stereo images, the poor estimation of the point positions and the short baseline between the stereo cameras which is limited in order to fit the user's eyes position are the major sources of the registration problems. However, because stereoscopic computation can be used to collect more information about the scene, the information accessible with stereoscopic computation must not be neglected in binocular system, even if the stereoscopic computation is not adequate to perform the registration. Table 1.1 resumes the important characteristics of the monocular vision-based registration performed with 3 or 4 points and the stereoscopic vision-based registration.

Based on the previous observation about the monocular vision-based and the stereoscopic vision-based registrations, a new registration method which fully exploits the strengths of both techniques is needed. In this research, a binocular vision-based system is proposed to fulfil this need. Consequently, our system aims to increase the depth registration of binocular systems, to perform a correction of the marker positions, to create a coherent pair of registrations, to enhance the robustness during a fast user's motion and to use a minimal number of points in order to perform the registration. In addition to those five goals, the system is also expected to run in real-time. The following sections introduce those six goals in more detail. The required characteristics of the proposed method are shown in Table 1.2.

Table 1.2: Required characteristics of the proposed method

	Proposed method
Number of points	3
Independent of distance	Yes
Computation time	2x (real-time)
Estimation of the 3D position of any points in the scene	Yes
Single pose solution	Yes
Coherence between left and right registrations	Yes
Special conditions	<ul style="list-style-type: none"> • Two cameras • Distances between the points are needed

1.4.1 Increasing the registration depth of stereoscopic systems

If binocular systems use the stereoscopic vision, the registration in the binocular system cannot be correctly performed when the user is far located from the visual markers. In other words, the registration depth, which is defined by the maximum distance allowing a correct registration between the user and the fiducial markers, is very limited in stereoscopic systems.

Nowadays, most binocular augmented reality systems are composed of two independent monocular vision-based registration modules since the registration depth associated with stereoscopic systems is too short. In fact, stereoscopic systems are limited to only some specific types of applications. Since the positions of the two cameras of a binocular system are correlated, the relation between the positions of the two cameras should be used to improve the registration. Therefore, a registration method is proposed to produce a stable registration independently of the distances between the markers and the cameras using both the stereoscopic relation between the two cameras of a binocular system and the perspective pose estimation solutions of the cameras. Consequently, the registration depth of the system using the proposed method is increased considerably. Without depth registration limitation, stereoscopic systems can be used in a larger field of applications.

1.4.2 Correction of point positions

In vision-based systems, the position of some points in the images can be used to compute the geometric registration. The point positions must be computed accurately. However, the accuracy of the position computed by vision-based processes is influenced by many sources of errors. One source of errors is that the shapes and colors of the markers may be altered by the optical devices. Another source is that the camera gives a discrete view of the scene. The pixel size of the digitalized images limits the image definition. That is, the positions of the marker points in the camera images are difficult to extract with precision since edges are smoothed and small details disappear.

Using camera calibration processes, vision-based systems are generally able to accurately compensate the effect of optical devices on the digitalized images. But, the image resolution gives some problems to the vision-based processes and the lack of resolution can't be adequately compensated. Consequently, a point position computed by a vision-based process generally includes a considerable error component caused by the lack of resolution in the images. As a result, misestimating the point positions is the first source of registration error in vision-based systems.

Therefore, registration methods are often associated to an optimization method in order to improve the quality of the registration. Usually, least square minimization based optimizations are used such as State et al.'s optimization method[SHC+96]. Bajura and Neumann put forward a correction based optimization[BN95]. In this thesis, the developed system uses a new correction based optimization. the system evaluates the consistency between the left and the right registrations and uses this information to optimize the registrations by performing a correction of the 2D feature positions.

1.4.3 Coherence between the left and right registrations

In augmented reality systems, the most important point is the feeling that the virtual objects are real. Obviously, we must not forget that the human vision is stereoscopic. Therefore, one of the conditions in order to give a realistic perception of the virtual objects to the user is that the left and right camera registrations are coherent. In other words, each user's eye must perceive that the virtual object as a unique position and orientation.

In a binocular video see-through augmented reality system, the user's eyes are replaced by two cameras. Since the position of each camera differs, each registration also differs. But, the two registrations must be correlated with their respective camera positions. The distance between the two cameras is defined by the physical setting of the head mounted device. The orientations of the cameras are also physically defined. Once the poses of the cameras have been computed by the system, those physical constants must be verified. If the physical constants are not verified, the registrations are then incoherent with the physical reality of the system setting. Consequently, the computed registrations may produce visual incoherences in the augmented images. If the incoherence is detectable by the user, the realistic perception of the virtual objects is lost. Therefore, our proposed method is asked to verify that the physical constants of the system setting are respected in order to continuously assure that the virtual object is perceived as real as possible.

1.4.4 Robustness to fast user's motion

Systems that process entirely every frame are more efficient than systems that use the tracking approach described in section 1.2.2 because systems that use the tracking approach hardly succeed to track the marker positions when a fast user's motion occurs. Therefore, some systems combine the tracking approach with additional sensors such as inertial sensors or accelerometers to estimate the marker positions in order to increase the robustness of the system when a fast user's motion occurs[KFT+00a].

When a system is robust to fast user's motion, the perception of the virtual objects in the scene increases significantly. Consequently, our proposed system aims to be robust to fast user's motions without the use of other sensors. Therefore, every frame should be

processed by our system even if our binocular system needs to process twice as many frames as a monocular system. Therefore, the proposed system must be able to quickly detect the marker positions using only image processing even in the presence of a fast user's motion in order to obtain a level of robustness similar to the robustness obtained in monocular systems.

1.4.5 Minimal number of points

At least three points are needed to perform the registration in a vision-based augmented reality system. Some systems using an increased number of points to perform an improved registration has already been proposed[KFT+00b][SHC+96]. Also, in monocular vision-based systems, using four points actually decrease the complexity of the system because the pose solution is then unique[OKT+98]. Recently, augmented reality systems tend to use square markers. Since the positions of the square corners are used as points, only one square marker is necessary to perform the registration[KBP+00][RS01].

Increasing the number of points is the easiest solution in order to achieve a better registration. Instead of improving the registration quality by using four points from the start, performing the registration using three points is studied in this research. The goal is to perform the best registration possible with only three points. Once the quality of the registration with three points has been optimized, the use of more than three points would be a possible solution to improve the registration to another level.

Since the registration of our developed system uses three points, like standard stereoscopic vision-based systems, the proposed registration method can be compared to the standard stereoscopic vision-based registration. Also, performing the registration with only three points is an important issue even for systems using four points, since those systems should also be able to produce a reasonable registration when one of the four points is missing. Consequently, a registration method using three points will also help to increase the robustness of systems using four point based registration[OKT+98].

1.4.6 Low cost hardware requirements

Many augmented reality systems need powerful and costly hardware. This fact is one of the principal reasons limiting the number of developed augmented reality applications. If we want augmented reality applications to reach our home, efforts are needed to create low cost systems. Unfortunately, head mounted devices and other associated hardwares are relatively expensive. However, an effort can be made to develop efficient algorithms which can run on home PCs.

1.5 Proposed methods

In our augmented reality system, three major tasks have to be performed. First, the positions in the frame images of the feature points used to perform the registration are computed during the extraction step. Then, the registration of the virtual objects is performed during the registration step. Finally, the correction of the feature points is applied during the optimization step. Every task needs to meet some requirements in order to fulfil the goals defined in the section 1.4.

First, the detection task must run in real-time. Also, the detection must be robust to fast user's motion and the detection must extract the positions of three feature points in the input stereoscopic frame images. Second, the registration method must perform a registration independent of the distance and the registration must be computed using only three points. Third, the optimization method must modify the extracted positions of the feature points in order to decrease the effect of detection errors. Also, the correction method must correct the registration incoherence between each stereoscopic augmented image. Figure 1.11 illustrates the three tasks and the requirements sought for every task.

In order to meet the requirements, a marker extraction strategy, a registration method and an optimization method have been developed. The developed marker detection strategy performs the extraction of the marker in every frame. The registration method produces the registration from three feature points using a combination of monocular vision-based and stereoscopic vision-based techniques. The optimization method is applied iteratively by modifying the 2D positions in the two stereoscopic frames of the feature points in order to validate the coherence between the two augmented image registrations.

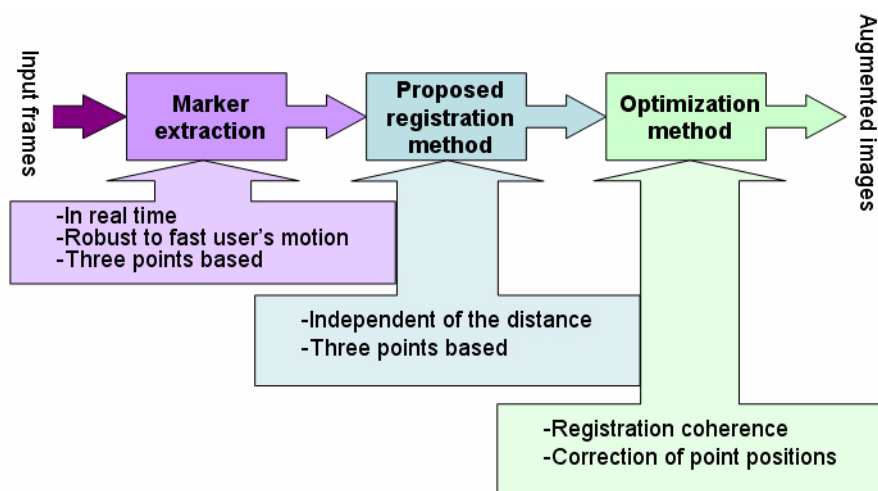


Figure 1.11 The three tasks of the system with their requirements.

In conclusion this work has mainly four original aspects. First, we use color information to extract markers as already used in other systems [KFT+00a][KIT+00][OKT+98][OSY+98], but we also use the color information to discriminate multiple markers instead of template matching or other techniques[ARTK][BKP01][KBP+00]. Second, a three point based registration method is studied. Only few papers talk about how to register virtual objects in this condition[KOT+00][OKT+98][SHC+96]. Third, we optimize the registration using correction of the 2D feature positions instead of using Bajura and Neumann's dynamic correction([BN95]) or typical least square minimization([SHC+96]). Fourth, we present a new way to evaluate quantitatively the registration stability instead of using already proposed evaluations[KIO00][NP98][SAY+01][SHC+96][SON01].

2. Robust registration method for vision-based augmented reality using monocular and binocular vision with feature position correction

This chapter presents a system developed to fulfill the goals presented in section 1.4. This chapter is divided in six sections. The first section overviews the system tasks and their requirements. The section 2.2 details the marker detection and the marker extraction strategies. Then, section 2.3 and section 2.4 respectively present the proposed registration and optimization methods.

2.1 Overview

The developed system should run in real-time on a common PC. In other words, the image processing needs to be kept as small as possible. Also, the system is required to be robust to fast user's motion. Therefore, instead of using typical image processes found in the literature, a quick and simple method based on color was developed to extract the marker in the images.

The goal of the proposed registration method is to combine monocular vision-based and stereoscopic vision-based processes to fully exploit the strengths of both techniques in order to produce a three point based registration for binocular augmented reality systems. In the first place, the proposed method must be independent of the distance. Consequently, the registration will principally be based on the monocular vision-based registration. Since stereoscopic computations give enough information to compute the registration using three points, only three points will be used to perform the monocular vision-based registration. The extended information obtained by stereoscopic computations will be used to discriminate the multiple pose solutions given by the monocular three point based registration.

In a second time, stereoscopic computations are used to compute an error coefficient of the registration performed. The major source of registration error is attributed to the measurement errors of the 2D positions of the feature points in the image. Consequently, the error coefficient computed with stereoscopic computation is used to perform a correction of the 2D positions of the points. Thus, the correction method is a new optimization method based on the correction of the 2D positions of the feature points proposed as an alternative of existing optimization methods.

All in all, our robust registration method for vision-based augmented reality using monocular and binocular vision with feature position correction is divided in three steps.

The first step includes the image processing of the input frames to extract the positions of the marker feature points(section 2.2). The second step performs the registration using the proposed registration method(section 2.3). The third step optimizes the registration by applying the correction method(section 2.4). Figure 2.1 shows the global flow chart of our system.



Figure 2.1 Global flow char of the proposed system.

2.2 Marker extraction

With the method described in section 1.2.2, vision-based augmented reality systems process only some zones of the frames in order to decrease the processing time. Systems using this method estimate the positions of the markers in the current frame based on the positions of the markers in the previous frame. Then, the systems perform refine positions of the marker points of interest by processing only the pixels in a small zone centered at the estimated marker positions. When a fast user's motion occurs, those systems often fail to retrieve the position of the fiducial markers. Consequently, a tracking failure occurs. Therefore, some systems using the described tracking method add sensors such as inertial sensors or accelerometers to estimate the fiducial marker positions, even during fast user's motions[KFT+00a]. Our proposed system enhances the robustness during fast user's motions without the use of a motion sensor since the marker positions are extracted without an estimation of the marker position. Therefore, a detection strategy has been developed in order to extract the position of the markers inside entire frames within a reasonable delay of time.

Instead of tracking the positions of the features from one frame to the other using the hypothesis that the feature positions don't change significantly in two consecutive frames, the positions of the feature points are independently extracted in each frame. Therefore, the system is able to acquire the feature positions in all frames even if a fast user's motion occurs, but the feature points must be detected with a minimum of processing in order to extract the position of the three features in real-time. Moreover, the processing must be done twice since two frames need to be processed: the left camera and the right camera frames. Consequently, the feature shapes and colors need to be easy to detect.

Three prior interrogations must be answered about the image processing that performs the extraction of the markers: how the marker should be designed, how the marker features should be extracted and how multiple markers should be discriminated. A vast number of solutions may be proposed for each interrogation. According to the choice

made, the efficiency of the extraction varies. The next sections give the proposed solution to each interrogation.

2.2.1 Design of the markers

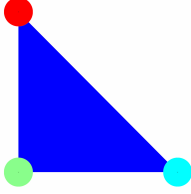
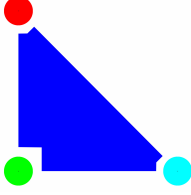
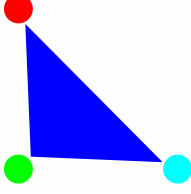
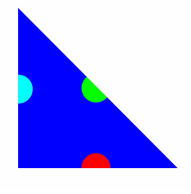
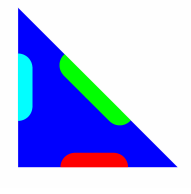
In vision-based systems, at least three points are needed to perform the registration. Systems using an increased number of points to perform a better registration have already been proposed[KFT+00b][SHC+96]. Also, in the case of monocular vision-based systems, the use of four points actually decreases the complexity of the pose estimation[OKT+98]. Therefore, augmented reality systems recently tend to use squares. Since the positions of the square corners are used as points, only one square is needed to perform the registration[KBP+00][RS01][ARTK]. Applications using multiple square models often discriminate each model by detecting specific patterns placed at the center of the squares[BKP01]. In contrast, our system aims to use only three points to perform the registration. Therefore, triangular markers have been used in the developed systems.

Three principal factors influence the design of the triangular markers. Firstly, the triangular markers must be easy to detect in the camera frames in order to quickly extract the position of the marker feature points. Secondly, the discrimination of multiple triangular markers must be allowed. Thirdly, the shape of the marker used is intimately linked to the registration method. Since the goal is to produce a three point based registration, triangular markers are used instead of squares in order to evaluate the proposed three point based registration method. However, with only few modifications, the extraction method can also be used to extract square marker.

The initial design of the markers is shown in the first row of Table 2.1. The marker printed on a white sheet of paper is composed of three colored circles positioned at the three corners of a blue triangle. The blue triangle is used only for a quick estimation of the three circular features in order to decrease the processing time of the feature detection. Multiple markers are discriminated from each other based on the color of the three circles.

The detection strategy used to detect the marker feature points of the initial design is divided in three main steps. Firstly, the triangle edges are quickly detected in both frames. The intersections of the edges give the estimations of the feature point positions. Next, refined positions of the feature points are computed. Finally, the marker is identified and orientated. From the initial design of the markers, the design was modified several times in order to improve the extraction. Table 2.1 illustrates the design evolution of the marker. The table also gives, for each marker design, the main reason why the design needs to be modified and the modification performed to solve the problem associated to the previous design.

Table 2.1: Evolution of the marker design

Marker design	Solution	Main problem
	<p>The initial design is composed of three colored circles positioned at the three corners of a blue triangle.</p>	<p>Since the regions are extracted based on the discrimination between the colors, borderline errors often occur between the circular features and the blue triangle under some lighting condition.</p>
	<p>The blue triangle is isolated from the features.</p>	<p>Since all the outline pixels of the blue triangle are edge pixels, the extraction of the triangle edges is ineffective.</p>
	<p>Instead of isolating the triangle from the feature, the triangle is shortened.</p>	<p>Since the intersections of the edges don't correspond to the center of the features anymore, the estimation of the feature positions is ineffective.</p>
	<p>Instead of using circular features, the corners of the triangle are now used. Also, three colored regions are inserted in the triangle to allow the discrimination between multiple markers.</p>	<p>Since the colored regions used to identify the marker are relatively small, the identification of the marker is ineffective when the marker is too small in the input images.</p>
	<p>The colored regions are enlarged.</p>	<p>This marker design is the design presently used in our system.</p>

The final design of the marker is shown in the last row of Table 2.1. The major modification between the initial and the final designs concerns the feature points. In the initial design, the centers of gravity of the three circular features are the points used to perform the registration. In the final design, the three corners of the triangle are now used as the feature points. The stability of the registration improves when using the final design since identifying the position of a corner is less influenced by detection error than identifying the center of gravity of a circular region. Another modification concerns the discrimination between multiple markers. Since the circular features are absent in the final design, colored zones are added inside the triangle in order to perform the discrimination between multiple markers. The discrimination between multiple markers is possible because every marker contains a unique set of three colors. Once a marker has been identified, the dimension of the marker, the position of the marker in the world coordinate system and the virtual object associated with the marker are known.

The detection strategy used to detect the marker feature points of the final design is also divided in three main steps. First, the triangle marker is quickly extracted in both stereoscopic frames (section 2.2.2). Second, the feature point positions are given by the intersections of the triangle marker edges (section 2.2.3). Finally, the marker is identified and orientated (section 2.2.4). Figure 2.2 shows the three main steps of the detection strategy.

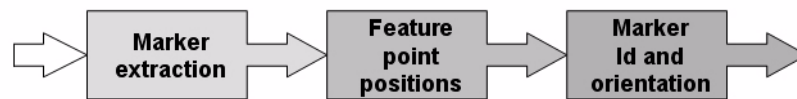


Figure 2.2 The main steps of the detection.

2.2.2 Marker extraction

Because systems that track the marker with a strategy using searching windows often fail to retrieve the marker when a fast user’s motion occurs, a detection strategy has been developed in order to extract the position of the marker feature points inside every entire frame within a reasonable delay of time. Using this extraction strategy, our system aims to be robust to rapid user’s motion.

Without time constraint, the basic strategy will be to extract all blue pixels of each stereoscopic frame in order to find all possible blue triangular markers in the frames (Figure 2.3b). Then, the blue pixels are segmented in blue regions. The three farthest pixels of every region are found (Figure 2.3c). If the segmented region corresponds to a marker, those pixels give the three corner positions of the triangular region. In other

words, those pixels give the feature points used to perform the registration. Figure 2.3 illustrates the basic strategy.

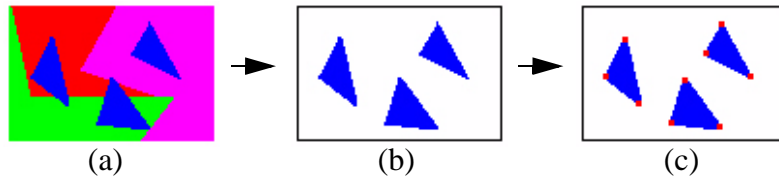


Figure 2.3 Illustration of the marker extraction strategy.

Extracting the marker position in both stereoscopic frames by processing all the pixels of each frame give accurate results, but the extraction is time consuming. Consequently, a new strategy is developed in order to extract the triangular marker. The first step of the developed strategy aims to decrease the processing time. Decreasing the processing time implies that the computed measurements are less accurate. Therefore, a second step is added to the strategy in order to improve the accuracy. Therefore, according to the new strategy, the maker extraction step is subdivided in two steps.

In a frame image, the neighborhood of a pixel is highly correlated with the color of this pixel. Therefore, only the central pixel of a neighborhood can be processed in order to evaluate the color of the neighborhood zone. The decreasing of the processing time is directly influenced by the definition of a neighborhood since more the neighborhood covers a large area, more the processing time decreases. On the other hand, more the neighborhood covers a large area, less the color identifying a neighborhood is accurate.

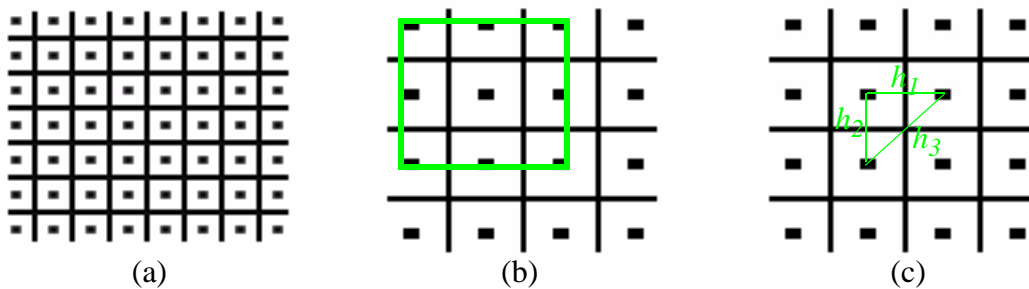


Figure 2.4 Illustration of the square neighborhood division pattern.

Based on some experimental observations, a neighborhood has been defined as an area of 25 pixels. In other words, only one pixel out of 25 pixels is processed. Consequently, the processing time is decreased by a factor 25. One possibility for the neighborhood shape is a 5x5 square. The neighborhood color is then represented by the color of the processed pixel located at the center pixel of a neighborhood. Figure 2.4a illustrates the

square pattern. At least one pixel of a region must be processed in order to extract the region. Figure 2.4b shows a square region detected by the square pattern. The distances h_1 , h_2 and h_3 between two processed adjacent neighbor pixels are different as shown in Figure 2.4c. The distance h_1 and h_2 are equal in a square pattern but these distances may have different values in order to create a rectangular pattern.

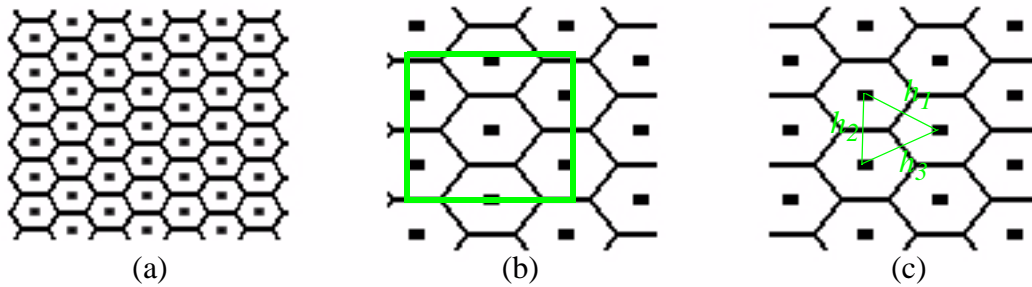


Figure 2.5 Illustration of the hexagonal neighborhood division pattern.

Instead of using the square neighborhood pattern, a hexagonal pattern is used. The hexagonal neighborhood pattern is shown in Figure 2.5a. Compared to the square pattern, a hexagonal pattern extracts a same surface using less processed pixels (Figure 2.5b). Since less pixels are processed in order to extract the regions of interest in an image, the time spent to extract the marker regions decreases. Furthermore, the distances h_u between two processed adjacent neighborhood pixels are now constant (Figure 2.5c). Then, the value of the distances h_u defines the density parameter H of the hexagonal pattern. If H is small, the density of pixels is high, and if H is high, the density of pixels is low.

Once the pixels to process have been identified in the image from the input frame (Figure 2.6a et Figure 2.6b), the blue pixels among the processed pixels are extracted and segmented in regions (Figure 2.6c). Then, the three farthest pixels are computed among \mathcal{S}_u where \mathcal{S}_u is the set including all the n_u processed pixels of the segmented region u (Figure 2.6d). Those pixels give an estimation of the corner positions of the triangular region. The algorithm used to find the three farthest pixels ρ_a , ρ_b and ρ_c is described and illustrated in Table 2.2.

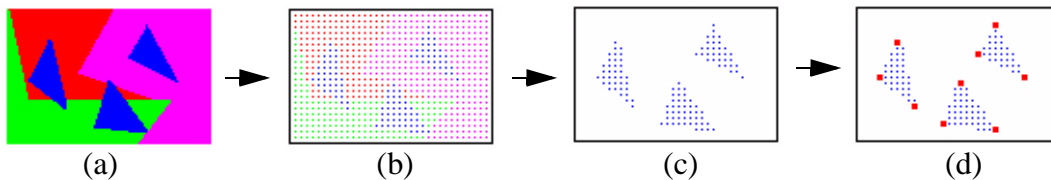


Figure 2.6 Example of extraction using the hexagonal pattern.

Table 2.2: Algorithm to find the three farthest pixels p_a , p_b and p_c

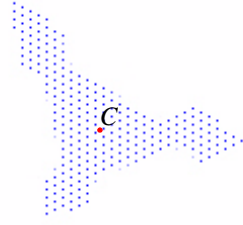
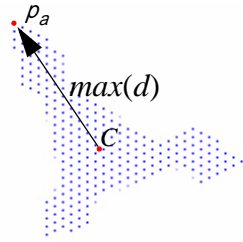
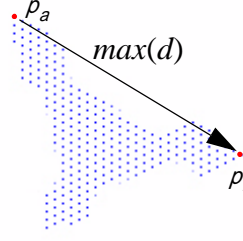
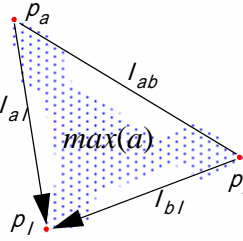
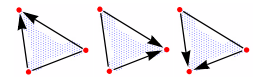
step	description	illustration
1	<p>Find with equation 2.1 the center of gravity $C_g(c_x, c_y)$ from the processed blue pixels $p_j(x_j, y_j)$ of the segmented region S_u.</p> $c_x = \sum_{v=1}^{n_u} \frac{x_v}{n_u}, \quad c_y = \sum_{v=1}^{n_u} \frac{y_v}{n_u} \quad (2.1)$	
2	<p>Find the pixel $p_a(x_a, y_a)$ that maximize the euclidian distance d between the center $C_g(c_x, c_y)$ and a pixel $p_v(x_v, y_v)$ among S_u (equation 2.2).</p> $p_v(x_v, y_v) \in S_u$ $p_a(x_a, y_a) = p_v(x_v, y_v) \quad (2.2)$ <p>if $\max(d) = \sqrt{(c_x - x_j)^2 + (c_y - y_j)^2}$</p>	
3	<p>Find the pixel $p_b(x_b, y_b)$ that maximize the euclidian distance d between the pixel $p_a(x_a, y_a)$ and a pixel $p_j(x_j, y_j)$ among S_k (equation 2.3).</p> $p_v(x_v, y_v) \in S_u$ $p_b(x_b, y_b) = p_v(x_v, y_v) \quad (2.3)$ <p>if $\max(d) = \sqrt{(x_a - x_j)^2 + (y_a - y_j)^2}$</p>	
4	<p>Find the pixel $p_c(x_c, y_c)$ that maximize the area A of the triangle of edge lengths l_{ab}, l_{aI} and l_{bI} created from the pixel $p_a(x_a, y_a)$, $p_b(x_b, y_b)$ and a pixel $p_v(x_v, y_v)$ among S_u (equation 2.4).</p> $p_v(x_v, y_v) \in S_u$ $p_c(x_c, y_c) = p_v(x_v, y_v) \quad (2.4)$ <p>if $\max(A) = \sqrt{s(s - l_{ab})(s - l_{aI})(s - l_{bI})}$</p> <p>with $s = \frac{l_{ab} + l_{aI} + l_{bI}}{2}$</p>	

Table 2.2: Algorithm to find the three farthest pixels ρ_a, ρ_b and ρ_c

step	description	illustration
5, 6, 7	Verify if the pixels ρ_a, ρ_b and ρ_c are really the pixels that maximize the area A of the triangle by repeating step 4.	

The three farthest pixels give an estimation of the three triangle corners. The accuracy of the estimations obtained with the algorithm of Table 2.2 is indirectly proportional to the density H defining the hexagonal pattern (more H is small, more the accuracy is high). Figure 2.7 illustrates the relation obtained between the corner estimations of a triangle marker and the actual positions of the triangle corners.

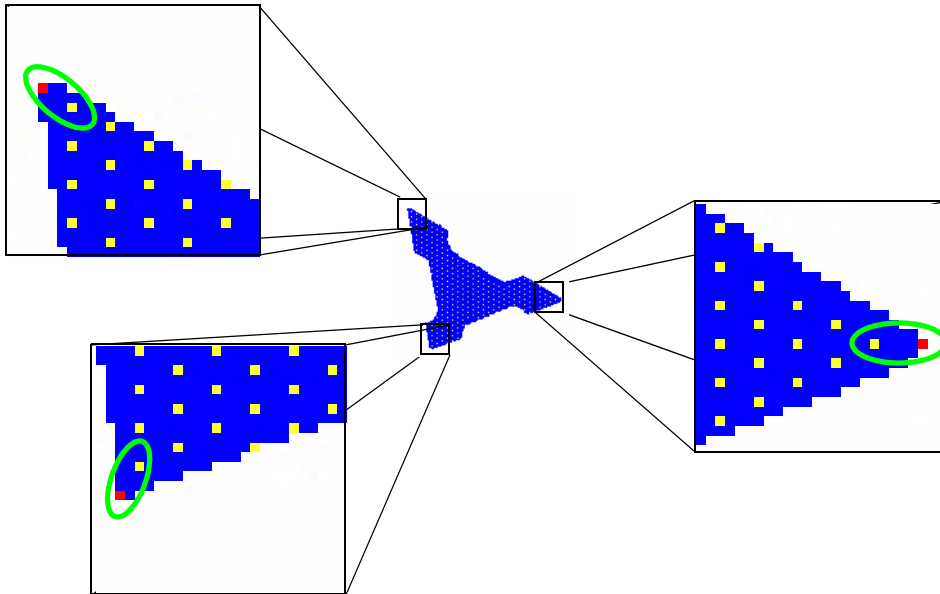


Figure 2.7 Relation between the corner positions and the estimations.

Then, the three edge equations of the triangular marker have to be computed. If the equations of the marker edges are determined, the position of the marker feature points is easily calculable. Therefore, two pixels on each triangle edge need to be found in order to deduce the three edge equations of the triangle. From the three corner estimations, three different groups of two points can be created : (ρ_a, ρ_b) , (ρ_a, ρ_c) and (ρ_b, ρ_c) . For each group of two points (ρ_u, ρ_v) , the normal vector \vec{N}_{uv} of the line passing by the two points is computed using equation 2.5. The sign of the vector \vec{N}_{uv} in equation 2.5 is chosen in order that \vec{N}_{uv} is oriented toward the outside of the triangular marker. Then, from each of the two points of the group, the pixels of the frame are scanned in the normal vector direction in order to locate a pixel on the edge of the triangle. The length L_N of the

scanning, or the length of a normal \vec{N}_{uv} , depends on the dimension H defining the hexagonal pattern. The edge is positioned on the first white pixel encountered. A simple one dimension edge detector is used to locate the position of the edge. That is, the edge detector $D(u)$ returns true only if the pixel u is white and the pixel $u-1$ is not (equation 2.6). Figure 2.8 illustrates the finding of the six pixels that give the edges of the triangular marker.

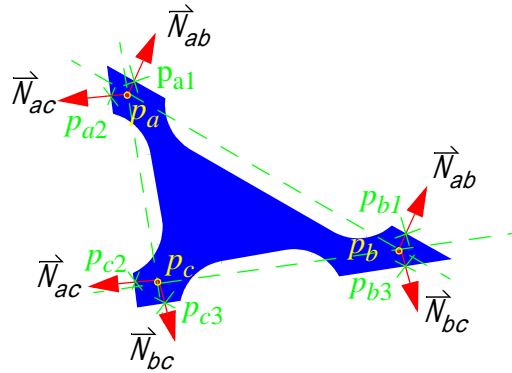


Figure 2.8 Localization of the six edge pixels.

$$\vec{N}_{uv} = \pm \begin{bmatrix} \frac{(\rho_{uy} - \rho_{vy})}{|\rho_v - \rho_u|} \\ \frac{(\rho_{vx} - \rho_{ux})}{|\rho_v - \rho_u|} \end{bmatrix} \quad (2.5)$$

$$D(u) = \begin{cases} \text{true} & \text{if } \begin{cases} \text{pixel } u = \text{white} \\ \text{pixel } u-1 \neq \text{white} \end{cases} \\ \text{false} & \text{otherwise} \end{cases} \quad (2.6)$$

Two pixels are then found for every edge. The equation of an edge is computed using equation 2.7 where $(\rho_u(x_u, y_u), \rho_v(x_v, y_v))$ take the values (ρ_{a1}, ρ_{b1}) , (ρ_{a2}, ρ_{c2}) or (ρ_{b3}, ρ_{c3}) depending on the case. The equation of an edge is defined using this equation form in order to avoid division by zero.

$$(x - x_k)(y_l - y_k) - (y - y_k)(y_l - y_k) = 0 \quad (2.7)$$

2.2.3 Feature point positions

The feature point positions which are the three marker corners have to be computed in order to perform the registration. If the equations of the three edges have previously been successfully computed, one corner of the triangle marker is given by the intersection of two edge lines. The equation 2.8 gives the systems of equations to solve in order to compute the three corners $\vec{f}'_a(a_x, a_y)$, $\vec{f}'_b(b_x, b_y)$ and $\vec{f}'_c(c_x, c_y)$ of the marker.

$$\begin{aligned}
 \vec{f}'_a: & \begin{cases} (a_x - x_{a1})(y_{b1} - y_{a1}) - (a_y - y_{a1})(x_{b1} - x_{a1}) = 0 \\ (a_x - x_{a2})(y_{b2} - y_{a2}) - (a_y - y_{a2})(x_{b2} - x_{a2}) = 0 \end{cases} \\
 \vec{f}'_b: & \begin{cases} (b_x - x_{a1})(y_{b1} - y_{a1}) - (b_y - y_{a1})(x_{b1} - x_{a1}) = 0 \\ (b_x - x_{a3})(y_{b3} - y_{a3}) - (b_y - y_{a3})(x_{b3} - x_{a3}) = 0 \end{cases} \quad (2.8) \\
 \vec{f}'_c: & \begin{cases} (c_x - x_{a2})(y_{b2} - y_{a2}) - (c_y - y_{a2})(x_{b2} - x_{a2}) = 0 \\ (c_x - x_{a3})(y_{b3} - y_{a3}) - (c_y - y_{a3})(x_{b3} - x_{a3}) = 0 \end{cases}
 \end{aligned}$$

The three corner positions are computed with sub-pixel resolution, but the edge equations used are not precise enough to validate the use of a sub-pixel precision. In fact, the precision of the three corners is not significant since the six pixels used to compute the edge equations have been obtained using an edge detection with a resolution of one pixel. Consequently, the quality of the registration will be badly influenced by this lack of precision. Therefore, the equations of the triangle edges need to be extracted with more accuracy.

For two corners $\vec{f}'_u(x_u, y_u)$ and $\vec{f}'_v(x_v, y_v)$ obtained with the equation 2.8, the equation of the line segment joining the two corners is computed using equation 2.7 where $\rho_u(x_u, y_u) = \vec{f}'_u$ and $\rho_v(x_v, y_v) = \vec{f}'_v$. The length L_{uv} of the segment is given by the euclidian distance between ρ_u and ρ_v . Once more, the outside oriented normal vector \vec{N}_{uv} of length L_N is defined for the edge defined with ρ_u and ρ_v (see equation 2.5). Then, n pixels of the edge are extracted from the frame image. First, an approximation of the edge position e_k is obtained from the equation 2.9. From this pixel approximation, a edge pixel E_k is located by scanning the pixels belonging to the set of pixels U (see equation 2.10) with a classic sub-pixel edge detector.

$$e_k = \frac{k}{L_{uv}}(\vec{P}_v - \vec{P}_u) \quad \text{where } k \in \mathfrak{N} \text{ and } 0 < k < n \quad (2.9)$$

$$U = \left\{ e_{k - \frac{L_N \vec{N}_{uv}}{2}} \dots, e_k + v \vec{N}_{uv} \dots, e_{k + L_N \vec{N}_{uv}} \right\} \quad \text{where } \begin{cases} v \in \mathfrak{N} \\ -\frac{L_N}{2} < v < L_N \end{cases} \quad (2.10)$$

for $u \in U$, $E_k = u$ if $D(u) = \text{true}$

Once the n edge pixels have been located, the new edge equation of the edge previously defined by p_u and p_v is given by the line that best fit all the n edge pixels E_k . A least square minimization is used to fit a straight line on the n pixels (see Appendix D). Since fitting a line on a large number of pixels leads to a line equation with significant sub-pixel precision, the newly computed equation is enough precise to calculate the corners of the triangle marker with significant sub-pixel precision. However, n must be large enough. When the three edge equations of the triangle have been obtained with significant accuracy, the three corners $\vec{f}_a(a_x a_y)$, $\vec{f}_b(b_x b_y)$ and $\vec{f}_c(c_x c_y)$ of the marker are computed with the equation 2.8 using the newly defined equation of the edges. The three corners $\vec{f}_a(a_x a_y)$, $\vec{f}_b(b_x b_y)$ and $\vec{f}_c(c_x c_y)$ have significant sub-pixel precision to perform an accurate registration.

2.2.4 Marker identification and orientation

When the three corners of the marker have been found, the triangular marker must be identified. Different marker identities allow the system to discriminate between multiple markers. The colored regions inserted in each triangular marker must be recognized. The identity of a marker is given by the unique group of three colors associated with it. Four colors are used: green, yellow, cyan and magenta. The green is reserved to identify the hypotenuse of the triangle marker. The position of the hypotenuse is an important information in order to orientate the marker, or in other words, to locate the right angle corner of the triangle.

First, the three colored regions are localized from the positions of the corners. The positions of the colored regions are given by equation 2.11. Also, the center of the triangle marker is computed using equation 2.13. In addition, two increment values Δ_{ux} and Δ_{uy} are defined for every region. The equation 2.12 computes those six increment values. The factor 6 in the equations refers to the fact that the width of the colored region is about $\frac{1}{6}$ of the distance between a pixel $\vec{r}_u(r_{ux} r_{uy})$ and the center \vec{C}_f .

$$\vec{r}_a = \frac{\vec{f}_b + \vec{f}_c}{2}, \vec{r}_b = \frac{\vec{f}_a + \vec{f}_c}{2}, \vec{r}_c = \frac{\vec{f}_a + \vec{f}_b}{2} \quad (2.11)$$

$$\begin{aligned}\Delta_{ax} &= \frac{C_x - r_{ax}}{6l}, \Delta_{ay} = \frac{C_y - r_{ay}}{6l} \\ \Delta_{bx} &= \frac{C_x - r_{bx}}{6l}, \Delta_{by} = \frac{C_y - r_{by}}{6l} \\ \Delta_{cx} &= \frac{C_x - r_{cx}}{6l}, \Delta_{cy} = \frac{C_y - r_{cy}}{6l}\end{aligned}\quad (2.12)$$

$$\vec{C}_f(C_x, C_y) = \frac{\vec{r}_a + \vec{r}_b + \vec{r}_c}{3} \quad (2.13)$$

Then, the color of every region is identified. More than one pixel are used to identify a region color in order to decrease the effect of the noise inducted in the input images. The variable l of equation 2.12 corresponds to the number of points used to identify a region. The l pixels form a set of pixels \mathcal{S} where the pixels are positioned using equation 2.14.

$$\begin{aligned}\mathcal{S} &= \{s_0, \dots, s_{l-1}\} \\ \text{where } s_v &= (r_{ux} + v\Delta_{ux}, r_{uy} + v\Delta_{uy})\end{aligned}\quad (2.14)$$

The RGB colors of the l pixels are transformed in the HSV color system (see Appendix A). The hue value of a pixel color is compared with the hue ranges of Table 2.3. If the hue value h fits inside the range of one of the four colors, the corresponding color score one point. The color that scores the more number of points identifies the color of the region.

Table 2.3: Hue range for the four identification colors

color	range of hue values
green (G)	$90^\circ \leq h \leq 175^\circ$
magenta (M)	$280^\circ \leq h \leq 350^\circ$
yellow (Y)	$25^\circ \leq h \leq 75^\circ$
cyan (C)	$175^\circ \leq h \leq 225^\circ$

Once the colors of the three regions located by \vec{r}_a , \vec{r}_b and \vec{r}_c have been identified, the marker identity is coded into three individual values v_a , v_b and v_c , one value for every region color. The values depend on the color and the region as shown in Table 2.4. The global code \mathcal{G} given to a marker corresponds to the sum of the v_u . Under some

illumination conditions, the color of a region may be mismatched. Furthermore, an identification error occurs if one of the three regions has an unknown color.

Table 2.4: Codification of the colors

color	value v_a located by \vec{r}_a	value v_b located by \vec{r}_b	value v_c located by \vec{r}_c
green (G)	0x000010	0x001000	0x100000
magenta (M)	0x000001	0x000100	0x010000
yellow (Y)	0x000002	0x000200	0x020000
cyan (C)	0x000003	0x000300	0x030000
unknown (U)	0x000000	0x000000	0x000000

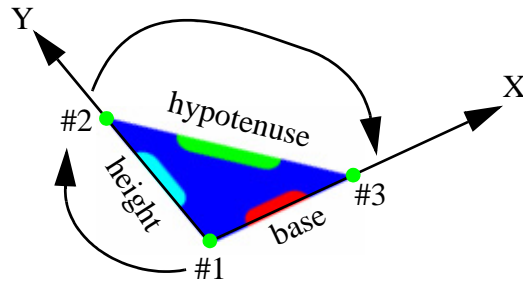


Figure 2.9 Illustration of the wanted orientation of corners.

Next, an attempt to orient the marker is performed. By listing the three corners \vec{f}_a , \vec{f}_b and \vec{f}_c of the marker in a defined order $\vec{f}_1 \vec{f}_2 \vec{f}_3$, the local marker coordinate system M is easily retrieved. By definition (see set of equations 1.4), the axis \vec{M}_x of the marker coordinate system is given by the normalized vector joined \vec{f}_1 and \vec{f}_2 . The axis \vec{M}_y is given by the normalized vector joined \vec{f}_1 and \vec{f}_3 . The axis \vec{M}_z is then given by the cross product of the axes \vec{M}_x and \vec{M}_y . Therefore, \vec{f}_1 corresponds to the right angle corner opposed to the hypotenuse. Then, the equation 2.15 is used to determine \vec{f}_2 and \vec{f}_3 . Among the two remaining corners \vec{q}_1 and \vec{q}_2 , \vec{f}_2 corresponds to the corner given a positive value s and \vec{f}_3 corresponds to the corner given a negative value s in equation 2.15 where \vec{h}_ρ is the middle of the hypotenuse. Also, the global code G is reorganized to correspond to the order $\vec{f}_1 \vec{f}_2 \vec{f}_3$. Figure 2.9 illustrates the wanted orientation of $\vec{f}_1 \vec{f}_2 \vec{f}_3$.

$$\vec{h}_p = \frac{\vec{q}_1 + \vec{q}_2}{2} \quad (2.15)$$

$$s = (q_x - h_{px})(f_{1y} - h_{py}) - (q_y - h_{py})(f_{1x} - h_{px})$$

The orientation of the marker succeeds only if the hypotenuse is successfully identified. In other words, only one green region must be identified (equation 2.16). In all the other cases, the orientation of the marker is undetermined. In equation 2.16, the operator $|$ refers to the bitwise “OR” operator.

$$\mathcal{G} | 0x101010 \in \{0x000010, 0x001000, 0x100000\} \quad (2.16)$$

If a triangle has been correctly identified and oriented, the global code \mathcal{G} of the marker must be one of the valid code presented in Table 2.5. The number of discriminable marker is limited to nine markers when using three colors for the identification of the marker and one color for the identification of the hypotenuse. The number of discriminable markers can be increased if other colors like black, brown and red are used, but the range of markers which could be robustly discriminated across widely varying illumination conditions is going to be small compared to other proposed methods such as template matching[BKP01][ARTK]. Although, the range of markers should be sufficiently large for usual augmented reality implementations.

Table 2.5: Valid global code for \mathcal{G}

0x100101, 0x100102, 0x100103, 0x100201, 0x100202, 0x100203, 0x100301, 0x100302, 0x100303
--

Each marker is observed twice, once by each of the stereoscopic camera. Therefore, the correspondence between the markers detected in the left camera image and the markers detected in the right camera image is achieved. In order to achieve the correspondence between the two stereoscopic images, an identification process creates, fills and analyses a set of marker structures. The element included in a marker structure is given in Table 2.6.

The identification process is divided into three steps. First, a marker structure is created and filled for every extracted marker in the left image. Next, if a marker extracted in the right image has the same code \mathcal{G} of a marker extracted in the left image, the marker structure associated to the left image marker is completed with the information obtained from the right image. Otherwise, a new marker structure is created and filled. Finally, the marker structures are analyzed.

Table 2.6: Marker structure

variable	use
global code (\mathcal{G})	Keep the global code \mathcal{G} of the marker.
validity (V)	Say if the marker has been successfully oriented or not.
in left (L)	Say if the marker exists in the left image.
in right (R)	Say if the marker exists in the right image.
positions in left (f_{l_1} , f_{l_2} and f_{l_3})	Keep the positions of \vec{T}_1 , \vec{T}_2 and \vec{T}_3 in the left image.
positions in right (f_{r_1} , f_{r_2} and f_{r_3})	Keep the positions of \vec{T}_1 , \vec{T}_2 and \vec{T}_3 in the right image.

Table 2.7: Description flags

flag	meaning
complete	A complete marker is an oriented marker (V is set) with a valid global code (\mathcal{G} is in Table 2.5) that exists in both images (L and R are set).
half-left	A half-left marker is an oriented marker (V is set) found only in the left image (L is set, R is cleared). The global code \mathcal{G} can be valid or invalid.
half-right	A half-right marker is an oriented marker (V is set) found only in the right image (R is set, L is cleared). The global code \mathcal{G} can be valid or invalid.
invalid left	An invalid left marker is a half-left marker not oriented (L is set, R and V are cleared).
invalid right	An invalid right marker is a half-right marker not oriented (R is set, L and V are cleared).

During the analyze step, all the created marker structures are analyzed. Each marker structure receives a description flag. The possible flags are listed in Table 2.7. When identification errors occur, a complete marker may be extracted as a group of half-left and

half-right markers. By comparing the markers of both images, identification and orientation errors may be corrected and two half marker structures may be merged into a complete marker structure.

Two rules control the merging between marker structures. The first rule specifies that only a couple of left and right marker structures can be merged together, and the second rule specifies that two invalid marker structures cannot be merged. Based on these rules, only three types of merging is possible: half-left with half-right, half-left with invalid right and invalid left with half-right.

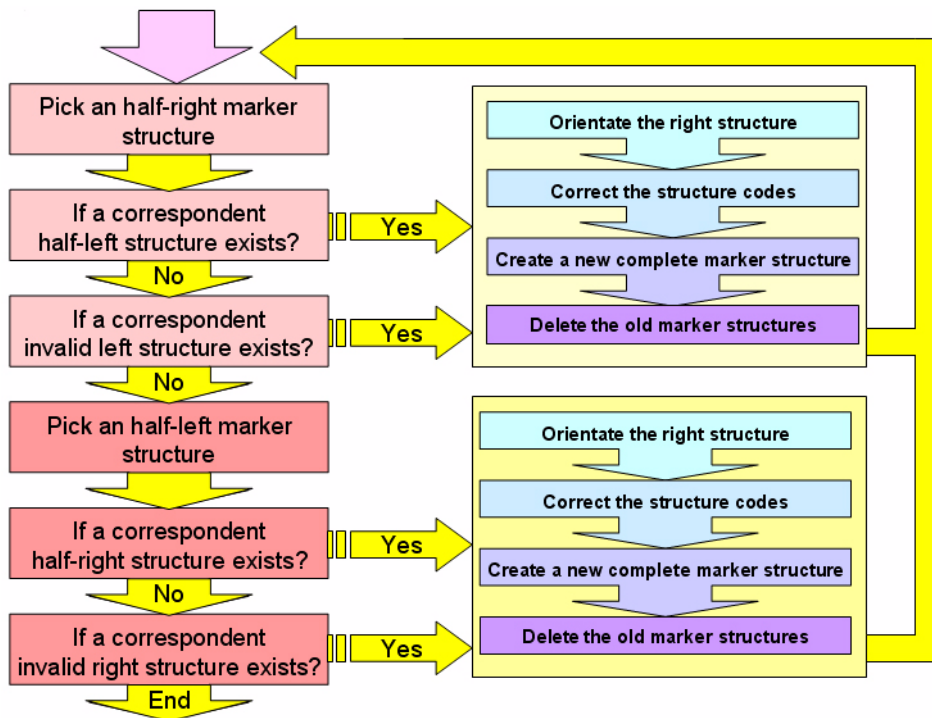


Figure 2.10 Algorithm for merging incomplete marker structure.

Two conditions must be verified to justify the merging of two marker structures. First, the epipolar constraint is used to verify if the positions of the markers correspond. Second, the global code G of the two marker structures must be sufficiently similar to identify and correct the errors inside the global code. Figure 2.10 illustrates the algorithm used to merge incomplete marker structure. An extracted blue region is taken as a non-marker region when the marker identity and orientation cannot be determined. Also, all the remaining half and invalid marker structures are ignored.

2.3 Registration method

In vision-based augmented reality systems, the relationship between the virtual and the real worlds needs to be estimated to perform the registration of the virtual objects. This section presents a registration method which aims to increase the accuracy of 3D position estimations in video see-through augmented reality systems using stereo-paired cameras. The three point based registration performed using triangular markers aims to be independent of the distance between the cameras and the markers.

2.3.1 Overview

As already mentioned, the registration method aims to produce correct registration from three feature points. Most of the systems fail to produce a correct registration if only three points are available. This registration method is an alternative method to keep producing the registration when only three points are available. Therefore, one goal of the method is to allow system using square markers to continue to produce the registration when one of the square corners is not available.

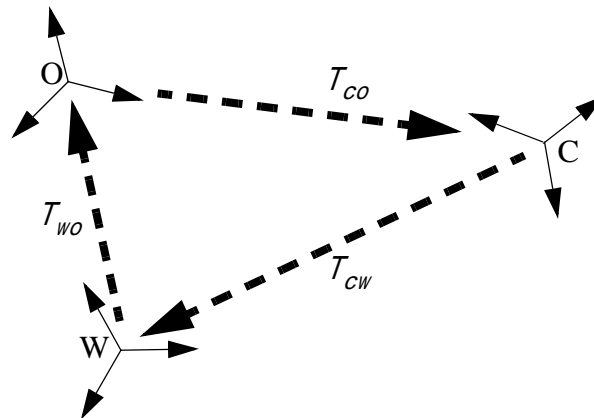


Figure 2.11 Relations between the coordinate systems.

To perform the registration, the relations between the positions of the cameras and the positions of the virtual objects are needed. The transformation matrix T_{CO} from an object coordinate system O to a camera coordinate system C represented in Figure 2.11 must be deduced from the scene. The relation T_{WO} between a virtual object coordinate system O and the world coordinate system W is usually known from the wanted topology of the scene. If the relation T_{CW} between the camera coordinate system and the world coordinate system is retrieved, the relation T_{CO} can be deduced with the equation 2.17. Therefore, the pose (the position and the orientation) T_{CW} of the cameras in the world coordinate system W can be computed.

$$T_{co} = T_{cw} \times T_{wo} \quad (2.17)$$

A camera pose can be retrieved when a marker is observed by the cameras. The three feature points \vec{F}_1 , \vec{F}_2 and \vec{F}_3 of a marker define a local marker coordinate system M given by the set of equations 1.4 which can be rewritten as the set of equations 2.18 with respect to the right angle of the triangular marker. The relation T_{wm} between a marker coordinate system and the world coordinate system is measurable for each marker placed in the environment by the user. Since the equation 2.19 stands, the missing relation T_{cw} can be retrieved if the relation T_{cm} between an observed marker and a camera is calculated from the observed scene. Figure 2.12 illustrates the decomposition of T_{cw} into T_{cm} and T_{mw} .

$$\vec{M}_x = \begin{bmatrix} M_{xx} \\ M_{xy} \\ M_{xz} \end{bmatrix} = \frac{\vec{F}_2 - \vec{F}_1}{|\vec{F}_2 - \vec{F}_1|}$$

$$\vec{M}_y = \begin{bmatrix} M_{yx} \\ M_{yy} \\ M_{yz} \end{bmatrix} = \frac{\vec{F}_3 - \vec{F}_1}{|\vec{F}_3 - \vec{F}_1|} \quad (2.18)$$

$$\vec{M}_z = \begin{bmatrix} M_{zx} \\ M_{zy} \\ M_{zz} \end{bmatrix} = \vec{M}_x \times \vec{M}_y$$

$$T_{cw} = T_{cm} \times T_{mw} \quad (2.19)$$

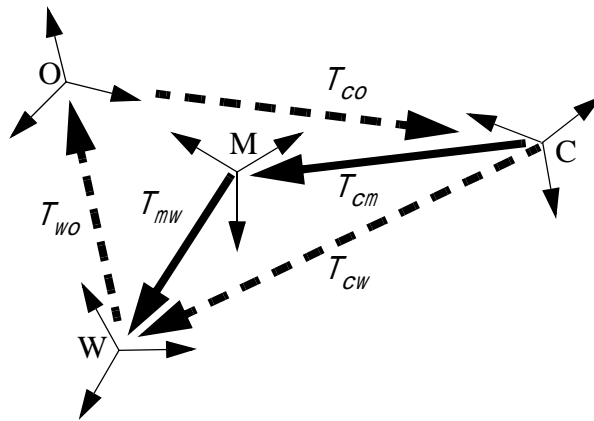


Figure 2.12 Decomposition of the camera-world relation.

Thus, the relation T_{cm} must be evaluated. The 3D positions of the features in the camera coordinate system are estimated with the 2D positions in the images of the three marker features to evaluate the relation T_{cm} . As already said, a stereoscopic registration may be used to compute the 3D position of the features in the camera coordinate system (see section 1.2.4). Those systems highly suffer from miscomputation of feature positions in the frame. Because of the low resolution of the camera frames, binocular systems are therefore unable to perform accurate registration when the features are too far from the cameras. The Figure 2.13 illustrates the effect of an error of one pixel on the 3D position computed for a feature according to the distance. In the example of Figure 2.13, the position of the feature in the left image is errorless. If the correct position of the feature in the right image is the pixel a , the error E_{a-1} computed when the feature position is mismatched by one pixel to the left and the error computed E_{a+1} when the feature position is mismatched by one pixel to the right are relatively negligible since the feature 3D position is near the cameras. But, in the case that the correct position of the feature in the right image is the pixel b , the error E_{b-1} and E_{b+1} become relatively important little by little when the distance between the 3D position of the feature and the cameras increases. Therefore, the registration depth of binocular systems using stereoscopic vision is very limited in depth. A solution to this problem is to increase the baseline b between the two cameras, but this solution is not allowed in the case of augmented reality systems because the cameras must be positioned as close as possible to the position of the user's eyes. Monocular registration may also be used to compute the 3D positions of the features in the camera coordinate system from a single camera image(see section 1.2.3). Monocular systems have an advantage over stereoscopic systems since monocular systems don't have limited registration depth.

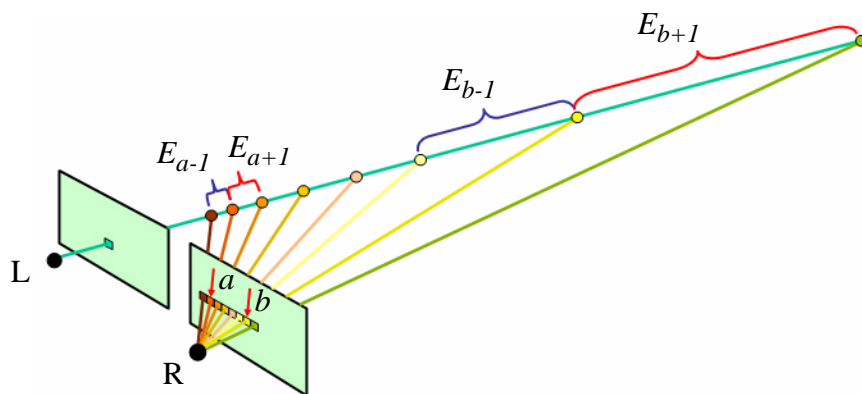


Figure 2.13 Influence of the distance on the 2D positions.

A method which combines monocular registration and stereoscopic registration computations is proposed. The monocular registration computations aim to increase the registration depth of our system. On the other hand, since binocular system has richer

information than monocular system, the stereoscopic registration computations aim to compensate the lack of information caused by the use of three features by selecting the correct solution among the multiple solutions obtained from the monocular registration computations. State et al. have suggested the use of stereoscopic projections to disambiguate the solutions, but they actually used a registration method based on minimization in their system[SHC+96]. In contrast, our registration method uses stereoscopic projections to disambiguate the solutions.

Therefore, the registration method is divided into two steps. Firstly, all estimations of the feature positions are computed using a monocular method by solving the three point space resection problem (section 2.3.2). Secondly, every solution is evaluated using stereoscopic projection and geometric considerations (section 2.3.3 and section 2.3.4).

2.3.2 Three point space resection problem (monocular)

The three point resection problem states that the 3D coordinates of three points (\vec{F}_1 , \vec{F}_2 and \vec{F}_3), constituting the vertices of a known triangle in the 3D space, can be determined from the observed 2D positions (\vec{f}_1 , \vec{f}_2 and \vec{f}_3) of those three points in an image providing that the distances d_{12} , d_{13} and d_{23} between the vertices in the space are known. In other words, the three point resection problem refers to solve the inverse perspective projection from the 2D positions in an image of three features. The Figure 1.8 in section 1.2.3 illustrates the three point resection problem.

The system of equations (equation 1.1, equation 1.2 and equation 1.3) gives the mathematical expression of the three point space resection problem. This system of equations needs to be solved for each camera. The distances d_{12} , d_{13} and d_{23} are known and correspond respectively to the length of the base, the height and the hypotenuse of a triangular marker. Also, the 2D positions \vec{f}_1 , \vec{f}_2 and \vec{f}_3 of the features have been extracted. Therefore, the angles α , β and γ can be computed. Consequently, the system of 3 equations possesses 3 variables: s_1 , s_2 and s_3 . Since the equations of the system are second degree polynomials, multiple solutions are obtained when solving the system. Different techniques may be used to solve the system[HLO+91]. Using Finsterwalder's approach, the computation of the 3D feature positions requires finding the roots of a cubic polynomial and two quadratic polynomials (see Appendix B). Up to twelve solutions may be computed. The Finsterwalder's method is explained with more details in Appendix A. In brief, the solutions for the 3D positions \vec{F}_1 , \vec{F}_2 and \vec{F}_3 of the 2D feature positions \vec{f}_1 , \vec{f}_2 and \vec{f}_3 are obtained by solving the equation 2.20 where $s_2 = |\vec{F}_2| = us_1 = u|\vec{f}_1|$ and $s_3 = |\vec{F}_3| = vs_1 = v|\vec{f}_1|$.

$$s_1^2 = \frac{d_{23}^2}{u^2 + v^2 - 2uv\cos\alpha} = \frac{d_{13}^2}{1 + v^2 - 2uv\cos\beta} = \frac{d_{12}^2}{1 + u^2 - 2uv\cos\gamma} \quad (2.20)$$

As many as 24 solutions are computed for the 3D positions \vec{F}_1 , \vec{F}_2 and \vec{F}_3 of the features (12 solutions by camera). The sets of solutions computed from the left image are noted \mathcal{S}_l and the sets of solutions computed from the right image are noted \mathcal{S}_r . At first, all the solutions \mathcal{S}_l and \mathcal{S}_r containing at least one imaginary component are eliminated. Then, only the solutions located in front of the camera centers of perspective are kept. In other words, the components F_{z_i} of the 3D feature positions \vec{F}_i (where $i = \{1, 2, 3\}$) must be positive values since the observed marker features are situated in front of the user and not behind. Since only real solutions in front of the center of perspectivity are kept, only four solutions generally remains (two solutions by camera). The set of solutions \mathcal{R}_l contains the remaining solutions of the left camera and the set of solutions \mathcal{R}_r contains the remaining solutions of the right camera. When the three 3D feature positions \vec{F}_i have been computed from a left camera image, the 3D feature positions are noted \vec{F}_{l_i} . Similarly, the 3D feature positions \vec{F}_i are noted \vec{F}_{r_i} when the three 3D positions of the features have been computed from a right camera image.

Two approaches are proposed to distinguish the true solution among the remaining solutions of \mathcal{R}_l and \mathcal{R}_r . The two approaches are called the best solution approach and the paired solution approach. The best solution approach is the natural approach based on the choice of the solution giving the best concordance between the observed positions in both frame images. The paired solution approach has been developed in order to decrease the frequency of wrong solution selection.

2.3.3 Evaluation using the best solution approach (binocular)

In the best solution approach, each 3D position \vec{F}_i of a solution is projected in an image to obtain the 2D positions ρ_i of the projection. The equation performing the projection differs if the \vec{F}_i belong to the set \mathcal{R}_l or the set \mathcal{R}_r . The equation 2.21 projects the F_i in the left image to compute the 2D positions ρ_{l_i} of the projection. The equation 2.22 projects the F_i in the right image to compute the 2D positions ρ_{r_i} of the projection. The variable b in equation 2.21 and in equation 2.22 refers to the baseline between the left and the right cameras (see Figure 2.15). The Figure 2.14 illustrates the projection.

The projection error E_p for the 3D positions F_i of a solution is given by the sum of euclidean errors between the projections ρ_i and the feature positions \vec{F}_i . The equation 2.23 computes the projection error E_p where the ρ_{l_i} are the positions in the left image of the projected \vec{F}_i , the ρ_{r_i} are the positions in the right image of the projected \vec{F}_i , the f_{l_i} are the positions in the left image of the marker features and the f_{r_i} are the positions in the right image of the marker features.

$$\left\{ \begin{array}{l} \rho_{l_i}(f_{ix}, f_{iy}) \\ \rho_{r_i} \left(\text{M2Px} \left(\frac{f(F_{ix} - b)}{F_{iz}} \right), \text{M2Py} \left(\frac{fF_{iy}}{F_{iz}} \right) \right) \end{array} \right. \quad (2.21)$$

$$\left\{ \begin{array}{l} \rho_{r_i} \left(\text{M2Px} \left(\frac{f(F_{ix} + b)}{F_{iz}} \right), \text{M2Py} \left(\frac{fF_{iy}}{F_{iz}} \right) \right) \\ \rho_{r_i}(f_{ix}, f_{iy}) \end{array} \right. \quad (2.22)$$

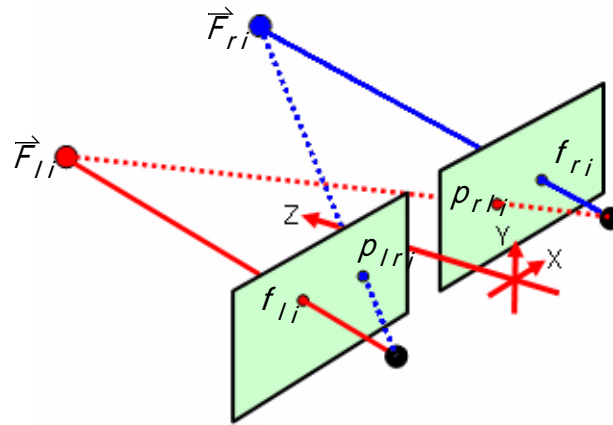


Figure 2.14 Illustration of the projection.

$$E_p = E_l + E_r = \sum_{i=1}^3 |\rho_{li} - f_{li}| + \sum_{i=i}^3 |\rho_{ri} - f_{ri}| \quad (2.23)$$

The position ρ_1, ρ_2 and ρ_3 obtained by projection will perfectly fit the position of the extracted features \vec{P}_1, \vec{P}_2 and \vec{P}_3 in the image used to compute the solution of \vec{F}_1, \vec{F}_2 and \vec{F}_3 . But, in the other image, the positions ρ_1, ρ_2 and ρ_3 obtained by projection will differ from the positions f_1, f_2 and f_3 of the features even for the true solution because of the errors coming from the image processing that extracts the feature positions. In brief, E_l is zero if the \vec{F}_i belong to R_l and E_r is zero if the \vec{F}_i belong to R_r . The hypothesis that the true solution gives the smallest projection error E_p is assumed. Consequently, the solution of R_l or R_r which minimizes the projection error E_p is considered as the true solution of the 3D feature positions \vec{F}_s . The 3D feature positions \vec{F}_s will be used to perform the registration.

Unfortunately, the hypothesis which assumes that the true solution gives the smallest projection error E_p is not always verified. The errors in the 2D positions \vec{F}_i of the features computed by the image processing may result in mistaken the true solution. Therefore, the paired solution approach is proposed to decrease the frequency of wrong solution selection.

2.3.4 Evaluation using the paired solution approach (binocular)

The paired solution approach searches for the best pair of \vec{F}_{li} and \vec{F}_{ri} where the paired solution component \vec{F}_{li} belongs to R_l and the paired solution component \vec{F}_{ri} belongs to R_r . A maximum of 144 paired solutions is possible (the number of pair created with two sets of 12 elements), but the number of possible solutions is less since R_l and R_r contain only the real solutions in front of the center of perspectivity. The set of possible paired solution P is given by the equation 2.24. Based on the set theory, the number of possible paired solutions n_p is then given by equation 2.25. Only four paired solutions are usually dealt with since each of the sets R_l and R_r generally contains two groups of \vec{F}_i . Once more, the projection error E_p is used to disambiguate between the possible paired solutions.

$$P = R_l \times R_r \quad (2.24)$$

$$n_p = \text{card}(R_l) \times \text{card}(R_r) \quad (2.25)$$

$$E_p = \begin{cases} E_{pl} + E_{pr} = E_{ll} + E_{lr} + E_{rl} + E_{rr} \\ 0 + \sum_{i=1}^3 |\rho_{lri} - f_{li}| + \sum_{i=i}^3 |\rho_{lri} - f_{ri}| + 0 \end{cases} \quad (2.26)$$

For a paired solution constituted of a solution \vec{F}_{li} from the left image and a solution \vec{F}_{ri} from the right image, the projection error E_p associated to a paired solution is given by the sum of the E_{pl} given by the E_p of equation 2.23 applied to the \vec{F}_{li} and the E_{pr} given by the E_p of the equation 2.23 applied to the \vec{F}_{ri} . Four projections are possible: the projection of \vec{F}_{li} into the left image plane and into the right image plane, and the projection of \vec{F}_{ri} onto the left image plane and into the right image plane. The projection error E_p of a paired solution is then given by the sum of all projection errors E_{ll} , E_{lr} , E_{rl} and E_{rr} (equation 2.26). When the equation 2.21 and the equation 2.22 are applied on \vec{F}_{li} and \vec{F}_{ri} , four projection positions are obtained: ρ_{lli} , ρ_{lri} , ρ_{rli} and ρ_{rri} . The projection error E_p of a paired solution is given by the projection error E_{lr} of the left image solution \vec{F}_{li} projected onto the right image plane plus the projection error E_{rl} of the right image solution \vec{F}_{ri} projected onto the left image (equation 2.26). The two terms

E_{ll} and E_{rr} are null and are simplified in equation 2.26 because the projection positions ρ_i and the feature positions \vec{F}_i are identical if the $\vec{\rho}_i$ and the \vec{F}_i belong to the same image.

Also, to reduce the possibility of mistaken the true paired solution when an ambiguity arises from the lack of disparity between the feature positions, geometric considerations are used. First, the 3D positions \vec{F}_{li} and \vec{F}_{ri} of the features obtained from the left and the right images are compared. A 3D feature position in the space is unique, therefore the 3D positions \vec{F}_{li} computed from the left image and the 3D position \vec{F}_{ri} computed from the right image must be similar. If the feature positions are too different, the paired solution is automatically rejected. The sum D of the euclidean distances between the left 3D feature positions \vec{F}_{li} and the right 3D feature positions \vec{F}_{ri} is used to test the similarity of the feature positions of a paired solution (equation 2.27). The sum D is compared with a threshold value T_D defined by experimentation. If the sum of the euclidean distances is more than the threshold, the paired solution is said to give the positions of two different features. The paired solution is consequently rejected. Second, the three feature positions \vec{F}_i in the 3D space define a plane. For a solution of \vec{F}_i , the normal vector \vec{N}_ρ of the computed plane is given by equation 2.28. Since solutions from the left and the right images should be identical, the two plane normal vectors computed from the \vec{F}_{li} and the \vec{F}_{ri} of a paired solution can be compared. If the two plane normal vectors $\vec{N}_{\rho l}$ and $\vec{N}_{\rho r}$ associated to the paired solution are really different, the paired solution is automatically rejected. The equation 2.29 is used to determine the angle between the two normals. The paired solution is valid only if the angle φ between the two normals must be inferior to a threshold value T_φ . The value of the threshold is obtained by experimentation.

$$D = \sum_{i=1}^3 |F_{li} - F_{ri}| \quad (2.27)$$

$$\vec{N}_\rho = \frac{(\vec{F}_2 - \vec{F}_1) \times (\vec{F}_3 - \vec{F}_1)}{|(\vec{F}_2 - \vec{F}_1) \times (\vec{F}_3 - \vec{F}_1)|} \quad (2.28)$$

$$\varphi = \arccos\left(\frac{\vec{N}_{\rho l} \cdot \vec{N}_{\rho r}}{|\vec{N}_{\rho l}| |\vec{N}_{\rho r}|}\right) \quad (2.29)$$

The paired solution among the paired solutions of \mathcal{P} that satisfies the geometric considerations and that minimizes the projection error E_ρ is selected as the true paired solution which gives the 3D positions of the marker features for both left and right camera images.

Thus, a paired solution has been identified as the true paired solution, but the registration still needs to be performed using one of the two paired solution components.

Without errors, the left component \vec{F}_{L_i} and the right component \vec{F}_{R_i} of the true paired solution should be identical. So, any of the two may be arbitrarily selected to perform the registration. Unfortunately, the two components are slightly different. A way to perform the registration is to perform the right image registration with the right component \vec{F}_{R_i} and the left image registration with the left component \vec{F}_{L_i} of the true paired solution.

2.3.5 Three point based registration

The initial goal is to retrieve the relation T_{CM} in order to perform the registration. The 3D positions of a marker features (\vec{F}_S , \vec{F}_L or \vec{F}_R) have been computed. The 3D positions of the features give the axes of the marker coordinate system in term of the coordinate system of one camera.

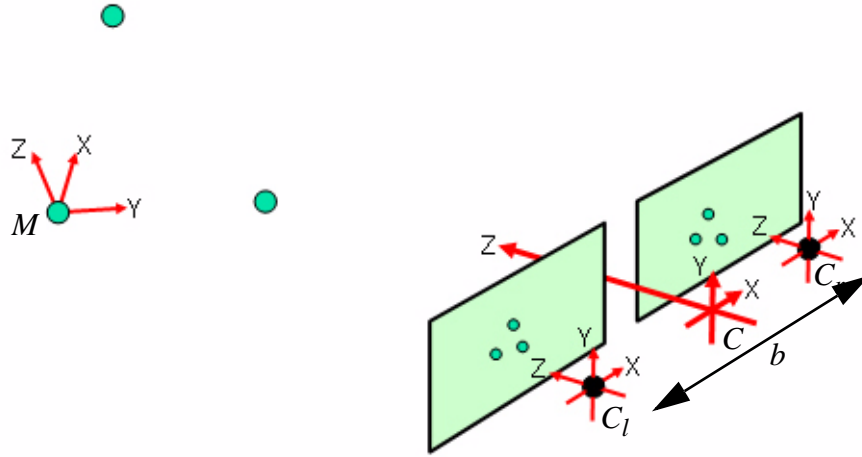


Figure 2.15 Illustration of the camera coordinate system.

Each camera has its own coordinate system (C_l or C_r) centered on the center of perspective with the \vec{Z} axis in the direction of the optical axis of the camera and the \vec{X} axis in the same direction than the baseline vector which is oriented from the left camera to the right camera. Each coordinate system of the cameras is transformed to the global camera coordinate system which corresponds to the user's head coordinate system. The origin of the global camera coordinate system C is placed at the middle point between the two camera centers of perspective. The axis \vec{C}_X is set along the baseline of the cameras and the axis \vec{C}_Z is set to the direction parallel to the optical axes of the cameras. The global camera coordinate system C is illustrated in Figure 2.15.

In terms of the camera coordinate system C , the coordinate system origin of the left camera is $(-\frac{b}{2}, 0, 0)$ and the coordinate system origin of the right camera is $(\frac{b}{2}, 0, 0)$. Therefore, the 3D position \vec{F}_{C_i} of a feature i in the global camera coordinate system C

computed from a feature position \vec{F}_i in the coordinate system of one of the two cameras is given by equation 2.30.

$$\vec{F}_{Ci} = \begin{cases} \vec{F}_i - \frac{b}{2}\vec{C}_X & \text{if } \vec{F}_i \in R_l \\ \vec{F}_i + \frac{b}{2}\vec{C}_X & \text{if } \vec{F}_i \in R_r \end{cases} \quad (2.30)$$

The relation T_{cm} is the transformation from a point coordinate in the marker coordinate system M to the coordinate of the same point in the camera coordinate system C . The marker coordinate system is given by equation 2.18. A transformation between two coordinate systems can be represented as a 4x4 matrix[KIT+00][KOT+00].

Thus, the relation T_{cm} is encoded in a 4x4 matrix. The matrix contains the translation and the rotations to move from the marker coordinate system M to the camera coordinate system C . Therefore, the following equation stands where T_{cm} is the 4x4 matrix, m is the position of a point in the marker coordinate system and c is the equivalent position of the point in the camera coordinate system.

$$c = T_{cm}m \quad (2.31)$$

The matrix T_{cm} is defined from the 3D position of the three features \vec{F}_{Ci} in the camera coordinate space C . The matrix can be decomposed into a rotation matrix T_{Rcm} and a translation vector T_{Tcm} as in equation 2.32. The translation T_{Tcm} is defined as the translation from the camera coordinate origin to the origin of the marker coordinate system given by the 3D position \vec{F}_{C1} of the feature \vec{F}_1 associated to the right angle corner of the marker(equation 2.33). The rotation matrix T_{Rcm} is obtained by concatenating the normalized axes \vec{M}_x , \vec{M}_y and \vec{M}_z of the marker coordinate system (equation 2.34).

$$T_{cm} = \left[\begin{array}{ccc|c} & & & \\ & T_{Rcm} & & T_{Tcm} \\ & \hline 0 & 0 & 0 & 1 \end{array} \right] \quad (2.32)$$

$$T_{Tcm} = \vec{F}_{C1} \quad (2.33)$$

$$T_{Rcm} = \begin{bmatrix} \vec{M}_x & \vec{M}_y & \vec{M}_z \end{bmatrix} = \begin{bmatrix} M_{xx} & M_{yx} & M_{zx} \\ M_{xy} & M_{yy} & M_{zy} \\ M_{xz} & M_{yz} & M_{zz} \end{bmatrix} = \begin{bmatrix} T_{Rcm}[0] & T_{Rcm}[3] & T_{Rcm}[6] \\ T_{Rcm}[1] & T_{Rcm}[4] & T_{Rcm}[7] \\ T_{Rcm}[2] & T_{Rcm}[5] & T_{Rcm}[8] \end{bmatrix} \quad (2.34)$$

To compute the relation T_{cw} between the camera coordinate system and the world coordinate system using equation 2.19, the relation T_{mw} is also needed. The relation T_{mw} gives the transformation between the marker coordinate system and the world coordinate system. When the markers are placed in the scene, this transformation can be measured for each marker.

$$c = T_{co}o \quad (2.35)$$

$$\rho(\rho_x, \rho_y) = \begin{cases} \rho \left(\text{M2Px} \left(\frac{f \left(c_x - \frac{b}{2} \right)}{c_z} \right), \text{M2Py} \left(\frac{f c_y}{c_z} \right) \right) & \text{in } L \\ \rho \left(\text{M2Px} \left(\frac{f \left(c_x + \frac{b}{2} \right)}{c_z} \right), \text{M2Py} \left(\frac{f c_y}{c_z} \right) \right) & \text{in } R \end{cases} \quad (2.36)$$

The relation T_{co} obtained with the equation 2.17 is named the model-view matrix. The model-view matrix gives the translation and the rotations to move from the camera coordinate system C to a virtual object coordinate system O . Therefore, the equation 2.35 stands where T_{co} is the model-view matrix, o is a point of the virtual object in the virtual object coordinate system and c is the equivalent point in the camera coordinate system. Using the equation 2.35, any point o of the virtual object can be localized in the camera coordinate system. Then, the projection $\mathcal{V}(\nu_x, \nu_y)$ into the camera image planes L and R of the equivalent position c in the camera coordinate system of every virtual object point is performed to create the augmented images. The equation 2.36 where f is the focal length of the cameras and b is the base line between the cameras performs the projection of a point $c(c_x, c_y, c_z)$ into an image plane. In equation 2.36, the functions M2Px and M2Py refer to functions that transfer a metric value to a pixel coordinate (x, y) in an image.

However, each relation T_{wo} between a virtual object and the world coordinate system is needed to compute the relation T_{co} . Every relation T_{wo} can be determined beforehand since the wanted position and orientation of the virtual object is measurable in the scene. Therefore, the position \vec{T}_{TWO} and the orientation T_{RWO} of the virtual objects which define the matrix T_{wo} must be determined by the user according to the wanted augmented scene.

2.3.6 Simplification of the registration

The matrix T_{wo} and T_{mw} need to be determined by the user for every wanted virtual object in the scene during the set-up of the system. Determining those matrix is time consuming and needs some extra equipments. Therefore, some simplifications are made in order to facilitate the set-up.

First, instead of having multiple marker coordinate systems relating to a global world coordinate system, each marker coordinate system is taken as a local world coordinate system. That is, the matrix T_{mw} is the identity matrix and the matrix T_{cw} is always equal to T_{cm} . Second, the virtual object coordinate system O is always matched with the marker coordinate system C . Therefore, the matrix T_{co} equals the matrix T_{cm} with the translation vector in the opposite direction.

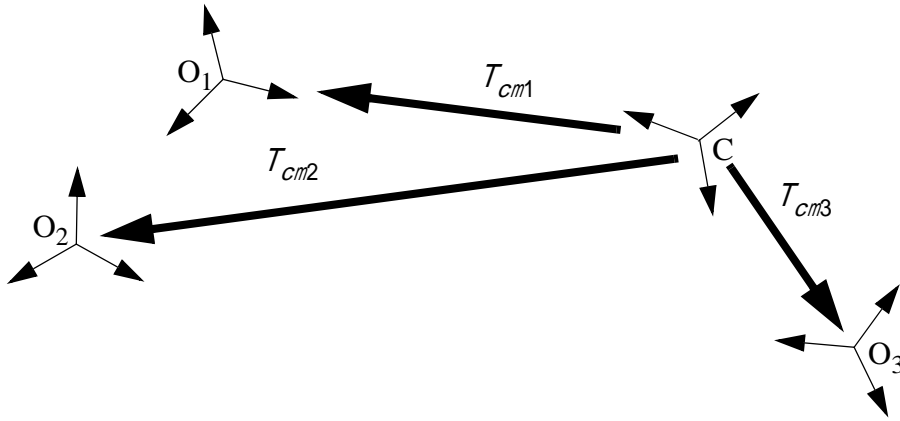


Figure 2.16 Simplified relations between the coordinate systems.

All in all, the three coordinate systems C , O and W are perfectly superposed with those two simplifications. As a result, equation 2.37 and equation 2.38 stand. The model-view matrix T_{co} needed to complete the registration is now given by a simple function of the transformation T_{cm} . The resulting relations between the camera coordinate system and the multiple virtual object coordinate systems are shown in Figure 2.16.

$$T_{wm} = T_{wo} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.37)$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \times T_{cm} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \times T_{cw} = T_{co} \quad (2.38)$$

2.4 Correction method

As already mentioned, misestimating the feature positions in the camera images is the first source of registration error in vision-based augmented reality systems. Performing a dynamic correction is an interesting approach to optimize a registration [BN95]. In order to increase the robustness and the accuracy of the registration, a position correction method is proposed. Therefore, a correction is applied on the 2D feature positions because we aim to correct the main source of the registration error instead of optimizing the 3D positions of the features. The correction method starts with the results obtained from the paired solution approach. Instead of realizing the registration of the right camera image using the three feature positions \vec{F}_{Ri} and the registration of the left camera using the three feature positions \vec{F}_{Li} , the correction method refines the 3D positions \vec{F}_{Li} and \vec{F}_{Ri} of a feature i , where $i \in \{1, 2, 3\}$, by applying a correction to the feature point positions. All in all, any registration methods producing a different registration for each stereo-paired camera can be improve using the proposed correction method, in particular, monocular registration using three or four feature points.

With equation 2.30, the feature positions \vec{F}_{Li} and \vec{F}_{Ri} , expressed in terms of the coordinate system of one of the cameras, are expressed into the equivalent positions \vec{F}_{Cli} and \vec{F}_{Cri} in terms of the camera coordinate system \mathcal{C} .

Theoretically, the 3D positions \vec{F}_{Cli} and \vec{F}_{Cri} of a feature i must be identical. Figure 2.17 shows the theoretical case when the detected feature positions \vec{F}_{Cli} and \vec{F}_{Cri} are errorless. Because of the detection errors of the vision-based tracking, the two solutions are expected to be different. Figure 2.18 illustrates the general case. The idea behind the correction method is to modify the positions \vec{F}_{Cli} and \vec{F}_{Cri} of a feature i to diminish the difference $|\vec{F}_{Cli} - \vec{F}_{Cri}|$. When the errors ε associated to the detection of the positions diminish, both left and right components \vec{F}_{Cli} and \vec{F}_{Cri} of the paired solutions should tend to the true positions of the three feature positions. In other words, the difference between the two positions of a feature decreases when the position errors ε diminish (equation 2.39). The hypothesis made is that the inverse implication is also true. That is, diminishing the difference $|\vec{F}_{Cli} - \vec{F}_{Cri}|$ tends to decrease the detection errors ε (equation 2.40).

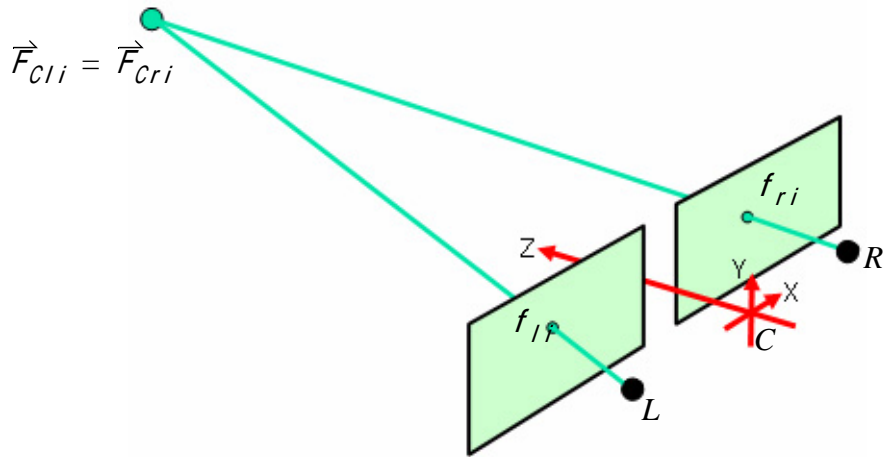


Figure 2.17 Theoretical case: the 3D positions are errorless.

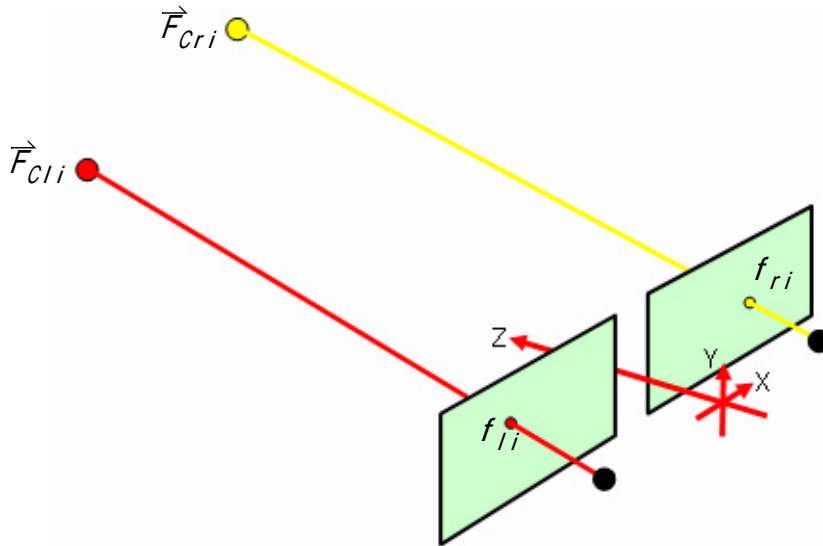


Figure 2.18 General case: the 3D positions differ.

$$\varepsilon \rightarrow 0 \Rightarrow |\vec{F}_{cli} - \vec{F}_{cri}| \rightarrow 0 \quad (2.39)$$

$$\varepsilon \rightarrow 0 \Leftarrow |\vec{F}_{cli} - \vec{F}_{cri}| \rightarrow 0 \quad (2.40)$$

The 2D position $f_i(x, y)$ in one camera image of a feature i is modified according to the projected feature position p_i computed with the equation 2.41 from the 3D position of the feature i given by the other image. The corrected 2D position n_{li} of a feature i in the left image is given by equation 2.43 where p_{li} is the projection of the position \vec{F}_{cri} of the feature i in the left image. Similarly, the corrected 2D position n_{ri} of a feature i

in the right image is given by equation 2.42 where p_{ri} is the projection of the position \vec{F}_{cli} of the feature i in the right image.

$$p_i(p_x, p_y) = \begin{cases} p_{li} \left(\text{M2Px} \left(\frac{f\vec{F}_{crix}}{\vec{F}_{criz}} \right), \text{M2Py} \left(\frac{f\vec{F}_{criy}}{\vec{F}_{criz}} \right) \right) & \vec{F}_{cri} \text{ projected in } L \\ p_{ri} \left(\text{M2Px} \left(\frac{f\vec{F}_{clix}}{\vec{F}_{cliz}} \right), \text{M2Py} \left(\frac{f\vec{F}_{cliy}}{\vec{F}_{cliz}} \right) \right) & \vec{F}_{cli} \text{ projected in } R \end{cases} \quad (2.41)$$

$$n_{li} = \begin{pmatrix} n_{lx} \\ n_{ly} \end{pmatrix} = \begin{pmatrix} x_l + \mu_i(p_{lix} - x_l) \\ y_l + \mu_i(p_{liy} - y_l) \end{pmatrix} \quad (2.42)$$

$$n_{ri} = \begin{pmatrix} n_{rx} \\ n_{ry} \end{pmatrix} = \begin{pmatrix} x_r + \tau_i(p_{rix} - x_r) \\ y_r + \tau_i(p_{riy} - y_r) \end{pmatrix} \quad (2.43)$$

The correction factors μ and τ take a value between 0 and 1. The correction factors can be used to characterize the confidence in the 2D positions of the features computed by the system. The factor value may be reduced when a feature position in the image seems errorless and may be increased when a feature position in the image seems erroneous. Furthermore, the correction factors control the importance of the correction. Consequently, the correction applied can be controlled for each registration using variable correction factors calculated either from image analyses or geometrical computations. This control is an advantage of the proposed correction method over other optimization methods such as the least square minimization.

To update the registration according to the corrected 2D positions n_{ri} and n_{li} of the features, Finsterwalder's method is applied again. Using the paired solution approach, the corrected 3D positions of the features \vec{F}'_{cli} and \vec{F}'_{cri} are computed from the corrected 2D positions. Then, a new model-view matrix T_{co} is defined from the positions \vec{F}'_{cli} and \vec{F}'_{cri} of the features. Figure 2.19 illustrates the correction method applied to the feature i illustrated in Figure 2.18. The position of f_{ri} and f_{li} are corrected in order to obtain corrected 3D position \vec{F}'_{cli} and \vec{F}'_{cri} .

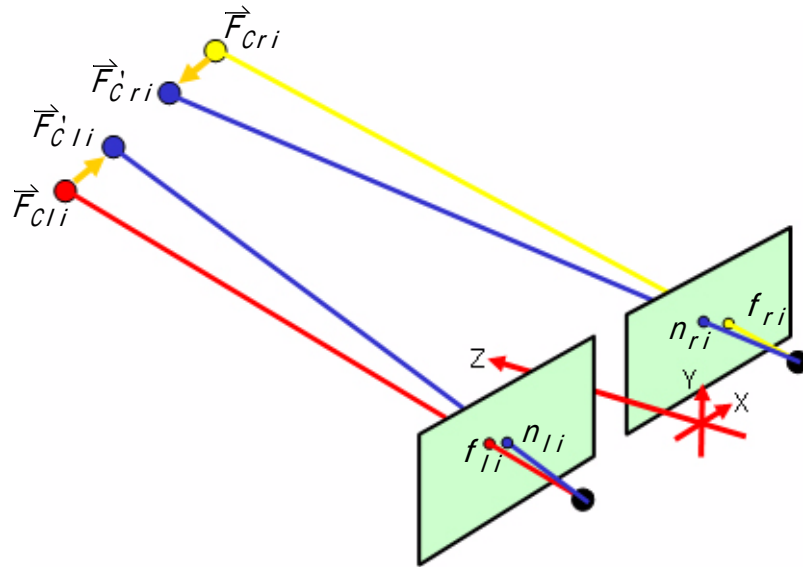


Figure 2.19 Illustration of the correction method.

The correction method is based on equation 2.42 and equation 2.43. These equations are called the correction equations because they dictate the rules which lead to the corrected 3D positions of the features. Different equations may be proposed for the correction equations as long as the correction equations verify the hypothesis defined in equation 2.40. If the equation 2.44 is proven to be always valid, the correction equations verify the hypothesis and are able to perform a valid correction.

$$|\vec{F}'_{C'li} - \vec{F}'_{C'ri}| < |\vec{F}_{C'li} - \vec{F}_{C'ri}| \quad (2.44)$$

When $\mu_j + \tau_j \leq 1$ in equation 2.40, the differences $|\vec{F}'_{C'li} - \vec{F}'_{C'ri}|$ have necessarily decreased compared to the differences $|\vec{F}_{C'li} - \vec{F}_{C'ri}|$ computed with the original positions. If $\mu_j + \tau_j > 1$, the correction are not guaranteed to converge. Thus, the equation 2.42 and the equation 2.43 are valid correction equations only when $\mu_j + \tau_j \leq 1$.

In summary, from the 2D positions of three features in left and right images, Finsterwalder's method is used to estimate the 3D feature positions. Among the estimated solutions, a paired solution constituted of a solution for each image is selected. Using this paired solution, a correction of the 2D feature positions of both images is performed. Once the correction is applied, the corrected 2D positions of the features are used to compute with the Finsterwalder's method a corrected paired solution which is closer to the true 3D positions of the features.

Still, the differences $|\vec{F}_{C_{li}}^i - \vec{F}_{C_{ri}}^i|$ may be significant. However, the correction procedure can be repeatedly applied until the differences between the $\vec{F}_{C_{li}}^t$ and $\vec{F}_{C_{ri}}^t$ at iteration t is less than a threshold. When the differences $|\vec{F}_{C_{li}}^t - \vec{F}_{C_{ri}}^t|$ is less than the threshold, the positions $\vec{F}_{C_{li}}^t$ and $\vec{F}_{C_{ri}}^t$ are considered identical. Therefore, any of the two $\vec{F}_{C_{li}}^t$ or $\vec{F}_{C_{ri}}^t$ can be associated to the true positions of the features. As a result, the feature positions in both frames have been successfully corrected. Finally, the registration is performed using the 3D feature positions. The 3D feature positions are given either by the solutions $\vec{F}_{C_{li}}^t$ and $\vec{F}_{C_{ri}}^t$ computed with Finsterwalder's method or by stereoscopic vision computed using the corrected feature positions n_{ri}^t and n_{lj}^t .

2.4.1 Correction by true paired progression

During each iteration of the correction method, the true paired solution is identified among the possible solutions given by Finsterwalder's method. Then, the correction process restarts to correct the feature positions based on the identified true paired solution. This progression method is called the true paired progression because the true paired solution identified at each iteration is used to continue the correction process. The Figure 2.20 illustrates the correction by true paired progression. First, at each iteration, the invalid solutions are rejected. Then, the true paired solution is selected among the valid paired solutions. Finally, the correction restarts with this selected true paired solution until the final true paired solution that has identical components is obtained.

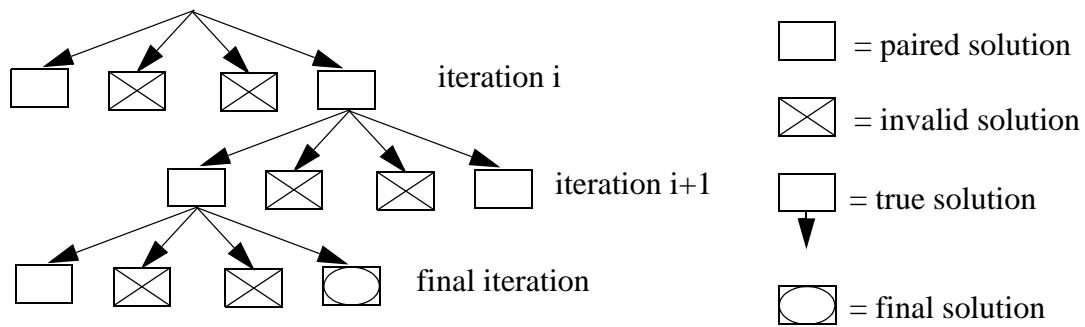


Figure 2.20 Illustration of the correction by true paired progression.

However, this progression is affected by any wrong identification of the true paired solution at any iteration. If a wrong paired solution is selected instead of the true paired solution at an iteration, the correction process continues and may at last converge to a wrong final solution.

2.4.2 Adjustment of the correction factors

The correction factors μ and τ are used to control the correction. This control aims to improve the quality of the registration by reducing the effect of error detection. Different considerations can be used to determine the correction factors. Independently of the type and the number of considerations used to evaluate the factors, two principal evaluations must be computed to allow a relative complete control of the correction.

First, a number of visual features extracted from the images are used to perform the registration. The confidence in the feature positions is evaluated in order to estimate the level of the correction associated with each feature. Therefore, an optimal correction that mainly corrects the positions of features associated with considerable error components and that leaves almost unchanged the positions considered accurate can be defined.

Second, two 2D positions are obtained for each visual feature: one from the left camera image and one from the right camera image. The two positions of a feature must be evaluated in order to identify if the correction should be stronger in the left image or in the right image.

As a result, the correction factors are determined from two measures : one measure m_e indicating which feature asks for a stronger correction and one measure m_j indicating if this correction should be stronger in the left image or in the right image. A correction factor is then given by $\Upsilon(m_e, m_j)$, a function of the measures m_e and m_j . The function used is a simple multiplication of the two measures m_{ej} and m_{jj} for each feature j .

Evaluation of the measure m_e

A stereoscopic consideration is used to evaluate the different variables m_{ek} where k varies between 1 and the number of features used to perform the registration. If the 3D positions of a same feature is known in the left and in the right camera coordinate systems, the value of the angles θ_l and θ_r (see Figure 1.10a) can be experimentally retrieved. In the case of a parallel axis setup, the retrieved angles will be zero.

To evaluate the values of the angles $\theta_l = \theta/2$ and $\theta_r = \theta/2$, the equation 2.45 is solved using an iterative process. In equation 2.45, θ is the calculated variable, \vec{F}_l is the 3D position of a feature in the left camera coordinate system, \vec{F}_r is the 3D position of the same feature in the right camera coordinate system and b is the baseline between the two cameras. The equation says that an angle θ that assures a correspondence between the x components of each 3D position of the feature exists. In other words, we are looking for

the angle θ of the configuration of the HMD when the 3D positions of the feature is considered errorless.

$$0 = F_{I_x}(F_{r_z}\cos(-\theta) - F_{r_x}\cos(-\theta) - B\sin(-\theta/2)) - F_{I_z}(F_{r_x}\cos(-\theta) - F_{r_z}\cos(-\theta) - B\cos(-\theta/2)) \quad (2.45)$$

Since the angle θ_c , given by the internal configuration of the HMD, is known, the computed angle θ_j for a feature j can be compared with θ_c . More the θ_j is similar to θ_c , more the error associated to the feature j is considered negligible. Therefore, the measure m_{ej} is given by the equation 2.46.

$$m_{ej} = \frac{(\theta_c - \theta_j)^2}{j_{max} \sum_{k=1} (\theta_c - \theta_k)^2} \quad (2.46)$$

Evaluation of the measure m_j

In order to evaluate whether the left or the right positions of a feature requires a more important correction, a photometric consideration is used. If a 2D position is correctly extracted from the image, the photometric contents of the image around the detected feature should correspond to a defined template given by the theoretical representation of the feature. Consequently, the quantification of the similarity between the theoretical template and the observed photometric contents is used to evaluate the factors m_{j_l} and m_{j_r} associated to a feature.

In the specific case where the extracted features are the corners of a determined polygon, a measure of variation in intensity is suggested. The variation in intensity is calculated between the photometric value associated to the 2D position F of a corner and the photometric value to a neighbor pixel located in a significant direction. The direction \vec{D} is chosen in order to maximize the variation in intensity $\Delta_j = Y(F, \vec{D})$ when the 2D position perfectly corresponds to a corner of the polygon. The direction of the normal vector of the polygon edges estimated at each feature position is used to calculate a direction \vec{D} for each feature j .

Then, the factors m_{j_l} and m_{j_r} are given by equation 2.47 where Δ_{j_l} is the variation in intensity for the feature j in the left image and Δ_{j_r} is the variation in intensity for the feature j in the right image. If Δ_{j_l} is bigger than Δ_{j_r} , m_{j_r} will be greater than m_{j_l} in order to perform a more significant correction on the position in the right image.

$$m_{i_l} = \frac{\Delta_{k_r}}{\Delta_{k_l} + \Delta_{k_r}}, \quad m_{i_r} = \frac{\Delta_{k_l}}{\Delta_{k_l} + \Delta_{k_r}} \quad (2.47)$$

2.5 Conclusion

In this chapter, a marker extraction strategy, a registration method and a correction method have been developed to fulfill the requirements specified by the goals enumerated in section 1.4. In the next chapter, the different methods are evaluated.

3. Experiment

In this chapter, the results obtained with the proposed system are presented. In section 3.1, the setup of the developed system is explained. Then, in section 3.2, a quantitative evaluation method is given to evaluate a registration stability. The developed system should fulfill all the goals described previously in section 1.4. Since the system uses only three features to perform the registration, the minimum number of features motivation is already verified. However, the system need to be evaluated to verify if the other goals are fulfilled. Therefore, stability and timing results observed for the developed system are presented from section 3.3 to section 3.6. Finally, section 3.7 compares our proposed system with a widely used development tool called the ARToolKit.

3.1 System description

The developed system runs on a 800MHz SGI PC and uses Canon's video see-through head mounted device. The two cameras of the head mounted device provide a stereoscopic view of the scene and the two monitors of the head mounted device are used to show the augmented world to the user. Figure 3.1 illustrates schematically the Canon's head mounted device. Figure 3.2 shows the Canon's head mounted device with the two cameras located in the front and the two monitors located in the back of the head mounted device. The focal lengths of the cameras are $f = 0.005$ m and the baseline between the camera is $b = 0.065$. The toed-in angles measured for the Canon's head mounted device are $\theta_l = 1.05^\circ$ and $\theta_r = -1.05^\circ$ (see section 1.2.5).

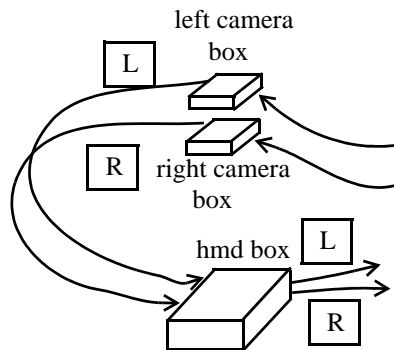
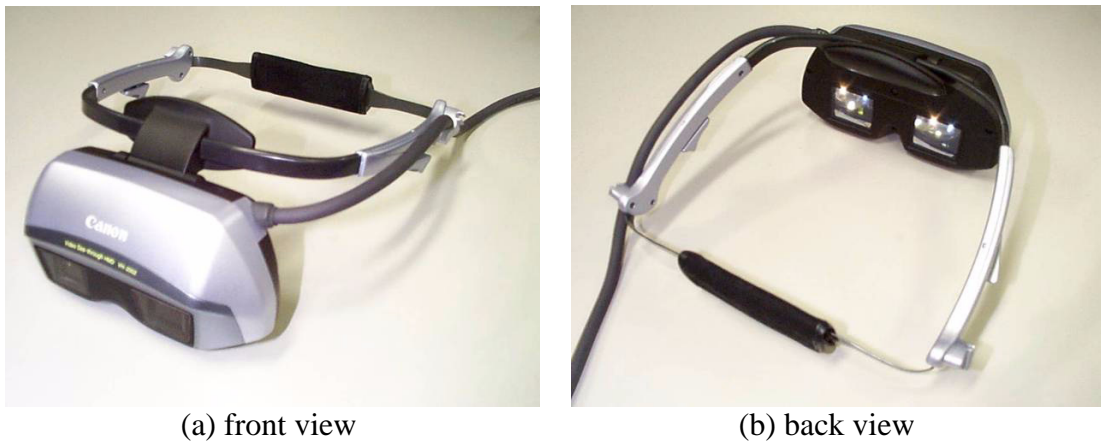


Figure 3.1 Schema of the Canon's HMD.

The configuration of the head mounted device illustrated in Figure 3.1 needs to be modified to realize an augmented reality system. The link between the camera boxes and the HMD box is eliminated and replaced by different devices used to transform the video sequences of the camera into augmented video sequences. Since only one video capture card is used, both video sequences ("L" and "R") are first merged into a single video

sequence (“LR”) using a composite - field sequential converter (model FC-55 made by Kastam). The converter divides by two the height of the input frames and merges the two resized frames into a single frame with the same height of the two original frames. Then, the converted frames (“LR”) are grabbed by the PC using a video capture card. The PC splits the frames into the left and right frames and adds the virtual objects to both frames. The augmented frames (“aL” and “aR”) are shown on the screen. The VGA output is also sent to scan converters (model DSC05d-HR Type 05 made by Digital Arts). The scan converters create the left and right augmented sequences by cropping the VGA signal in order to extract the augmented images (“aL” and “aR”) from every monitor image. Finally, the outputs of the converters are input into the HMD box. The HMD box will transmit the augmented sequences to the head mounted device monitors. Figure 3.3 illustrates the configuration of our augmented reality system.



(a) front view (b) back view

Figure 3.2 Pictures of the Canon’s HMD.

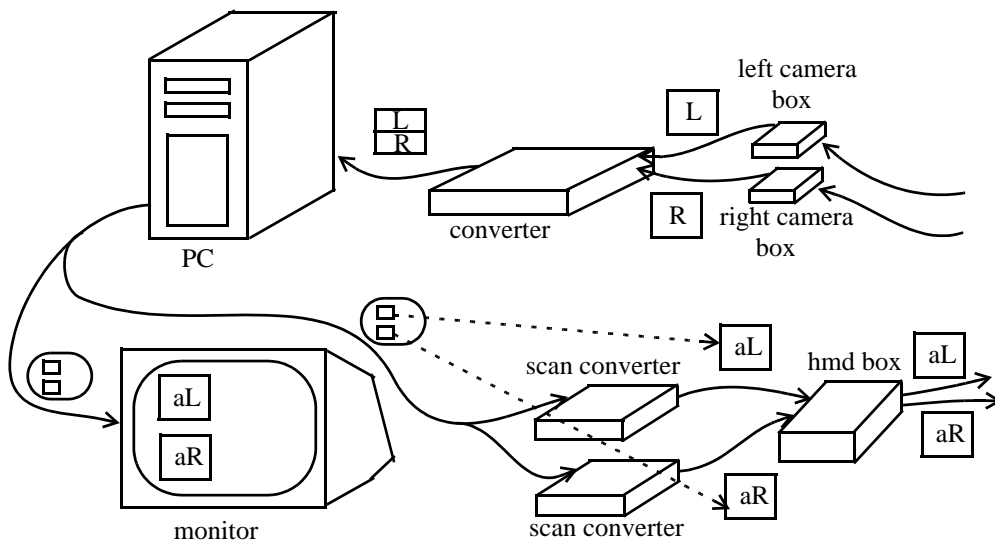


Figure 3.3 Configuration of our augmented reality system.

As already mentioned, the system is wanted to run in real-time on an affordable PC computer. Therefore, a 800MHz SGI PC computer is used instead of a more powerful computer such as a SGI Onyx 3000. The hardware specification of the SGI PC is equivalent to typical hardware specifications retrieved in a home personal computer. Thus, the system could be used with personal application at a relatively low cost.

The results presented in the following sections have been computed with $h_u = 5$ pixels between two adjacent neighborhood pixels of the hexagonal pattern. Consequently, the density parameter H of the hexagonal pattern is also $H = 5$ which gives a total number of 7084 processed pixels for a region of 700 pixels by 200 pixels centered on every camera frame of 768 pixels by 240 pixels. Also, the length L_N of a normal \vec{N}_{uv} is set to 15 pixels ($3 \times h_u$). The threshold value T_D is fixed to 0.25m and the threshold value T_ϕ , to 30° . Each colored region of a triangular marker is identified from 5 pixels ($I = 5$). The correction factor μ and τ have been set to 0.5. The conversion functions are $M2Px(x) = 165000x$ and $M2Px(y) = 140000y$.

3.2 Measures to evaluate the registration quality

In order to evaluate different registration methods, the quality of a registration must be quantified. The accuracy of a registration method is difficult to measure because the registration performed must be compared with the expected theoretical registration. Since the theoretical registration is difficult to obtain, a new way to quantify different registration methods is proposed. The evaluation is based on the stability of the registration instead of comparing the registration. In most cases, a difference between the theoretical registration and the performed registration is acceptable as long as the virtual object position and orientation are stable. Therefore, a measure of stability can give a good idea of the robustness of a registration. The main advantage of the proposed evaluation is the simplicity to implement the evaluation in an AR system.

In the developed system, the registration is performed with a model-view matrix retrieved by the system from the 3D positions of the marker feature points. Also, in a video sequence, the performed registrations must be static when the user's viewpoint is fixed. Therefore, the model-view matrix must not change in order to draw the virtual object at the same position and with the same orientation in every frame of the sequence. Consequently, the stability of a registration method can be characterized by the amount of fluctuations in the model-view matrix. The level of stability of a registration method represents the quality of the performed registration. When the fluctuations are weak, the quality of the registration performed with a registration method is good. However, the quality of the registration is poor if the fluctuations are strong.

Since the model-view matrix T_{CO} , like any transformation matrix, can be divided into an orientation component T_R and in a translation component \vec{T}_T (see equation 2.32), two stability values can be computed: stability in orientation and stability in position. The degree of stability associated with the orientation of an augmented virtual object is given by the fluctuation in orientation S_o . To compute a fluctuation in orientation S_o , the position of a virtual point s_t in a coordinate system created from the elements of the orientation matrix T_{Rt} at the current frame t (equation 3.1) is compared to the virtual point s_{t-1} in a coordinate system created from the elements of the orientation matrix T_{Rt-1} of the previous frame $t-1$. The Figure 3.4 illustrates the positions of the virtual point s_t in the coordinate system of axes \vec{X} , \vec{Y} and \vec{Z} . The distance between the two virtual points in the camera coordinate system \mathcal{C} is transformed into an angle value (in degree) that gives the fluctuation in orientation S_o between two successive frames (equation 3.2). Less the fluctuation in orientation S_o is, more the registration is stable. Figure 3.5 shows the geometric representation of equation 3.2. An average value of fluctuation in orientation is obtained by averaging the fluctuations in orientation S_o computed from a video sequence.

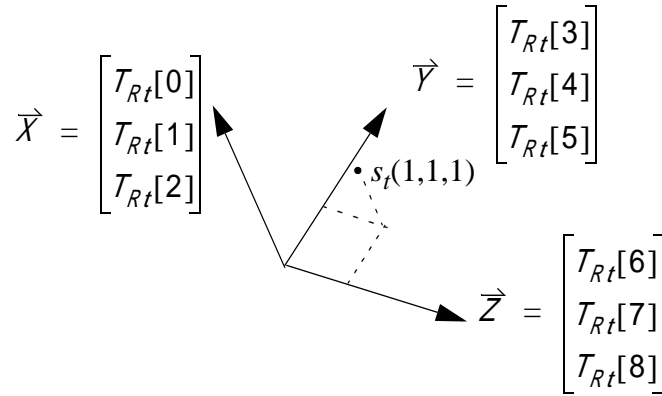


Figure 3.4 Illustration of the virtual point position.

$$s_t = \begin{bmatrix} s_{tx} \\ s_{ty} \\ s_{tz} \end{bmatrix} = \begin{bmatrix} T_{Rt}[0] + T_{Rt}[3] + T_{Rt}[6] \\ T_{Rt}[1] + T_{Rt}[4] + T_{Rt}[7] \\ T_{Rt}[2] + T_{Rt}[5] + T_{Rt}[8] \end{bmatrix} \quad (3.1)$$

$$S_o = \frac{180}{\pi} \arccos \left(\frac{6 - |s_t - s_{t-1}|^2}{6} \right) \quad (3.2)$$

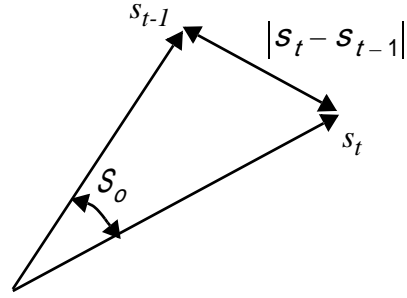


Figure 3.5 Distance between two virtual points.

Similarly, an average value of fluctuation in position is obtained by averaging the fluctuations in position \mathcal{S}_p computed from the same video sequence. The fluctuation in position between two successive frames is given by the length of the vector associated to the difference of two consecutive translation vectors $\vec{T}_{T_{t-1}}$ and \vec{T}_{T_t} (equation 3.3). Like for the fluctuation in orientation, less the fluctuation in position \mathcal{S}_p is, more the registration is stable in position.

$$\mathcal{S}_p = \sqrt[3]{\sum_{k=1} (\vec{T}_{T_t}[k] - \vec{T}_{T_{t-1}}[k])^2} \quad (3.3)$$

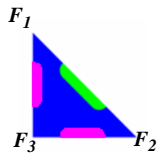
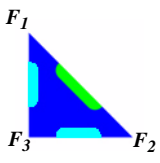
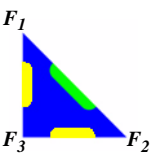
Also, the coherence between the left and the right registration can be computed. The coherence in position is giving by the difference between the translation vector \vec{T}_{T_l} of the left model view matrix and the translation vector \vec{T}_{T_r} of the right model view matrix (equation 3.4). Similarly, the coherence in orientation given by equation 3.5 is given by the angle between the vectors defined by the virtual point s_l computed from the left model view orientation matrix T_{R_l} and the virtual point s_r computed from the right model view orientation matrix T_{R_r} .

$$K_p = \sqrt[3]{\sum_{k=1} (\vec{T}_{T_l}[k] - \vec{T}_{T_r}[k])^2} \quad (3.4)$$

$$K_o = \frac{180}{\pi} \arccos\left(\frac{6 - |s_l - s_r|^2}{6}\right) \quad (3.5)$$

Three different sizes of markers have been used to study the stability of the different methods. The Table 3.1 shows the dimension of the three markers.

Table 3.1: Dimension of the small, middle and large size markers

small size (S size)	middle size (M size)	large size (L size)
		
$d_{12} = 0.070\text{m}$	$d_{12} = 0.161\text{m}$	$d_{12} = 0.322\text{m}$
$d_{13} = 0.099\text{m}$	$d_{13} = 0.227\text{m}$	$d_{13} = 0.455\text{m}$
$d_{23} = 0.070\text{m}$	$d_{23} = 0.161\text{m}$	$d_{23} = 0.322\text{m}$

3.3 Evaluation of the robustness to fast user's motion

In order to evaluate the robustness to fast user's motion, the proposed detection method using hexagonal pattern (Hexagonal) described in section 2.2 is compared to the usual tracking strategy found in literature (Tracking) and presented in section 1.2.2. These two methods of feature extraction have been integrated in a prototype system in order to compare the detection results of both methods during the extraction of a blue triangular marker.

First, the average processing time needed to extract a triangular marker is computed for each method. To compute the average processing time, nine sequences of 450 frames are processed by each system. Each sequence shows the small size triangular marker placed at a different distance from the cameras.

The processing time computed for the proposed method is stable since the method is almost not influenced by the distance or the size of the marker in the images. The extraction of a marker in one frame takes about 0.03 seconds. In stereoscopic systems, two frames need to be processed at each iteration, so the maximum frame rate reachable for the developed system when using our SGI PC is about 16.67 frames by second. The average timing results for one frame are shown in Figure 3.6.

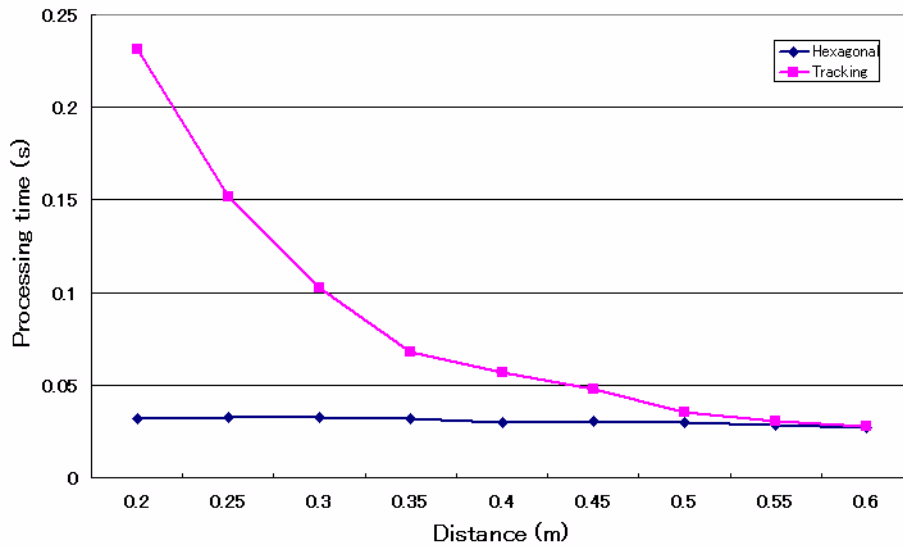


Figure 3.6 Average processing time.

On the other hand, the tracking strategy is influenced by the size of the processing windows. More the marker is near the camera, more the marker is big in the image and more the surface of the processing windows is enlarged to cover the marker. This fact is clearly shown in Figure 3.6. The processing time obtained with the tracking strategy decreases progressively when the distance between the marker and the camera increases. During a motion, the frame rate obtained with the tracking method can change drastically from 18.1 frames/sec at 0.60cm to 2.16 frames/sec at 0.20cm.

Another measure used to compare the two detection methods is the number of frames where the marker has been successfully extracted. Three sequences of 450 frames are processed and the number of successful extraction of the markers are counted for both the tracking strategy of section 1.2.2 and the proposed method using hexagonal pattern of section 2.2. The small size marker is placed at a distance of about 0.40m from the cameras for the three sequences, but the speed of the head motion executed by the user is different for each of the three sequences. The second sequence mainly contains head motion at natural speed. In contrast, the first and the third sequences respectively contain relatively slow motion and relatively fast motion of the user's head. Figure 3.7 presents the percentage of successful extractions for each sequence and each method.

Both methods successfully extract the marker in every frame of the slow motion and normal motion sequences. But, the number of successful extraction decreases significantly in the fast motion sequence when the tracking method is used. In contrast,

the proposed method using hexagonal pattern still succeeds to extract the marker in every frame. Therefore, the proposed method is proven to be robust to fast user's motion.

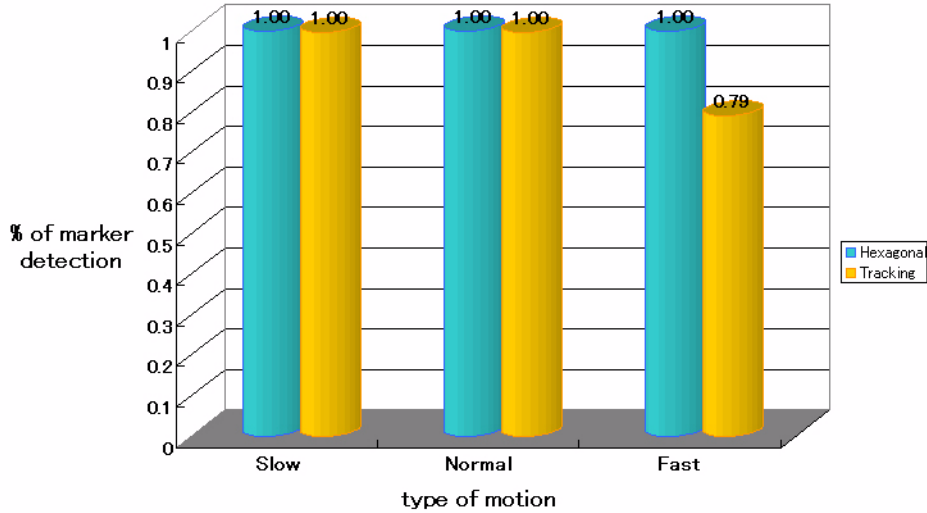


Figure 3.7 Percentage of successful marker extractions (1).

The frame rate obtained with the three sequences may also be compared for both methods. Table 3.2 details the average time spent to process one frame with both methods. The frames processed by the tracking method are divided in two groups: the frames where the full frame is processed and the frames where only the searching zone is processed. Table 3.2 gives the number of each frame and the average time spent to process one frame of each type. An average frame rate is computed from those values. In contrast, the proposed method using hexagonal pattern always processes the entire frame.

Table 3.2: Average frame rates of the marker extraction for one frame

type of motion	Tracking					Hexagonal		
	searching zones		entire frames		average frame rate	entire frames		average frame rate
	number	average time	number	average time		number	average time	
slow	449	0.0356s	1	0.141	27.9 f/s	450	0.0195s	51.3 f/s
normal	449	0.0363s	1	0.132	27.4 f/s	450	0.0199s	50.3 f/s
fast	356	0.0299s	94	0.136	19.2 f/s	450	0.0197s	50.8 f/s

The average frame rates observed for the proposed method are constant and lower than the average frame rates observed for the tracking method. Furthermore, the frame rate observed for the tracking method is influenced by the motion speed because the probability that a tracking failure occurs increases with the speed of the motion. When a tracking failure occurs, the entire frame must be processed to retrieve the lost position of the marker. Therefore, the method spends more time to extract the marker after a tracking failure. Since the number of entire frames processed increases significantly in the fast motion sequence, the average frame rate decreases for the fast motion sequence compared to the average frame rate observed for the two other sequences.

All in all, the proposed method using hexagonal pattern gives better results compared to the standard tracking method. First, the proposed method is robust to fast user's motion. Second, the proposed method spends less time to extract the marker. Third, the frame rate observed with the proposed method is constant. Therefore, the proposed method is better than the tracking method. However, the proposed method can only be used when the extraction of the marker is color based. Therefore, systems using only template matching are unable to use the proposed extraction strategy.

3.4 Evaluation of the registration method

In order to evaluate our registration method, the registration is performed with two different methods. The first method is the standard stereoscopic registration method already found in the literature [KIT+00][KFT+00a][KFT+00b]. The second method is the proposed registration method presented in section 2.3. Results of both methods are compared to evaluate the improvement associated to the proposed registration method. Figure 3.8 gives the block diagram of the two registration methods: the proposed method without correction ("Xnoc") and the stereoscopic method without correction ("Snoc").

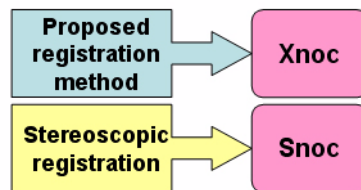


Figure 3.8 Block diagram of the registration methods.

3.4.1 Effect of the distance

The effect of the distance on the registration stability is measured for the three markers of Table 3.1. Therefore, the fluctuations in position and in orientation are computed for different distance values between the marker and the cameras. Then, the stability of the

methods are studied by comparing the fluctuations obtained with each registration method. The fluctuation data have been computed with an angle of 45° between the marker plane normal \vec{N}_m and the axis \vec{Z} of the camera coordinate system (see Figure 3.19).

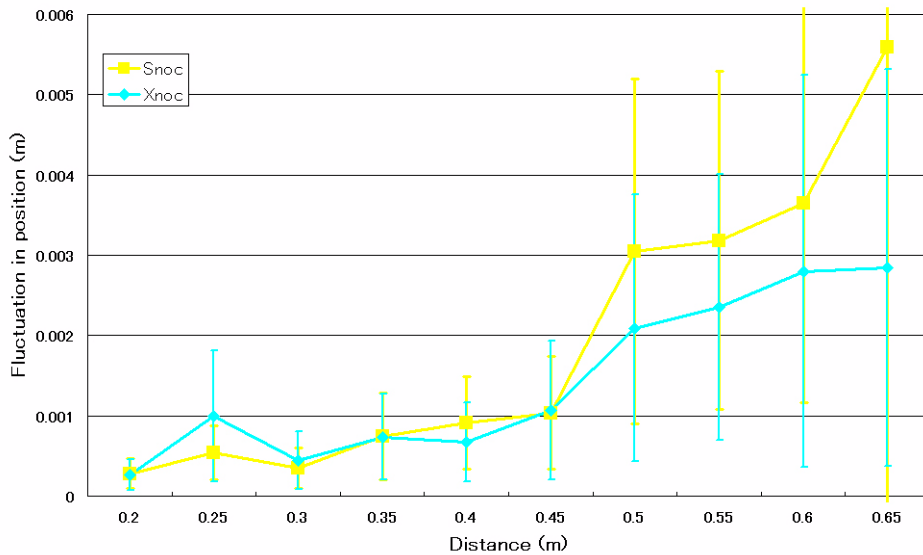


Figure 3.9 Fluctuation in position without correction (marker S).

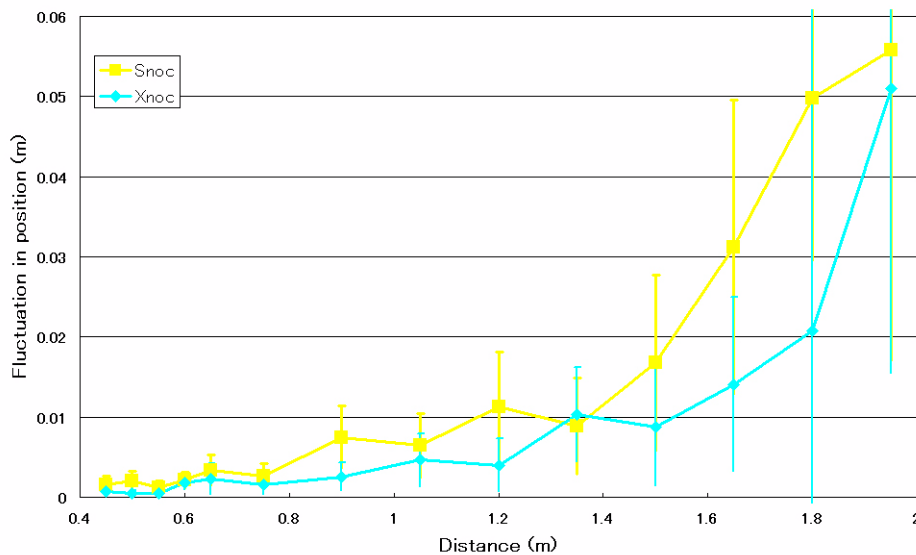


Figure 3.10 Fluctuation in position without correction (marker M).

Figure 3.9, Figure 3.10 and Figure 3.11 give the average fluctuations in position obtained with the small size (S), middle size (M) and large size (L) markers, respectively. In the figures, the stereoscopic registration method is noted “Snoc” and the proposed registration method is noted “Xnoc”.

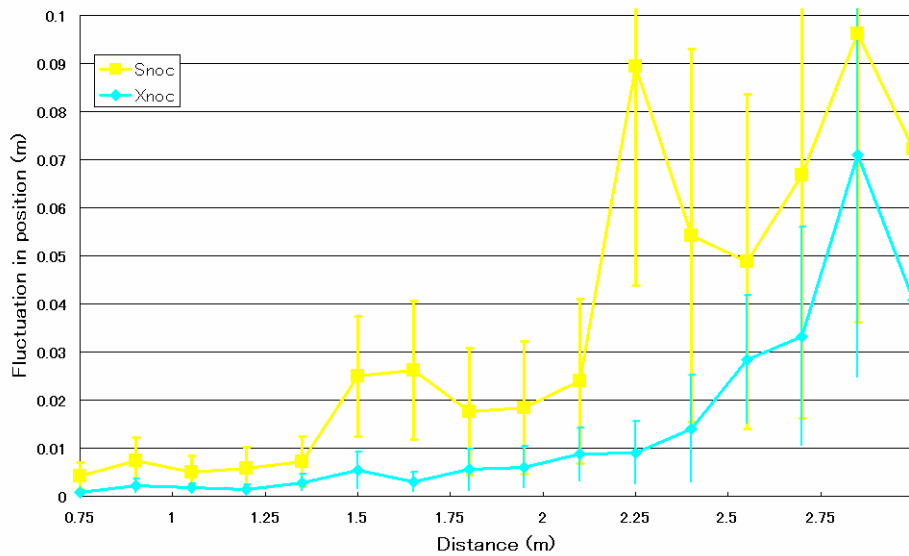


Figure 3.11 Fluctuation in position without correction (marker L).

According to the observed fluctuation data, the position of the virtual object is generally more stable when using the proposed registration method instead of the registration method. The registration depth is then less limited when using the proposed registration method. As shown in Figure 3.12, the fluctuations in position observed with the stereoscopic method are mainly influenced by the distance between the marker and the cameras. The fluctuation in position can be evaluated for any distance using an approximation function shown as the dotted line in the figure. The quality of the position stability decreases with the distance when using the stereoscopic method.

On the other hand, the distance is not the only factor influencing the proposed registration method. As shown in Figure 3.13, the fluctuations in position observed for the three size markers are associated to disjoint stability curves (shown by dotted lines in the figure). The size of the marker also influences the registration quality. That is, the registration can be performed at any distance from the marker at the condition that the marker size is appropriate. Consequently, the registration depth is not limited when using the proposed registration method. In a practical application, the distance range of the application must be evaluated beforehand in order to find the best compromise for the size

of the marker. If the size of the marker is not correctly chosen, the quality of the registration may be affected.

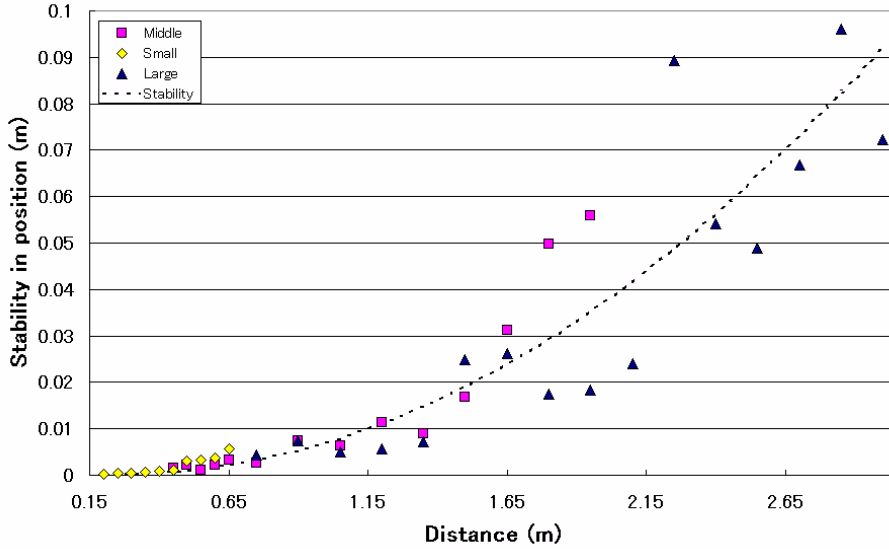


Figure 3.12 Fluctuation in position for the stereoscopic method (Snoc).

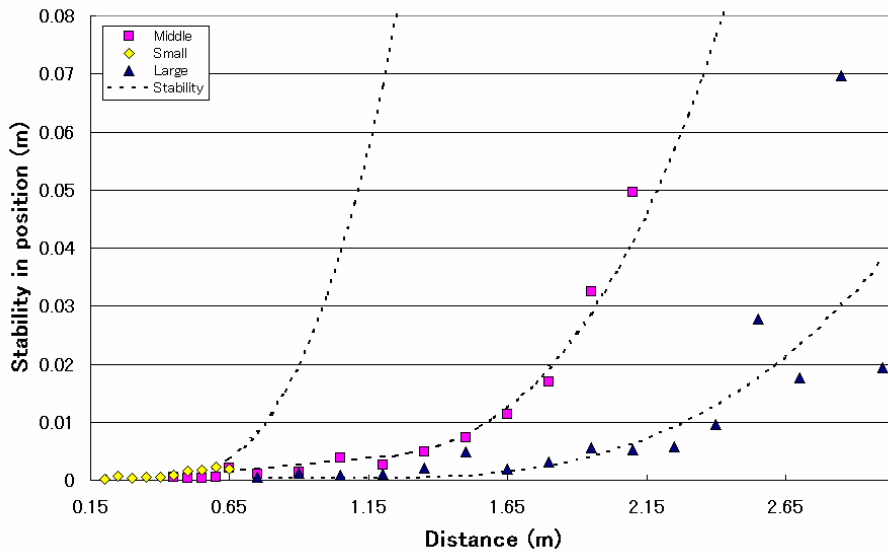


Figure 3.13 Fluctuation in position for the proposed method (Xnoc).

Similar results are observed for the orientation stability. Figure 3.14, Figure 3.15 and Figure 3.16 presents the fluctuations in orientation computed for each of the three size

markers. The orientation stability is clearly better with the proposed method. The average fluctuation in orientation associated to the registration method can be up to twelve times more stable than the average fluctuation in orientation observed for the stereoscopic method.

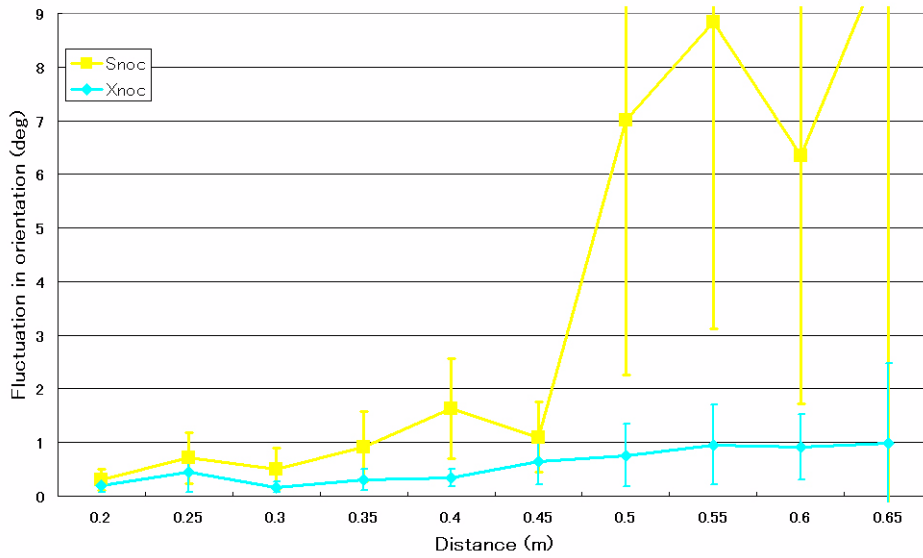


Figure 3.14 Fluctuation in orientation without correction (marker S).

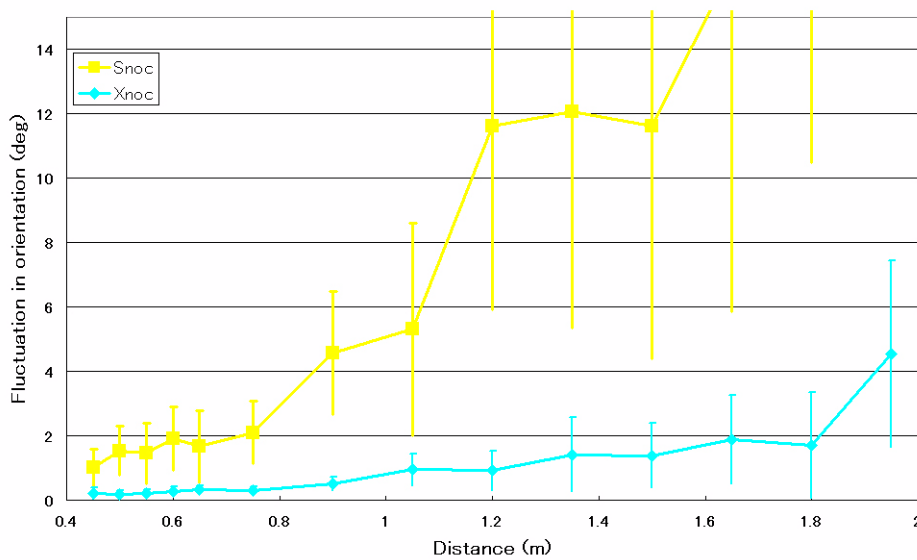


Figure 3.15 Fluctuation in orientation without correction (marker M).

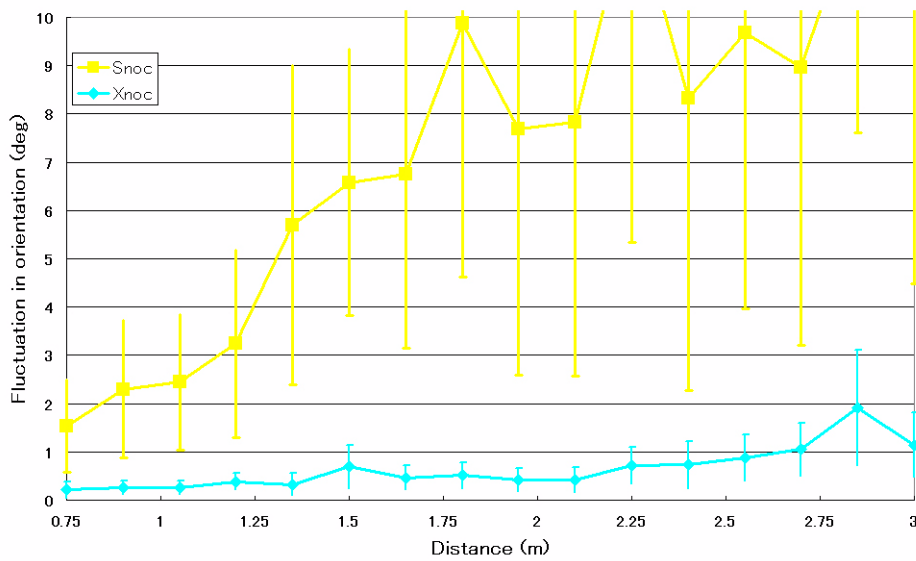


Figure 3.16 Fluctuation in orientation without correction (marker L).

Although the strict influence of the distance is less evident than in the case of the fluctuations in position, most of the observed fluctuations in orientation seem to follow a stability equation (the dotted line in Figure 3.17). Consequently, the stability in orientation is principally influenced by the distance, but the size of the marker may also have an influence on the stability observed for the stereoscopic method.

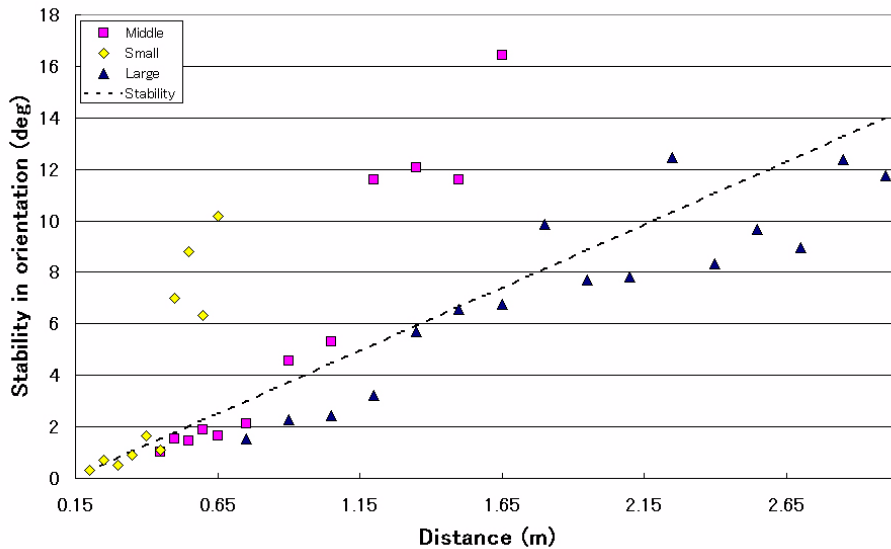


Figure 3.17 Fluctuation in orientation for the stereoscopic method (Snoc).

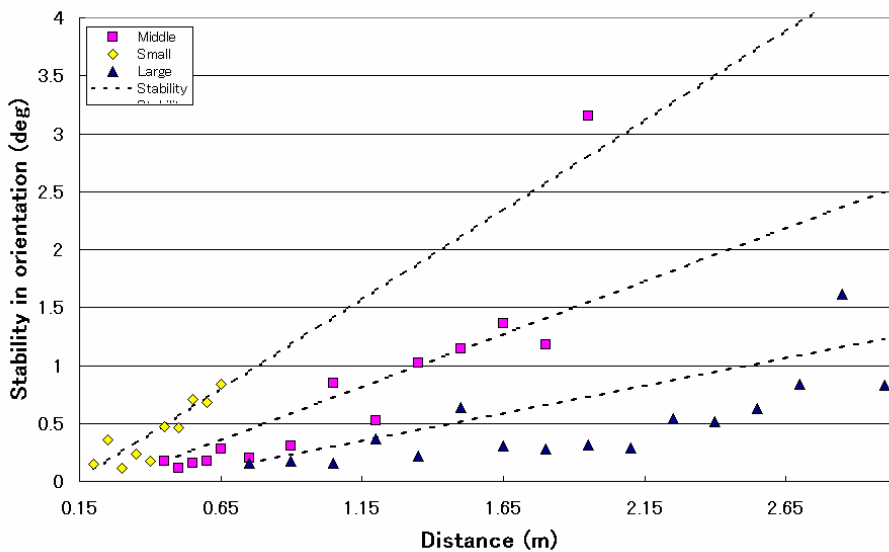


Figure 3.18 Fluctuation in orientation for the proposed method (Xnoc).

However, the fluctuations in orientation, like the fluctuations in position, are clearly influenced by both the distance and the size of the marker when using the proposed method. Figure 3.18 shows that a different stability equation can be used to approximate the fluctuations in orientation for each of the three size markers. As for the fluctuations in position, the quality of the registration is obtained if the size of the marker is sufficiently large according to the distance.

3.4.2 Effect of the angle

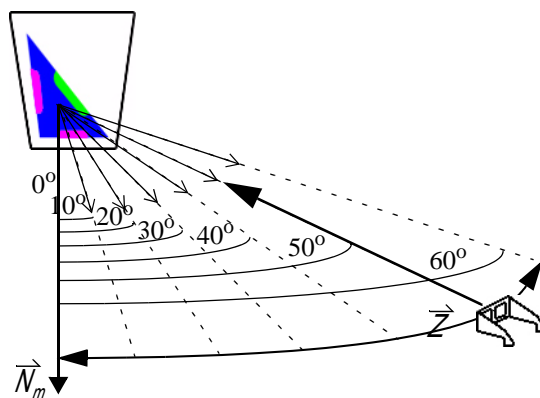


Figure 3.19 Evaluation of the angle effect on the stability.

The angle between the marker plane normal \vec{N}_m and the axis \vec{Z} of the camera coordinate system also influence the quality of the registration. The fluctuations data are obtained for different angles using the small size marker as shown in Figure 3.19. The values have been computed at a distance of 0.35m between the marker and the cameras. The stabilities in position and in orientation observed are given in Figure 3.20 and Figure 3.21, respectively.

The angle has no evident effect on the position stability for both methods. The observed variation of the fluctuations data is less than 1mm. However, the angle has a visible effect on the orientation stability. At 0° , the quality of the registration slightly decreases, especially for the proposed registration method. Finsterwalder's approach usually gives 2 different solutions for each frame (a maximum of 4 paired solutions). Because the number of solutions given by Finsterwalder's approach increases to 12 for each frame when the angle is near 0° , the algorithm must choose between 144 paired solutions instead of only 16. Among those 144 paired solutions, many solutions are very similar and the small difference in 3D positions of the triangle corners is only discernable from a slight orientation difference of the triangle marker. The probability to select the correct solution among the possible ones decreases since many solutions are only slightly different. As a result, the fluctuations in orientation values increase. However, the frame rate is not influenced by the excess of processing added by the larger number of solutions to evaluate (about 0.01 second slower at 0° compared to the processing time at 60°).

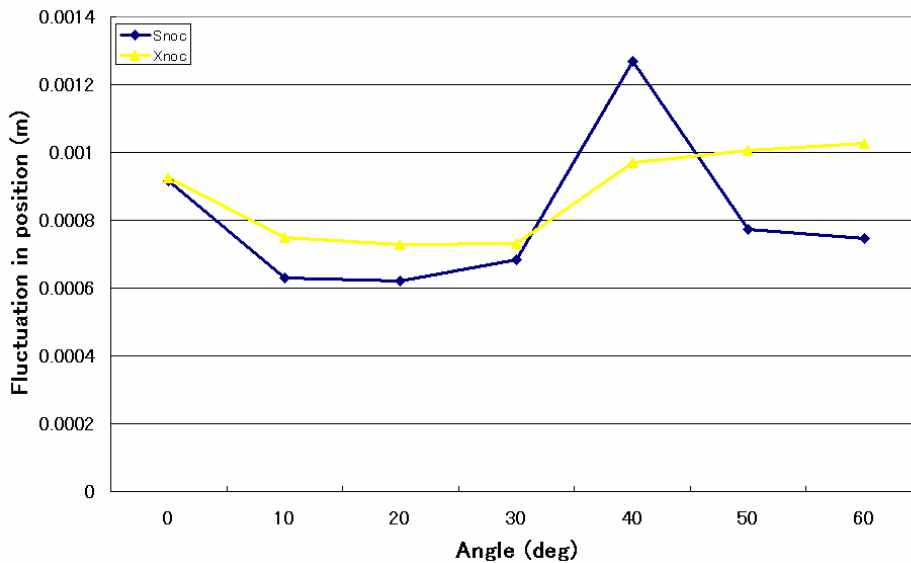


Figure 3.20 Angle effect on the fluctuations in position.

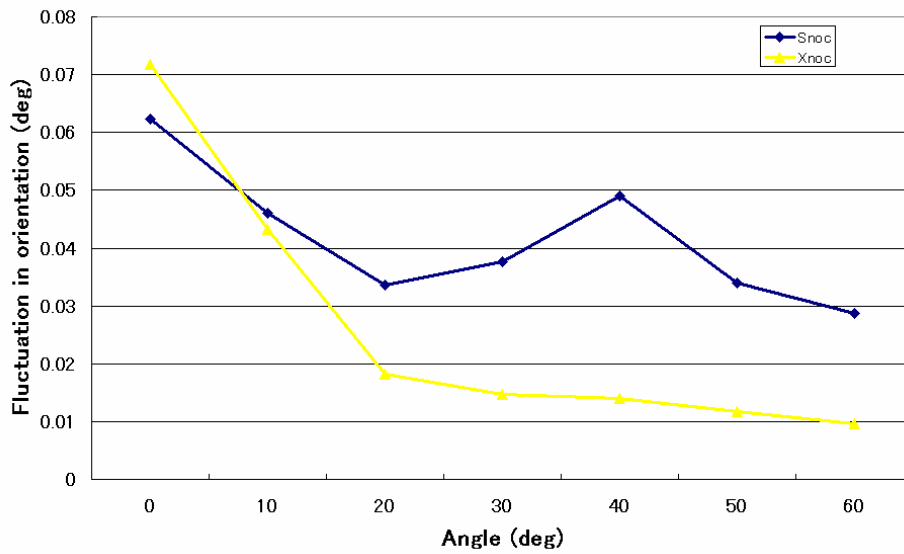


Figure 3.21 Angle effect on the fluctuations in orientation.

3.4.3 Registration coherence

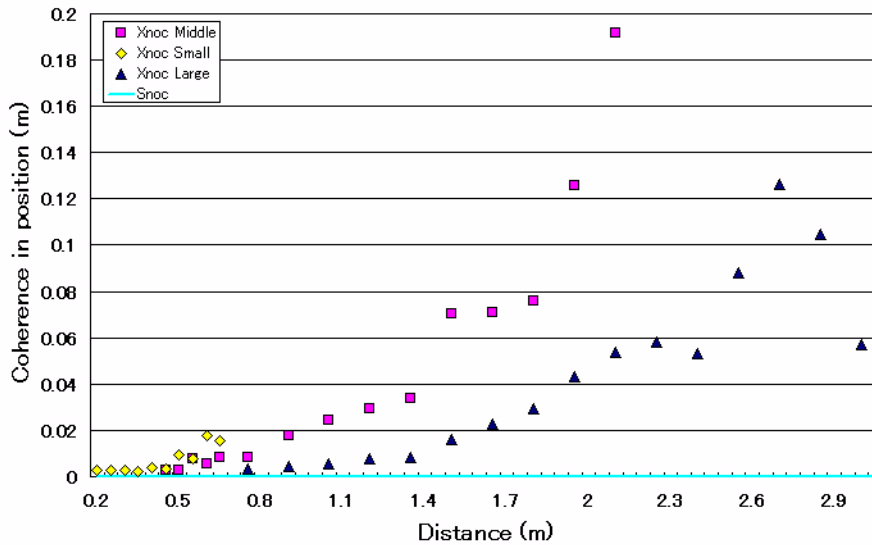


Figure 3.22 Coherence in position without correction.

The last aspect studied is the coherence between the stereoscopic augmented images. When using the stereoscopic method, the position computed for each triangle corner is unique. Therefore, the two generated augmented images are perfectly coherent.

Consequently, the coherence in position and the coherence in orientation given by equation 3.4 and equation 3.5 are always zero. However, incoherence exists between the two stereoscopic augmented images created with the proposed registration method since the 3D positions of the triangle corners are computed independently for the left and right frames. Figure 3.22 and Figure 3.23 respectively gives the observed position coherence values and the observed orientation coherence values.

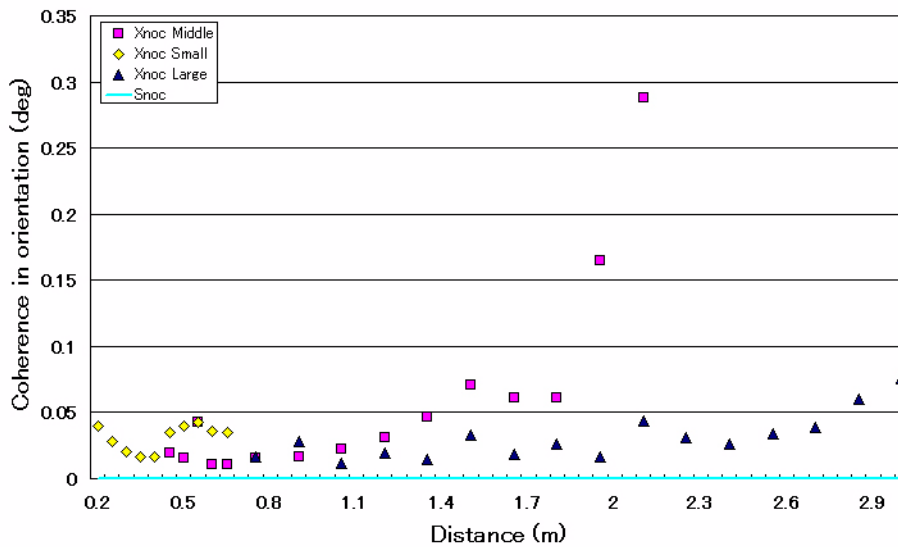


Figure 3.23 Coherence in orientation without correction.

As expected, the coherence values observed for every size marker at every distance are always zero for the stereoscopic method. The coherence values obtained with the proposed method are closely related to the fluctuation data previously presented. Both position and orientation coherence values observed are significant. Consequently, the user’s perception of the scene can be confused by the incoherence between the stereoscopic images.

3.4.4 Augmented image results

Table 3.3 shows some examples of augmented images produced using the stereoscopic method (“Snoc”). Two successive frames of three augmented sequences are shown to give an idea of the stability observed for the virtual object. In the case of the middle size and the large size markers, the augmented images have been cropped and resized. Similarly, Table 3.4 shows the augmented images produced using the proposed registration method (“Xnoc”) for the same frames shown in Table 3.3. The stability of the registration is

clearly better with the proposed registration method when comparing the augmented images produced by both methods.

Table 3.3: Examples of augmented images using the “Snoc” method


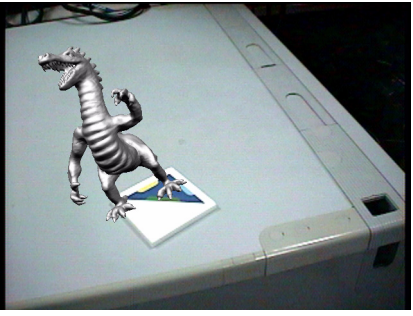


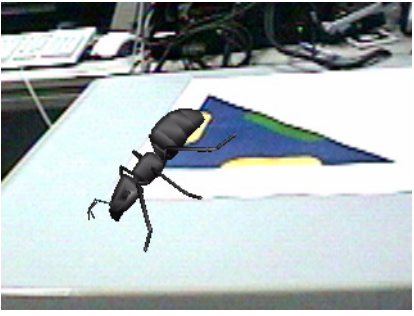
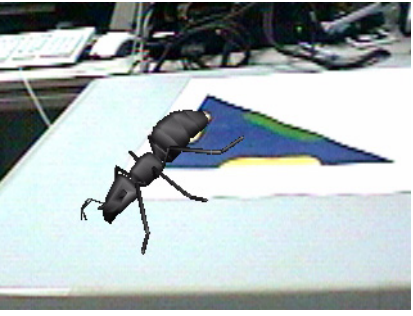

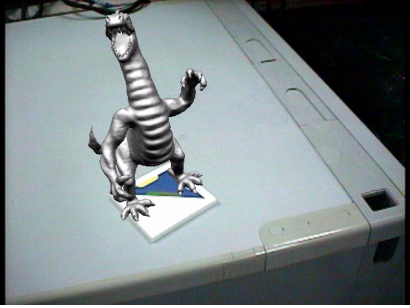
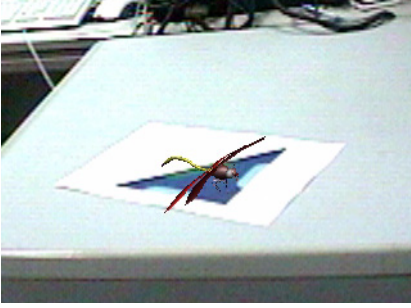

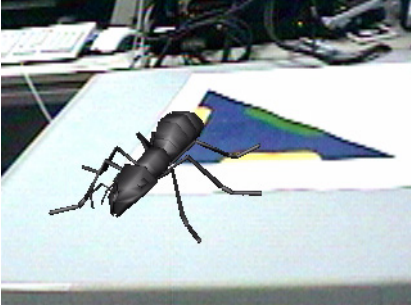
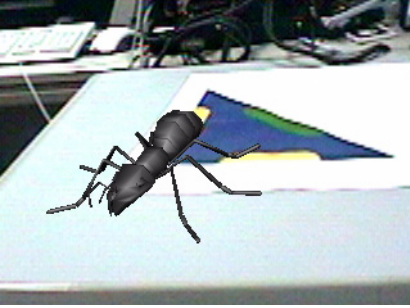
Information	Frame #1	Frame #2
<p>method : Snoc marker : S size distance : 0.5m</p>		
<p>method : Snoc marker : M size distance : 1.2m</p>		
<p>method : Snoc marker : L size distance : 1.2m</p>		

Table 3.4: Examples of augmented images using the “Xnoc” method

Information	Frame #1	Frame #2
method : Xnoc marker : S size distance : 0.5m		
method : Xnoc marker : M size distance : 1.2m		
method : Xnoc marker : L size distance : 1.2m		

All in all, the proposed registration method shows better registration results compared to the stereoscopic method. However, significant incoherence can occur between both stereoscopic views of the augmented scene.

3.5 Evaluation of the correction method

The correction method is applied to the proposed registration method in order to evaluate its effect on the stability of the registration. Once the correction method has been done, the corrected positions of the features can also be used to produce the registration using the stereoscopic registration method. Therefore, the effect of the correction method can be evaluated for the proposed registration method and for the stereoscopic method. Figure 3.24 gives the block diagram of the correction method applied to the registration proposed registration method (“Xcor”) and to the stereoscopic registration method (“Scor”).

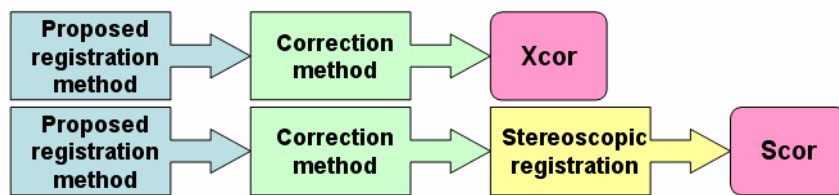


Figure 3.24 Block diagram of the registration methods with correction.

3.5.1 Effect on the registration stability

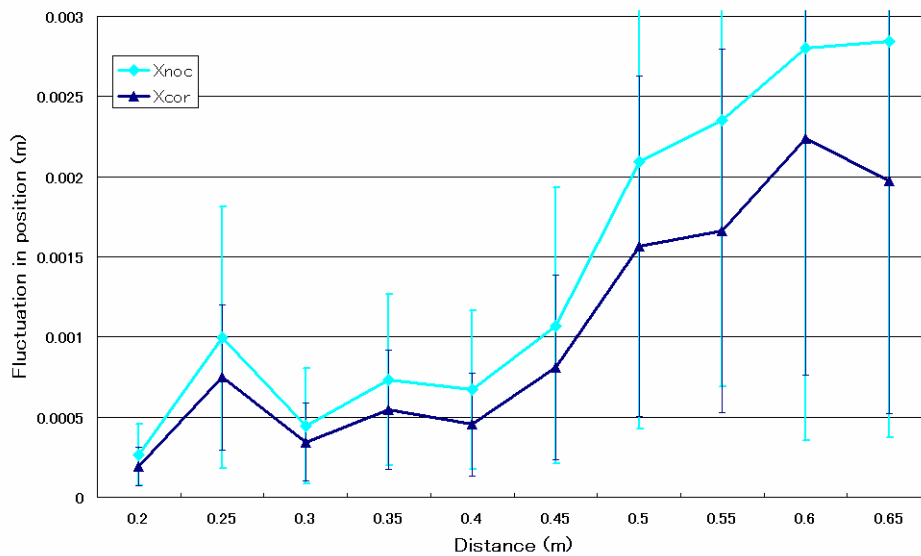


Figure 3.25 Fluctuation in position for the proposed method (marker S).

The effect of the correction on the fluctuation in position computed with the proposed registration method is presented for each size marker in Figure 3.25, Figure 3.26 and

Figure 3.27. The correction has a positive effect on the stability. Typically, the error of position decreases of few centimeters.

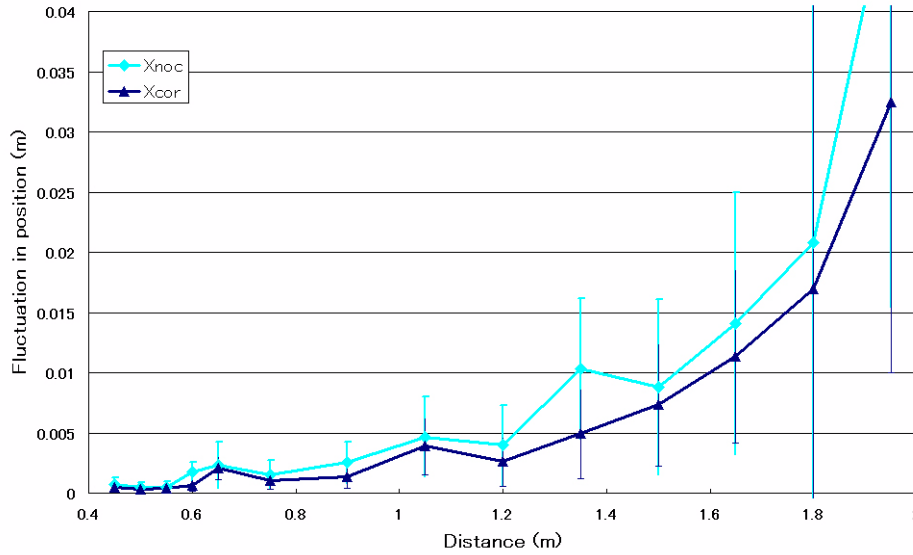


Figure 3.26 Fluctuation in position for the proposed method (marker M).

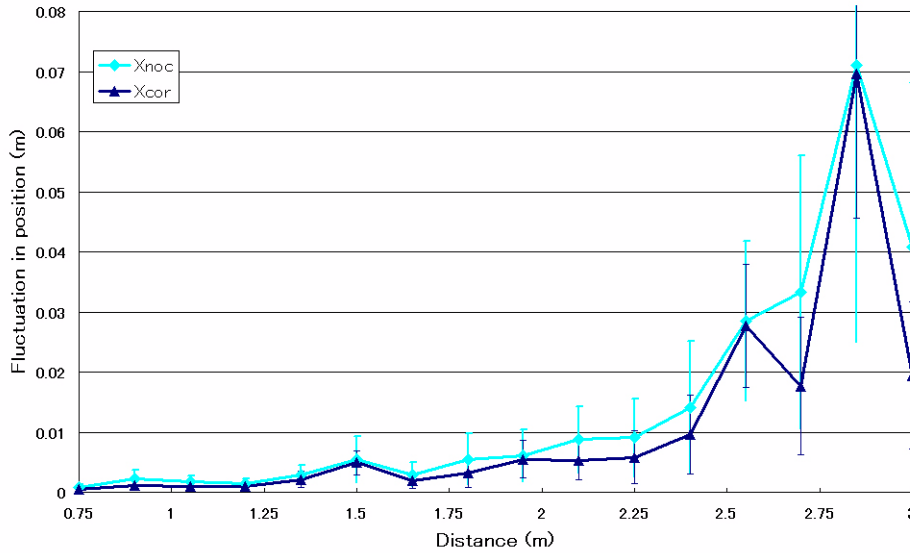


Figure 3.27 Fluctuation in position for the proposed method (marker L).

The effect of the correction on the fluctuations in orientation is also positive. The orientation gains in stability when the correction is applied. Figure 3.28, Figure 3.29 and Figure 3.30 give the fluctuation data for each size marker.

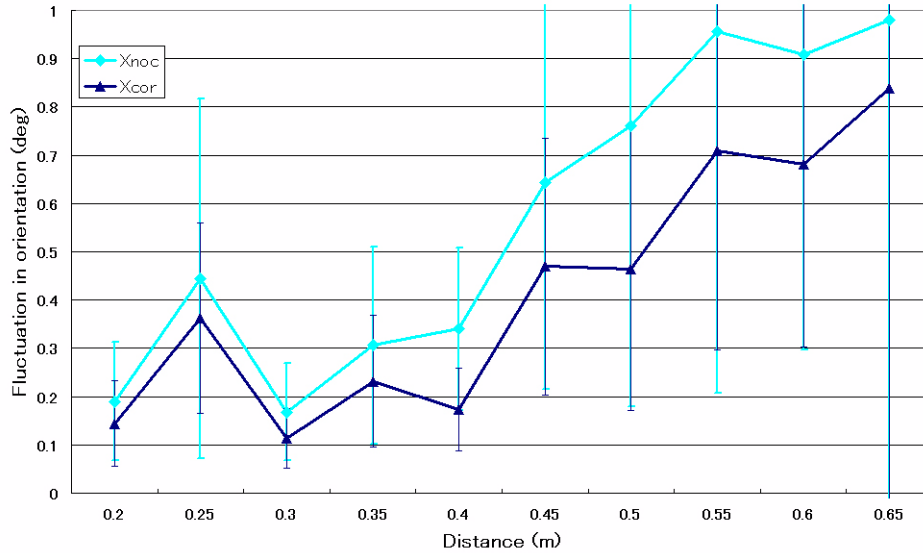


Figure 3.28 Angular fluctuations for the proposed method (marker S).

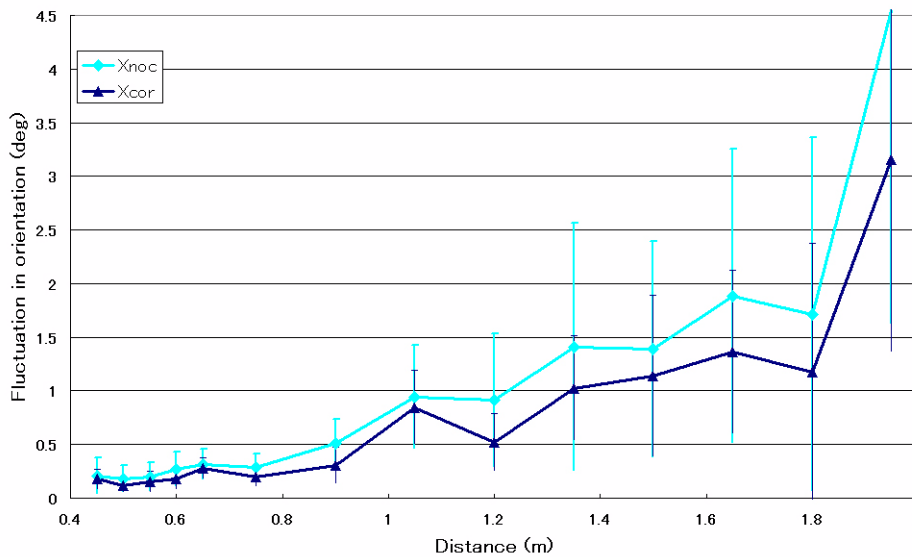


Figure 3.29 Angular fluctuations for the proposed method (marker M).

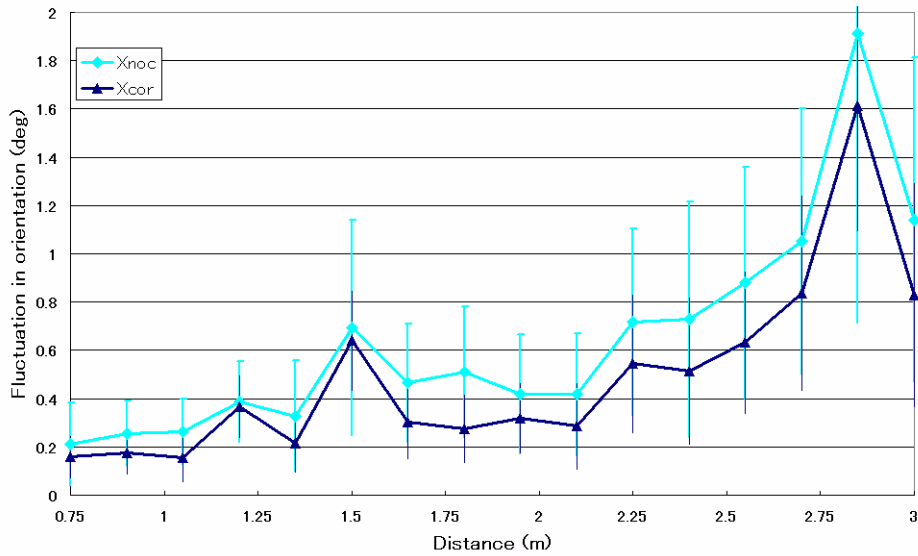


Figure 3.30 Angular fluctuations for the proposed method (marker L).

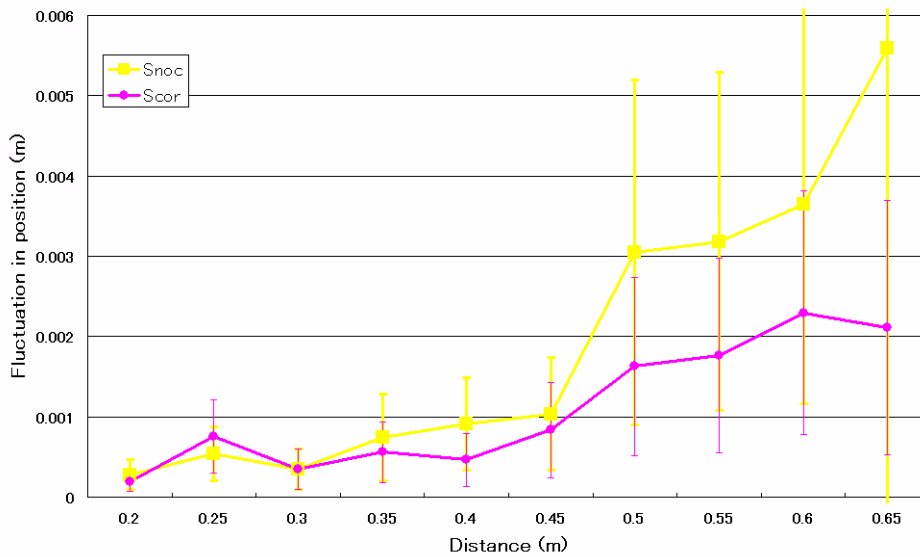


Figure 3.31 Position fluctuations for the stereoscopic method (marker S).

The correction takes about 0.1msec to execute, independently of the distance. Also, the correction seems to be more important when the marker is more distant from the cameras. The correction method gives better stability results with the proposed registration method, but if the correction is really valid and improves the positions of the feature points in the

image frames, the stereoscopic method must also be positively influenced by the correction. The figures from Figure 3.31 to Figure 3.36 give the fluctuation data observed with the stereoscopic method for each of the three size markers.

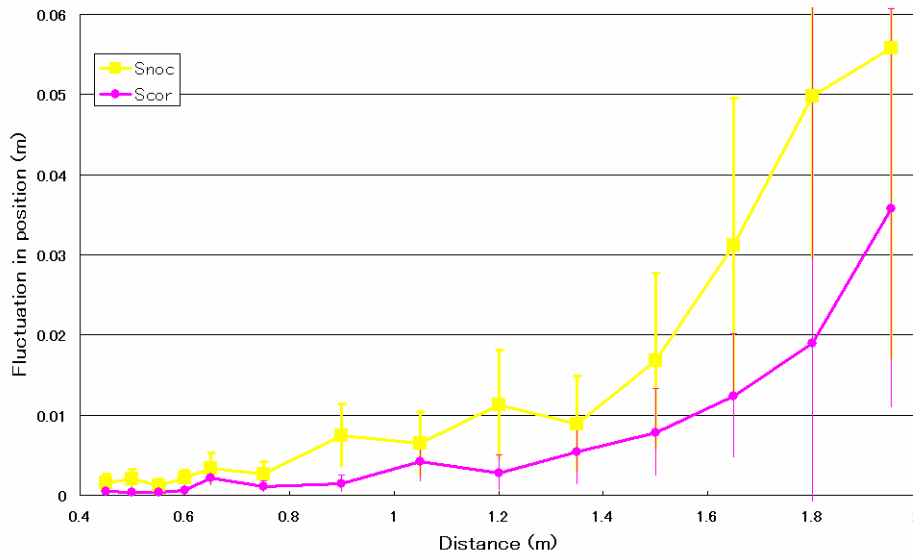


Figure 3.32 Position fluctuations for the stereoscopic method (marker M).

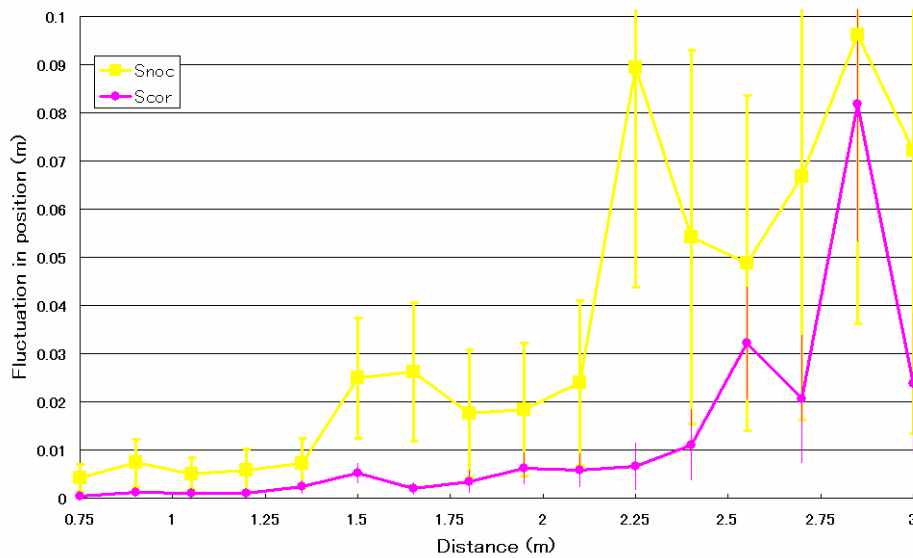


Figure 3.33 Position fluctuations for the stereoscopic method (marker L).

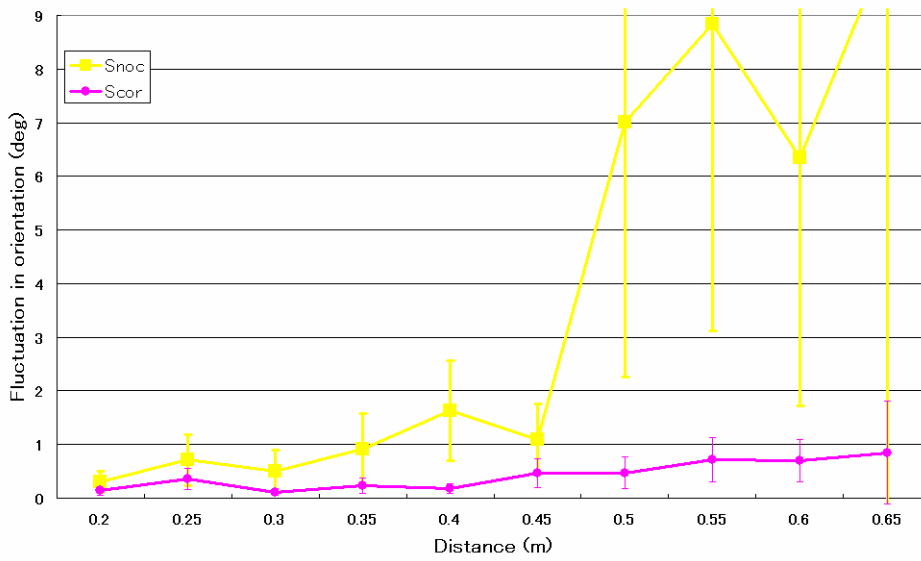


Figure 3.34 Angular fluctuations for the stereoscopic method (marker S).

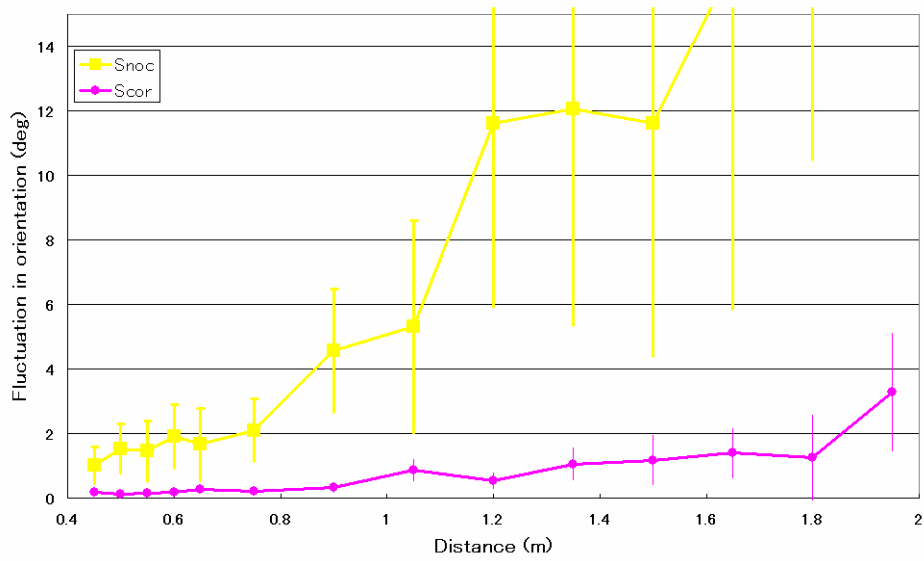


Figure 3.35 Angular fluctuations for the stereoscopic method (marker M).

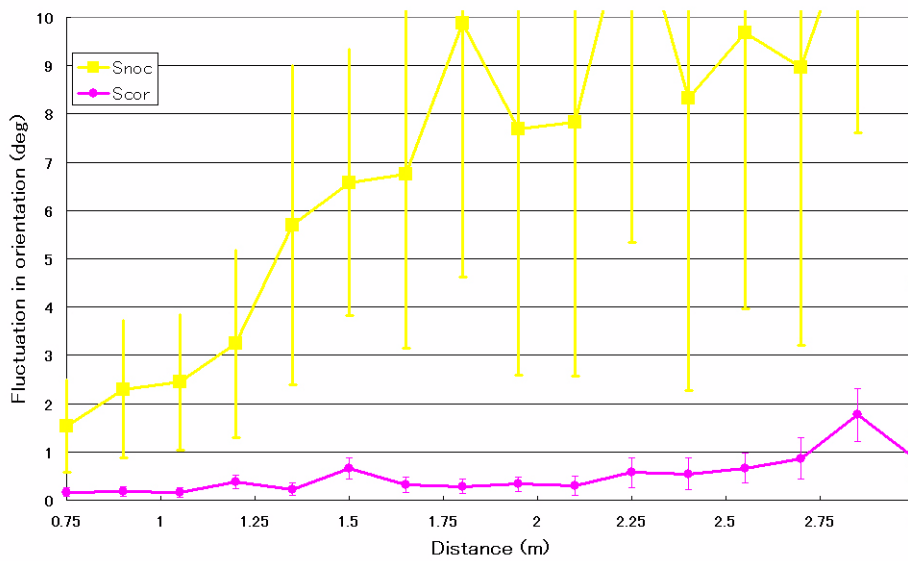


Figure 3.36 Angular fluctuation for the stereoscopic method (marker L).

The correction has a significant positive effect on the registration stability of the stereoscopic method. The correction decreases the error in position up to 8cm and the error in orientation up to 20°. The gain in stability is more important when the marker is distant from the cameras. Therefore, after the correction is applied, the stereoscopic method is not critically influenced anymore by the distance. Consequently, the correction method is proven to be valid and efficient.

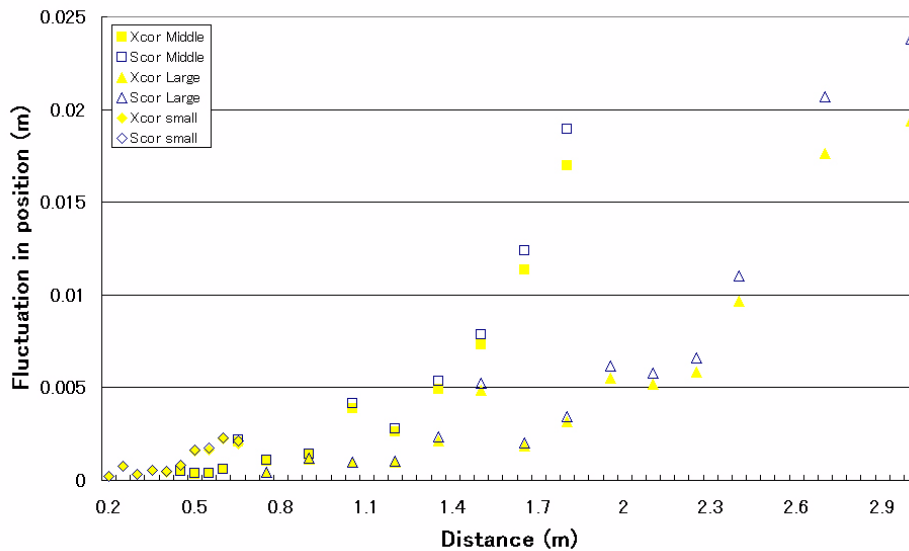


Figure 3.37 Fluctuations in position with correction.

Both registration methods are positively affected by the correction. However, the stability of the registration is still better with the proposed method as shown in Figure 3.37 and Figure 3.38. The proposed registration with the correction method is then the method that should be used to create augmented images, but the correction method permits that the stereoscopic method now reaches the same level of registration stability than the proposed registration method.

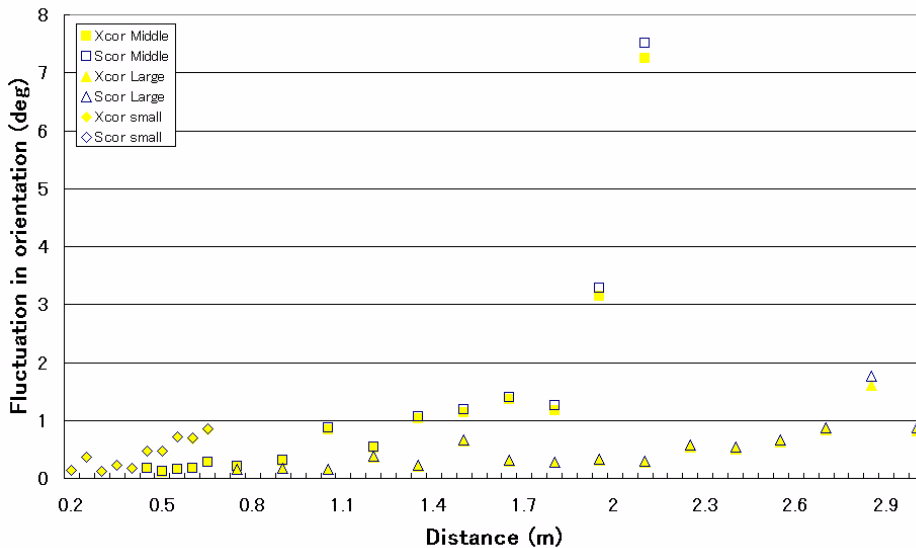


Figure 3.38 Fluctuations in orientation with correction.

3.5.2 Effect on the registration coherence

Another important aspect of the correction concerns the coherence between the left and right augmented images. As said previously in the last section, the proposed registration method may produce incoherent stereoscopic augmented images since the two frames are processed independently. Because the correction relates on the stereoscopic relation between the two frames, the 3D positions of the marker features are now dependent of both frames. Therefore, the coherence of the stereoscopic augmented images is significantly improved compared to the coherence observed without the correction. The fluctuations in position and the fluctuations in orientation are compared for the proposed method with and without correction and the stereoscopic method in Figure 3.39 and Figure 3.40. As already mentioned, the stereoscopic method guaranties a perfect coherence between both stereoscopic augmented images. Also, the coherence values observed for the proposed method without correction have previously been commented. As shown in the figures, the coherence values observed for the proposed method with correction are sufficiently negligible to conclude that the coherence between the two stereoscopic augmented images is assured.

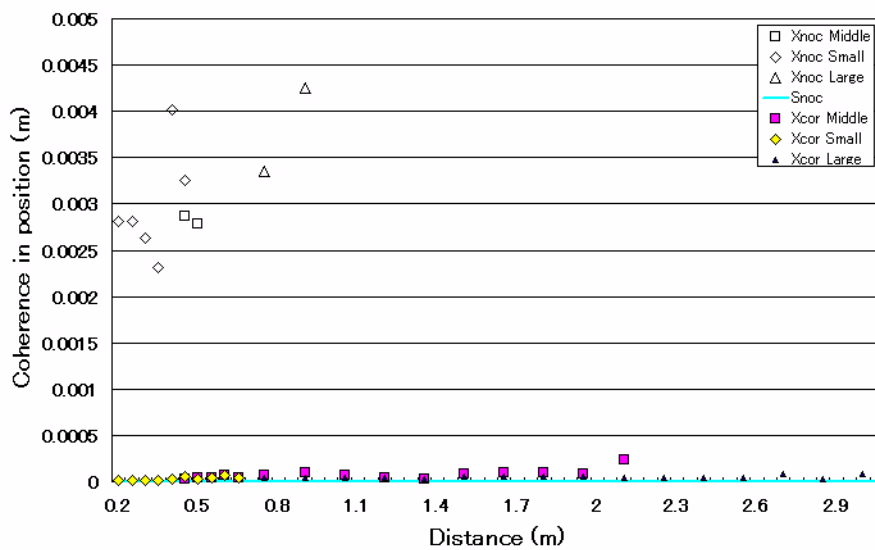


Figure 3.39 Coherence in position with correction.

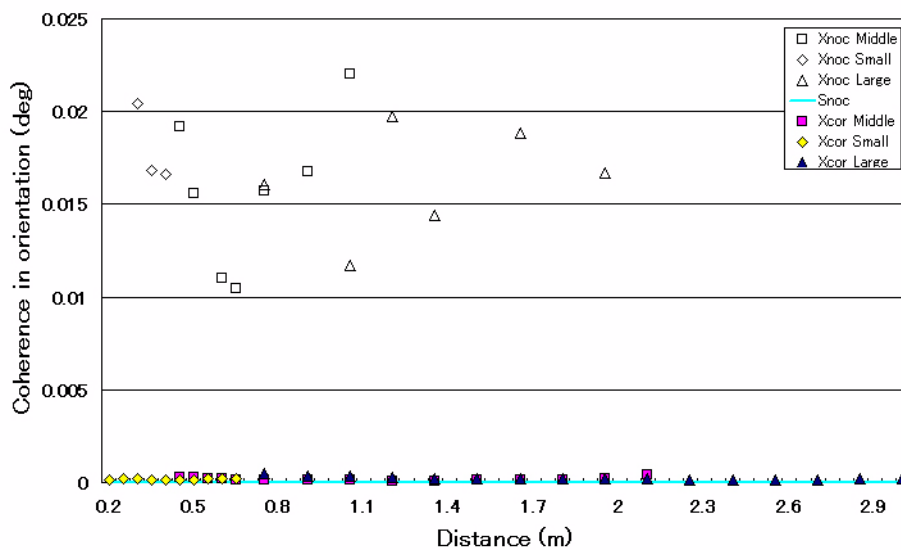


Figure 3.40 Coherence in orientation with correction.

3.5.3 Augmented image results

For the same frames of Table 3.3 and Table 3.4, the augmented images produced using both the stereoscopic and the proposed registration with the correction applied are shown in Table 3.5 and in Table 3.6. The effect of the correction is hardly perceivable when

comparing the different augmented images presented for the “Xnoc”, “Xcor” and “Scor” methods because the fluctuation data of these methods have a same magnitude level.

Table 3.5: Examples of augmented images using the “Scor” method



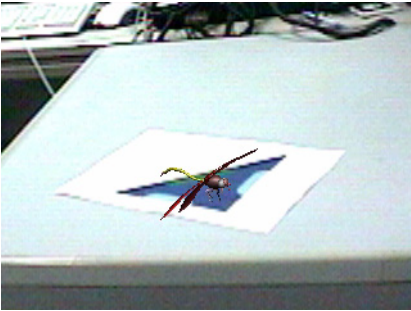

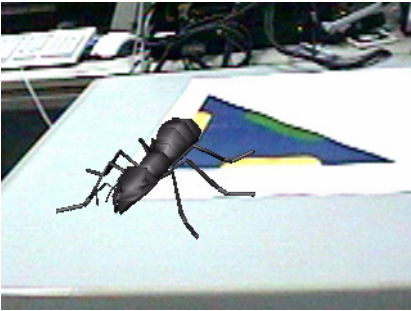
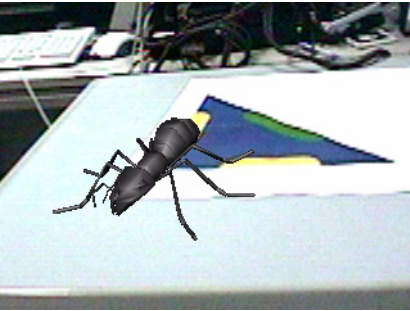

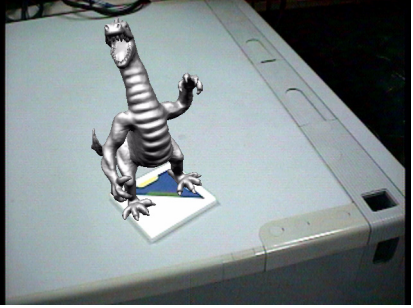
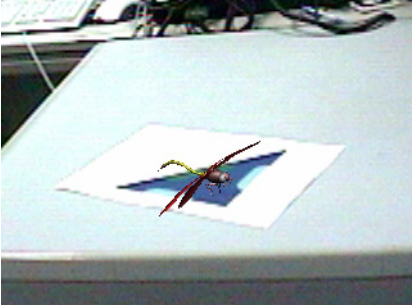
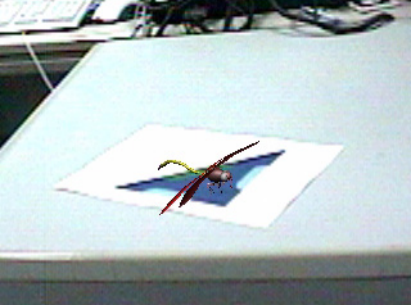
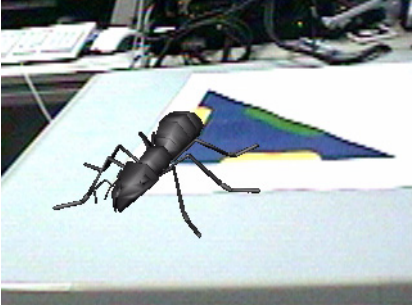
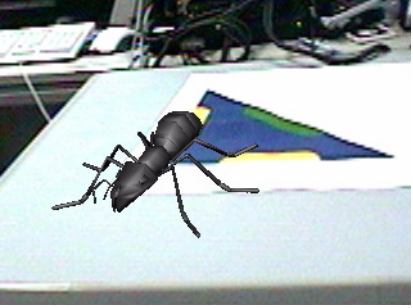
Information	Frame #1	Frame #2
<p>method : Scor marker : S size distance : 0.5m</p>		
<p>method : Scor marker : M size distance : 1.2m</p>		
<p>method : Scor marker : L size distance : 1.2m</p>		

Table 3.6: Examples of augmented images using the “Xcor” method

Information	Frame #1	Frame #2
method : Xcor marker : S size distance : 0.5m		
method : Xcor marker : M size distance : 1.2m		
method : Xcor marker : L size distance : 1.2m		

All in all, the correction method improves the stability of the registration. Also, in the case of the proposed method, the correction assures that the coherence condition between the left and right registrations is met.

3.6 Evaluation of the processing time

The processing time needed for the system to create the two augmented images is measured for the developed system using the proposed registration method with correction (Xcor). Two elements influence the processing time: the complexity of each individual virtual object and the number of virtual objects to render.

To evaluate the processing time in terms of the complexity of the virtual object, the system is asked to register four different 3D objects. The 3D objects are presented in Table 3.7.

Table 3.7: 3D objects registered by the system.


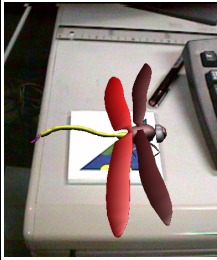

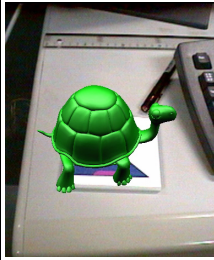
	“dragon”	“dragonfly”	“monster”	“turtle”
image				
number of vertices	54831	3855	5873	6642
number of faces	108588	7570	9556	13120

Table 3.8 presents the time spent to create two stereo-images using one of the 3D objects presented in Table 3.7. The time are given according to different steps of the process. Also, the table presents the average frame rates observed for the system. The markers were placed at about 0.30m of the cameras. As already observed in section 3.3, the time needed by the identification of the blue pixels using the hexagonal pattern and the time needed by the extraction of the triangular marker are constant independently of the complexity of the 3D object. An interesting point observed is the negligible time asked to complete the registration and the correction. The results also show that, during the merging of the “dragonfly”, the “monster” and the “turtle”, the major part of the processing time is spent to map the camera images onto a rectangular surface using the texture mapping functions of OpenGL. However, the time spent to map the camera images is not influenced by the complexity of the 3D object. In contrast, the time needed to render the 3D object is directly influenced by the complexity of the 3D object. The “dragonfly”, the “monster” and the “turtle” have a limited number of polygon and the time needed to render those objects is relatively short compared to the time needed to render

the “dragon” which possesses a large number of polygons. Therefore, the time needed to render the “dragon” object is significant and the frame rate reached by the system drops under the 10 frames by second level. Otherwise, the system is able to run in real-time on a 800MHz SGI PC. In other words, the system succeeds to create at least a pair of 10 augmented images each second when the augmented virtual object is relatively simple to render. The frame rate of the system should increase if more powerful rendering hardwares are used. Consequently, more complex 3D objects would also be registered in real-time.

Table 3.8: Time for merging different 3D objects on stereo images.

Object 3D	Identify blue pixels	Extract the marker	Registration and correction	Mapping of the camera images	Render the 3D object	Frame rate
dragonfly	7.04ms	4.27ms	0.34ms	35.49ms	1.71ms	14.74 f/s
monster	7.05ms	4.24ms	0.31ms	35.96ms	2.08ms	14.69 f/s
turtle	7.05ms	4.24ms	0.24ms	35.44ms	2.52ms	14.70 f/s
dragon	6.88ms	4.22ms	0.33ms	36.24ms	31.26ms	9.56 f/s

Multiple markers are inserted in the field of view of the camera to evaluate the consequence of the number of markers on the processing time. Table 3.9 gives the frame rate reach by the system and the time spent in the interesting step of the process according to the number of markers extracted. The distance between the camera and the markers was about 0.30m and a 3D “turtle” object is rendered for each extracted marker. The identification of the blue pixels in the image is slightly influenced by the number of blue pixels in the two images. The extraction of the marker is linearly influenced by the number of markers since the extraction of one marker takes about 4 ms. The time spent to perform the registration and the correction is also influenced by the number of markers, but the time spent is negligible (about 0.30ms by marker). The mapping of the camera images is not influenced by the number of markers because the mapping of the two camera frames is independent of the content of the frames. However, a 3D object needs to be rendered for each extracted marker. Thus, the time spent in rendering the 3D objects increases considerably with the number of markers extracted. The system runs with more than 10 frames by second. If the number of markers continues to increase, the frame rate will eventually drop under 10 frames by second. If the frame rate drops too far under the level of 10 frames by second, the user will be perturbed by the slowing down of the system. In this case, a solution should be to use more powerful rendering hardwares.

Table 3.9: Time for merging multiple 3D objects on stereo images.

Number of markers	Identify blue pixels	Extract the marker	Registration and correction	Mapping of the camera images	Render the 3D objects	Frame rate
0 marker	6.79ms	0.00ms	0.03ms	35.52ms	0.07ms	14.94 f/s
1 marker	7.05ms	4.24ms	0.34ms	36.44ms	2.52ms	14.70 f/s
2 markers	7.23ms	7.18ms	0.54ms	35.40ms	3.34ms	11.96 f/s
3 markers	7.26ms	9.98ms	0.77ms	36.27ms	4.54ms	11.91 f/s
4 markers	7.32ms	14.21ms	0.96ms	35.64ms	5.61ms	11.16 f/s

All in all, the system succeeds to run in real-time if the complexity of the 3D objects to render and if the number of markers in the image are limited. The timing results have been measured on a 800MHz SGI PC. Also, the graphic card used to render the virtual objects is relatively slow. Obviously, the system will speed up if a more powerful PC or a more powerful graphic card are used.

3.7 Comparison with the ARToolKit

The ARToolKit is a software library that can be used to calculate camera position and orientation relative to physical markers in real-time[ARTK]. This library is widely used because it is distributed free of charge and the algorithms contained in the library corresponds to the standard algorithms used in augmented reality systems. Therefore, many developers and searchers use this development tool to quickly develop fully functional augmented reality applications.

The ARToolKit also processes the entire frame in order to extract the markers (when the full frame option is set). The ARToolKit extraction of square markers is based on monochromatic and template matching processes. In contrast, the proposed system uses triangular markers extracted from color information. The percentage of successful marker extraction observed for the ARToolKit have also been measured for slow, normal and fast motion sequences. The three sequences of each system are equivalent to the sequence used in Figure 3.7. The percentages of successful extraction of the marker in the sequences are presented in Figure 3.41. The ARToolKit system succeeds to extract the marker in every frame during a slow and a normal motion of the user's head. However, the ARToolKit system fails to detect the marker in about 6% of the frames during a fast motion. Consequently, the movement of the virtual object is smoother with the proposed system when fast motion occurs.

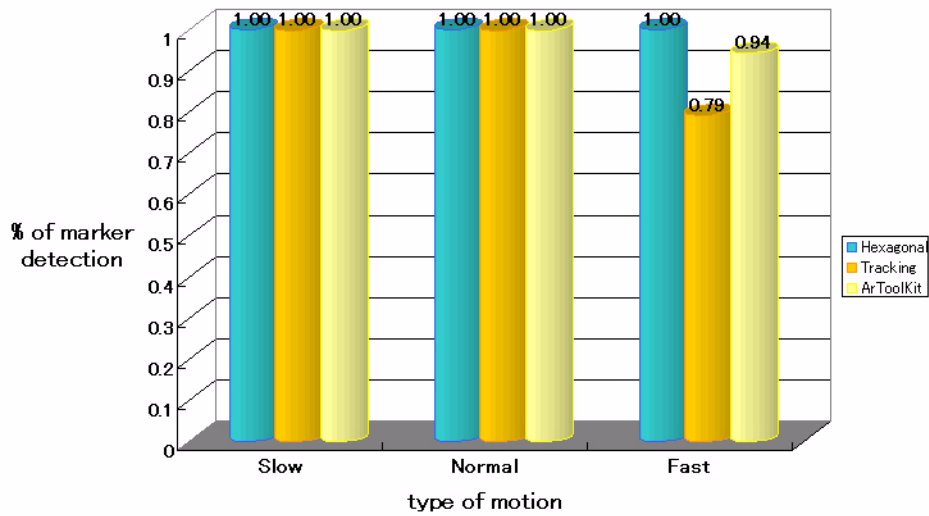


Figure 3.41 Percentage of successful marker extractions (2).

Table 3.10 compares the processing time for the developed system using the proposed registration and correction (Xcor), the developed system using the standard stereoscopic registration (Snoc) and the ARToolKit system. Two versions of the ARToolKit have been evaluated since the ARToolKit possesses the option of processing either the entire frame (100%) or the frame reduced to 25% of its original size. In order to measure the different times, all the systems have been executed on one 800 Mhz SGI PC. The measured time corresponds to the time spent to perform the registration of the 3D turtle object of Table 3.7 when the marker is distant of about 0.30m from the head mounted device. In the table, the ARToolKit timing corresponds to the registration of only the left camera since the ARToolKit system tested deals only with a single camera.

Each system is asked to capture one video image (the left camera image for the ARToolKit and a merging of the left and right images for the developed system). As shown in the table, the capture time of the camera image differs for the developed system and the ARToolKit system since the two systems used different capture functions. The table also shows that the proposed system succeeds to extract the marker faster than the ARToolKit. The proposed system spends about 5ms to extract the marker in one frame compared to 7ms and 14ms for the ARToolKit based system. The registration and correction method (Xcor) takes about 0.15ms by frame; slower than the stereoscopic registration method (Snoc) with 0.05ms (both frames are done in the same time), but faster than the monocular registration of the ARToolKit with 4.36ms by frame. However, it is important to say that the proposed registration method (Xcor) and the stereoscopic registration method (Snoc) employ three point based registrations in contrast to the

ARToolKit registration method which is based on a four point based registration method using square marker. Also, the proposed registration method (Xcor) includes an optimization method, which is not the case of the stereoscopic registration method (Snoc) and the ARToolKit registration method. Since all the systems use texture functions to map the camera images, the times needed for the systems are equivalent when the number of frames processed is considered. The same result is observed for the time asked to render 3D objects. In conclusion, when we consider the number of processed frames, the developed system is slightly faster than both versions of the ARToolKit system. In this experiment, the proposed system is able to discriminate between six markers and the ARToolKit between four markers.

Table 3.10: Timing of the systems during the registration of the monster.

System	Number of frames by iteration	Capture camera image	Extract the marker	Registration	Mapping of the camera image(s)	Rendering the 3D object	Frame rate
proposed system (Xcor)	2	15.24ms	10.45ms	0.31ms	36.47ms	2.55ms	14.7 f/s
stereo system (Snoc)	2	15.32ms	10.50ms	0.05ms	35.88ms	2.51ms	14.7 f/s
ArToolKit (100%)	1	25.67ms	14.54ms	4.36ms	20.44ms	1.49ms	15.0 f/s
ArToolKit (25%)	1	9.52ms	7.02ms	4.35ms	21.54ms	1.65ms	22.6 f/s

The registration results of the proposed registration method (Xcor) have been compared with the registration results of an ARToolKit based system (the full frame version). For different distances between the marker and the cameras, we record stereoscopic video sequences showing an ARToolKit compatible square marker with a side length of 6.6cm. From each sequence, the ARToolKit performs the registration of the left frames using a four point based monocular method. Then, from the same sequence, the proposed registration method (Xcor) performs the registration of the stereoscopic frames with three of the four corners of the ARToolKit square marker. The fluctuations in position and orientation of both methods are given in Figure 3.42 and Figure 3.43.

Because the information given by three points observed by stereo-paired camera is sufficient to compute accurate 3D position of a marker, the fluctuations in position observed for both methods are similar as shown in Figure 3.42. Furthermore, the proposed registration gives slightly better results when the distance between the marker and the

camera increases because the correction method improves the accuracy of each corner position and the effect of the correction is more considerable when the distance increases. In contrast, the ARToolKit registration doesn't integrate an optimization method.

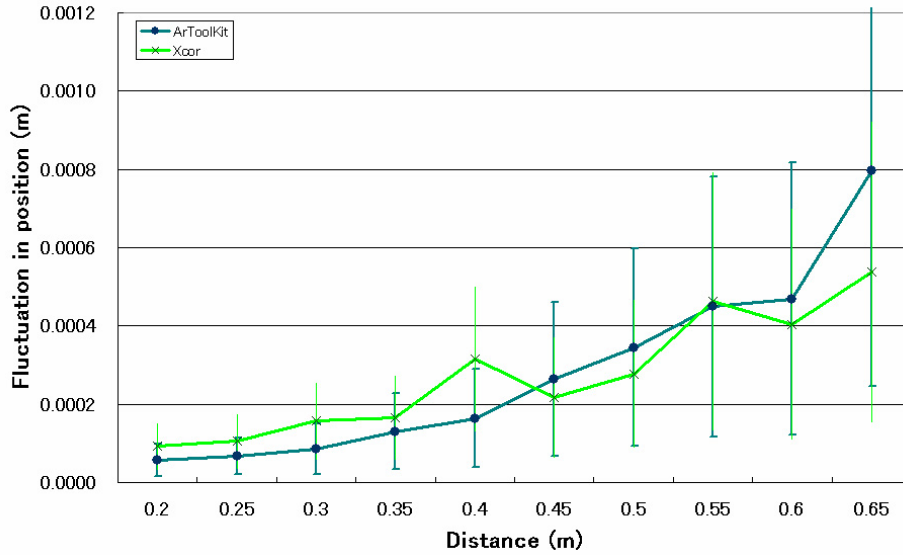


Figure 3.42 Fluctuation in position with the ARToolKit (marker S).

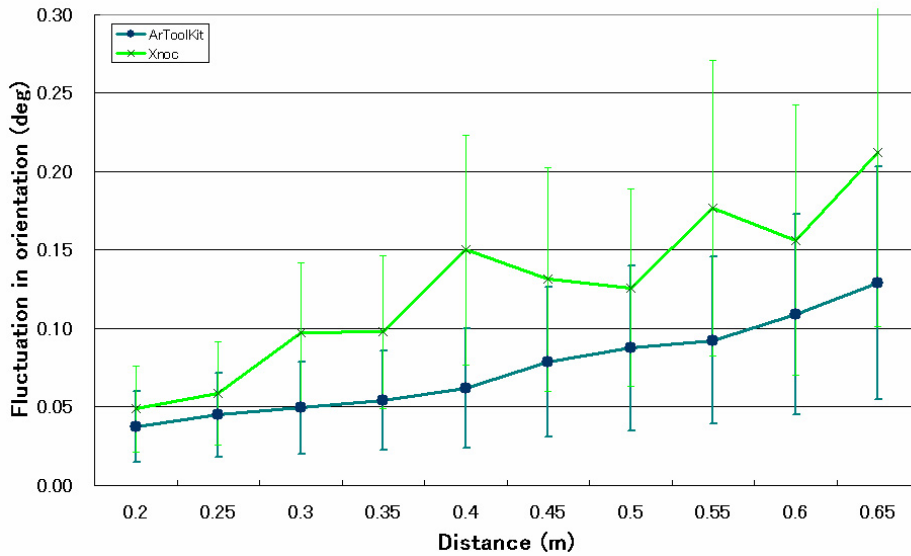




Figure 3.43 Fluctuation in orientation with the ARToolKit (marker S).

However, the four point based registration of the ARToolKit is more stable in orientation than the proposed three point based registration method (Xcor) as shown in Figure 3.43. Retrieving the orientation requires more information than retrieving the position. The use of a fourth point decreases the fluctuations in orientation since the four point based inverse perspective projection is significantly more accurate than the three point based inverse perspective projection or the three point based standard stereoscopic computation. Furthermore, the improvement from the optimization with the correction method is not sufficient to compensate the use of a fourth point.

Globally, the four point based registration of the ARToolKit gives better stability results than the proposed three point based registration method. However, the stability difference between both registrations is usually hardly perceptible by the user. Table 3.11 presents an example of augmented images produced with each of the systems.

In conclusion, the ARToolKit system and the proposed system have their own advantages and disadvantages. The proposed method is slightly faster than the ARToolKit and shows better extraction results. However, the proposed system slightly suffers from a lack of stability compared to the ARToolKit system.

Table 3.11: Augmented images of a teapot.

ARToolKit	Developed system
	

3.8 Effect of the correction factors

In order to evaluate the effect of the correction factors on the quality of the registration, fluctuation data given by the proposed registration method without correction “Xnoc” and by the proposed registration method with correction are compared when fixed factors values are used “Xcor” ($\mu = 0.5$ and $\tau = 0.5$) and when variable factors are calculated “Xcor_var” using the method explained in section 2.4.2. The Figure 3.44 and

the Figure 3.45 present the fluctuation data for the three methods: “Xnoc”, “Xcor” and “Xcor_var” when performing the registration using the small marker.

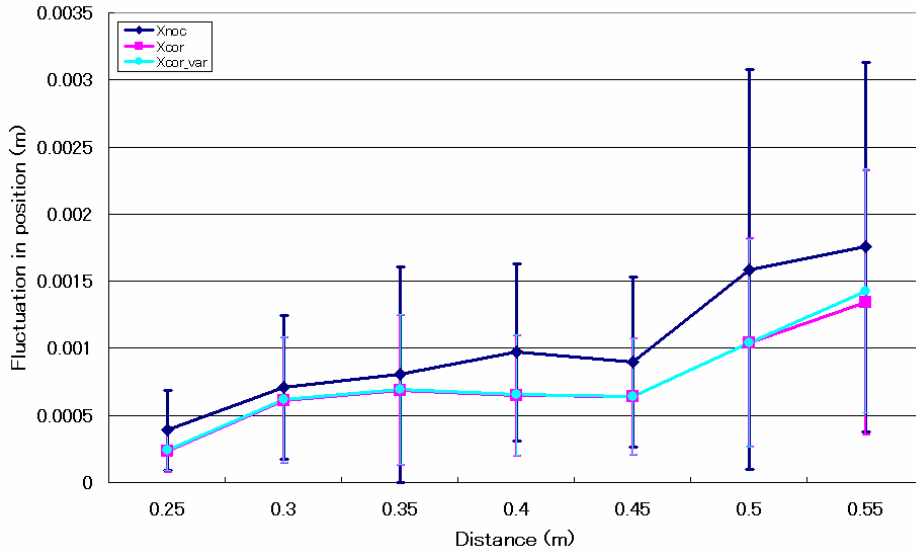


Figure 3.44 Fluctuation in position using variable factors (marker S).

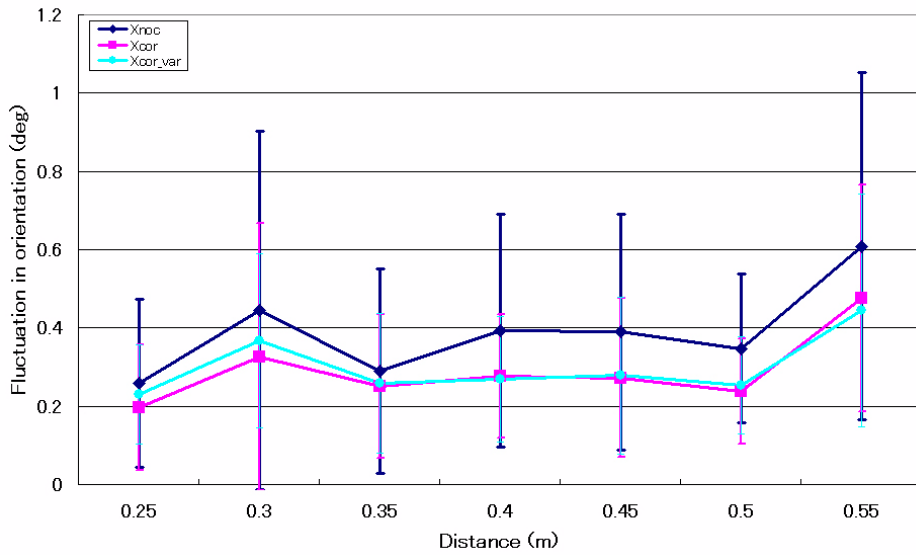


Figure 3.45 Fluctuation in orientation using variable factors (marker S).

The figures show that the fluctuations in position and the fluctuations in orientation decrease when the proposed registration method is used with or without variable factors. However, the use of variable factors doesn't improve the average stability results, but the standard deviations of the fluctuations are slightly more limited. In other words, the use of variable factors doesn't necessary improve the stability, but narrows the range of fluctuation data.

The advantage of the use of variable factors appears when an extraction error of a 2D position of one feature becomes significant. In some specific cases, like when one of the corners is obstructed by a foreground object, the 2D position of a feature may be unable to be extracted with sufficient precision, and consequently, the registration is perturbed.

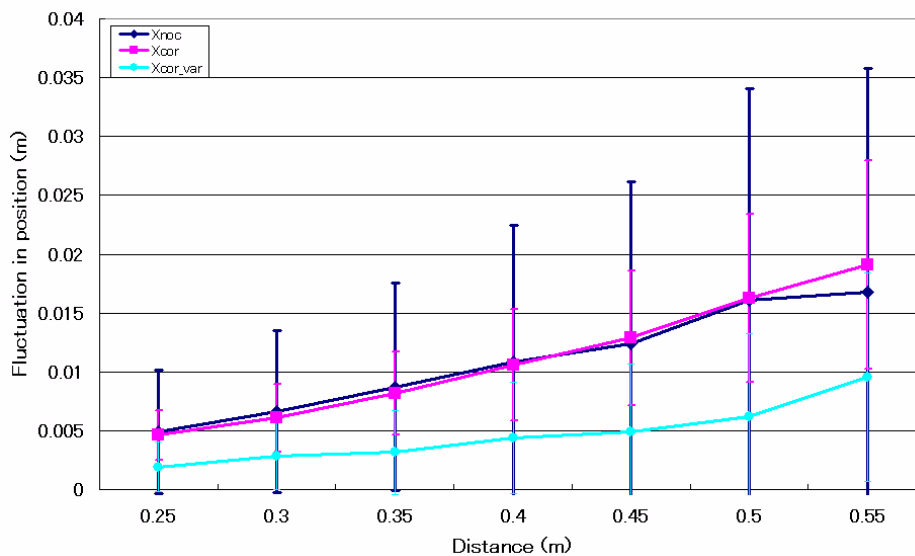


Figure 3.46 Fluctuation in position with random errors (marker S).

In order to simulate this event, an error component is randomly added to the left 2D positions extracted for one feature in each even frame of the video sequence of the left camera. The registrations produced with the proposed registration method without correction, the proposed registration method with constant factor and the proposed registration method with variable factor corrections are compared when the error component is added. The Figure 3.46 and Figure 3.47 present the stability results. The norm of the error vector varies from 1 to 15 pixels.

In presence of extraction errors, the stability of the registration is significantly perturbed. When the correction method with variable factors is used, the perturbations caused by the extraction errors are significantly decreased compared to the two other

methods. However, the methods used to evaluate the correction factors don't allow to retrieve the same level of stability initially obtained without extraction errors.

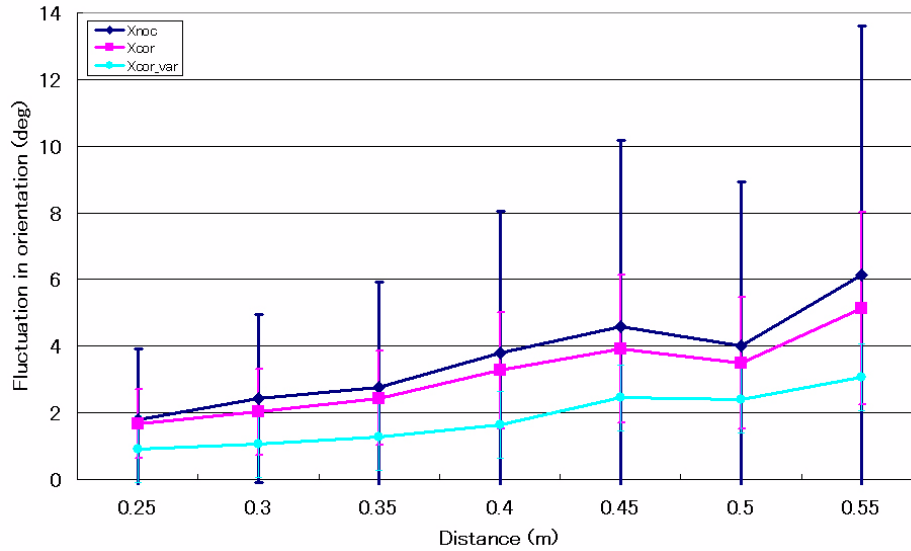


Figure 3.47 Fluctuation in orientation with random errors (marker S).

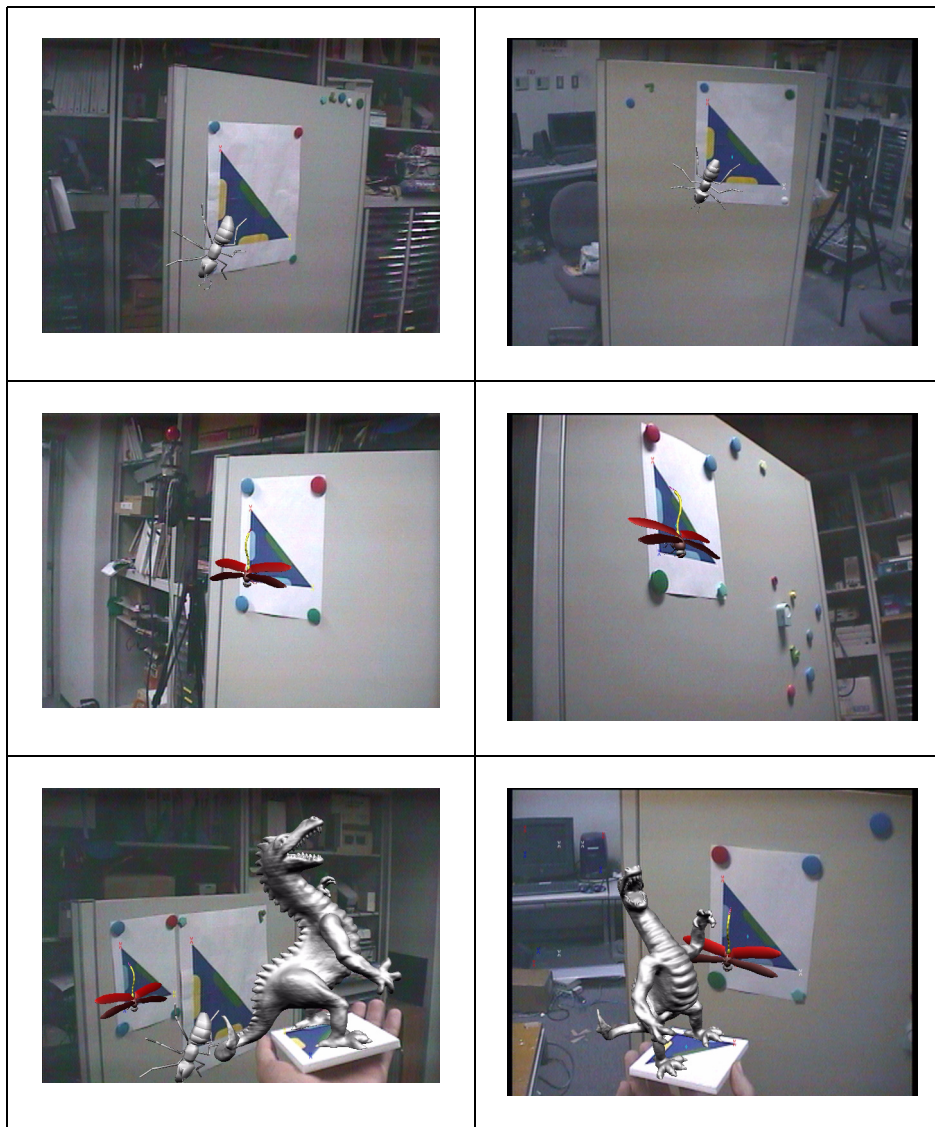
3.9 Conclusion

In this chapter, the stability of different methods has been evaluated using fluctuation coefficients. The experimentation proves that the requirements of each task have been met and that the developed system produces stable three point based registration. To conclude this chapter, several augmented images obtained with the developed system are presented in Table 3.12.

Table 3.12: Augmented images produced with the developed system.



Table 3.12: Augmented images produced with the developed system.



4. Conclusion

This thesis describes a stable registration method for vision-based augmented reality using monocular and binocular vision with feature position correction. In the field of augmented reality, two major vision-based registration methods are used: monocular registration and stereoscopic registration. The main goal of this research is to improve the stability of the registration by fully exploiting the strengths of both registration methods. Consequently, a method using both monocular registration and stereoscopic registration elements have been proposed. Several other goals have been defined to fill the lack observed in other related vision-based augmented reality systems. The other goals are: to increase the registration depth of stereoscopic systems, to correct the point positions, to assure the coherence between the left and right registrations, to be robust to fast user's motion, to use a minimal number of points for the registration and to minimize the cost of the hardware.

An experimental system which aims to fulfil those goals has been developed. The system accomplishes three main tasks: marker extraction, registration, and optimization. Each task has been specified to satisfy some requirements in order to fulfil the defined goals. The marker extraction strategy is asked to run in real-time, to be robust to fast user's motion. Also, triangular markers are extracted by the system in order to perform the registration with three feature points as specified by the goal of using a minimal number of points. The three point based registration method is asked to be independent of the distance. Finally, the correction method is asked to perform a correction of the feature points of the marker extracted in the images. Also, the correction assures that the coherence between the left and right registration is met.

The developed marker extraction strategy uses color-based processing and extracts the markers inside the entire frames. Consequently, the system is able to extract the markers even if a fast user's motion occurs. To decrease the time spent for the marker extraction, the markers are detected using a hexagonal pattern that diminish the number of pixels processed in order to find the marker in the frames. Using colored region inserted in the blue triangular marker, each marker can be discriminated and oriented.

After, the feature points of each extracted marker are used to register the virtual object associated with the marker. The Finsterwalder's approach is used to solve the three point based inverse perspective projection. The multiple solutions of the left and the right frames are compared two by two using stereoscopic projections and geometric considerations. The best left-right solution is selected to perform the registration.

Once the best solution has been identified, the 2D positions of the marker features can be corrected based on the fact that the left and right solutions are supposed to be identical.

The correction is applied by projecting the left solution into the right image plane and the right solution into the left image plane. The positions obtained by projection are compared to the extracted 2D positions of the feature points. A correction is applied on each point position based on the positions obtained by the projection.

The developed system using the proposed marker extraction, the registration and the correction methods is evaluated. First, the percentage of successful extractions of the marker is calculated for three sequences containing respectively slow, normal and fast user's motions. The results observed with the proposed marker extraction strategy are compared to the results observed with the standard tracking method based on searching windows. The proposed method using hexagonal pattern gives better results compared to the standard tracking method for three principal reasons: the proposed method is robust to fast user's motion, the proposed method spends less time to extract the marker, and the frame rate resulting of the proposed method is constant.

Second, the fluctuations in position and the fluctuations in orientation observed with the proposed registration method is compared to the fluctuations observed for the standard stereoscopic registration method. Since the proposed registration method is not influenced by the distance and the stereoscopic registration is dramatically influenced by the distance, the proposed registration method is more stable than the stereoscopic method. However, significant incoherence can occur between both stereoscopic views of the augmented scene generated by the proposed method.

In a third time, the effect of the correction is evaluated. Using the stereoscopic registration method, the correction of the feature positions allows the system to compute successfully the 3D feature positions independently of the distance between the user's head and the features. Therefore, the correction is proven to be efficient. The resulting positions of the features are then more accurate. Consequently, the correction method improves the stability of the registration. Also, the correction assures that the coherence condition between the left and right registrations is met when using the proposed method.

All in all, the fluctuation data have been computed for four different registration methods: the fluctuations obtained with the stereoscopic method ("Snoc"), the fluctuations obtained with the proposed registration method ("Xnoc"), the fluctuations obtained with the proposed registration method after correction of the feature positions ("Xcor"), and the fluctuations obtained with the stereoscopic method after correction of the feature positions ("Scor"). The best registration results are observed with the proposed method after correction ("Xcor").

If the number and the complexity of the 3D objects to render are limited, the developed system succeeds to run in real-time on an 800MHz PC. The frame rate of the system is

not influenced by the size of the marker and the distance between the marker and the camera.

Also, the stability results obtained with the developed system are compared to the results obtained with an ARToolKit based system. Both systems have their own advantages and disadvantages. Since the ARToolKit registration is four points based, the ARToolKit performs more stable registration than the proposed method. However, the proposed method is slightly faster than the ARToolKit and shows better successful extraction results.

At last, a correction method using variable correction factors is implemented and the effect of the use of variable factors on the registration stability is studied. The use of variable factors limits the effect of the perturbations on the stability of the registration when significant extraction errors occur.

In conclusion, a stable three point based registration method for vision-based augmented reality using monocular and binocular vision with feature position correction have been presented in this thesis. A developed system using the proposed registration method and the correction method produces stable three point based registration. This has been successfully proven with experiments. Also, the registration produced is robust to fast user's motion. The system is able to run in real-time on a PC with relatively low specifications. In future works, the developed system would be updated to perform occlusion of the virtual objects from the scene information collected by stereoscopic vision-based computation. Also, four points based registration would be implemented in the system. The registration would be performed using three or four points according to the marker visibility. Also, new correction rules by developing more robust algorithm to determine the variable correction factors will be investigated and the effect of the correction method on monocular registration using 4 points will be evaluated.

References

- [ABB+01] R. T. Azuma, Y. Baillet, R. Behringer, S. Feiner, S. Julier and B. MacIntyre, "Recent advances in augmented reality", *IEEE Computer Graphics and Application*, Vol. 21, No. 6, pp. 34-47, 2001.
- [AHN+99] R. Azuma, B. Hoff, H. Neely III and R. Sarfaty, "A Motion-Stabilized Outdoor Augmented Reality System", *Proc. IEEE Virtual Reality 1999(VR'99)*, pp. 252-259, 1999.
- [Azu97] R. T. Azuma, "A survey of augmented reality", *Presence*, Vol. 6, No. 4, pp. 355-385, 1997.
- [Beh99] R. Behringer, "Registration for Outdoor Augmented Reality Applications Using Computer Vision Techniques and Hybrid Sensors", *Proc. IEEE Virtual Reality 1999(VR'99)*, pp. 244-251, 1999.
- [BFA+95] F. Betting, J. Feldar, N. Ayache and F. Devernay, "A new framework for fusing stereo images with volumetric medical images", *Proc. Conference on Computer Vision, Virtual Reality and Robotics in Medicine(CVRMed'95)*, pp. 30-39, 1995.
- [BFO92] M. Bajura, H. Fuchs and R. Ohbuchi, "Merging virtual objects with the real world: Seeing ultrasound imagery within the patient", *Proc. SIGGRAPH'92*, pp. 203-210, 1992.
- [BKP01] M. Billinghurst, H. Kato and I. Poupyrev, "Project in VR: The MagicBook - Moving Seamlessly between Reality and Virtuality", *IEEE Computer Graphics and Applications*, pp. 2-4, 2001.
- [BN95] M. Bajura and U. Neumann, "Dynamic Registration Correction in Video-Based Augmented Reality Systems", *Proc. IEEE Virtual Reality 2000(VR2000)*, pp. 52-60, 1995.
- [FTV01] V. Ferrari, T. Tuytelaars and L. Van Gool, "featureless Augmented Reality with a Real-time Affine Region Tracker", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 87-96, 2001.
- [HLO+91] R. M. Haralick, C.-N. Lee, K. Ottenberg and M. Nolle, "Analysis and Solutions of The Three Point Perspective Pose Estimation Problem", *Proc. CVPR'91*, pp. 592-598, 1991.

- [KBP+00] H. Kato, M. Billinghurst, I. Poupyrev, K. Imamoto, K. Tachibana, “Virtual object manipulation on a table-top AR environment”, *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 111-119, 2000.
- [KFT+00a] M. Kanbara, T. Fujii, H. Takemura and N. Yokoya, “A Stereo Vision-based Augmented Reality System with an Inertial Sensor”, *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 97-100, 2000.
- [KFT+00b] M. Kanbara, T. Fujii, H. Takemura and N. Yokoya, “A Stereo Vision-based Mixed Reality System with Natural Feature Point Tracking”, *Proc. 2nd Int. Sympo. on Mixed Reality(ISMR2001)*, pp. 56-63, 2001.
- [KIO00] T. Kobayashi, G. Inoue and Y. Ohta, “A Unified Linear Algorithm for a Novel View Synthesis and Camera Pose Estimation in Mixed Reality”, *Proc. IEEE Virtual Reality 2000(VR2000)*, pp. 263-270, 2000.
- [KIT+00] M. Kanbara, H. Iwasa, H. Takemura and N. Yokoya, “A Stereo Vision-based Augmented Reality System with a Wide Range Registration”, *Proc. 15th IAPR Int. Conf. on Pattern Recognition(ICPR2000)*, Vol. 4, pp. 147-151, 2000.
- [KKO+00] K. Kiyokawa, Y. Kurata and H. Ohta, “An optical see-through display for mutual occlusion of real and virtual environments”, *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 60-67, 2000.
- [KOT+00] M. Kanbara, T. Okuma, H. Takemura and N. Yokoya, “A Stereoscopic Video See-through Augmented Reality System Based on Real-time Vision-based Registration”, *Proc. IEEE Virtual Reality 2000(VR2000)*, pp. 255-262, 2000.
- [LB00] V. Lepetit and M. Berger, “Handling occlusion in augmented reality systems: A semi-automatic method”, *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 137-146, 2000.
- [Moh79] H. Moravec, “Visual Mapping by a Robot Rover”, *Proc. 6th International Joint Conference on Artificial Intelligence(IJCAI'79)*, pp. 599-601, 1979.
- [MC99] P. Milgram and H. Colquhoun Jr., “A Taxonomy of Real and Virtual World Display Intergration”, *Mixed Reality - Merging Real and Virtual Worlds*, Ohmsha & Springer-Verlag, pp. 5-30, 1999.

- [NIH01] J. Newman, D. Ingram and A. Hopper, "Augmented Reality in a Wide Area Sentient Environment", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 77-86, 2001.
- [NP98] U. Neumann and Jun Park, "Extendible Object-Centric Tracking for Augmented Reality", *Proc. IEEE Virtual Reality 1998(VR'98)*, pp. 148-155, 1998.
- [OKT+98] T. Okuma, K. Kiyokawa, H. Takemura and N. Yokoya, "An augmented reality system using a real-time vision based registration", *Proc. 14th IAPR Int. Conf. on Pattern Recognition(ICPR'98)*, Vol. 2, pp. 1226-1229, 1998.
- [OSI+01] Y. Ohta, Y. Sugaya, H. Igarashi, T. Ohtsuki and K. Taguchi, "Share-Z: Client/server depth sensing for see-through head-mounted displays", *Proc. 2nd Int. Sympo. on Mixed Reality(ISMR2001)*, pp. 64-72, 2001.
- [OSY+98] T. Ohshima, K. Satoh, H. Yamamoto and H. Tamura, "AR2 hockey: A case study of collaborative augmented reality", *Proc. IEEE Virtual Reality Annual International Symposium(VRAIS'98)*, pp. 14-18, 1998.
- [PYN98] J. Park, S. You and U. Neumann, "Extending Augmented Reality with Natural Feature Tracking",
- [RS01] G. Retmayr and D. Schmalstieg, "Mobile collaborative augmented reality", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 114-123, 2001.
- [SAY+01] K. Satoh, M. Anabuki, H. Yamamoto and H. Tamura, "A hybrid registration method for outdoor augmented reality", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 67-76, 2001.
- [SFZ00] G. Simon, A. W. Fitzgibbon and A. Zisserman, "Markerless tracking using planar structures in the scene", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 120-128, 2000.
- [SH00] Y. Seo and K. Hong, "Weakly calibrated video-based augmented reality: Embedding and rendering through virtual camera", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 129-136, 2000.
- [SHC+96] A. State, G. Hirota, D. T. Chen, W. F. Garrett and A. Livingston, "Superior augmented reality registration by integrating landmark tracking and magnetic tracking", *Proc. SIGGRAPH'96*, pp. 429-438, 1996.

- [SK01] D. Stricker and T. Kettenbach, "Real-time and featureless Vision-Based Tracking for Outdoor Augmented Reality Applications", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 189-190, 2001.
- [SKB+01] F. Sauer, A. Khamene, B. Bascle, L. Schimmang, F. Wenzel and S. Vogt, "Augmented reality visualization of ultrasound images: System description, calibration and features", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2001)*, pp. 30-39, 2001.
- [SLG+96] A. State, M. A. Livingston, W. F. Garrett, G. Hirota, M. C. Whitton, E. D. Pisano and H. Fuchs, "Technologies for Augmented Reality Systems: Realizing Ultrasound-Guided Needle Biopsies", *Proc. SIGGRAPH'96*, pp. 439-446, 1996.
- [SON01] K. Sawada, M. Okihara and S. Nakamura, "A wearable attitude measurement system using a fiber optic gyroscope", *Proc. 2nd Int. Sympo. on Mixed Reality(ISMR2001)*, pp. 35-39, 2001.
- [SWV+00] F. Sauer, F. Wenzel, S. Vogt, Y. Tao, Y. Genc and A. Bani-Hashemi, "Augmented Workspace: designing an AR testbed", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 47-53, 2000.
- [TYK01] H. Tamura, H. Yamamoto and A. Katayama, "Mixed reality: Future dreams seen at the border between real and virtual worlds", *IEEE Computer Graphics and Application*, Vol. 21, No. 6, pp. 64-70, 2001.
- [TYS+00] A. Takagi, S. Yamazaki, Y. Saito and N. Taniguchi, "Development of a Stereo Video See-through HMD for AR Systems", *Proc. IEEE/ACM Int. Sympo. on Augmented Reality(ISAR2000)*, pp. 68-77, 2000.
- [UK95] M. Uenohara and T. Kanade, "Vision-based object registration for real-time image overlay", *Proc. Conference on Computer Vision, Virtual Reality and Robotics in Medicine(CVRMed'95)*, pp. 13-22, 1995.
- [YN01] S. You and U. Neumann, "Fusion of vision and gyro tracking for robust augmented reality registration", *Proc. IEEE Virtual Reality 2001(VR2001)*, pp. 71-78, 2001.
- [YNA99] S. You, U. Newmann and R. T. Azuma, "Hybrid inertial and vision tracking for augmented reality registration", *Proc. IEEE Virtual Reality 1999(VR'99)*, pp. 260-267, 1999.

- [YSY00] Y. Yokokohji, Y. Sugawara and T. Yoshikawa, "Accurate image overlay on see-through head-mounted displays using vision and accelerometers", *Proc. IEEE Virtual Reality 2000(VR2000)*, pp. 247-254, 2000.
- [YTO+99] N. Yokoya, H. Takemura, T. Okuma and M. Kanbara, "Stereo Vision Based Video See-through Mixed Reality", *Mixed Reality - Merging Real and Virtual Worlds*, Ohmsha & Springer-Verlag, pp. 131-145, 1999.
- [ARTK] "Shared Space / ARToolKit Download Page", http://www.hitl.washington.edu/research/shared_space/download/.

List of publications

Journal papers

S. Vallerand, M. Kanbara, N. Yokoya, "Three Point Based Registration for Augmented Reality Using Binocular and Monocular Vision", *IEICE Trans. Information and System*, 2004. [accepted]

S. Vallerand, M. Kanbara, N. Yokoya, "Optimisation de l'alignement geometrique par correction de positions 2D pour les systemes binoculaires de realite augmentee", *Canadian Journal of Electrical and Computer Engineering*, 2004. [submitted]

International conferences

S. Vallerand, M. Kanbara, N. Yokoya, "Vision-Based Registration for Augmented Reality System Using Monocular and Binocular Vision", *Proc. SPIE: Electronic Imaging (Science and Techonolgy)*, Vol. 5006, pp 487-498, 2003.

S. Vallerand, M. Kanbara, N. Yokoya, "Binocular Vision-Based Augmented Reality System with an Increased Registration Depth Using Dynamic Correction of Feature Positions", *Proc. IEEE Virtual Reality 2003 (VR2003)*, pp 271-272, 2003.

S. Vallerand, M. Kanbara, N. Yokoya, "Amélioration de l'alignement en réalité augmentée", *Journée Science et Technologie du Japon*, pp. 55-56, 2002. (in French)

Domestic conferences

S. Vallerand, M. Kanbara, N. Yokoya, "Video See-through Augmented Reality System With Tracking Regions", *Proceedings of the 6th Virtual Reality Society of Japan*, pp 465-468, 2001.

S. Vallerand, M. Kanbara, N. Yokoya, "Video See-through Augmented Reality With Tracking Regions", *Technical Report of IEICE*, PRMU2001-227, pp 41-46, 2002.

Other non-related publications

S. Vallerand, A. Darabi, X. Maldague, "Defect Detection in Pulsed Thermography: A Comparison of Kohonen and Perceptron Neural Networks", *SPIE: Thermosense XXI (SPIE: Society of Photo-Optical Instrumentation Engineers)*, R.N. Wurzbach, D.D. Burleigh eds., vol. 3700, pp 20-25, 1999.

C. Fortin, F. Galmiche, S. Vallerand, X. Maldague, "Surface defect detection on thin metal sheets", *5th Conf. on QCAV 1999 (Quality Control by Artificial Vision)*, Trois-Rivières, pp 271-273, 1999.

S. Vallerand, X. Maldague, "Defect Characterization in Pulsed Thermography: A Statistical Method Compared with Kohonen and Perceptron Neural Networks", *Nondestructive Testing and Evaluation International*, 33[5]: 307-315, 2000.

F. Galmiche, S. Vallerand, X. Maldague, "Wavelet Transform Applied to Pulsed Phased Thermography", *V Workshop on Advances in Infrared Technology (5th AITA)*, E. Grinzato, P.G. Bison, A. Mazzoldi eds, 117-122, CNR, 2000.

F. Galmiche, S. Vallerand, X. Maldague, "Pulsed Phase Thermography with the Wavelet Transform", *EReview of Progress in Quantitative NDE*, D.O. Thompson et D.E. Chimenti eds., Am. Institute of Physics, 19A:609-615, 2000.

Appendix A : Finsterwalder's manipulations and equations

The three point space resection problem, or three point inverse perspective projection, asks to retrieve the 3D positions of three points from the known perspective projections of those points. With the condition that the three points constitute the vertices of a known triangle in 3D space, the 3D positions of the three vertices can be determined. The Figure A.1 illustrates the three point space resection problem.

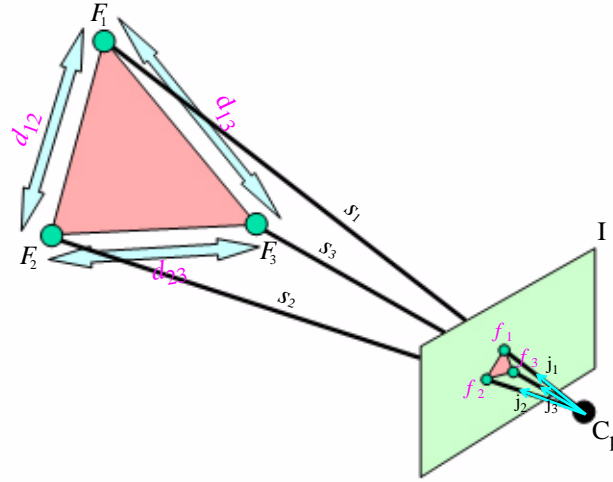


Figure A.1 Geometry of the three point space resection problem.

The length of the three sides of a known triangle are d_{12} , d_{13} and d_{23} . The vectors j_1 , j_2 and j_3 are the unit vectors pointing from the center of perspective C_p to the perspective projection f_1 , f_2 and f_3 of the triangle vertices F_1 , F_2 and F_3 . Consequently, the angles at the center of perspective are given by $\alpha = \text{acos}(j_2 \cdot j_3)$, $\beta = \text{acos}(j_1 \cdot j_3)$ and $\gamma = \text{acos}(j_1 \cdot j_2)$.

To solve the three point space resection problem, the distance from the center of perspective C_p to each of the three vertices s_1 , s_2 and s_3 must be retrieved. By the law of cosines, the three point space resection problem is written as follow:

$$s_2^2 + s_3^2 - 2s_2s_3\cos\alpha = d_{23}^2 \quad (\text{A.1})$$

$$s_1^2 + s_3^2 - 2s_1s_3\cos\beta = d_{13}^2 \quad (\text{A.2})$$

$$s_1^2 + s_2^2 - 2s_1s_2\cos\gamma = d_{12}^2 \quad (\text{A.3}).$$

By substituting s_2 by us_1 and s_3 by vs_1 , the system is then rewritten as:

$$s_1^2 = \frac{d_{23}^2}{u^2 + v^2 - 2uv\cos\alpha} = \frac{d_{13}^2}{1 + v^2 - 2uv\cos\beta} = \frac{d_{12}^2}{1 + u^2 - 2uv\cos\gamma} \quad (\text{A.4}).$$

From which, the following system of two equations is obtained:

$$u^2 + \frac{d_{13}^2 - d_{23}^2}{d_{13}^2} v^2 - 2uv\cos\alpha + \frac{2d_{23}^2}{d_{13}^2} v\cos\beta - \frac{d_{23}^2}{d_{13}^2} = 0 \quad (\text{A.5})$$

$$u^2 - \frac{d_{12}^2}{d_{13}^2} v^2 + 2v\frac{d_{12}^2}{d_{13}^2}\cos\beta - 2u\cos\gamma + \frac{d_{13}^2 - d_{12}^2}{d_{13}^2} = 0 \quad (\text{A.6}).$$

Finsterwalder multiplies the equation A.5 by λ and adds the result to the equation A.6 to produce:

$$Au^2 + 2Buv + Cv^2 + 2Du + 2Ev + F = 0 \quad (\text{A.7})$$

where the coefficients are:

$$A = 1 + \lambda, \quad C = \frac{d_{13}^2 - d_{23}^2}{d_{13}^2} - \lambda \frac{d_{12}^2}{d_{13}^2}, \quad E = \frac{d_{23}^2}{d_{13}^2} + \lambda \frac{d_{12}^2}{d_{13}^2},$$

$$B = -\cos\alpha, \quad D = -\lambda\cos\gamma, \quad F = -\frac{d_{23}^2}{d_{13}^2} + \lambda \frac{d_{13}^2 - d_{12}^2}{d_{13}^2}.$$

The equation A.7 is considered by Finsterwalder as a quadratic equation in v . The solution of the equation is consequently:

$$v = \frac{-(Bu + E) \pm \sqrt{(B^2 - AC)u^2 + 2(BE - CD)u + E^2 - CF}}{C} \quad (\text{A.8}).$$

Then, Finsterwalder searches for the value λ which makes $(B^2 - AC)u^2 + 2(BE - CD)u + E^2 - CF$ a perfect square in order to express v as a first order polynomial in terms of u . After, this first order polynomial is substituted back into the equation A.5 or the equation A.6 to obtain a quadratic solution that can be solved to determine the value of u . For one value of λ , four solutions are produced since there are two first order expressions for v and two solutions for u exists when each of the two order expressions is substituted back.

Therefore, the value of λ which produces a perfect square is needed. In others words, the value of λ must satisfy where S and T are real number:

$$(Su + T)^2 = (B^2 - AC)u^2 + 2(BE - CD)u + E^2 - CF \quad (\text{A.9}).$$

Consequently, the following equation must be solved:

$$G\lambda^3 + H\lambda^2 + I\lambda + J = 0 \quad (\text{A.10}).$$

The coefficients G , H , I and J are:

$$\begin{aligned} G &= d_{12}^2(d_{12}^2 \sin^2 \beta - d_{13}^2 \sin^2 \gamma), & J &= d_{23}^2(d_{23}^2 \sin^2 \beta - d_{13}^2 \sin^2 \alpha), \\ H &= \frac{d_{13}^2(d_{13}^2 - d_{23}^2) \sin^2 \gamma + d_{12}^2(d_{12}^2 + 2d_{23}^2) \sin^2 \beta}{+ 2d_{13}^2 d_{12}^2 (\cos \alpha \cos \beta \cos \gamma - 1)}, \\ I &= \frac{d_{13}^2(d_{13}^2 - d_{12}^2) \sin^2 \alpha + d_{23}^2(d_{23}^2 + 2d_{12}^2) \sin^2 \beta}{+ 2d_{23}^2 d_{13}^2 (\cos \alpha \cos \beta \cos \gamma - 1)}. \end{aligned}$$

Once the three roots of equation A.10 have been founded, the coefficient A , B , C , D , E and F are determined. Therefore, the v of equation A.8 is obtained in terms of the variable u for each of the three roots. For each root, two first order polynomial equations for v are produced. Consequently, a total of six polynomial solutions of v have been produced. Substituting back a polynomial solution of v in into the equation A.5 or the equation A.6, two values are found for u . That is, 12 values are found for u . The 12 corresponding values of v can be obtained by substituting back the values of u in the polynomial solutions of v . Once the values of u and v have been determined, the distance s_1 is determined using equation A.4. Finally, the distance s_2 and s_3 are determined using $s_2 = us_1$ and $s_3 = vs_1$.

Appendix B : Roots of a third degree polynomial equation

A third degree polynomial is a cubic equation usually written on the following form:

$$Ax^3 + Bx^2 + Cx + D = 0 \quad (\text{B.1}).$$

Let the roots be denoted by x_1 , x_2 and x_3 . The cubic then has the form:

$$a(x - x_1)(x - x_2)(x - x_3) = 0 \quad (\text{B.2}).$$

By multiplying out, we obtain:

$$ax^3 - a(x_1 + x_2 + x_3)x^2 + a(x_1x_2 + x_1x_3 + x_2x_3)x - ax_1x_2x_3 = 0 \quad (\text{B.3})$$

which is the cubic equation on his usual form.

The first operation is to depress the cubic equation ($B \Rightarrow 0$) and to normalize the depressed cubic ($A \Rightarrow 1$). In other words, the cubic equation is transformed on the form:

$$y^3 + Qy + R = 0 \quad (\text{B.4}).$$

By depressing the cubic, the sum of the roots is now zero ($y_1 + y_2 + y_3 = 0$). To produce the depressed form of a cubic equation, Tartaglia's substitution is performed. Therefore,

$$x = y - \frac{B}{3A} \quad (\text{B.5})$$

is substituted in equation B.1 in order to obtain the depressed equation:

$$Ay^3 - \left(C - \frac{B^2}{3A}\right)y + \left(D + \frac{2B^2}{27A^2} - \frac{BC}{3A}\right) = 0 \quad (\text{B.6}).$$

Then, the depressed equation is normalized.

$$y^3 + \left(\frac{3AC - B^2}{3A^2}\right)y + \left(\frac{27A^2D + 2B^2 - 9ABC}{27A^3}\right) = 0 \quad (\text{B.7})$$

Consequently, the terms Q and R of equation B.4 are respectively:

$$\frac{3AC - B^2}{3A^2} \quad \text{and} \quad \frac{27A^2D + 2B^2 - 9ABC}{27A^3} \quad (\text{B.8}).$$

Now, we set:

$$\delta^2 = \frac{-Q}{3} \text{ and } h^2 = 2\delta^6 \quad (\text{B.9}).$$

The quantity h^2 plays a major role since the quantity $R^2 - h^2$ tells us how many real roots the cubic will have. If $R^2 - h^2 > 0$, the cubic will have one real root and two imaginary roots. On the other side, if $R^2 - h^2 < 0$, the cubic will have three real roots.

Three real roots ($R^2 - h^2 < 0$)

Transcendental functions are used to find the roots of the cubic equation. The trigonometric substitution

$$y = 2\delta \cos \theta \quad (\text{B.10})$$

is substituted in the cubic equation B.4. The substitution yields

$$8\delta^3 \cos^3 \theta + 2Q\delta \cos \theta + R = 0 \quad (\text{B.11}).$$

Using $\delta^2 = -\frac{Q}{3}$, the equation B.11 can be rewritten as:

$$2\delta^3(4\cos^3 \theta - 3\cos \theta) + R = 0 \quad (\text{B.12}).$$

Next, observing that deMoivre's formula is

$$4\cos^3 \theta - 3\cos \theta = \cos(3\theta) \quad (\text{B.13})$$

and remembering that $h = 2\delta^3$, the cubic equation can be written as:

$$\cos(3\theta) = -\frac{R}{h} \quad (\text{B.14}).$$

The first root of equation B.14 is given by:

$$\theta_1 = \arccos\left(-\frac{R}{h}\right) \quad (\text{B.15}),$$

producing the first root of the depressed cubic equation

$$y_1 = 2\delta \cos \theta_1 \quad (\text{B.16}).$$

The other two roots are given by

$$y_2 = 2\delta \cos\left(\frac{2\pi}{3} - \theta_1\right) \text{ and } y_3 = 2\delta \cos\left(\frac{2\pi}{3} + \theta_1\right) \quad (\text{B.17}).$$

From equation B.5, the roots of the original cubic equation are then:

$$x_1 = y_1 - \frac{B}{3A}, x_2 = y_2 - \frac{B}{3A} \text{ and } x_3 = y_3 - \frac{B}{3A} \quad (\text{B.18}).$$

One real and two imaginary roots ($R^2 - h^2 > 0$)

We are left with the depressed cubic equation given by equation B.4. To solve the depressed cubic, s and t will be find so that:

$$3st = Q \quad (\text{B.19})$$

$$t^3 - s^3 = R \quad (\text{B.20}).$$

It turns out that $y = t^3 - s^3$ will be a solution of the depressed cubic. Solving the equation B.19 for s and substituting into equation B.20 yields:

$$\left(\frac{Q}{3t}\right)^3 - t^3 + R = 0 \quad (\text{B.21}).$$

After simplification, this turns into a “tri-quadratic” equation

$$t^6 - Rt^3 - \frac{Q^3}{27} = 0 \quad (\text{B.22})$$

which using the substitution $u = t^3$ becomes the quadratic equation

$$u^2 - Ru - \frac{Q^3}{27} = 0 \quad (\text{B.23}).$$

Using the quadratic formula, we obtain a value for u . Then, t is given by

$$t = \sqrt[3]{u} \quad (\text{B.24})$$

or, equivalently, by

$$t = \exp\left(\frac{\ln u}{3}\right) \quad (\text{B.25}).$$

Afterward, we obtain s from the equation B.19 and we find the real root of the cubic equation using $y_1 = t^3 - s^3$. The two imaginary roots y_2 and y_3 can be retrieved with the quadratic formula after dividing the cubic equation by $(y - y_1)$.

Again, from equation B.5, the roots of the original cubic equation are then:

$$x_1 = y_1 - \frac{B}{3A}, x_2 = y_2 - \frac{B}{3A} \text{ and } x_3 = y_3 - \frac{B}{3A} \quad (\text{B.26}).$$

Appendix C : Stereoscopic equations

Figure C.1 illustrates the geometry of the stereoscopic vision.

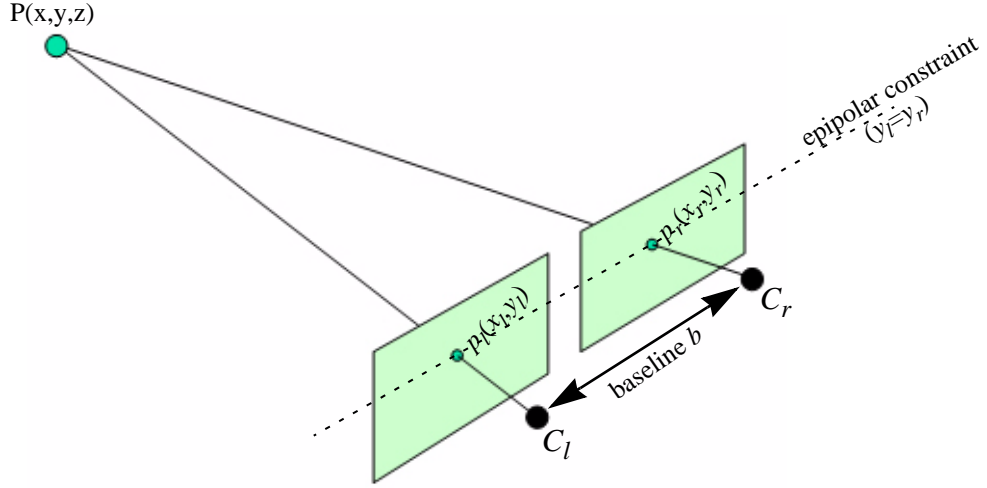


Figure C.1 Geometry of the stereoscopic vision.

The origin of the global camera coordinate system \mathcal{C} is placed at the middle point between the two camera centers of perspective \mathcal{C}_l and \mathcal{C}_r . The axis \vec{C}_x of the global camera coordinate system is set along the baseline of the cameras and the axis \vec{C}_z is set to the direction parallel to the optical axes of the cameras. The position of the left camera center of perspective \mathcal{C}_l and the position of the right camera center of perspective \mathcal{C}_r in the global camera coordinate system are then:

$$\mathcal{C}_l((c_{lx}, c_{ly}, c_{lz})) = \left(-\frac{b}{2}, 0, 0\right) \quad (\text{C.1})$$

$$\mathcal{C}_r((c_{rx}, c_{ry}, c_{rz})) = \left(\frac{b}{2}, 0, 0\right) \quad (\text{C.2}).$$

Consequently, the 3D position P_l in the global camera coordinate system of a pixel $p_l(x_l, y_l)$ from the left image is:

$$P_l(p_{lx}, p_{ly}, p_{lz}) = \left(-\frac{b}{2} + x_l, y_l, f\right) \quad (\text{C.3})$$

and the 3D position P_r of a pixel $p_r(x_r, y_r)$ from the right image is:

$$P_r(p_{rx}, p_{ry}, p_{rz}) = \left(\frac{b}{2} + x_r, y_r, f\right) \quad (\text{C.4})$$

where f is the focal length of the cameras.

The goal is to retrieve the 3D position of the point which has the perspective projections p_l and p_r .

First, the equation of the line joining the points C_l and P_l is:

$$\frac{x - p_{lx}}{c_{lx} - p_{lx}} = \frac{y - p_{ly}}{c_{ly} - p_{ly}} = \frac{z - p_{lz}}{c_{lz} - p_{lz}} \quad (\text{C.5}).$$

Similarly, the equation of the line joining the point C_r and P_r is:

$$\frac{x - p_{rx}}{c_{rx} - p_{rx}} = \frac{y - p_{ry}}{c_{ry} - p_{ry}} = \frac{z - p_{rz}}{c_{rz} - p_{rz}} \quad (\text{C.6}).$$

The 3D position $P(x, y, z)$ of the point observed in both camera images is located at the intersection between the two lines defined by equation C.5 and by equation C.6. As a result, to retrieve the 3D position of the point, the following system of equations must be solved:

$$\frac{x + \frac{b}{2}}{x_l} = \frac{y}{y_l} = \frac{z}{f} \quad (\text{C.7})$$

$$\frac{x - \frac{b}{2}}{x_r} = \frac{y}{y_r} = \frac{z}{f} \quad (\text{C.8}).$$

The component x of the point $P(x, y, z)$ position is obtained by solving the following equation:

$$x_r \left(x + \frac{b}{2} \right) = x_l \left(x - \frac{b}{2} \right) \quad (\text{C.9}).$$

The solution for x is then:

$$x = \frac{b(x_l + x_r)}{2(x_l - x_r)} \quad (\text{C.10}).$$

The component z is produced by solving the next system of equations:

$$\text{from equation C.7 } \frac{x + \frac{b}{2}}{x_l} = \frac{z}{f} \quad (\text{C.11}),$$

$$\text{from equation C.8 } \frac{x - \frac{b}{2}}{x_r} = \frac{z}{f} \quad (\text{C.12}).$$

Solving the system produces the following solution for z :

$$z = \frac{fb}{(x_l - x_r)} \quad (\text{C.13}).$$

Finally, the next system of equations is solved to find the component y :

$$\text{from equation C.7 } x = y \frac{x_l}{y_l} - \frac{b}{2} \quad (\text{C.14}),$$

$$\text{from equation C.8 } x = y \frac{x_r}{y_r} + \frac{b}{2} \quad (\text{C.15}).$$

The equation for y is then:

$$y = \frac{b}{\left(\frac{x_l}{y_l} - \frac{x_r}{y_r}\right)} \quad (\text{C.16}),$$

simplified as:

$$y = \frac{b(y_l + y_r)}{2(x_l - x_r)} \quad (\text{C.17})$$

if the epipolar constraint which stipulates that theoretically $y_l = y_r$ is verified.

Appendix D : Straight line fitting using least square minimization.

Let's P a set of I points (x_i, y_i) where $i \in [1, I]$. The goal is to find the equation of the straight line which best fit the set of points.

The equation of a straight line may be written as:

$$y = ax + b \quad (\text{D.1}).$$

Consequently, two variables have to be computed in order to define the straight line: the gradient a and the y-intercept b .

For a point i of the set P , the error E_i observed for this point is given by:

$$E_i = y_i - (ax_i + b) \quad (\text{D.2}).$$

The best straight line will be the straight line which minimizes the squared error on all the points of P . The error is squared in order to obtain a positive error value for each point. Otherwise, a negative and positive error values may cancel each other and the minimization will be mismatched. As a result, the global error E_g is given by:

$$E_g = \sum_{i=1}^I E_i^2 = \sum_{i=1}^I (y_i - (ax_i + b))^2 \quad (\text{D.3}).$$

Minimizing the error E_g means that:

$$\frac{\partial E_g}{\partial a} = 0 \quad \text{and} \quad \frac{\partial E_g}{\partial b} = 0 \quad (\text{D.4}).$$

Applying the partial derivatives on equation D.3 yields:

$$\frac{\partial E_g}{\partial a} = \sum_{i=1}^I (2x_i^2 a + 2x_i b - 2x_i y_i) \quad (\text{D.5})$$

and

$$\frac{\partial E_g}{\partial b} = \sum_{i=1}^I (2b + 2x_i a - 2y_i) \quad (\text{D.6}).$$

Rewriting equation D.5 and equation D.6 gives the following system of two equations:

$$\left. \sum_{i=1}^l x_i^2 \right\} a + \left. \left(\sum_{i=1}^l x_i \right) \right\} b = \sum_{i=1}^l x_i y_i \quad (\text{D.7}),$$

$$\left. \sum_{i=1}^l x_i \right\} a + l b = \sum_{i=1}^l y_i \quad (\text{D.8}).$$

Solving the system of equations, the direct solutions of a and b are respectively given by:

$$a = \frac{\left(\sum_{i=1}^l x_i \right) \left(\sum_{i=1}^l y_i \right) - l \left(\sum_{i=1}^l x_i y_i \right)}{\left(\sum_{i=1}^l x_i \right) \left(\sum_{i=1}^l x_i \right) - l \left(\sum_{i=1}^l x_i^2 \right)} \quad (\text{D.9})$$

and

$$b = \frac{\left(\sum_{i=1}^l y_i \right) - \left(\sum_{i=1}^l x_i \right) a}{l} \quad (\text{D.10}).$$

Appendix E : The HSV color system.

The Hue/Saturation/Value model was created by A. R. Smith in 1978. The major characteristic of the HSV color space is that both intensity (V) and hue (H) informations are disjoint. The hue value H which classes the color as red, yellow, green, blue, or an intermediate between any contiguous pair of these colors runs from 0° to 360° . The saturation S which ranges from 0.0 to 1.0 is the degree of strength or purity. The purity of a color is how much white is added to the color, so $S=1.0$ makes the purest color (no white). Brightness value V also ranges from 0.0 to 1.0, where 0.0 gives black. All in all, the HSV color space is a hexcone as illustrated in Figure E.1.

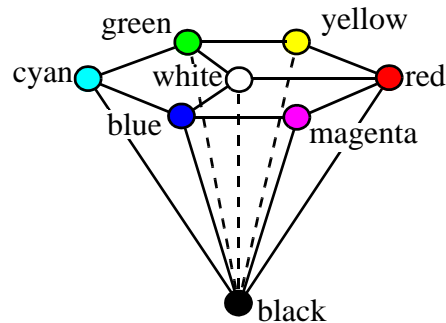


Figure E.1 Illustration of the HSV color space.

No matrix transformation directly performs the transformation between the RGB color space to the HSV color space. The algorithm performing the transformation from RGB components to the HSV components is presented in the Table E.1.

Table E.1: RGB to HSV transformation algorithm

```
variable r // red component
variable g // green component
variable b // blue component
variable h // hue component
variable s // saturation component
variable v // brightness value component

variable min = MINIMUM BETWEEN r, g, b
variable max = MAXIMUM BETWEEN r, g, b

variable delta = max - min

// brightness value
v = max

// saturation
IF max = 0
    s = 0
    h = UNDEFINED
    QUIT
ELSE
    s = delta / max

// hue
IF r = max // between yellow & magenta
    h = (g - b) / delta
ELSE IF g = max // between cyan & yellow
    h = 2 + (b - r) / delta
ELSE // between magenta & cyan
    h = 4 + (r - g) / delta

h = h * 360 // in degree

IF h < 0 // if hue is negative
    h = h + 360

QUIT
```