3-D Vision for an Autonomous Underwater Vehicle

Matthew Bryant

June 2000

Supervisors
Dr David Wettergreen
Mr Samer Abdallah
Dr Alex Zelinsky

A thesis submitted for the degree of
Bachelor of Engineering
Australian National University

Abstract

To navigate and explore underwater environments, an Autonomous Underwater Vehicle (AUV) must be capable of computing properties of a 3-dimensional environment in real time. This thesis details the development of a 3-D vision system for this purpose.

The Australian National University is developing a AUV named Kambara, equipped with two fixed cameras and a pan/tilt/zoom camera. All three cameras must be calibrated to establish their geometric and lens projection properties. Standard calibration algorithms are challenged by the behaviour of light in underwater environments, in particular the reduction of image contrast. A robust calibration strategy, based on invariant indexing, has been developed to overcome these underwater imaging problems. The algorithm has been demonstrated to improve upon the reliability of standard calibration algorithms by up to 80%.

Evaluating the accuracy of camera calibration is a challenging task due to a shortage of ground truth for comparison. A methodology for accuracy evaluation has been developed to overcome this problem, judging calibration accuracy by the accuracy of object dimension estimates. Applying this methodology to the evaluation of Kambara's calibration system found calibration was sufficiently accurate for range estimation of targets within 3 metres with 95% accuracy.

The suitability of Kambara's zoom lens for 3-D applications has been investigated. It was found that the zoom lens is well modelled by polynomials, but that calibration of the lens is not practical over the entire zoom range, which consequently limits the zoom capabilities for 3-D vision. The investigation also found that that zoom lens model of the camera is too complex to be used in conjunction with pan and tilt for real-time vision applications

One of Kambara's objectives is to track dynamic 3-D targets. A Feature Tracker software module has been designed for this purpose. The module has been carefully designed to ensure flexibility for future research. Included in the design is a robust range estimation algorithm useable with arbitrarily oriented cameras. The module has successfully been tested to track a target with the pan/tilt/zoom camera.

Acknowledgments

I am very grateful for the quality of supervision in this project. I would like to thank David Wettergreen for the many hours spent answering my questions, discussing ideas, reviewing my work, and for the quality of each of these contributions. Thanks also to Samer Abdallah for his advice and perceptive comments, and Alex Zelinsky for his advice and constant encouragement.

I consider myself lucky to have worked with the gifted and helpful Kambara project team, and am thankful for their tireless commitment and willingness to identify and solve technical problems. I would like specifically like to thank:

- Chris Gaskett, for always being ready to share his depth and breadth of technical know-how;
- Chanop Silpa-Anan, for his patience, which I often tested during our collaborative research efforts;
- Harley "Cable Guy" Truong, for his technical wizardry, advice, and company;
- Peter Brown, for answering many stupid questions before I started asking intelligent ones;
- Luke Fletcher, for his ready assistance with technical problems.
- Ian Fitzgerald, for his willingness to help me get started with the project.

I would also like to thank my fellow Honours students for their company, their ideas, and their encouragement. I specifically must thank:

- Seungjun Oh, foremost for his company during long nights in the robotics lab, and for always offering to drive me home;
- Jordan Cvetznovski, for helping keep the challenges of my project in perspective;
- Sunil Rao, for humouring me with many a game of chess.

I am indebted to my family for their support and patience, and particularly grateful for the reviewing expertise of Pauline and Kevin Bryant. Finally, I would like to thank Beth Jennings for her endless patience and constant support.

Table of Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1 Introduction

The Australian National University is currently developing an Autonomous Underwater Vehicle (AUV), named Kambara, for the task of underwater exploration and observation. To autonomously navigate and explore underwater environments, Kambara must be capable of computing geometric and dynamic properties of a three-dimensional environment in real time. This thesis describes the computer vision approach that has been taken to develop such capabilities.

Computer vision techniques have been used extensively for a variety of surface vehicle applications. We wish to establish whether these same standard techniques can be successfully applied in underwater environments, where light transmittance and image quality are considerably different.

Figure 1.1: Kambara

The following section introduces the standard computer vision techniques used for the visual guidance of autonomous vehicles, and discusses the properties of underwater vision which challenge these techniques. Section 1.2 briefly outlines the vision components used by Kambara, followed by a summary and outline of the rest of this thesis.

1.1 3-D Computer Vision for an AUV

Computer vision is an extensive research area which has spawned many different techniques for a wide variety of applications. The common objective of these techniques is to extract 3-D information from digital images captured by one or more cameras [Trucco, 1998].

For the navigation of an autonomous vehicle, the 3-D information we require is the relative position of objects in the vehicle's environment. The traditional computer vision techniques used for such applications are known as *range estimation* and *feature tracking*; together they allow the real-time tracking of an object's trajectory through space.

1.1.1 Feature Tracking

An autonomous vehicle needs to keep track of moving objects in its environment — failing to do so may lead to collision. This requires tracking the object's projection through a sequence of images. *Correlation-based* feature tracking involves taking a small image window of the object projection, called a *template*, and using it to search successive image frames. This approach is challenged by the fact that as the position and orientation of the object changes relative to the camera, the object's projection will gradually deviate more and more from the original template.

1.1.2 Range Estimation

Range estimation is the technique of using stereo images of an object to estimate the object's 3-D position. The first step of range estimation is to accurately locate the projection of the object in the left and right stereo images. This is a similar problem to feature tracking, only now the feature is being matched between stereo images instead of sequential images from the same camera. This can be challenging, as the left and right projections are often considerably different, depending on the relative position and orientation of the stereo cameras.

The second step is to determine how the pixel coordinates of the object in each image correspond to 3-D coordinates. This mapping can be mathematically calculated using information characterising the stereo cameras. The task of characterising the cameras is known as camera calibration.

1.1.3 Stereo Camera Calibration

Most range estimation algorithms require a knowledge of how image pixel coordinates correspond to point coordinates in 3-D space. The purpose of *camera calibration* is to determine a set of *camera parameters* which defines this correspondence.

Every camera has two types of parameters: intrinsic and extrinsic. The internal workings of a camera are characterised by a set of *intrinsic parameters*, specifying the internal camera geometry and dimensions. *Extrinsic parameters* define the position and the orientation of a camera relative to a world reference frame. A stereo pair of cameras has an associated set of *stereo extrinsic parameters*, defining the relative position and orientation of the cameras.

The stereo extrinsic parameters change every time the cameras are repositioned, and must be recalibrated every time the cameras are adjusted. The intrinsic parameters are independent of position and orientation, but vary across differing light mediums. The focal length of camera, for example, is greater in an underwater environment than for an air medium.

1.1.4 Underwater Vision

The behaviour of light under water challenges the application of 3-D vision techniques to AUVs. The most significant problems that must be overcome are underwater visibility, and the quality of underwater imaging. Wide angle camera lenses significantly reduce these problems, and are standard for underwater vision applications.

1.1.4.1 Visibility

Water limits the frequency spectrum of transmitted light, by absorbing low frequency light (e.g. red, yellow) more readily than high frequency light (e.g. blue, green). Furthermore, the bandwidth of transmitted light decreases with water depth. This severely limits the extraction of colour information from underwater environments, and therefore restricts the use of many standard image processing algorithms.

The light spectrum that is not absorbed tends to diffuse quickly, scattered by tiny suspended particles. This reduces the penetration of light, with the intensity of a beam of light decreasing exponentially with distance [Glover et al., 1977].

1.1.4.2 Image Quality

Suspended underwater particles also compromise the quality of camera images. The scattering effect decreases image contrast, which is widely used as an information source by many image processing algorithms.

The diffusion of light by suspended particles also tends to make pictures blurry.

1.1.4.3 Lens Properties

Wide angle lenses tend to be used under water because having a wider field of view allows a camera to get closer to the subject than standard lens. This therefore minimises the light transmittance and colour loss problems which increase with distance.

These benefits come at the cost of increased image distortion. Wide angle lenses cause elongation of objects on the edges of the image, and curvature of straight lines.

1.2 Kambara's Vision System

Kambara's vision system is comprised of three cameras and a digitizer. Stereo vision is accomplished using two Pulnix wide angle lens cameras, housed in moveable waterproof containers mounted on the Kambara frame [Reynolds, 1998], as shown in Figure 1.2. A Sony pan/tilt/zoom camera is mounted in a watertight enclosure, capturing images for Kambara's user interface [McPherson, 1999]. (See Appendix B for the camera manufacturer's specifications.) An Imagenation framegrabber multiplexes the three camera signals, delivering image frames to the onboard computer.

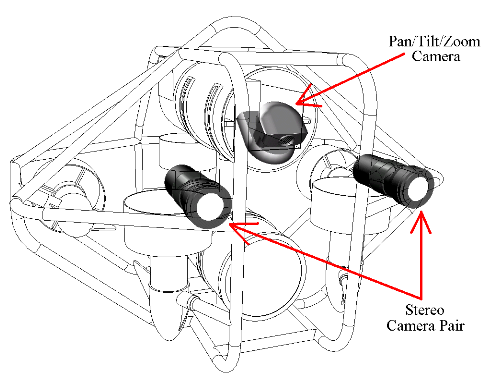


Figure 1.2: Kambara's camera suite

1.3 Project Objectives

The overall goal of this project is to contribute to the development of Kambara's vision system, and in doing so contribute to the 3-D computer vision discipline. Four specific objectives are made the focus of this project:

1) **Improve the robustness of Kambara's camera calibration system.** Reliability was the biggest problem left unsolved from previous development of a calibration system. Since calibration is a prerequisite to range estimation, this problem must be solved before further vision research can proceed.

2) **Evaluate the accuracy of Kambara's calibration system.** A thorough investigation into the accuracy of the calibration system has not previously been possible, since reliability problems have prevented extensive calibration.

3) **Investigate the calibration of Kambara's automated zoom lens camera.** The manoeuvrability of the Sony EVI-D30 pan/tilt/zoom camera could potentially be used for feature tracking over a wider field of view. The first step towards this goal is to investigate the practical issues of calibrating Kambara's zoom lens.

4) **Design and develop Kambara's Feature Tracking module.** The objective is to lay the foundations for future feature tracking development, by developing a flexible, extendable, and conceptually simple design.

1.4 Thesis Outline

Chapter 2 outlines the underlying theory of camera calibration, providing the background for the rest of the thesis. The conventional mathematical models used to describe cameras are explained, and the standard algorithms used to calibrate these models. Underwater environments present challenges to the reliability and accuracy of these algorithms. The relevant problems are examined together with the impact these have had on the previous development of a camera calibration system.

Chapter 3 describes a robust approach to underwater calibration, using invariant indexing to overcome the underwater unreliability of image *point detection*, one of the cornerstones of standard calibration algorithms. The concepts and algorithms of this approach are outlined in some detail, followed by an experimental evaluation of the robustness of this approach.

Having examined the reliability issues of underwater calibration, Chapter 4 moves on to the issue of calibration accuracy. The difficulties of evaluating calibration accuracy are examined, and a methodology developed to overcome these difficulties is outlined. This methodology was used to evaluate Kambara's calibration accuracy, and the results from these experiments are presented and discussed.

Chapter 5 investigates the suitability of Kambara's zoom lens camera to 3-D vision applications, by examining the feasibility of zoom lens calibration. A mathematical model describing a zoom lens camera is outlined, and previous work in calibrating this model is discussed. These concepts are then applied to the calibration of Kambara's zoom lens camera, and the success of this approach is evaluated.

Chapter 6 outlines the design of Kambara's Feature Tracker Module. The fundamental concepts of feature tracking are briefly explained, followed by an examination of the requirements of the module. The design approach is discussed, and the central elements of the design are outlined. The results of a feature tracking test, integrating the components of the Feature Tracker, is detailed.

Appendix A provides a glossary of terms. Appendix B provides the manufacturer's camera specification. Appendix C derives the full set of stereo extrinsic parameters, while Appendix D lists the derivation of a robust range estimation algorithm. Appendix E provides more details on the Feature Tracker software design.

Chapter 2 Camera Calibration Theory

Range estimation algorithms calculate the distance of a point in 3-D space using information extracted from 2-D images. This usually requires knowledge of how pixel coordinates in a 2-D image plane correspond to point coordinates in 3-D space. *Camera parameters* define this correspondence, and the procedure of determining the parameters is known as *camera calibration*.

This chapter introduces the fundamental concepts of camera calibration. We start by examining the camera components we hope to characterise, then outline the mathematical model and camera parameters which together describe the geometry and functionality of these components. Section 2.4 outlines the standard calibration technique, while Section 2.5 discusses how this technique is challenged by the properties of underwater imaging. Finally we describe the past development of Kambara's calibration system, which is based on the standard technique, and the underwater imaging problems that have affected its performance.

2.1 Camera Components

The standard camera used in computer vision applications is the Charge Coupled Device CCD camera. The essential elements of the CCD camera consist of a lens that focuses incoming light rays onto a rectangular grid of photosensors, each outputting a voltage proportional to the detected intensity of light. A framegrabber digitizes these voltages into numerical values, and stores them as a 2-D array in a memory buffer. Each element of the array is called a *pixel*.

2.2 The Camera Model

To determine the mapping between pixels in frame memory and points in 3-D space, we need a mathematical model of the camera and its components, describing:

- the geometry of image projection;
- the position and orientation of the camera;
- the geometry of the CCD array and lens.

The first of these properties is characterised by the projective geometry of the camera lens, while the other properties can be specified using reference frames assigned to the camera and its components.

2.2.1 Reference Frames

The standard camera model uses three reference frames, shown in Figure 2.1. The *camera reference frame* is used to specify the position and orientation of the camera, relative to a world reference frame, or another camera. Position vectors of objects found by range estimation algorithms are usually defined in this reference frame. The origin of the frame is centred on the camera's focus of projection, with the z-axis collinear with the optical axis.

The *sensor reference frame* is a 2-D frame used to specify coordinates of images projected onto the CCD grid. This coordinate system is centred on the intersection between the optical axis and the CCD grid, with the x-y plane parallel to the x-y plane of the camera coordinate system.

The *image reference frame* is used to specify pixel coordinates of images stored in computer memory. The origin of this frame is centred on the upper left pixel of the image.

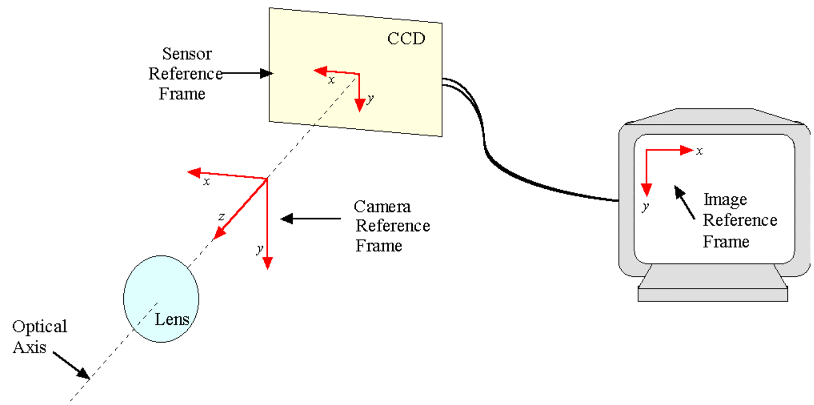


Figure 2.1: The three reference frames of the camera model.

2.2.2 Projective Geometry

The simplest model of projection is the *pin-hole model*, which assumes all light rays focused by the camera lens pass through the focus of projection (the camera reference frame origin). This is illustrated in Figure 2.2, which shows the vertical projection of a triangle onto the CCD. If the triangle apex is located with respect to the camera reference frame by (${}^{c}X\,,{}^{c}Y$), then from similar triangles the corresponding projected point ($X_u\,, Y_u$), is given by[1]:

$$\frac{X_u}{f} = -\frac{{}^{c}X}{{}^{c}Z} \qquad \text{(2-1a)}$$

$$\frac{Y_u}{f} = -\frac{{}^{c}Y}{{}^{c}Z} \qquad \text{(2-1b)}$$

The pin-hole model is an *ideal* model, because it produces perfectly undistorted image projections. With real cameras, however, projected image distortion is unavoidable, because no camera lens is capable of focusing all incoming light rays through the same point. This is also illustrated in Figure 2.2, where we see the sensor coordinates of the distorted projection, (X_d, Y_d), deviate from the undistorted sensor coordinates (X_u, Y_u) of the pin-hole model.

---

[1] The sign of this relation depends on the chosen orientation of the camera and sensor reference frames.

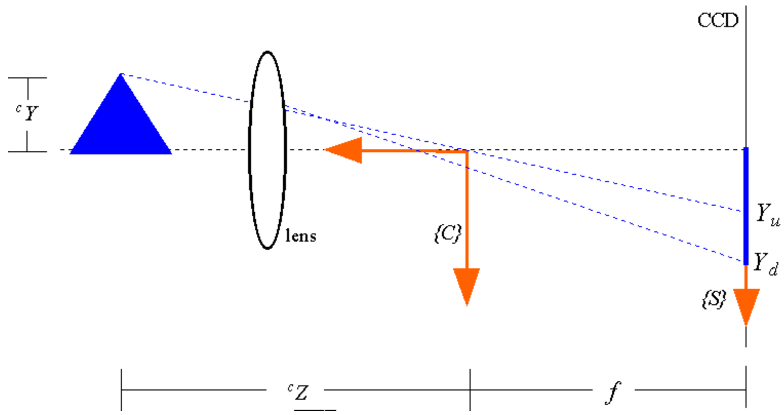


Figure 2.2: Undistorted and distorted image projection.

Lens distortion can be accurately modelled as *radial lens distortion*, with the distortion increasing radially from the image centre [Trucco, 1998]. In Figure 2.2, we see the undistorted and distorted coordinates are related by:

$$X_d + D_x = X_u \tag{2-2a}$$

$$Y_d + D_y = Y_u \tag{2-2b}$$

The well-used Tsai Camera Model [Tsai, 1987] models the distortion vector (D_x, D_y) as increasing radially from the origin of the sensor reference frame:

$$D_x = X_d(\kappa_1 r^2 + \kappa_2 r^4 + \ldots) \tag{2-3a}$$

$$D_y = Y_d(\kappa_1 r^2 + \kappa_2 r^4 + \ldots) \tag{2-3b}$$

where $r = \sqrt{X_d^2 + Y_d^2}$ [Tsai, 1987]. The higher the order of equations (2-3a) and (2-3b), the more accurate the model. A first order model is found to be sufficiently accurate in most applications [Trucco, 1998].

This model enables us to calculate undistorted coordinates from distorted coordinates, which can then be used in the simple equations of the pin-hole model (equations (2-1a) and (2-1b)).

2.3 Camera Parameters

The camera model outlined in Section 2.2 provides a generic mathematical framework for describing a camera. Individual cameras are characterised by specifying a set of model parameters, known as camera parameters. These parameters are usually grouped into *extrinsic* and *intrinsic* parameters, specifying the external and internal properties of the camera. For 3-D vision applications where a suite of cameras must be modelled, another set of parameters called *stereo extrinsic parameters* are used to specify the relative geometry of the stereo cameras.

2.3 1 Extrinsic Parameters

Extrinsic parameters specify the position and orientation of the camera reference frame relative to a world reference frame. Denoting the camera and world reference frames as {C} and {W} respectively, the relative position of the frames is represented by a 3-D translation vector[2] ${}^{C}T_{Worg}$, while the relative orientation is represented by a 3x3 rotation matrix ${}^{C}_{W}R$:

$$ {}^{C}T_{Worg} = \begin{bmatrix} T_x & T_y & T_z \end{bmatrix}^T ; \quad {}^{C}_{W}R = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} $$

The relative rotation of the reference frame is often equivalently expressed as the Euler angles R_x R_y R_z describing the relative rotation about the *x*, *y*, and *z* axes of the camera reference frame.

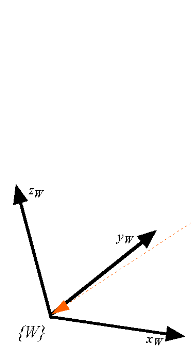


Figure 2.3: The extrinsic parameters of a camera define the position and orientation of the camera reference frame {C} relative to a world reference frame {W}.

2.3.2 Intrinsic Parameters

Intrinsic parameters define the transformations between the three reference frames, and specify the projective geometry and dimensions of the CCD array. Only some of these parameters need to be calibrated across different operating environments. These parameters are:

- *f, the focal length of the camera*. This also specifies the distance between the camera reference frame and the CCD reference frame, along the optical axis.
- *(C_x , C_y), the image pixel coordinates of the image centre*. These specify the CCD frame origin (the image centre) relative to the pixel frame origin.
- κ_1, *the first order radial distortion coefficient*. This is the first order coefficient of the radial lens distortion equation (2-3). A first order model

[2] This vector and reference frame notation is adopted from [Craig, 1989]. ${}^{C}T_{Worg}$ is defined as the translation vector from the origin of frame {C} to the origin of frame {W}. ${}^{C}_{W}R$ defines the rotation of {W} relative to {C}. Superscript *T* denotes the matrix transpose operation.

is usually sufficient for the characterisation of lens distortion in most camera applications.

- s_x, *a horizontal scale factor*. This defines how the horizontal scales of the pixel and CCD reference frames are related.

The remaining intrinsic parameters specify the dimensions of the CCD array, and are fixed for all applications. These are usually supplied by manufacturer's specifications, and do not need to be calibrated.

- (d_x, d_y), the "centre-to-centre" distance between the adjacent CCD sensor elements in the X and Y directions, given in terms of the sensor reference frame.
- N_{cx}, the number of sensor elements in the X direction.
- N_{fx}, the number of pixels in the X direction sampled by the computer.

2.3.3 Stereo Extrinsic Parameters

Stereo extrinsic parameters define the relative position and orientation of a stereo pair of cameras. This is accomplished using a translation vector and rotation matrix to specify the transformation between the camera reference frame of each camera. If the left camera frame is used as the base frame, then the translation vector is ${}^{L}T_{Rorg}$, and the rotation vector is ${}^{L}_{R}R$.

Stereo extrinsic parameters are related to but different from *extrinsic parameters,* which define the relative position and orientation of a single camera relative to a world reference frame. This relationship is illustrated in Figure 2.4. The stereo parameters can be calculated from the extrinsic parameters of the left and right cameras, using the following equations:

$$ {}^{L}T_{Rorg} = {}^{L}T_{Worg} - {}^{L}_{W}R\ {}^{R}T_{Worg} \tag{2-4}$$

$$ {}^{L}_{R}R = {}^{L}_{W}R\ {}^{R}_{W}R^{T} \tag{2-5}$$

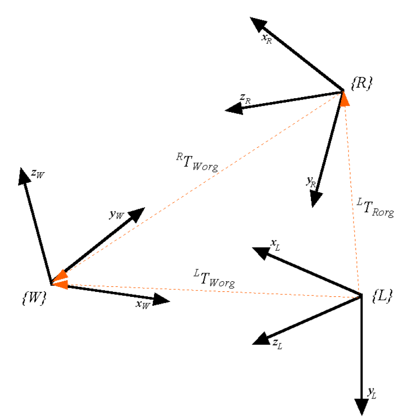


Figure 2.4: The stereo extrinsic parameters describe the relative position and orientation of a stereo camera pair.

The derivation of the stereo extrinsic parameters from the extrinsic parameters of each camera is given in Appendix C.

2.4 Standard Calibration Algorithm

Although calibration algorithms vary across different applications, they are all based on the same fundamental concept: camera parameters can be calculated using information extracted from the correspondence between accurately located points in 3-D space, and their corresponding projected image pixel coordinates [Trucco, 1998].

This section outlines the standard algorithm for accomplishing this, including the image processing techniques of *point detection* and *point identification,* required to match pixels to points. The performance measures by which a calibration algorithm can be evaluated is also briefly discussed.

2.4.1 Standard Algorithm

The standard method of accurately locating points in 3-D space is to use a 3-D *calibration pattern*, which supplies a set of points with an accurately known geometry. A typical calibration pattern consists of one or two planar grids of rectangles, or *boxes*, on a contrasting background [Tsai, 1987], such as the one shown in Figure2.5.

The standard algorithm for calculating the intrinsic and extrinsic parameters of a camera is listed in Algorithm 2.1. The following sections describe the point detection and point identification steps of the algorithm.

1) Capture image of the calibration target

2) Detect box corner-points:

   a) Detect edges

   b) Find box edges

   c) Fit lines to each box edge

   d) Intersect lines to find corner-points

3) Identify points

4) Calculate parameters

Algorithm 2.1: Generic algorithm for the calibration of a single camera

2.4.2 Point Detection

The box corner-points are located in the pattern image by first detecting the box edges. This is done using an *edge detection* algorithm, which locates pixels in regions where the image intensity undergoes sharp variations [Trucco, 1998], called *edgels*. Adjacent edgels are grouped together into *edgel chains*, so that there is one chain for every detected box.

In addition to detecting the box edges, the edge detection algorithm may detect the edges of other objects in the calibration environment with similarly sharp intensity variations. The total set of detected edges must therefore be filtered to eliminate all extraneous edges.

Lines are then fitted to the detected edgel chains using least squares fitting techniques. These lines are intersected to find the pixel coordinates of each box corner-point.

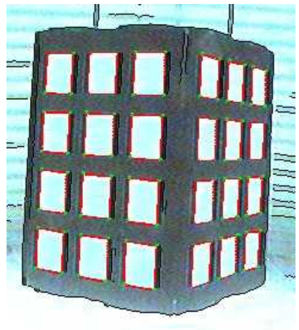


Figure 2.5: A typical calibration pattern

2.4.3 Point Identification

The detected box corner-points must be identified in order to be matched with their corresponding 3-D coordinates. Standard point identification algorithms identify points on the basis of their horizontal and vertical ordering in the image. This approach is only reliable, however, if every box corner-point in the image has been detected. The point ordering algorithm must therefore only proceed once the point detection algorithm has detected every corner-point.

2.4.4 Calculating Parameters

The points identified in the projected image are able to be matched with their corresponding 3-D points. Information extracted from the correspondence between these 2-D and 3-D coordinates is used to calculate the set of camera parameters. A variety of calculation techniques have been developed, perhaps the most popular being Tsai's Algorithm [Tsai, 1987].

2.5 Calibration Challenges in an Underwater Environment

Underwater environments present challenges to the reliability and accuracy of calibration algorithms. Accuracy is challenged because difficulties in underwater vision cause errors in point detection algorithms, which propagate through to inaccuracies in the calculated camera parameters. Reliability is challenged because unreliable point detection can limit the success of point identification algorithms.

2.5.1 Reliability Problems

As discussed in Section 1.1.4, suspended underwater particles tend to scatter light, decreasing image contrast. This reduces the intensity variations between the boxes and the background on the calibration pattern, meaning that some box edges, and therefore box corner-points, may not be detected in an image. This causes simple point ordering algorithms to fail, because they require the full set of points to work.

Identifying an incomplete set of projected points is a challenging task. Because the pattern image is distorted under projection, most of the geometric information relating the box corner-points is lost. In particular, the relative distance between points, the area of each box, and parallelism are not preserved under projective transformations [Mundy and Zisserman, 1992].

2.5.2 Accuracy Problems

Images captured underwater tend to be blurred from particle light scattering, as discussed in Section 1.1.4. This reduces the sharpness of intensity variation, making it difficult for edge detection algorithms to locate box edges precisely. Lines fitted to inaccurately located edges will deviate from the true box edge, leading to inaccurately located corner-points, which propagate through to errors in the calculated camera parameters.

2.6 Previous Work on Kambara's Calibration System

The calibration system developed for Kambara is based upon Algorithm 2.1, and uses the standard calibration pattern shown in Figure 2.5. Both planes of the pattern contain 12 boxes, each with dimensions 60mm x 75 mm, providing a total of 96 points [Fitzgerald, 1999].

The calibration pattern images captured by the stereo cameras have a 640 x 480 resolution. A Canny edge detection algorithm [Canny, 1996] is used to extract the edges of each box in the image, followed by orthogonal regression line fitting. Lines are then fitted to the detected edge of each box, and are then intersected to extract the corner-points of each box. Tsai's Algorithm is used to calculate the parameters from the identified corner-points and their corresponding 3-D coordinates.

The point identification step of Algorithm 2.1 was initially implemented with simple point ordering algorithms [Fitzgerald, 1999], assuming that all the corner-points can be detected reliably. However, this assumption was found to be unreliable in underwater testing. This unreliability motivated the development of a new algorithm.

2.7 Conclusion

The standard calibration algorithm calculates the camera parameters based on the assumption of reliable and accurate box corner-point detection. While this assumption may be valid for surface applications, it must be accepted that in underwater environments the behaviour of underwater light can significantly reduce this reliability — experience in the previous development of Kambara's calibration system confirms this. This in turn causes standard point identification algorithms, based on ordering the points by their horizontal and vertical positions, to fail, because they depend on *perfect* point detection. This points to the need for the development of a more robust point identification algorithm, which accommodates for unreliable point detection. This is the topic of the next chapter.

Chapter 3 Robust Underwater Camera Calibration

In Section 2.5 it was seen that the robustness of standard camera calibration techniques is challenged in underwater applications. This is because standard point ordering schemes often fail as a consequence of unreliable corner-point detection in underwater images. This motivates the development of a robust point identification scheme which is not reliant on the detection of every corner-point.

3.1 Robust Point Identification

To identify points in an image of the calibration target, what we need is a property of the points which stays constant under projection. We have seen that *geometric* properties, such as distance and area, are not preserved under a projective transformation. There are, however, *algebraic* properties of points which are invariant under projection. These properties are called *projective invariants*.

This section describes a robust point identification scheme based on the concept of *planar* projective invariants, that is, invariants which are calculated from a planar set of points.

3.1.1 Planar Projective Invariant Theory

A projective invariant is an algebraic property of a set of points which remains constant under projection. Two projective invariants can be calculated for five coplanar points x_i, $i \in \{1,..,5\}$:

$$I_1 = \frac{\left|\begin{bmatrix} x_4 & x_2 & x_1 \end{bmatrix}\right| \left|\begin{bmatrix} x_5 & x_3 & x_2 \end{bmatrix}\right|}{\left|\begin{bmatrix} x_4 & x_3 & x_2 \end{bmatrix}\right| \left|\begin{bmatrix} x_5 & x_2 & x_1 \end{bmatrix}\right|}; \quad I_2 = \frac{\left|\begin{bmatrix} x_4 & x_2 & x_1 \end{bmatrix}\right| \left|\begin{bmatrix} x_5 & x_3 & x_1 \end{bmatrix}\right|}{\left|\begin{bmatrix} x_4 & x_3 & x_1 \end{bmatrix}\right| \left|\begin{bmatrix} x_5 & x_2 & x_1 \end{bmatrix}\right|}; \qquad (3\text{-}1)$$

where the points x_i are represented as homogeneous coordinate vectors [Rothwell, 1995]. There are two important collinearity properties of planar projective invariants which affect the choice and ordering of planar points [Rothwell, 1995]:

1) If any three points are collinear, then *either* I_1 or I_2 is singular;
2) If both x_1 and x_2 are collinear with any of the other three points, then *both* I_1 and I_2 are singular.

These properties demonstrate the invariants' dependence on point ordering. The second property leads to a constraint where any set of five points must be chosen such that x_1 and x_2 are not collinear with a third point.

3.1.2 Identifying Box-Pairs with Planar Projective Invariants

The invariant formulae in (3-1) can be used to identify pairs of boxes. Consider two boxes A and B: we can take x_1 and x_2 from A, and the remaining points from B to calculate I_1 and I_2. The selection of points used in our algorithm is shown in Figure 3.1, and was chosen to avoid having both I_1 and I_2 singular for any possible combination of boxes.

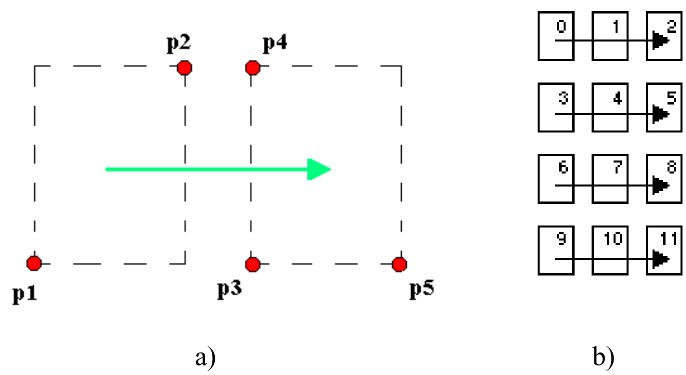


Figure 3.1: a) The corners selected for invariant calculations.
b) Box-pairs identical by translation share the same invariant.

Two invariants I_1 and I_2 calculated from boxes A and B can be collectively termed as a *box-pair invariant* $I_{A\text{-}B}$, graphically represented in Figure 3.1 as an arrow from box A to box B. Each possible box-pair will have an associated box-pair invariant.

Box-pair invariants have the following uniqueness properties:

1) For any two boxes A and B, $I_{A\text{-}B} \neq I_{B\text{-}A}$;

2) Box-pairs identical under translation share the same invariant.

Figure 3.1 (b) illustrates these uniqueness properties. Numbering the boxes 0 through 11 for one plane of the calibration target, we see box-pairs 0-2, 3-5, 6-8, and 9-11 all share the same box-pair invariant.

3.1.3 Invariant Indexing

A box-pair invariant index was constructed to aid the identification of box-pairs in images of the calibration target. Given a box-pair invariant, the index returns a list of the corresponding box-pairs.

The index has an entry for each unique box-pair invariant. For a 4 x 3 grid of boxes there are 34 unique invariants associated with 132 box-pair combinations.

The invariants in the index were calculated from precise measurements of the calibration target. The invariants used to key the index are calculated from images of the calibration target, and will always differ slightly from the index invariants. There are two main causes of this disparity:

1) Projective invariants are based on an assumption of pinhole projection. Radial lens distortion in the cameras means the pinhole model is not an accurate model for projection;

2) The edge detection and line fitting of boxes has inaccuracies which propagate through to box corner-point calculations, which in turn leads to errors in invariant calculations.

This disparity requires the use of invariant tolerances. When searching the index, a match is found if the image-based invariant falls within the tolerance range of the index invariants. If the tolerance is too small then there will be many false negatives, resulting in many points not being identified. If the tolerance is too big, then false positives will be made, corrupting calculation of the camera parameters.

A tolerance of ±20% was empirically determined to work reliably without false positives, and with only a small number of false negatives.

3.1.4 Invariant Index Search Algorithm

The invariant index is used to identify a set of planar boxes. Keying the invariant index with an invariant $I_{A\text{-}B}$, calculated from two of these boxes, will return a list of possible box-pairs. Only one of these is the true box-pair. Calculating invariants between the other boxes can help eliminate possibilities, until a single unique combination of boxes can been identified.

1) Begin with a list of n unidentified boxes, B_1, B_2....B_n
2) Calculate the invariant $I_{1\text{-}2}$ from B_1 to B_2.
3) Look up the invariant index for $I_{1\text{-}2}$ and find the corresponding list of box-pairs.
4) Create a search structure with one branch allocated to each of the box-pairs found from the index.
5) For the remaining boxes B_i, $i \in \{3...n\}$:
    a) Calculate the invariant $I_{1\text{-}i}$
    b) Use $I_{1\text{-}i}$ to look up the invariant index
    c) For each box-pair BP_j, $j \in \{1...k\}$, found in the index:
        i) Search the structure for a branch having the same base box as BP_j.
        ii) If such a branch is found, then add BP_j to it
    d) Prune any branch which does not have a base box corresponding to any of the box-pairs.
6) The algorithm succeeds if only 1 branch remains in the search structure, otherwise the boxes B_1.....B_n are not sufficient for identification.

Algorithm 3.1: Identifying boxes using a search structure and an invariant index

This is the basis for the invariant indexing algorithm stated in Algorithm 3.1. The input to the algorithm is the set of planar boxes that have been identified from edge detection and line intersection. The algorithm uses a search structure, with branches of the structure representing possible box combination scenarios. As more invariants are calculated, branches from the search structure are pruned as possible scenarios are eliminated, until one branch is left which identifies the boxes.

3.1.4.1 Index Search Algorithm Example

The index searching algorithm is illustrated in Figure 3.2. In this example only four boxes have been detected from a planar grid: A, B, C, and D. The box-pair invariant $I_{A\text{-}B}$ between boxes A and B is calculated and used to search the invariant index. A match is found, finding A and B could be one of three possible box-pairs. A search structure is created by dedicating a branch to represent the three scenarios: 1) Box A is 3, 2) Box A is 6, and 3) Box A is 9.

Next the invariant $I_{A\text{-}C}$ between boxes A and C is calculated. Searching the invariant index finds that A and C could be one of six possible box-pairs. Only two of these share the same base boxes with branches of the search tree, namely box-pairs 6-0 and 9-3. These are added to the corresponding branches. No possible box-pairs are found consistent with scenario 1), so it is pruned.

Next the invariant $I_{A\text{-}D}$ between boxes A and D is calculated. Searching the invariant index finds that A and D could be one of six possible box-pairs. Only one box-pair shares the same base box with a branch in the search tree, namely 6-10, and this is added to the corresponding branch. None of the box-pairs were found consistent with scenario 3), so this branch is pruned.

Only one branch remains, revealing the true identity of the boxes: box A is 6, box B is 5, box C is 0, and box D is 10.

3.1.5 Preprocessing

The points detected by the point detection algorithm must be processed before they can be used by the invariant indexing algorithm. There are three preprocessing steps required, grouping detected points into boxes, identifying the points within the boxes, and sorting the boxes into left and right planes.

3.1.5.1 Grouping Points into Boxes

The corner-points detected in the image of the calibration pattern must be grouped into boxes for use by the invariant algorithm. This is accomplished by a straightforward extension of the point detection algorithm outlined in section 2.4.2, so that points originating from the same edgel chain are grouped together.

3.1.5.2 Sorting the corner-points of a box

As outlined in section 3.1.2, the invariant algorithms takes five points from a pair of boxes according to Figure 3.1(a). To ensure the correct five points are used, each corner-point within a box must be identified as top-left, top-right etc. The sophistication of the algorithm to do this depends on the constraints placed on the orientation of the calibration pattern: the task is very difficult if we require successful calibration for arbitrary orientations, but is very easy if the pattern is held reasonably horizontal.

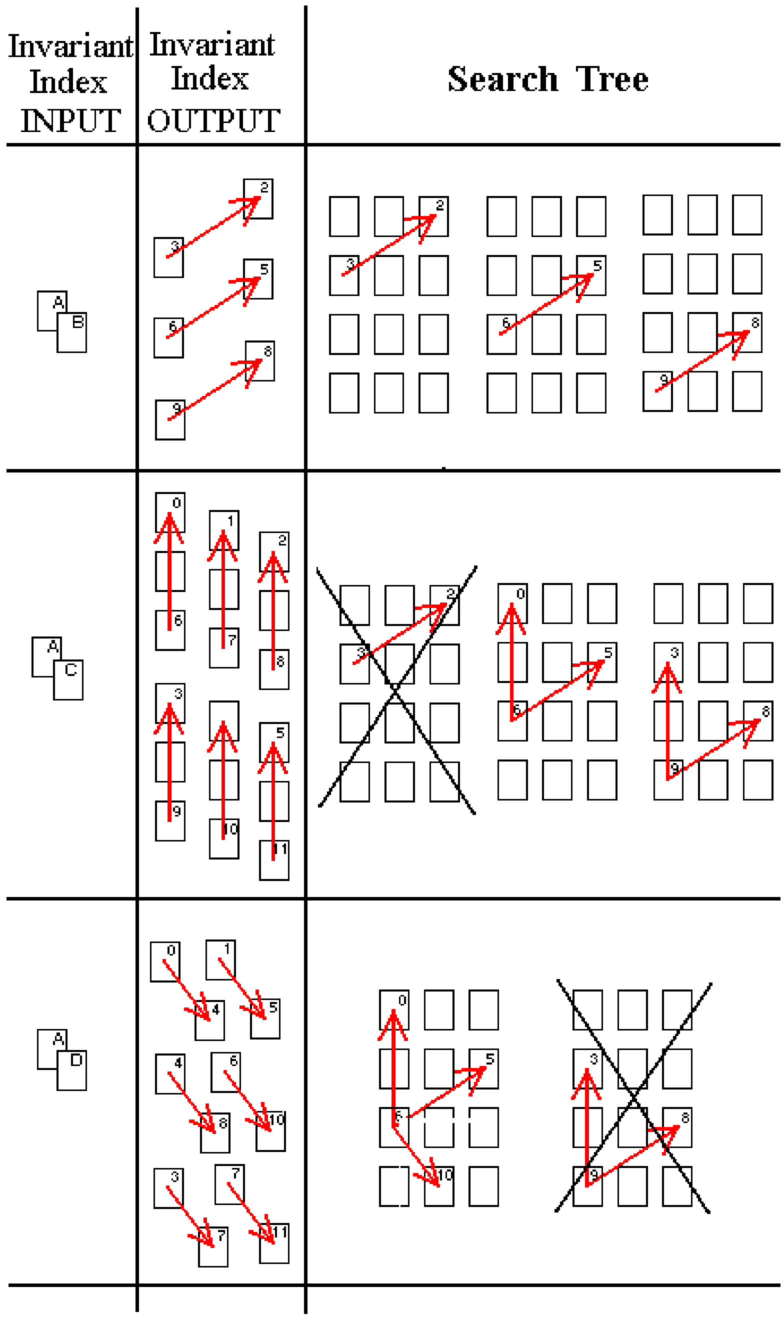


Figure 3.2: An example illustrating Algorithm 3.1. Boxes A, B, C and D have been detected in the image of a calibration pattern, but their identity is unknown. The invariant index is used together with the search tree to identify the boxes. The search tree contains branches representing different possible identities. Arrows represent box-pair invariants, while a cross represents a pruned branch of the search structure.

To keep things simple it was assumed that the calibration target will always be oriented within $\pm 30^{o}$ from the horizontal. This allows the use of the simple algorithm outlined in Algorithm 3.2, which sorts the points into top, bottom, left and right categories. Each point is then identified by the two labels that is has been assigned: top/bottom, and left/right.

1)Label the 2 points with the smallest vertical coordinates as TOP

2) Label the 2 points with the largest vertical coordinates as BOTTOM

3) Label the 2 points with the smallest horizontal coordinates as LEFT

4) Label the 2 points with the largest horizontal coordinates as RIGHT

Algorithm 3.2: Sorting the corner-points of a box.

Figure 3.3 shows the target when oriented on too much of an angle, so that Algorithm 3.2 will confuse the top right and bottom left corner-points for some of the boxes.

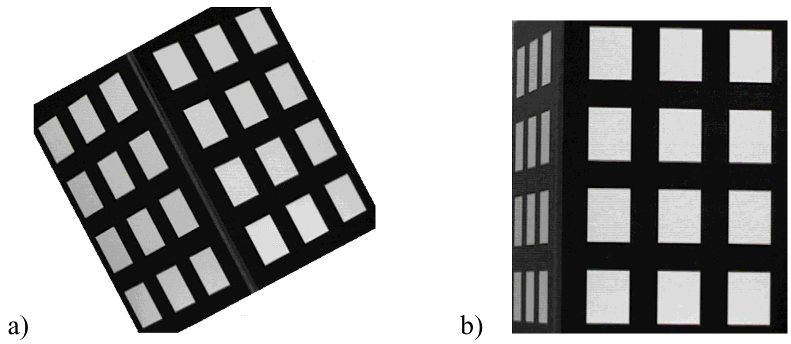


Figure 3.3: Target orientations for which calibration will fail. In a) the target is tilted too far, causing the sorting of corner-points within each box to fail. In b) the algorithm sorting boxes into the left and right planes will fail, because one plane is held too close to the camera.

3.1.5.3 Sorting Boxes into Planes

The invariant point identification algorithm works only for planar sets of boxes, and must therefore be applied separately to the left and right plane of the calibration pattern. Before this can be done, the list of detected boxes must be sorted into the left and right planes.

A variety of simple techniques were devised to accomplish this. The most reliable is listed in Algorithm 3.3. Boxes are sorted into the left and right planes depending on how close the box's average x-coordinate is to the minimum and maximum x-coordinates of the points.

Figure 3.3(b) illustrates how the algorithm fails when one plane of the target is angled too far away from the camera, causing the left column of boxes in the right plane to be attributed to the left plane. Step 3 of the algorithm minimises the occurrence of these false positives, by checking no more than 12 boxes have been attributed to each plane.

The algorithm therefore constrains the calibration target to be oriented such that the projection of each plane fills approximately the same image area. This is actually desirable for accurate point detection, because it ensures there are enough detected edge pixels to accurately fit lines to: in Figure 3.3 (b), the horizontal box edges detected in the left image plane contain too few pixels for accurate line fitting.

1) Search through all the points of the detected boxes to find:

x_{min} - the minimum x pixel coordinate of all the detected points

x_{max} - the maximum x pixel coordinate of all the detected points

2) For each detected box B_i, $i \in \{1...n\}$:

i) find the average x coordinate of the box, x_{avg}, using the x-coordinates of the corner-points $x_j \in \{1..4\}$:

$$x_{avg} = \sum_{j=1}^{4} x_j / 4$$

ii) if $| x_{avg} - x_{min} | < | x_{avg} - x_{max} |$

B_i belongs to the left plane

else

B_i belongs to the right plane

3) Check that the number of boxes in each plane < 12

Algorithm 3.3: Sorting boxes detected in a pattern image into the left and right planes

3.2 Robustness Evaluation

The robustness of a calibration system is difficult to measure — possibly the best metric for calibration robustness is the frustration level of the users of the calibration system! Using this metric the invariant indexing algorithm can be considered successful: frustration noticeably decreased once integrated into Kambara's calibration system.

It is, of course, important to quantify the reliability improvement of the invariant algorithm. The following subsection outlines the evaluation of the invariant based calibration system, using a point-ordering based calibration system as a benchmark.

3.2.1 Experimental Comparison with Standard Point Identification Schemes

The performance of the invariant algorithm was tested by comparing the reliability of the calibration system using 1) the invariant indexing algorithm, and 2) simple point ordering algorithms reliant on all 96 corner-points being detected.

3.2.1.1 Procedure

Calibration testing used underwater stereo image sets of the calibration target. The image sets were collected by placing Kambara underwater, capturing images from the stereo cameras as the calibration target was moved around the stereo FOV. The stereo camera pair, apart from being slightly verged to enable a close stereo range, were arbitrarily oriented. The calibration target was oriented so as to be approximately horizontal in the FOV of each camera.

Not all the captured image sets were fit for calibration. Image sets were discarded if 1) both planes of the calibration target were not seen clearly in the FOV of both cameras; or 2) the calibration target was more than 2 metres away from the cameras — if the calibration target is further away than this then each box edge is seen with too few pixels for accurate line fitting. Figure 3.4 gives an example of an accepted image set.

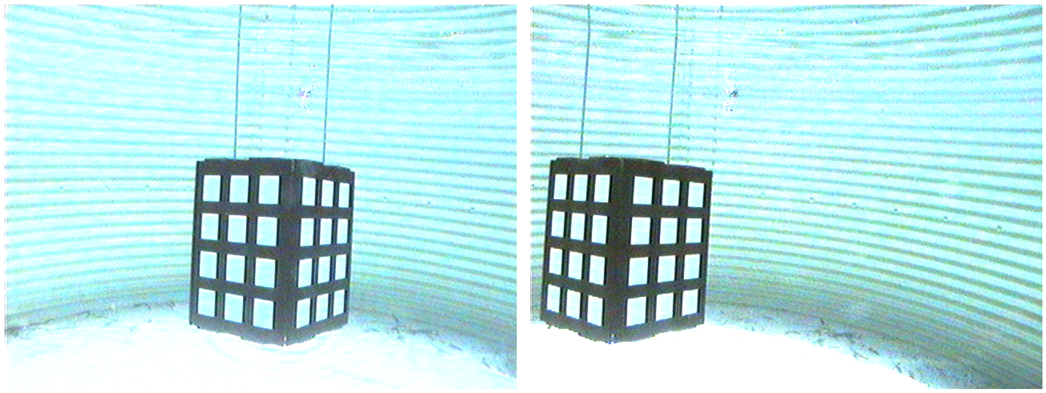


Figure 3.4: Stereo image set used to evaluate calibration robustness

In the end a total of 43 image sets satisfying these requirements were used for testing the calibration technique. Each image set was used once each for the point ordering and invariant indexing algorithms.

3.2.1.2 Results

The invariant indexing algorithm was found to be far superior to simple point ordering. The results of comparison are summarised in Table 3.1.

Although the point ordering algorithm led to the successful calibration of *one* of the stereo cameras for 40% of the image sets, calibration of *both* cameras never occurred. This result emphasises the difficulty of detecting all 96 corner-points in an underwater environment.

Point Identification Algorithm	% of images sets for which at least 1 camera calibrated successfully	% of images sets for which BOTH cameras calibrated successfully
Point ordering	40%	0%
Invariant indexing	81%	80%

Table 3.1: Evaluation of calibration system reliability with point ordering and invariant indexing point identification algorithms.

3.2.2 Performance Limits

The invariant indexing algorithm has been shown to be significantly more reliable than simple point ordering schemes. Like all algorithms, however, it does have some performance limits, which must be identified to provide a starting point for further underwater calibration research.

3.2.2.1 Invariant Indexing

A requirement that must be satisfied for the invariant algorithm to work is that at least one box from the outer boundaries of the planar grid must be detected. The number of detected boxes required for identification is then dependent on which boxes are detected. A box on two opposite grid corners is sufficient for identification, since each corner box shares two boundaries. The maximum number of boxes required in a 4 x 3 grid is 10.

3.2.2.2 Calibration Target Orientation

The plane sorting and corner-point sorting algorithms outlined in Section 3.1.5 constrain the orientation of the calibration target during calibration:

- The plane sorting algorithm requires the target to be oriented so that the projection of each plane covers approximately the same image area;
- The corner-point sorting algorithm requires the target to be held $\pm 30^{o}$ to the horizontal.

The invariant point identification algorithm itself does not impose any constraints on the target orientation. To reap the full benefits of this, more sophisticated plane sorting and box corner-point algorithms would need to be developed.

3.2.2.3 Radial Distortion Problems with the Invariant Indexing Algorithm

Radial lens distortion prevents the target being placed too close to the cameras, as the true invariants increasingly deviate from the pin-hole model invariants. This conflicts with the aim of accurately characterising radial distortion, because the more distortion captured by the image points the better the accuracy of that parameter estimate.

Fortunately this is not a serious problem under water, since the air–water interface refracts light so as to reduce lens distortion. For the given calibration target dimensions, it was found the calibration target can be brought close enough to occupy most of the vertical FOV of each camera.

3.3 Conclusion

The invariant indexing algorithm was developed to accommodate for unreliable point detection underwater. This algorithm has shown to be up to 80% more reliable in this respect than the simpler, more conventional point ordering techniques.

Chapter 4 Evaluating the Accuracy of Camera Calibration

Having previously dealt with calibration reliability, this chapter examines the other performance criterion of calibration: the accuracy of the calibrated camera parameters. Evaluating parameter accuracy is crucial for underwater 3-D vision applications. Underwater vision challenges the accuracy of standard calibration algorithms, as discussed in Section 2.5, so it is important to test for the impact these inaccuracies may have on camera parameter accuracy.

Unfortunately, evaluating the accuracy of camera calibration is not straightforward. The challenge of accuracy evaluation is discussed in Section 4.1. Section 4.2 outlines an accuracy evaluation methodology that has been developed, based on range estimation. This methodology has been successfully applied in evaluating Kambara's camera calibration accuracy, a task which has been previously been difficult due to reliability problems. The details and results of this experiment are outlined in Section 4.3.

4.1 The Difficulties of Evaluating Calibration Accuracy

Evaluating the accuracy of camera parameters is challenging because of the difficulty in obtaining ground truth for comparison.

Since intrinsic parameters characterise the internal workings of a camera, it might be expected that camera manufacturers, having constructed the camera, should be capable of providing intrinsic parameter specifications. However most intrinsic parameters are dependant on the vision environment, preventing the tabulation of comprehensive specifications.

A camera's extrinsic parameters can in principle be compared with ruler measurements made between camera and world reference frames. Similarly, the *stereo* extrinsic parameters of a camera suite could in principle be compared with ruler measurements between the camera reference frames of each camera. In practice, however, such measurements are impractical, especially in underwater environments. Even if this were not an issue, such measurements will always be inaccurate, and therefore inappropriate for comparison. These inaccuracies are consequences of :

a) the camera reference frame being located within the camera, and therefore inaccessible to ruler measurements; and
b) even if inaccessibility were not a problem, the origin of the camera reference frame would be very difficult to locate accurately.

Since camera parameters are calibrated for use by range estimation, another approach might be to determine the accuracy of range estimation, using this to infer the accuracy of the camera parameters. Unfortunately, the inaccessibility of the camera reference again prevents the accurate measurement of ground truth for comparison.

4.2 Accuracy Evaluation Through Object Length Estimation

We need a measure of camera parameter accuracy that can be compared with ground truths that are 1) accurate, and 2) practically obtainable in an underwater environment. One such measure is 3-D object length. The length of an object in the FOV of both stereo cameras can be calculated using range estimation to

1) estimate the range vectors from the two endpoints of the object;
2) use simple vector arithmetic to find the distance between the points.

If the 3-D object is chosen carefully, the calculated length estimates can be compared with measurements made above water and to the required level of precision. From this comparison we can make inferences on the accuracy of the camera parameters: accurate length estimation infers accurate range estimation, which in turn infers accurate camera parameters.

4.3 Evaluating the Accuracy of Kambara's Calibration System

Evaluation of the accuracy of Kambara's original calibration system was limited, due to the reliability problems discussed in Sections 2.5 and 2.6. Reducing these problems through robust point identification, as outlined Chapter 3, now allows a more thorough evaluation of calibration accuracy.

Section 4.3.1 outlines the problems facing the previous accuracy evaluation. This is followed by the evaluation procedure that was followed, based upon the object length methodology outlined in Section 4.2. The results of the evaluation are given in Section 4.3.3.

4.3.1 Previous Calibration Accuracy Evaluation

The methodology employed for previous calibration accuracy evaluation suffered from the limited ground truth previously mentioned. Intrinsic parameter accuracy was judged by examining the variation of parameter values from a mean value [Fitzgerald, 1999]. This approach establishes the *precision* of the camera parameters, but without ground truth the accuracy of the parameters remains unknown.

Extrinsic parameter accuracy was judged by calculating the stereo camera *baseline*, the distance between the camera reference frame origins of the two cameras, and comparing it with a ruler measurement. This approach is inadequate for two reasons. Firstly, the baseline can only be used to draw conclusions about the accuracy of the translation vector between reference frames, not the rotation matrix. This is because the baseline is independent of rotation, and dependant only on the translation vector between frames. Secondly, accurate measurements of the baseline are not possible due to the inaccessibility of the camera reference frame origin, as discussed in Section 4.1.

4.3.2 Experimental Procedure

The experimental procedure to evaluate Kambara's calibration accuracy was based upon the object length estimation method outlined in Section 4.2. The calibration target was chosen as the 3-D object, as it provides a large set of points (i.e. box corner points) whose relative geometry is known to a high accuracy (±0.1mm).

It is important to ensure the object's end points are accurately located in the stereo images. The calibration graphical user interface was adapted to allow the manual correlation of points, that is the user uses a mouse to locate the chosen point in each stereo image.

Kambara was placed under water with an arbitrary camera position and orientation, and the cameras were calibrated five times. The calibration target was then moved around the mutual FOV of the stereo cameras, at a distance approximately 1metre from the cameras. At each target position, several pairs of corner points were manually selected from the stereo images. This was repeated again at 2 and 3 metre distances.

The distance between each point pair was later calculated for each set of calibrated camera parameters, using the range estimation algorithm outlined in Section 6.5.

4.3.3 Results

Figure 4.1 a) and b) illustrate the results of the length estimation tests at 1 and 3 metres respectively. Each shows the accuracy of length estimates across the FOV of the left camera. The accuracy of each length estimation is calculated as the percentage error of the true (measured) length. The plots show accuracy in length estimates decreasing with increased target distance.

Table 4.1 summarises the results of underwater tests. The mean standard deviation and maximum error are calculated across the length estimates generated by each collected set of calibration parameters. The results show calibration to be sufficiently accurate for range estimation of targets within 3 metres with a maximum mean error of 5.0%.

Approximate distance between AUV and target	Mean Error %	Std Dev %	Max Error %
1 metre	1.4 %	1%	6%
2 metres	3.6 %	2.7%	15%
3 metres	5.0 %	5.2%	30%

Table 4.1: Evaluation of calibration system accuracy

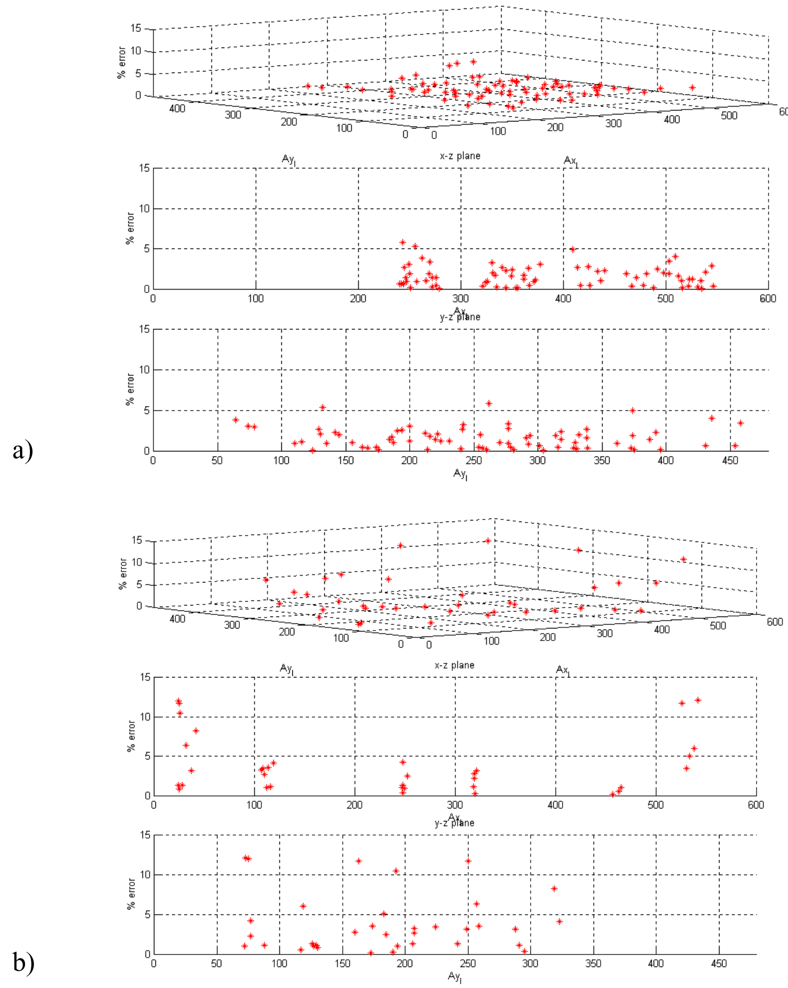


Figure 4.1: Variation of length estimate accuracy across the different camera parameter sets, at distances a) 1m, and b) 3m. Length estimate accuracy is calculated as a percentage error of the true value.

4.4 Run-time Evaluation of Calibration Accuracy

Kambara's calibration system was extended to allow accuracy evaluation immediately after calibration. 3-D objects in the stereo FOV can be selected using a mouse to manually correlate features of the object in each stereo image. The calculated length of the object is calculated using the algorithm outlined in Section 6.5, and displayed on the user interface. The calibration user can then use the length estimate to reject or accept the calibrated parameters.

4.5 Conclusion

A methodology for calibration accuracy evaluation was developed allowing accuracy to be judged using easily obtained ground truth. Applying this methodology to the evaluation of Kambara's calibration system found calibration was sufficiently accurate for range estimation of targets within 3 metres with 95% accuracy.

Chapter 5 Calibration of an Automated Zoom Lens Camera

Kambara's pan/tilt/zoom camera, originally intended to provide the user interface with a manoeuvrable view of the underwater environment, has the potential to be used as a valuable extension of the AUV's range estimation capabilities. Used together with the fixed stereo pair of cameras, the pan/tilt/zoom camera would provide a larger and variable field of view (FOV).

In order to use a camera for 3-D vision applications, a mathematical model describing the image properties of the camera is required. For a pan/tilt/zoom camera the model must be *adjustable*, with each model parameter a function of the position, orientation and lens control settings.

This motivates the development of a calibration strategy which determines how the camera model varies as the camera settings are adjusted. For Kambara's pan/tilt/zoom camera this is a considerable task because of the number of adjustable settings. This chapter outlines the more modest task of modelling and calibrating the camera's zoom facilities. In particular we aim to answer three questions:

1) Is calibration of an Automated Zoom Lens practical for an AUV?
2) Is the calibrated adjustable model practical for 3-D vision purposes?
3) Can autofocus be used without having to account for it in the adjustable camera model?

Section 5.1 examines the theory of modelling and calibrating an adjustable zoom lens. In section 5.2 this theory is applied to the calibration of Kambara's zoom lens. Section 5.3 analyses the calibrated model to provide answers for the three questions asked above.

5.1 Zoom Lens Theory

Little information about the internal zoom mechanisms is provided by manufacturers. What little is known must be developed into a generic adjustable model, and the relationship between the camera model and the lens control settings must be established empirically by calibration.

5.1.1 Zoom and Autofocus Mechanisms

Although we talk of an adjustable camera lens as though it were a single object, most are actually comprised of a flexible arrangement of separate lens parts. The position of the lens system components are controlled by motors, and the user can adjust the projected image by selecting various lens control settings which activate the motors. "Zoom" is achieved by motors moving these lens components along the optical axis of the camera: "zooming in" enlarges the projected image by moving the lens system away from the CCD array, while "zooming out" reduces the image by moving the lens system towards the CCD array.

Lens focusing also involves moving the lens components, only the movement associated with focus is usually much smaller than for zoom. Many cameras are equipped with autofocus capabilities which allow automatic focusing on a subject. There are two types of autofocus mechanisms, active and passive. Active autofocus adjusts the lens components according to estimates of the subject's distance from the camera. The distance is estimated by bouncing infrared pulse signals off the subject, using time, triangulation, or the intensity of the reflected signal to estimate the distance to the subject [Brown, 2000].

Kambara's Sony pan/tilt/zoom camera uses passive autofocus. Microprocessors inside the camera process image information collected from the CCD array, adjusting the lens system to find the maximum contrast between adjacent pixels [Brown, 2000].

5.1.2 The Adjustable Lens Camera Model

The projective geometry of an adjustable camera lens must be modelled if it is to be used for 3-D computer vision applications. The approach taken by Willson [Willson, 1994] was to assign a fixed camera model (described in Section 2.2) to the lens configuration at every lens control setting. The adjustable camera model therefore uses the same parameters as the fixed camera model, only now each parameter is a function of the lens control settings. For example, the focal length parameter f increases as the camera lens zooms in.

The complexity of the adjustable model depends on how many variables — lens control settings — each parameter is a function of. Willson modelled each parameter as a function of two variables: zoom and focus motor steps.

5.1.3 Calibration of the Adjustable Model

The task of calibrating the adjustable model is to determine exactly how the model parameters vary over the range of lens control settings. One way to do this might be to calibrate the fixed model at each lens setting. If, like Willson, we assume that each camera parameter is dependant on zoom and focus, then the number of calibrations required for each unique focus-zoom setting may run into the thousands or even tens of thousands. Even if we assume dependence only on zoom the task is too big — for the Sony pan/tilt/zoom camera this would require 1024 calibrations — especially for an AUV with limited power.

Another approach may be to calibrate only at sample zoom positions, and then estimate the parameters values through interpolation techniques. This approach significantly reduces the task of data collection, but means the calibrated model must be stored in memory as a large index.

A more manageable method would be to establish a model for the parameter variation. Then only enough calibration data needs to be collected to fix the model. Willson modelled the parameter variations of his adjustable camera lens as polynomials. An N^{th} order polynomial, this requires calibration at a minimum of $N+1$ lens settings. The polynomial orders used by Willson to characterise each parameter are listed in Table 5.1.

Camera Parameter	Polynomial Order
s_x, R_x, R_y, R_z, T_x, T_y	0
f, T_z, C_x, C_y	5
κ_1	2

Table 5.1: Polynomial orders used by Willson for adjustable camera model parameters

5.2 Calibration of Kambara's Automated Zoom Lens

Development of a calibration technique for Kambara's pan/tilt/zoom camera was guided by the research of Willson outlined in the previous section [Willson, 1994]. The following subsections evaluate the appropriateness of this approach for the Sony camera, and discuss the overall viability of automated zoom lens calibration for Kambara.

5.2.1 Calibration Data Collection

The VISCA protocol, integrated into Kambara's calibration system last year [Fitzgerald, 1999], was used to set the zoom settings of the Sony camera. The focus was kept at a constant motor position (position 6674). The camera was calibrated over a zoom step range of 0 to 775, with increments of 25 motor steps, out of a possible 0:1024 range, taking approximately six minutes. This sampling rate was empirically determined to be high enough to capture the variations in each parameter.

The calibrated data is shown in Figure 5.2 (together with fitted polynomials discussed in the following section).

5.2.2 Polynomial Fitting

Polynomials were fitted to the calibration data collected for each parameter using least squares fitting.

5.2.2.1 Polynomial Characteristics

Fitted polynomials can only be used for the interpolation of parameter values within the range of collected data. Parameter estimates taken outside the data range will generally be unpredictable and highly inaccurate, since polynomials, high order polynomials in particular, tend to lurch wildly outside the data range.

In general, the higher order polynomials provide a better fit to the data than lower order polynomials. If, however, the order is too high, the fitted polynomial can oscillate wildly in between data points. This is illustrated in Figure 5.1, where although the fitted polynomial passes accurately through each data point, any interpolation within the ranges 25 to 50, or 725 to 750 will provide grossly inaccurate parameter estimates.

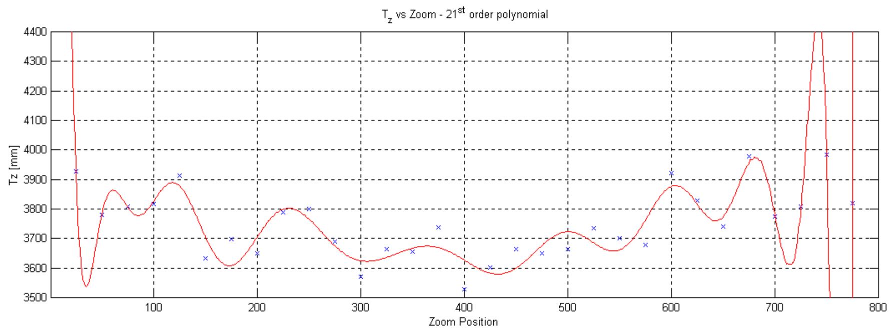


Figure 5.1: Polynomials with too high an order can oscillate widely between data. Shown is the T_z parameter fitted with a 21st order polynomial.

5.2.2.2 Measuring the Accuracy of the Fit

The approach taken to measuring the accuracy of the fitted polynomials was to calculate the mean absolute error of the fit:

$$\text{Mean absolute fit error} = \frac{1}{n}\sum_{i=1}^{n}\left|y_{data,i} - y_{fit,i}\right| \tag{5-1}$$

where $y_{data,i}$ *and* $y_{fit,i}$ are the measured data points and fitted polynomial value at the i^{th} sampled zoom position.

This metric does not always tell the whole story, as it only measures the accuracy of the fit to the collected data, and does not provide any insight into the interpoint accuracy. High order polynomials suffering from wild interpoint oscillations will therefore have a small mean absolute fit error even though interpolation will yield grossly inaccurate parameter values.

The fitted polynomial was therefore examined to ensure the polynomial between data points never deviated too far from neighbouring data points.

5.2.2.3 Optimum Polynomial Order

For the purpose of judging the best polynomial fit for each parameter, the *optimum polynomial order* was defined as *the order which minimises the mean absolute fit error without causing interpoint oscillation*. The optimum polynomial orders for each parameter are listed in Table 5.2, and the fitted polynomials for each parameter are shown in Figure 5.2.

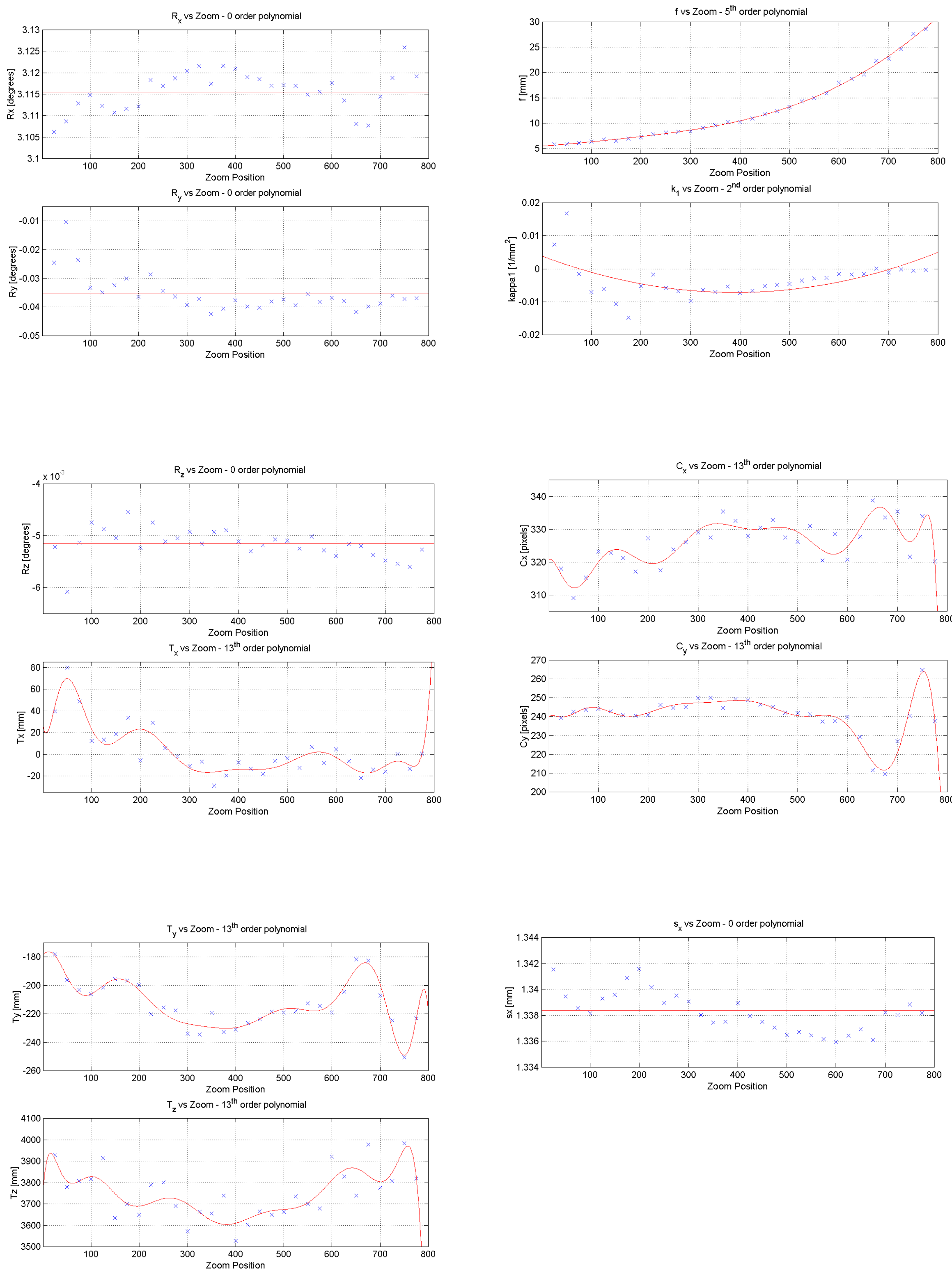


Figure 5.2: Calibration data and the optimum fitted polynomials for the camera parameters of the adjustable model. The order of each polynomial is given in Table 5.2.

Camera Parameter	Polynomial Order
$s_x, R_x, R_y, R_z,$	0
κ_1	2
f	5
C_x, C_y, T_x, T_y, T_z	13

Table 5.2: Optimum polynomial order for the Sony EVI-D30, found empirically at

5.2.2.4 Comparison with Willson's Model

Table 5.3 compares the fit accuracy of the empirically determined optimum polynomial order with Willson polynomial orders. It is seen for each of the listed parameters that the 13th polynomial fit significantly reduces the mean absolute error.

From this it can only be concluded that 13th order polynomials provide the best possible fit *for the tested zoom range and focus position*. Although repeated testing over this range confirmed the parameter variations are best modelled as 13th order systems, it perhaps doesn't tell the whole story. It is possible then lens is actually a lower order system, and that the parameter variations are caused by unidentified higher-order effects, which are themselves well modelled by 13th order polynomials.

Parameter	Empirically determined optimum polynomial order		Willson's Polynomial Order	
	Polynomial Order	Mean Absolute Fit Error	Polynomial Order	Mean Absolute Fit Error
C_x	13	3 pixels	5	4 pixels
C_y	13	2 pixels	5	4 pixels
T_x	13	7mm	0	18 mm
T_y	13	3mm	0	14 mm
T_z	13	52mm	5	65 mm

Table 5.3: Comparison of the absolute mean error for the optimum polynomial orders in Table 5.2 and Willson polynomial orders

Particularly noticeable in Figure 5.2 is that T_x and T_y are not at all well approximated as constants. In fact it is seen that T_x and T_y vary within a range of 110 mm and 73 mm respectively. Interpreted literally, this would seem to tell us the camera reference frame, and hence the camera lens, is being translated in the x and y directions by 10cm and 7cm respectively — despite the calibration target and the camera being held stationary for the entire range of zoom calibrations. This surprising result was confirmed by further calibrations.

This paradox is possibly best explained by remembering that the T_x and T_y form part of an *approximate* model, where the adjustable camera lens is approximated as though it were a moveable thin lens. The *real* camera uses a compound lens, whose imperfections result in translation of the focal point.

5.2.3 Evaluation of Automated Zoom Lens Calibration

5.2.3.1 Calibration Sensitivity to Autofocus

To determine whether focus has an appreciable effect on the calibrated parameters, the adjustable camera model was calibrated again with the autofocus switched on. The focus varied during calibration from motor positions 16 to 64290, almost the full focus range.

The calibrated model parameters did not deviate significantly from the constant focus parameters. Figure 5.3, for example, shows how the focal length parameter, calibrated with and without autofocus, results in almost identical polynomials. This preliminary investigation therefore suggests that the adjustable zoom lens model can be accurately approximated as being independent of focus. Further confirmation of this result would require calibrating over the full range of zoom and focus positions.

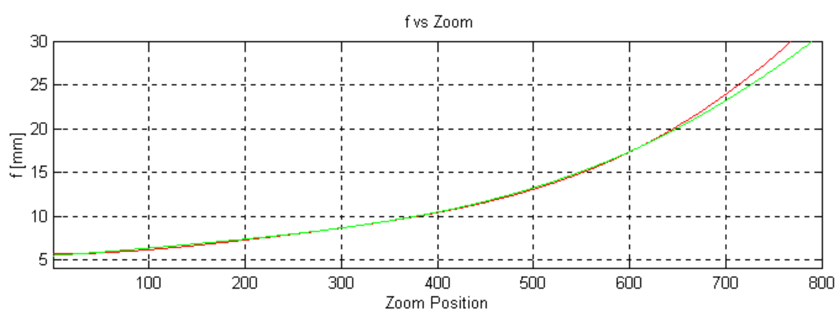


Figure 5.3: The calibrated focal length parameter with autofocus (green) and without (red)

5.2.3.2 Practicality of Calibration for an AUV

An adjustable lens model that is independent of focus is much easier to calibrate than a focus-dependent model, because it drastically reduces the size of the calibration data to be collected.

Figure 5.2 shows that the extrinsic parameters T_x, T_y, and T_z are a function of zoom. This is to the detriment of calibration practicality, because it means the calibration target must not be moved during calibration (to do so would drastically change the extrinsic parameters). This makes it difficult to calibrate over the full zoom range: we cannot zoom in too close because calibration is only possible when the full target projection is captured, and we cannot zoom out too far since calibration accuracy requires the target projection to fill most of the captured image.

5.2.3.3 Practicality of Adjustable Model for 3-D Vision Applications

Restriction of the calibrated zoom range in turn restricts the zoom range useable for 3-D vision, since parameter values extrapolated from polynomials outside the calibrated zoom range can be wildly inaccurate.

The calibrated adjustable model can be concisely described, and efficiently stored in memory, as a set of polynomial coefficients. 3-D vision algorithms requiring a zoom lens can either expand this representation into a lookup table, or quickly evaluate the polynomials at run time.

Ideally we would like to use the zoom lens together with the Sony camera's pan and tilt facilities. This would require extending the adjustable lens model, with the extrinsic parameters now a function of three variables: zoom position, pan angle, and tilt angle. The extrinsic parameters' dependence on each variable could be calibrated separately, to be later combined using the superposition of translation vectors and rotation matrices. Although the mathematics for this is straightforward, the calculations required would be too lengthy to be done at run time by 3-D tracking algorithms. The alternative would be to store all the calculations in a lookup table keyed by zoom, pan and tilt. Such a table would require over a million entries, which is not practical given Kambara's limited memory. If the extrinsic parameters depended only on pan and tilt, the table size would much more manageable: but dependence on zoom makes it impractical.

5.3 Conclusion

The zoom lens facilities of Kambara's pan/tilt/zoom camera can be practically calibrated and used for 3-D vision applications over limited zoom ranges. Autofocus can be used without increasing the complexity of the camera model or its calibration. The useability of the zoom lens for 3-D vision is severely limited by the extrinsic parameters' dependence on zoom, specifically:

1) calibration is limited to the zoom range for which the entire target is clearly visible in the zoomed images;
2) this limits the use of the zoom lens to the calibrated zoom range;
3) the task of combining the zoom lens with pan and tilt facilities is impractical due to the dimensionality of the required lookup tables

This preliminary investigation into zoom lens calibration suggests that using Kambara's pan/tilt/zoom camera for 3-D vision is more trouble than it is worth.

Chapter 6 Design and Development of a Feature Tracker Module

The extensive calibration research outlined in this and previous projects has been undertaken for one reason: to equip Kambara with 3-D stereo vision. There are a variety of possible 3-D vision applications; the one we are most interested in is the capability of tracking dynamic 3-D targets.

The task of target tracking has been allocated to Kambara's Feature Tracker Module. This module interacts with other software modules as part of an on-board and off-board software architecture, as illustrated in Figure 6.1. This chapter outlines the design and development of this feature tracking module. A brief overview of feature tracking is discussed, before outlining the requirements of the feature tracker specified in Kambara's top-level software design document [Wettergreen et.al., 2000]. This is followed by an explanation of the object-oriented approach to design, and an explanation of the key elements of the design. A constraint-free range estimation algorithm is also outlined, and finally the module testing is described.

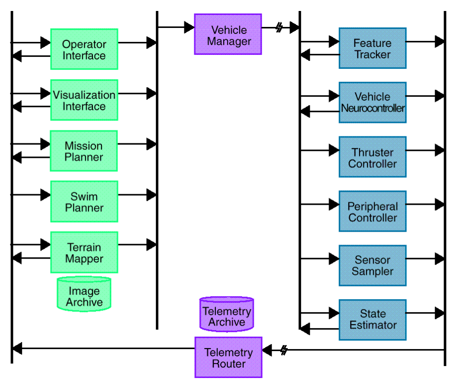


Figure 6.1: Kambara's software architecture [Wettergreen et.al, 1999]

6.1 Feature Tracking Overview

An image feature is any interesting or meaningful portion of an image. For 3-D target tracking, the features that are "meaningful" are the projected images of the target. Feature tracking is the task of matching these features from frame to frame in an image sequence [Trucco, 1998].

Kambara's feature tracking strategy is based on *correlation*, a strategy which involves extracting small image sections from the feature, called *templates*, and searching an image to find the portion of the image most similar to the template. There are many different correlation techniques, but most follow the basic steps shown in Figure 6.2. The first step is to determine the correlation search area, commonly called the *region-of-interest* (ROI). Because correlation is a computationally intensive process, the aim is to use the smallest possible ROI that can possibly contain the feature location.

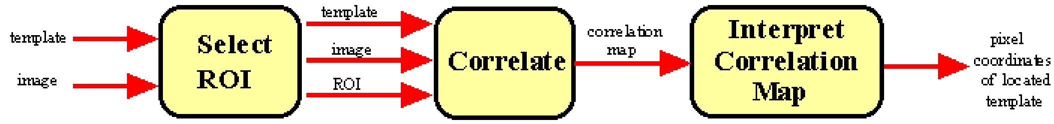


Figure 6.2: The basic steps of a correlation algorithm

Next the template is correlated across the ROI. This can be thought of as scanning the template over every pixel of the ROI, and at each pixel evaluating the similarity between the template and the image portion. Similarity is evaluated as a *correlation value*. Scanning and evaluating similarity at every pixel location within the ROI generates a map of correlation values, appropriately known as a *correlation map*. The next step is to interpret the correlation map to find the pixel location of the best match. The simplest possible interpretation is that the feature is located at the maximum correlation value in the map.

Tracking the 3-D position of a target can be accomplished by tracking the target's projected feature in left and right image sequences. The challenge is to ensure the features being tracked in each stereo image correspond to the same target.

6.2 Requirements of Kambara's Feature Tracker Module

The black box behaviour of Kambara's feature tracker module has previously been specified in [Wettergreen et al., 2000]. This section outlines these specifications, and briefly discusses the requirements of the internal module design.

6.2.1 Input and Output of the Feature Tracker Module

Figure 6.3 illustrates the input/output requirements of the feature tracker module. The module provides three public methods:

- ftTrackPoint() - track a feature at specified pixel coordinates;
- ftTrackTemplate() - track a feature using a specified template;
- ftPickFature() - identify and track a feature.

Each method starts the module tracking over an image sequence. The tracking information generated at each image is written to a global data structure, where it can be read by other software modules.

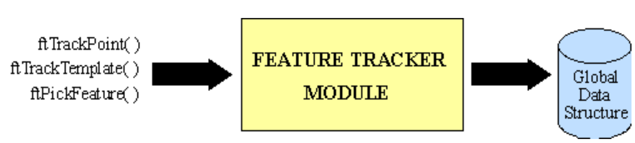


Figure 6.3: Input and output of the Feature Tracker module

6.2.2 Performance Requirements

The feature tracker module is required to be capable of tracking multiple targets at 10 Hz. The module must be capable of tracking composite targets. A fish, for example, might be tracked using both its head and its tail as targets.

6.2.3 Design Requirements

The feature tracker module is likely to be developed by different people having different research goals. It follows that the chief design requirement of the feature tracker is *extendibility*. This has two facets:

- *simplicity*: the design should capture and represent the concepts of feature tracking, and hence be easily understood by developers;
- *flexibility*: the tracker should be structured to facilitate a wide variety of different algorithms.

6.3 Design Approach

The chosen design approach has been based on the Object Oriented paradigm, an approach that facilitates both conceptual simplicity and flexibility.

6.3.1 Conceptual Simplicity with Object Oriented Design

An object oriented approach enables real world objects to be intuitively represented by software objects. Information processing can be neatly compartmentalised into "black boxes", and abstract concepts can be concisely represented. This allows the designer to consider firstly *what* each component does, before worrying about *how* it does it. This naturally leads to a design consisting of components defined by their functionality, not their implementation, and which will consequently be simpler to comprehend by future developers.

6.3.2 Flexibility with Inheritance and Polymorphism

Ideally the feature tracker should consist of a number of black box components, with each component interchangeable with other components of its type. This can be called the "toolbox" approach: each component comes from a toolbox of components all with the same functionality, but having different implementations. Various configurations can then be constructed by selecting the appropriate component from each toolbox.

As an example, consider the acquisition of images with which to track. For real-time applications, features will be tracked in video streams from each camera, but for development purposes it would also be convenient to use images stored on disk. For this purpose it would be convenient to have a toolbox of "framegrabber" objects. One of these might supply images directly from a camera; another might supply images from disk. Whichever framegrabber object is appropriate for an application can be slotted in, without having to adapt or even inform the other module components.

This is possible using the object-oriented concepts of inheritance and polymorphism. Each toolbox we wish to create can be represented as an abstract class, which defines the common behaviour of each component in the toolbox. Specific components are created by defining classes which are inherited from the abstract class. Polymorphism enables components from the toolbox to be known by the same name, eliminating the need for extensive modifications or messy conditional statements in other module components each type a component is interchanged.

Returning to the previous example of the "FrameGrabber" toolbox, the abstract FrameGrabber class may define all FrameGrabber objects to offer a *fetchImage()* method. A

File_FrameGrabber class can be derived from the FrameGrabber class, implementing the *fetchImage()* method by reading image files from disk, while Stream_Framegrabber may implement the method by capturing images from live video stream from a camera. Users of these components only know they are using a FrameGrabber, not the specific type of Framegrabber.

Singleton and *Factory Method* design patterns [Gamma et al., 1995] are particularly useful for this “toolbox” approach, confining the impact of interchanging components to the abstract class definition of a toolbox.

6.4 Software Design

This section briefly outlines the key components of the Feature Tracker design. A more detailed software design is provided in Appendix E.

6.4.1 Feature Tracker Class

The structure of the FeatureTracker class is shown in Figure 6.4. The basic components are:

- ***a Target array***: each 3-D target being tracked is represented by a Target object in this array;
- ***Vision System***: this encapsulates Kambara’s vision components, including cameras and a framegrabber, together with the stereo extrinsic parameters defining the configuration of the cameras;
- ***Utilities***: these are a collection of components which provide services to Target objects and their internal components.

Each of these components is discussed in more detail in the following sections.

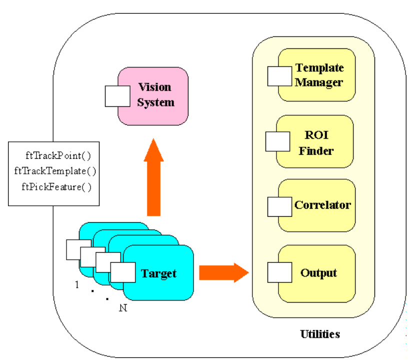


Figure 6.4: The feature tracking class. Arrows denote the “uses” relationship.

6.4.2 Targets

Each Target consists of an array of sub-targets, as illustrated in Figure 6.5, with each sub-target being an Object3D object. This allows composite targets to be tracked.

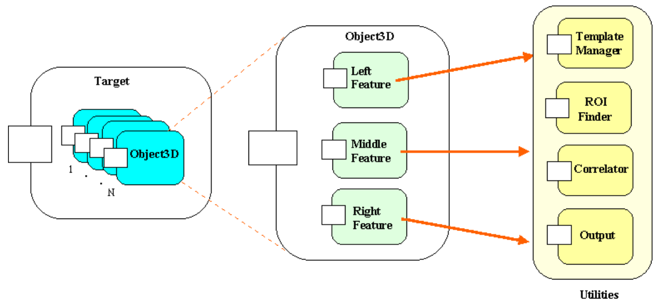


Figure 6.5: A Target, Object3D, and Feature components

Each Object3D object consists of three Feature objects, representing projections of the Object3D in the left, middle and right image streams. Each Feature object has a *locate()* method, which employs the utility components to locate the feature within an image.

6.4.3 The Vision System

The Vision System consists of the following components:

- 3 camera objects;
- A framegrabber. This is an interchangeable component similar to that described in Section 6.3;
- Stereo Extrinsic Parameters, storing the full set of stereo extrinsic parameters, and providing methods to map position vectors between camera reference frames.

6.4.4 Utilities

The utilities shown in Figure 6.4 provide services to the Feature components of each Target. Each utility has its own "toolbox" from which specific utilities can be selected.

6.4.4.1 Correlator

The Correlator provides correlation algorithms. Currently the only Correlator component in the toolbox is the NCC_Correlator, adapted from the Normalised Cross Correlation algorithm implemented by Chanop Silpa-Anan [Fitzgerald, 1999]. The algorithm correlates with gray-scale images, generating a correlation map with values ranging from -1 to 1, with 1 representing the best possible correlation value. Figure 6.6 shows a graphical representation of a correlation map, generated by the NCC_Correlator while searching for a template extracted from the left image in the right image. A clear peak is seen in the correlation map, locating the location of the template.

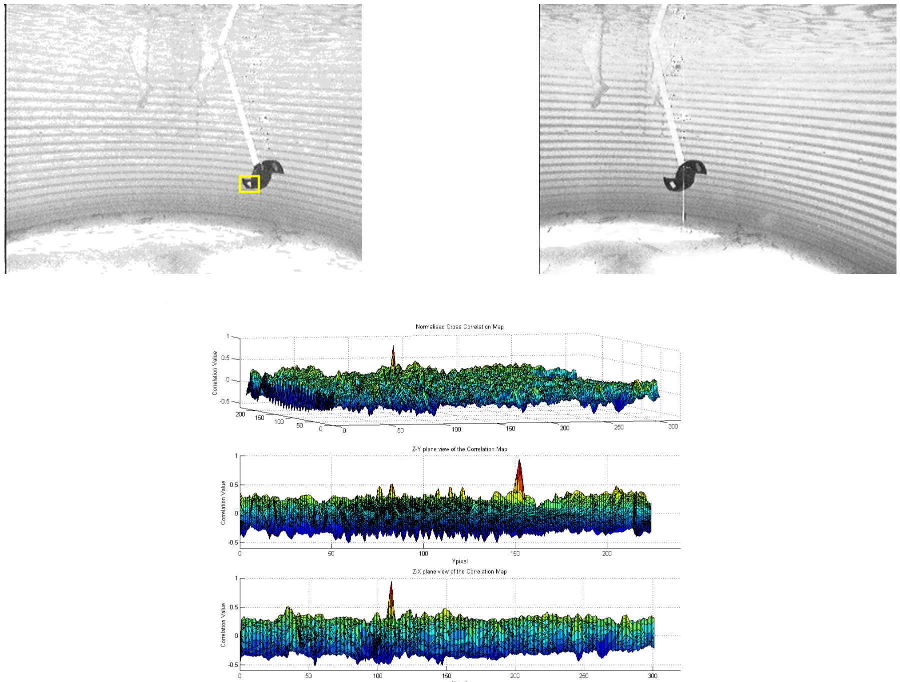


Figure 6.6: A graphical representation of a correlation map generated by normalised cross-correlation, when correlating a template extracted from the left camera image (shown with a yellow border), with the right camera image.

6.4.4.2 Template Manager

The role of the template manager is to store and supply templates for correlation. The simplest template manager will supply the same template for every correlation. More sophisticated managers might use range estimation and correlation values to make intelligent decisions on the best template to supply for correlation. These possibilities are discussed further in Section 7.3.3.

6.4.4.3 Region-of-Interest Finder

The ROI Finder determines the search area for correlation. The simplest ROI Finder will supply a ROI covering the entire image. More sophisticated ROI finders could find search areas using position and velocity information to predict a feature's future location. For optimisation of stereo correlation, the ROI Finder could supply ROIs on the basis of the stereo epipolar constraint [Faugeras, 1996].

6.4.4.4 Output

The Output utility, as its name suggests, handles the task of outputting feature tracking information such as located feature pixel coordinates, and target vectors. Currently the only Output object in its toolbox is a File_Output object which writes the information to a file. When integrated with the rest of Kambara's modules an Output object can be added to the toolbox that writes the appropriate information to a global data structure.

6.5 Range Estimation

Range estimation is the task of using the pixel coordinates of target's projection in stereo image, to calculate a position vector to the target relative to a camera reference frame.

6.5.1 Previous Work

Previous development of range estimation algorithms for the Kambara project has produced algorithms built upon inflexible and unrealistic constraints. [Reynolds, 1998] developed two range estimation algorithms, the first designed for use with parallel cameras. Requiring the stereo cameras to be parallel places severe restrictions on the flexibility of the stereo FOV. Furthermore, the task of aligning the optical axis of each camera is prohibitively difficult. The second algorithm was designed for cameras configured so that optical axes of each camera intersected. Again, it is impractical to do this accurately. A further weakness of these algorithms is that they used the pin-hole model of projection and did not take into account radial lens distortion.

[Fitzgerald, 1999] developed a more sophisticated range estimation algorithm, using the intrinsic parameters of each camera to find the undistorted coordinates corresponding to the observed distorted coordinates. The drawback of the algorithm is that it makes an implicit assumption that the optical axis of each camera intersects, which has already been explained as unrealistic and impractical. Another implicit assumption of the algorithm is that the cameras never move relative to a fixed world reference frame; a severe constraint for a mobile robot.

6.5.2 Range Estimation with Arbitrary Camera Orientation

The following range estimation algorithm, adapted from the work of [Horn, 1996] and [Tsai, 1987], escapes such restrictive assumptions. The only constraint which exists is one which applies to all fixed camera stereo vision: the FOV of each camera must overlap sufficiently for a target to be seen in both images.

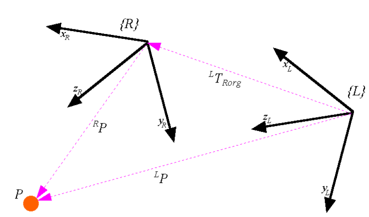


Figure 6.7: The left and right camera reference frames.

The algorithm can be used to calculate a range vector from the left and right reference frame to a target P, $^{L}P_T = \left[{^L}p_x \ {^L}p_y \ {^L}p_z \right]^T$ and $^{R}P = \left[{^R}p_x \ {^R}p_y \ {^R}p_z \right]^T$. There are two main steps involved:

1) In each image, map the observed distorted coordinates to undistorted coordinates using the intrinsic parameter of each camera;
2) Use the stereo extrinsic parameters to map the undistorted pixel coordinates in each image to range vector.

The following sub-sections outline each of these steps. Step 2 is derived in Appendix D.

6.5.2.1 Mapping Distorted Image Coordinates to Undistorted Pixel Coordinates

The intrinsic camera parameters for each camera can be used to map the target projection coordinates (X_f , Y_f) in the captured image to undistorted coordinates (x' , y'), using

$$x' = K_1 X_d^3 + K_1 X_d Y_d^2 + X_d \tag{6-1a}$$

$$y' = K_1 Y_d^3 + K_1 Y_d X_d^2 + Y_d \tag{6-1b}$$

where X_d and Y_d are found from the observed pixel values X_f and Y_f by:

$$X_d = \left(\left(X_f - C_x\right)/s_x N_{fx}\right) N_{cx} d_x \tag{6-2a}$$

$$Y_d = \left(Y_f - C_y\right) d_y \tag{6-2b}$$

6.5.2.2 Mapping Undistorted Image Coordinates to a 3-D Range Vector

Finding the undistorted image coordinates in the left and right stereo images, (x'_L , y'_L) and (x'_R , y'_R), allows use of the simple pin-hole model. Together with the stereo extrinsic parameters and the focal lengths of each camera, f_L and f_R, they are substituted into the following equation:

$${}^L_R R \begin{bmatrix} x'_R / f_R \\ y'_R / f_R \\ 1 \end{bmatrix} {}^R p_z = \begin{bmatrix} x'_L / f_L \\ y'_L / f_L \\ 1 \end{bmatrix} {}^L p_z \tag{6-3}$$

This vector equation is solved to find ${}^L p_z$ and ${}^R p_z$, which can then be substituted into the following equations to find the range vectors from the left and right cameras, ${}^L P$ and ${}^R P$.

$${}^L P = \begin{bmatrix} \frac{x'_L}{f_L} & \frac{y'_L}{f_L} & 1 \end{bmatrix}^T {}^L p_z \quad \text{and} \quad {}^R P = \begin{bmatrix} \frac{x'_R}{f_R} & \frac{y'_R}{f_R} & 1 \end{bmatrix}^T {}^R p_z \tag{6-4}$$

6.6 Testing

Each component of the feature tracker has been individually tested during development. The components were integrated and tested using a simple feature tracking loop. The aim of this test was to check that the components worked together to track the 3-D position of a moving target. A paper fish was moved across the FOV of Kambara's camera suite, and was tracked using a template of the fish's tail shown in Figure 6.8. The same template was correlated in each tracking cycle across both the left and right camera images. Range estimates calculated from the located template coordinates were used to pan and tilt the Sony pan/tilt/zoom camera.

A set of captured images taken during this testing is shown in Figure 6.9. The left and right columns show image sequences captured from the left and right cameras. A target "sight" in each image, superimposed on the image during tracking, demonstrates that the template was successfully located across a sequence of images.

The middle column of images shows the camera image sequence captured from the Sony pan/tilt/zoom camera. Range estimates calculated from the located feature pixel coordinates in the left and right images were used to send commands to the pan and tilt camera to follow the target. The zoom position of the lens was held constant during tracking. Locating the fish tail in each of these highly zoomed-in images implies high range estimate accuracy from the developed range estimation algorithm.

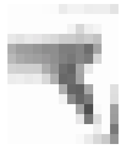


Figure 6.8: A template extracted from the tail of a paper fish

The tracking rate during these tests was calculated at approximately 0.5 Hz. This is slow, but acceptable for a preliminary test performed without any correlation optimisation.

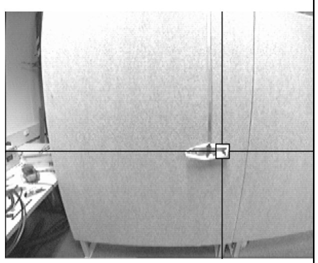
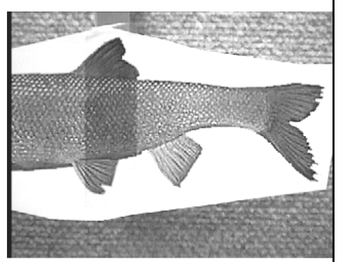
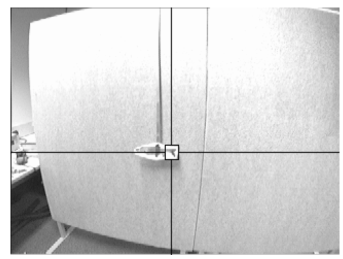
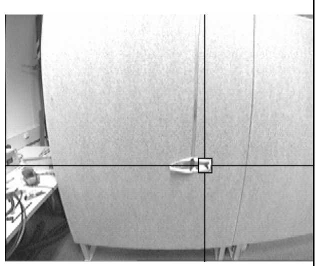
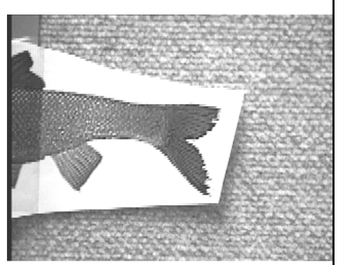
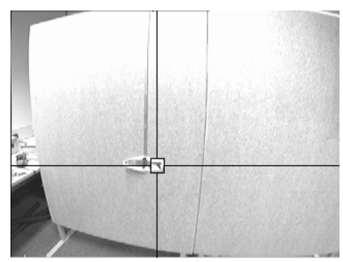
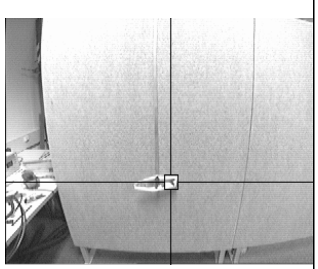
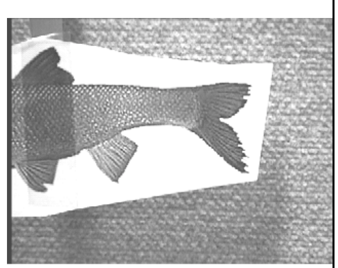
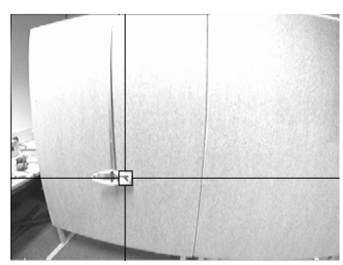
frame	LEFT CAMERA	MIDDLE CAMERA	RIGHT CAMERA
50			
55			
59			

Figure 6.9: Images captured during a feature tracking test. The left, middle, and right camera images are shown. The target “sights” were superimposed on the left and right camera images during tracking.

6.7 Conclusion

These tests indicate the Feature Tracker module is now ready to be used as a platform for feature tracking research. The object-oriented design of the module will provide the flexibility to research a variety of algorithms.

Chapter 7 Conclusion

The broad objective of this project was to contribute both to the Kambara project, and to the 3-D computer vision discipline. The following sections outline the achievements in both areas, followed by a brief discussion on the possible direction of future Kambara 3-D vision research.

7.1 Contributions to the 3-D Computer Vision Discipline

7.1.1 Robust Camera Calibration

Standard camera calibration techniques, traditionally used for surface vision applications, are unreliable in underwater environments. This is caused by the properties of underwater light preventing reliable point *detection* in a calibration pattern image, which in turn prevents simple point *identification* algorithms from being used.

A new point identification algorithm, based on planar projective indexing, was developed to accommodate unreliable point detection. For a calibration pattern with 96 points, the algorithm can theoretically succeed with as few as 16 detected points. Evaluated against a simple point ordering benchmark, the algorithm was found to improve the rate of successful calibration by up to 80%. The algorithm has the additional benefit of filtering out inaccurately located corner-points that may compromise the accuracy of the calibrated parameters.

Though developed for underwater applications, the invariant indexing algorithm will prove useful in any environment where camera images suffer from low contrast or blurriness.

7.1.2 A Calibration Accuracy Evaluation Methodology

Evaluating the accuracy of camera calibration is a challenging task, particularly in underwater applications. The main source of difficulty is the lack of available ground truth for comparison with calibrated camera parameters. Intrinsic camera parameters are a function of the visual medium, and hence there are no "true" parameter values available. Extrinsic parameters can be compared with measurements, but the accuracy of these measurements is limited, and their practicality is limited in underwater applications.

An evaluation methodology was developed to overcome these difficulties, using the calibrated parameters to estimate the dimensions of a 3-D object with an accurately known geometry. Using this approach, calibration accuracy can be inferred from accurate dimension estimates.

7.2 Contributions to the Kambara Project

7.2.1 Reliability Improvements of Kambara's Calibration System

The invariant indexing algorithm was integrated into Kambara's calibration system, replacing a point identification algorithm based on point ordering. Experimental evaluation of the algorithm demonstrated an increase in the success rate of stereo camera calibration from 0% to 80%.

7.2.2 Accuracy Evaluation

Improvements in calibration reliability enabled the accuracy of the system to be evaluated. Using an evaluation methodology based on object dimension estimates, the system was found to calculate camera parameters sufficiently accurate for dimension estimates of objects within a 3 metre range, to an accuracy of 95%.

Kambara's calibration system was adapted to allow object dimension estimates to be made after calibration, as a quick means of evaluating the accuracy of the calibrated parameters. A user selects the object to be measured using a point and click interface, manually correlating points across stereo images.

7.2.3 Zoom Lens Feasibility Study

The possibility of using Kambara's pan/tilt/zoom camera for 3-D vision purposes was investigated. Calibration of the zoom lens was found to be practical only over limited zoom ranges where the entire calibration target can be clearly seen within the captured images. Because a calibrated zoom lens can be only accurately used for 3-D vision applications over the calibrated zoom range, this also limits the functionality of the zoom lens for Kambara's 3-D vision requirements.

Preliminary investigation into lens focusing found that autofocus has a negligible effect on calibrated parameters, and can hence be used for 3-D vision applications without an increase in complexity of the camera model or its calibration procedure.

The unwelcome results of the investigation were that extrinsic parameters of the camera are a function of zoom. The implications of this are that for 3-D vision applications, in order to minimise computation complexity and memory requirements, the zoom lens should be used in isolation from the camera's pan and tilt facilities.

7.2.4 Laying the Foundations for Feature Tracking Module

With Kambara's camera calibration problems largely solved, the path was clear to start work on what calibration is designed to facilitate: range estimation and feature tracking. A flexible, extendible, and conceptually simple design for Kambara's feature tracking module was developed, with the primary goal to lay the foundation for future developers. The building blocks of the feature tracker were implemented, including image acquisition, template correlation, range estimation, and pan/tilt tracking components. These components were individually tested and integrated into a primitive feature tracking loop, which successfully tracks the 3-D position of simple templates, with the target followed by Kambara's pan/tilt camera.

7.3 Future Directions

The breadth of the 3-D computer vision discipline provides a multitude of directions for future development of Kambara's vision system. This section outlines research that logically follows from the work described in this thesis.

7.3.1 Finalising Kambara's Camera Calibration System

With research now completed on Kambara's calibration system, development efforts to follow must focus on integrating the calibration system with the rest of Kambara's software system. While not strictly research itself, this must be completed to facilitate the future research of 3-D algorithms for Kambara. There are two main tasks to be completed:

1) *The calibration software must be ported from its current Microsoft Windows 98 platform to VxWorks, the real time operating system used by Kambara's onboard computer*. This is a considerable task, due to the size of the software system, and its current dependence on Microsoft specific base classes. To be useable by Kambara's software system, the calibration system must also be adapted from its current event loop structure to satisfy the input/output behaviour expected by other Kambara modules.
2) *Memory leaks must be eliminated*. The calibration system has been plagued by memory leaks since its inception in 1999, a consequence of extensive run-time memory allocation/deallocation, associated with linked list processing. Preliminary efforts this year reduced the memory leaks from 400 Kb to 90 Kb per camera calibration. This must be reduced further to allow repeated calibration, and zoom lens calibration, without "eating" Kambara's limited onboard memory.

7.3.2 Pan/tilt Camera Calibration

The next step in preparing Kambara's Sony pan/tilt/zoom camera for 3-D vision applications is to calibrate the pan and tilt facilities of the camera, to determine how the camera parameters vary with pan and tilt motor positions. Because the intrinsic parameters are independent of pan and tilt angles, the camera need only be calibrated once, to determine the extrinsic parameters' dependence of pan and tilt.

This calibration task will not be practical over the full range of pan and tilt angles, since the requirement that the calibration target remain stationary will mean that the target will disappear from the FOV of the camera at large angles. This will not be a serious problem if it can be assumed, or proven, that that relationship between pan/tilt angles with pan/tilt motor positions is linear. The *rotation* extrinsic parameters need then only be calibrated over a small range of pan and tilt angles, with the angle/motor position relationship being used to determine the rotation of the camera reference frame at all other angles.

Because the Sony camera's pan and tilt rotation does not revolve about the camera reference frame, the *translation* extrinsic parameters will also be a function of pan and tilt motor position. The main task here will be to determine the axes of pan and tilt rotation, and may be complicated if the pan and tilt rotation axes are found not to intersect.

7.3.3 Feature Tracking

The focus of future 3-D vision research for Kambara will be the development of the feature tracking module. Further development of the feature tracker can be split into two main research goals:

1) *Robust tracking*, to track dynamic targets over extended image sequences; and

2) *Fast tracking*, to enable tracking of multiple, fast moving targets.

There are several paths to follow in the pursuit of both goals. The avenue of research with perhaps the most potential for robust tracking is the investigation of template management schemes. Such schemes aim to overcome the varying appearance of targets — the appearance of most real-world objects vary with orientation and position. Previous work in this area has focused on *template buffering*, where feature templates are stored and extracted in a buffer on the basis of satisfying a correlation threshold [Reynolds, 1998].

An extension of the template buffering concept is *template indexing*. There are variety of options to explore here: an index of templates could be constructed to be keyed by target range, or perhaps the area of a target's projection within an image. Another option might be to combine buffering and indexing so that keying a template index by target range returns a buffer of templates previously captured at that range.

Possibly the biggest challenge in exploring these techniques will be avoiding *template drift* [Wettergreen et al., 1997]. This problem arises from the accumulation of small correlation errors, or more precisely, in the interpretation of correlation maps. Image noise can often mean that simply picking the maximum correlation value in a correlation map will lead to incorrectly located features. This is motivation for future research into more sophisticated correlation map interpretation schemes.

The obvious place to start research for fast feature tracking is in correlation optimisation strategies, because correlation is the most computationally intensive component of feature tracking. Correlation optimisation strategies aim to minimise the correlation search window, by predicting the future location of the tracked features. In frame-to-frame correlation, i.e. tracking a feature through the image sequence from one camera, this could be accomplished using predictive state estimation techniques. For example, the position and velocity of a feature within a target image could be used to predict the future pixel coordinates of the feature.

In stereo correlation, there is much potential for the reduction of the correlation search area by utilising the *epipolar constraint* of stereo cameras [Faugeras, 1996]. This technique reduces the correlation search area to an *epipolar line*. The stereo extrinsic parameters found from calibration could be used to construct an epipolar index, which when keyed by pixel coordinates in one image, returns the corresponding epipolar line in the other.

References

[Brown, 2000] Brown, G. How Autofocus Cameras Work. http://www.howstuffworks.com/autofocus1.htm 19/6/00.

[Canny, 1996] Canny, J.*A Computational Approach to Edge Detection*. IEEE Trans. on PAMI, Vol. 8, No. 6, pp. 679–698.

[Craig, 1986] Craig, J.J. *Introduction to Robotics – Mechanics and Control.* 2nd Edition. Addison-Wesley Publishing Co., Reading, Massachusetts.

[Faugeras, 1996] Faugeras, O. *Three-Dimensional Computer Vision – A Geometric Viewpoint.* The MIT Press, London. 1996.

[Fitzgerald, 1999] Fitzgerald, I. *A Vision System for an Autonomous Underwater Vehicle*. Department of Engineering ANU. 1999.

[Gamma et. al., 1995] Gamma, E., Helm, R., Johnson, R. and Vlissides, J. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Massachusetts. 1995.

[Gaskett, 1998] Gaskett, C., Wettergreen, D. and Zelinsky, A. *Development of a Visually-Guided Autonomous Underwater Vehicle.* RSL RSISE ANU 1998

[Glover, 1997] Glover, T., Harwood, G.E. and Lythgoe, J.N. *A Manual of Underwater Photography*. Academic Press, London. 1977.

[Horn, 1996] Horn, B.K.P. *Robot Vision.* The MIT Press, Cambridge, Massachusetts. 1996.

[McPherson, 1999] McPherson, C. *Network-based Operator Interface for anUnderwater Robot*. Department of Engineering ANU. 1999.

[Mundy and Zisserman, 1992] Mundy, J. and Zisserman, A. *Geometric Invariance in Computer Vision*. MIT Press. 1992.

[Reynolds, 1998] Reynolds, J. *Autonomous Underwater Vehicle: Vision System*. Department of Engineering ANU. 1998.

[Rothwell, 1995] Rothwell, C. A. *Object Recognition Through Invariant Indexing*. Oxford University Press, New York. 1995.

[Trucco, 1998] Trucco, E. and Verri, A. *Introductory Techniques for 3-D Computer Vision*. Prentice-Hall. 1998.

[Tsai, 1987] Tsai, R. Y. *A Versatile Camera Calibration Technique for High Accuracy 3D Machine Vision Metrology Using Off-the-shelf TV Cameras and Lenses*. IEEE Trans. on Robotics and Automation,Vol. RA-3, No. 4, pp. 323–344. 1987.

[Wettergreen et al., 1997] Wettergreen, D., Thomas, H. and Bualat, M. *Initial Results from Vision-based Control of the Ames Marsokhod Rover.* IEEE International Conference on Intelligent Robots and Systems, Grenoble.September 1997.

[Wettergreen et al., 1999] Wettergreen, D., Gaskett, C. and Zelinsky, A. *Autonomous Guidance and Control for an Autonomous Robotic Vehicle.* 1999.

[Wettergreen, D et al., 2000] *Kambara Software Design*.http://www.syseng.anu.edu.au/rsl/sub/software_design.html 2000.

[Willson, 1987] Willson, R. G. *Modeling and Calibration of Automated Zoom Lenses.*

Appendix A Glossary of Terms

baseline the perpendicular distance between the optical axes of two cameras.

box a rectangle on the calibration pattern.

box corner point a corner point of a calibration pattern box.

box pair a pair of boxes on a calibration pattern (see **box**). The invariant indexing algorithm identifies box corner points by first identifying box pairs (see **box**, **calibration pattern**, **invariant indexing algorithm**).

calibration pattern a 3-D object supplying a set of points with an accurately known geometry. The standard calibration pattern supplies points from the corners of boxes printed on a contrasting background.

camera calibration the task of determining the camera parameters.

camera parameters the parameters which characterise the mathematical model of a camera (see **intrinsic parameters**, **extrinsic parameters**).

camera reference frame a 3-D reference frame used to specify the rotation and orientation of a camera relative to another frame.

charge coupled device a grid of photosensors outputting a voltage proportional to detected light intensity.

correlation the task of locating a template within an image.

correlation map the output of correlation, providing a correlation value for every pixel a template was scanned.

edge a feature of an image which borders sharp variations in image intensity.

edge detection the task of detecting pixels in an image where there is a sharp variation in image intensity.

edgel a pixel element on an edge.

edgel chain a chain of adjacent edgels.

epipolar line a line of possible projected image coordinates of a target in an image from one camera, corresponding to the pixel coordinates of the target in an image from another camera.

extrinsic parameters parameters specifying the rotation and translation of a camera reference frame relative to a world reference frame (see **camera reference frame**).

feature a portion of an image which is of interest. In the context of feature tracking a feature can be defined as the image projection of the 3-D target.

feature tracker module a software module which tracks the 3-D position of targets using feature tracking techniques.

feature tracking the task of tracking an image feature through a sequence of image frames.

FOV field of view. The 3-D area visible by a camera.

frame-to-frame correlation the task of locating a template in a sequence of image frames captured from the one camera.

image reference frame a 2-D reference frame with axes measured in discrete units called pixels. The conventional origin of the frame is the upper left pixel of an image.

intrinsic parameters parameters specifying the internal image projection mechanisms of a camera.

planar projective invariant a property of points which remains constant under projection.

point detection the task of detecting box corner points in an image of a calibration pattern.

point identification the task of identifying box corner points that have been detected in an image.

point ordering a standard point identification technique where points are identified on the basis of their horizontal, vertical, or radial ordering.

range estimation the task of finding the position coordinates of a 3-D target relative to a camera reference frame (see **camera reference frame**).

sensor reference frame a 2-D reference frame with an origin centred on the intersection between the optical axis of the camera lens with the CCD sensor plane.

singleton an object-oriented design pattern, used by classes to ensure an object is only ever instantiated once.

stereo correlation the task of locating a template taken from an image captured by one camera in the image captured by another.

stereo extrinsic parameters parameters specifying the relative rotation and translation of two or more cameras. They are derived from the extrinsic parameters of each camera (see **extrinsic parameters**).

stereo FOV the field of view that is seen by both stereo cameras.

template a small image window representing an image feature, used by correlation techniques.

Tsai's algorithm a standard calibration algorithm which calculates intrinsic and extrinsic parameters by extracting information from corresponding pairs of image and 3-D coordinates.

world reference frame an arbitrarily chosen reference frame external to a camera. For calibration of Kambara's camera suite the world reference frame has been chosen to be centred on a calibration target.

zero normalised cross correlation a standard correlation technique.

Appendix B Camera Specifications

Camera	Pulnix	Sony
Video Outputs	SVideo/Composite	SVideo/Composite
Power Requirements	12V DC, 190 mA	12V-14 V DC, 850 mA
Fixed Intrinsic Parameters	Ncx = 758 Nfx = 640 dx = 0.00635 dy = 0.0074	Ncx = 758 Nfx = 492 dx = 0.00625 dy = 0.00732
Additional Features	2.8mm focal length (approx)	100^{o} pan range 25^{o} tilt range 5.4 - 64.8 mm focal lengt

Table B.1: Camera specifications

Appendix C Derivation of Stereo Extrinsic Parameters

Consider Figure C.1, which illustrates the relationship between the camera reference frames of the left, middle, and right cameras — {L}, {M}, and {R} — and a world reference frame {W}. We wish to find the full set of stereo extrinsic parameters, defining the mapping between every combination of camera reference frames[1]:

$\{L\} \rightarrow \{R\}: ({}^{L}T_{Rorg}, {}^{L}_{R}R)$ $\{M\} \rightarrow \{L\}: ({}^{M}T_{Lorg}, {}^{M}_{L}R)$

$\{R\} \rightarrow \{L\}: ({}^{R}T_{Lorg}, {}^{R}_{L}R)$ $\{M\} \rightarrow \{R\}: ({}^{M}T_{Rorg}, {}^{M}_{R}R)$

$\{L\} \rightarrow \{M\}: ({}^{L}T_{Morg}, {}^{L}_{M}R)$ $\{R\} \rightarrow \{M\}: ({}^{R}T_{Morg}, {}^{R}_{M}R)$

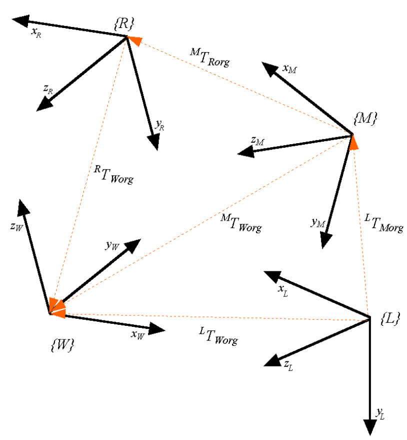


Figure C.1: The relationship between camera reference frames of the left, middle, and right camera — {L}, {M} and {R} respectively — and a world reference frame {W}.

These can be derived from the extrinsic parameters of each camera: $({}^{L}T_{Worg}, {}^{L}_{W}R)$; $({}^{M}T_{Worg}, {}^{M}_{W}R)$; and $({}^{R}T_{Worg}, {}^{R}_{W}R)$. From Figure C.1, we can see the stereo parameters relative the left reference frame {L} are given by:

[1] Notation is adopted from [Craig, 1986].

$$ {}^{L}T_{Rorg} = {}^{L}T_{Worg} - {}^{L}_{W}R \; {}^{R}T_{Worg} \tag{C-1} $$

$$ {}^{L}_{R}R = {}^{L}_{W}R \; {}^{R}_{W}R^{T} \tag{C-2} $$

$$ {}^{L}T_{Morg} = {}^{L}T_{Worg} - {}^{L}_{W}R \; {}^{M}T_{Worg} \tag{C-3} $$

$$ {}^{L}_{M}R = {}^{L}_{W}R \; {}^{M}_{W}R^{T} \tag{C-4} $$

These are the extrinsic parameters outputted by Kambara's calibration system. They can be used to calculate the remaining stereo extrinsic parameters, using the following relations:

$$ {}^{M}T_{Lorg} = -{}^{L}T_{Morg} \tag{C-5} $$

$$ {}^{M}_{L}R = {}^{L}_{M}R^{T} \tag{C-6} $$

$$ {}^{M}T_{Rorg} = -{}^{L}T_{Morg} + {}^{L}_{M}R^{T} \; {}^{L}T_{Rorg} \tag{C-7} $$

$$ {}^{M}_{R}R = {}^{L}_{M}R^{T} \; {}^{L}_{R}R \tag{C-8} $$

$$ {}^{R}T_{Lorg} = -{}^{L}T_{Rorg} \tag{C-9} $$

$$ {}^{R}_{L}R = {}^{L}_{R}R^{T} \tag{C-10} $$

$$ {}^{R}T_{Morg} = -{}^{M}T_{Rorg} \tag{C-11} $$

$$ {}^{R}_{M}R = {}^{M}_{R}R^{T} \tag{C-12} $$

Kambara's Feature Tracker module calculates (C-5) → (C-12) from (C-1) → (C-4) during its initialisation.

Appendix D Derivation of a Range Estimation Algorithm

Consider a stereo camera pair with left and right camera reference frames, denoted {L} and {R} respectively. We wish to find the range vectors

$$^{L}P = \left[{}^{L}p_x, {}^{L}p_y, {}^{L}p_z\right]; \qquad {}^{R}P = \left[{}^{R}p_x, {}^{R}p_y, {}^{R}p_z\right];$$

from the left and right camera reference frames to a point P, where P has *undistorted*[1] projected image coordinates in the left and right camera images at (x'_L, y'_L) and (x'_R, y'_R). The following derivation, based on the work of [Horn, 1996], makes no assumptions about the relative position and orientation of the camera reference frames — the only requirement is that P be positioned within the mutual FOV of both cameras.

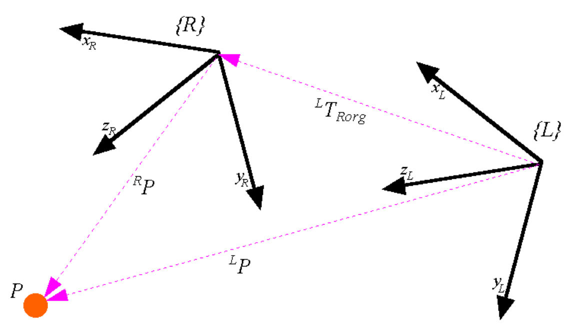


Figure D.1: The left and right camera reference frames and the target point P.

The relationship between *{L}*, *{R}*, and P, shown in Figure D.1, is given by:

$$^{L}P = {}^{L}_{R}R\, {}^{R}P + {}^{L}T_R \qquad \text{(D-1)}$$

where $$^{L}P = \begin{bmatrix} {}^{L}p_x \\ {}^{L}p_y \\ {}^{L}p_z \end{bmatrix}$$ is the translation vector from *{L}* to *{R}*

and $${}^{L}_{R}R = \begin{bmatrix} r1 & r2 & r3 \\ r4 & r5 & r6 \\ r7 & r8 & r9 \end{bmatrix}$$ is defines the rotation of {R} relative to {L}.

Both ${}^{L}T_R$ and ${}^{L}_{R}R$ are provided by camera calibration. (D-1) can be written equivalently as

---

[1] Undistorted coordinates can be calculate from the observed *distorted* coordinates using equations (6-1) and (6-2)

$$ {}^{L}p_{x} = r1.{}^{R}p_{x} + r2.{}^{R}p_{y} + r3.{}^{R}p_{z} + {}^{L}_{R}p_{x} \tag{D-2} $$

$$ {}^{L}p_{y} = r4.{}^{R}p_{x} + r5.{}^{R}p_{y} + r6.{}^{R}p_{z} + {}^{L}_{R}p_{y} \tag{D-3} $$

$$ {}^{L}p_{z} = r7.{}^{R}p_{x} + r8.{}^{R}p_{y} + r9.{}^{R}p_{z} + {}^{L}_{R}p_{z} \tag{D-4} $$

The pin-hole camera model provides:

$$ \frac{x_{L}^{'}}{f_{L}} = \frac{{}^{L}p_{x}}{{}^{L}p_{z}} \qquad \text{and} \qquad \frac{y_{L}^{'}}{f_{L}} = \frac{{}^{L}p_{y}}{{}^{L}p_{z}} \tag{D-5} $$

where f_L is the focal length of the left camera. Substituting (D-5) into equations (D-2), (D-3), and (D-4) we have:

$$ \left(r1\frac{x_{R}^{'}}{f_{R}} + r2.\frac{y_{R}^{'}}{f_{R}} + r3 \right) {}^{R}p_{z} + {}^{L}_{R}p_{x} = \frac{x_{L}^{'}}{f_{L}}.{}^{L}p_{z} \tag{D-6} $$

$$ \left(r4\frac{x_{R}^{'}}{f_{R}} + r5\frac{y_{R}^{'}}{f_{R}} + r6 \right) {}^{R}p_{z} + {}^{L}_{R}p_{y} = \frac{y_{L}^{'}}{f_{L}}.{}^{L}p_{z} \tag{D-7} $$

$$ \left(r7\frac{x_{R}^{'}}{f_{R}} + r8\frac{y_{R}^{'}}{f_{R}} + r9 \right) {}^{R}p_{z} + {}^{L}_{R}p_{z} = {}^{L}p_{z} \tag{D-8} $$

Solving any two of these equations gives ${}^{L}p_{z}$ and ${}^{R}p_{z}$. Applying (D-5) we can now find the required target vectors relative to {L} and {R}:

$$ {}^{L}P = \begin{bmatrix} \frac{x_{L}^{'}}{f_{L}} \\ \frac{y_{L}^{'}}{f_{L}} \\ 1 \end{bmatrix} {}^{L}p_{z} \quad \text{and} \quad {}^{R}P = \begin{bmatrix} \frac{x_{R}^{'}}{f_{R}} \\ \frac{y_{R}^{'}}{f_{R}} \\ 1 \end{bmatrix} {}^{R}p_{z} \tag{D-9} $$

Appendix E Software Design of Kambara's Feature Tracker Module

This appendix provides some further details of the Feature Tracker software design.

E.1 Vision System Class

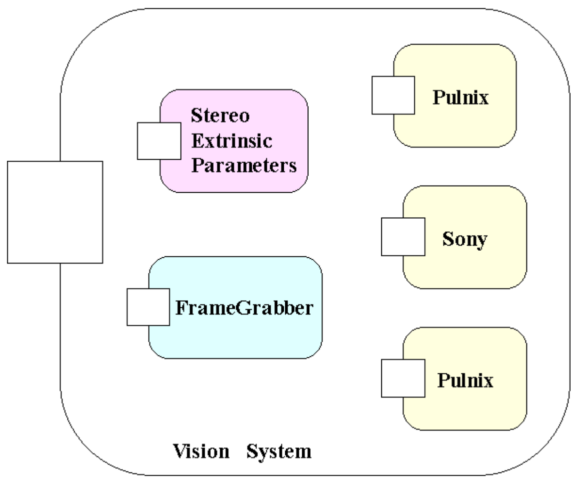


Figure E.1: Components of the Vision System

E.2 Camera Classes

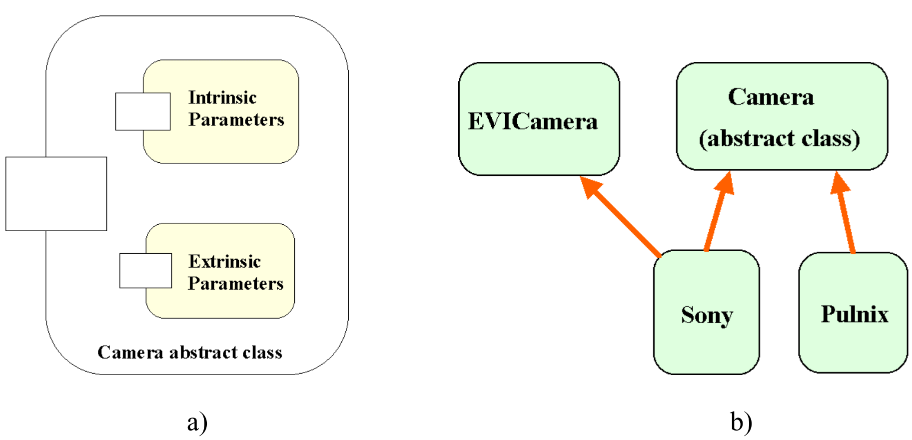


Figure E.2: a) Components of the abstract Camera class

b) class hierarchy for the camera classes

E.3 FrameGrabber Classes

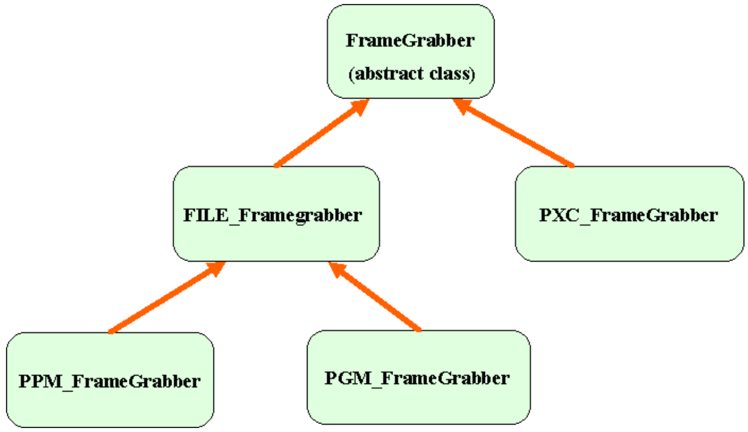


Figure E.3: Class hierarchy for the FrameGrabber classes