TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science

Institute of Artificial Intelligence
Chair of Machine Learning for Computer Vision

Student Research Project

Camera Calibration with a Reference bar

Rajasekar Sankar

Born on: 16th June 1997 in Chennai
Matriculation number: 4986158

1st May 2023

Referee
Prof. Dr. Bjoern Andres

Supervisor
Mr. Holger Heidrich

TECHNISCHE UNIVERSITÄT DRESDEN

Faculty of Computer Science Chair of Machine Learning for Computer Vision

Task for the preparation of a Student Research Project

Course:	M.Sc. Computational Modeling and Simulation
Name:	Rajasekar Sankar
Matriculation number:	4986158
Matriculation year:	2020
Title:	Camera Calibration with a Reference bar

Objectives of work

The first task of the project is to setup a system with at least two cameras to capture the videos of the bar. The second task is to capture the frames of the videos from both cameras on either synchronous or with precise time stamps. The third task is to detect the endings/markers of the reference bar. The fourth task is to do a create a bundle adjustment tool to calibrate both the cameras and to refine the camera parameters (intrinsic and extrinsic parameters).

Focus of work

- Recherche & Analyse
- Entwicklung eines Konzeptes & Anwendung der entwickelten Methodik
- Dokumentation und grafische Aufbereitung der Ergebnisse

Referee:	Prof. Dr. Bjoern Andres
Supervisor:	Mr. Holger Heidrich
Issued on:	20th November 2022
Due date for submission:	2nd May 2023

Mr. Holger Heidrich
Supervising professor

Statement of authorship

I hereby certify that I have authored this document entitled *Camera Calibration with a Reference bar* independently and without undue assistance from third parties. No other than the resources and references indicated in this document have been used. I have marked both literal and accordingly adopted quotations as such. There were no additional persons involved in the intellectual preparation of the present document. I am aware that violations of this declaration may lead to subsequent withdrawal of the academic degree.

Dresden, 1st May 2023

Rajasekar Sankar

Abstract

Camera calibration is an important process in the computer vision and photogrammetry, used for retrieving the information like orientation and position of the target objects. In many industrial applications, pinhole cameras are used for 3D measurements especially for depth estimation using the stereo system. But for accurate results it is very important to calibrate all the cameras using the calibration object. This paper describes a methodology to stereo-calibrate the intrinsic and extrinsic parameters (including distortion coefficients) of the two stationary pinhole cameras simultaneously using a 3D calibration object. The reference bar of known length is moved through the object space for N observations of the dynamic events with the detectable marker endings attached on both ends is captured using the left and right cameras for synchronous time stamps. An object detection algorithm is defined by masking the frame for the area of interest using HSV colour space to obtain the image points from the sphere centres, which is the marker endings on either side of the bar. The solution is developed using the bundle adjustment technique, Levenberg-Marquardt algorithm is used for minimising the least square error estimation. Experiments are performed for the planar pattern (checkerboard) and reference bar of known length on both same and different camera devices of the stereo system and the results were studied.

Contents

Symbols and Acronyms

WYSIWYG	What You See Is What You Get
CGV lab	Chair of Computer Graphics and Visualization
GPU	Graphics Processing Unit

1 Introduction

1.1 Brief overview

Camera calibration is the process of determining the internal camera geometric and optical characteristics (intrinsic parameters) and the 3D position and orientation of the camera frame relative to a certain world coordinate system (extrinsic parameters) [2]. The calibration goal is to establish the correspondence between the computer's perception of the 2D image coordinates and the 3D world coordinates. This relationship provides the 2D and 3D information that can be inferred from one another. Camera calibration should be carried out for any application that requires the relationship between a 2D image and a 3D environment. Examples include robotic vision-based 3D sensing and measuring, manufacturing inspection, automated assembly, etc. Due to this requirement, various calibration approaches have been developed for different calibration objects. This section provides a quick overview of the calibration techniques that have been developed throughout the years for camera calibration.

1.2 Literature review

Hans-Gerd Maas [2] presented an image sequence-based, fully automatic and rather flexible procedure for the calibration of stationary multi-camera systems for 3-D observation of dynamic events. While conventional close-range camera calibration techniques are either based on a stable point-field with known reference coordinates or on a temporarily stationary point-field with only approximately known 3-D coordinates, which is imaged from different locations and under different orientations with one single camera, the proposed technique is based on stationary cameras and moving targets, making use of the image sequence acquisition nature of most solid-state cameras. This single-marker method does not allow for the determination of the interior orientation. Thus, for full camera orientation and calibration, a reference bar of known length is moved through object space, with the problem of feature identification and establishment of multi-view correspondences being reduced to the tracking of two targets.

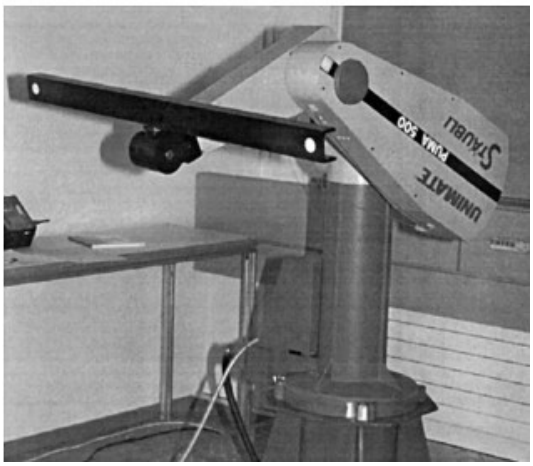

Figure 1.1: Moved reference bar

The advantages of this method over conventional self calibration techniques are the trivial establishment of multi-view correspondences, the fact that no temporarily stable target field was constructed, and each camera was set up only once. The necessity of the establishment of a temporarily stable target field is part of the actual task in most stationary applications of digital close-range photogrammetry but may be cumbersome in dynamic applications. Moreover, photogrammetric systems for 3-D data acquisition in dynamic processes will usually consist of multiple cameras imaging events simultaneously, with each camera to be calibrated individually. The term calibration of a multi-camera system is understood as the orientation and calibration of each individual camera of the system. The task of orientation can then be reduced to the establishment of correspondences sequentially at a number of image points, exploiting the redundancy of stereo imaging.

This method of “moved reference bar” is based on moving a bar of known length, which is imaged by all cameras at a number of locations or orientations over the observation volume. The self-calibrating bundle adjustment uses the constant length of the reference bar as additional geometric constraint information thus greatly strengthening the solution and enabling full calibration of each camera in a multicamera system, including the interior orientation parameters [2].

Several examples for the application of the “moved reference bar” method can be found in the literature: Heikkila 1990 shows via simulations, that the calibration of a four-camera system based only on intersections at 141 object points is not possible, while after the introduction of 38 distance observations the full interior orientation can be determined. In Pettersen 1992, the method is used with pre-calibrated cameras only for the determination of exterior orientation parameters, improving scale control over the network. Maas 1997a shows an application of the technique in photogrammetric industrial robot calibration with a reference bar moved to 27 random positions for the calibration of a three-camera system [2].

The “moved reference bar” method can be considered a versatile and reliable method for the calibration of photogrammetric systems consisting of multiple solid-state cameras. The multi-ocular image sequences of a reference bar acquire the known length of the reference bar can be used as additional observations in self-calibrating bundle adjustment. The analysis of simulations and a practical example has shown that good results can be achieved with

a total of 25-50 reference bar locations or orientations, which are preferably randomly distributed over the observation volume [2].

Zhengyou Zhang [3] in his paper proposes a new calibration technique using 1D objects (points aligned on a line), thus filling the missing dimension in calibration. In particular, the author says that camera calibration is not possible with free moving 1D objects but can be solved if one point is fixed. A closed-form solution is developed if six or more observations of such a 1D object are made. For higher accuracy, a nonlinear technique based on the maximum likelihood criterion is then used to refine the estimate. According to the dimension

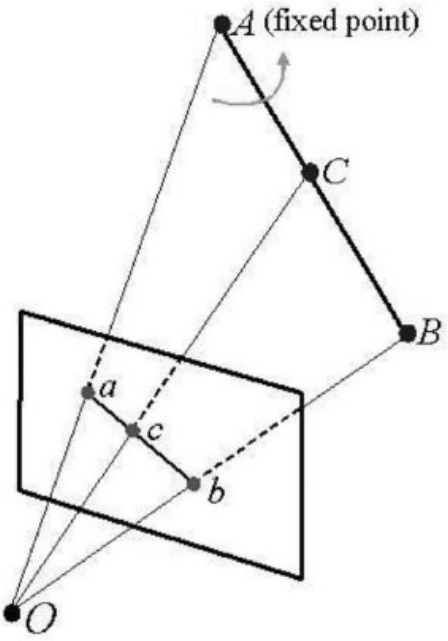

Figure 1.2: Illustration of 1D calibration objects

of the calibration objects the other classifies the calibration techniques into categories [3].

3D reference object-based calibration. Camera calibration is performed by observing a calibration object whose geometry in 3D space is known with very good precision gives very efficient calibration [4]. Typically, the calibration object is made up of two or three orthogonal planes. Sometimes, a plane undergoing a known precise translation is also employed [5], providing 3D reference points in an equivalent manner. This method necessitates a complex setup and an expensive calibrating device.

2D plane-based calibration. This group of techniques calls for the observation of a planar pattern displayed in a variety of orientations [6],[7]. This setup is easier for camera calibration.

Self-calibration. Techniques in this category require only image point correspondences and can be regarded as 0D approach since they do not use any calibration object. Thus, the rigidity of the static scene provides two constraints [8], [9] on the camera intrinsic parameters. The correspondences between three images are enough to recover the internal and external parameters, allowing us to rebuild the three-dimensional structure up to a similarity [10], [11]. A huge number of parameters must be estimated despite the absence of calibration objects, which creates a significantly more challenging mathematical task [12].

1D object-based calibration: It consist of three or more collinear points with known relative positioning. If one camera is put at the front of a room and another in the back, it is difficult to calibrate the cameras using 3D or 2D calibration object, but it becomes possible with the 1D object [3].

Zhengyou Zhang [6] in the paper "A Flexible New Technique for Camera Calibration" proposes a flexible technique to calibrate a camera. This method just calls for the camera to look at a planar pattern from at least two different orientations. Either the camera or the planar pattern can be moved, and it is not necessary to be aware of the motion. Modeling of radial lens distortion is also done. The method starts with a closed-form solution and then refines it nonlinearly using the maximum likelihood criterion.

Jose´Alexandre de Franca et al. [18] provides a novel calibration method that groups the cameras into binocular sets. The fundamental matrix of every pair of binoculars is then estimated, from the projective calibration of every camera. Then the procedure is transformed by updating the calibration for Euclidean space to determine the intrinsic and extrinsic parameters of the camera. Without limiting the pattern's freedom of movement or requiring any prior knowledge of the cameras or movements, the calibration is achievable.

The possibility to calibrate multiple cameras at once is the main benefit of employing 1D patterns for calibration. This is because that the 1D pattern points are captured simultaneously by cameras with widely different points of view. However, in this method it is necessary that at least one of the cameras, called the "reference camera", is already calibrated. Reference camera is marked as camer0 and other cameras are called as secondary cameras [18].

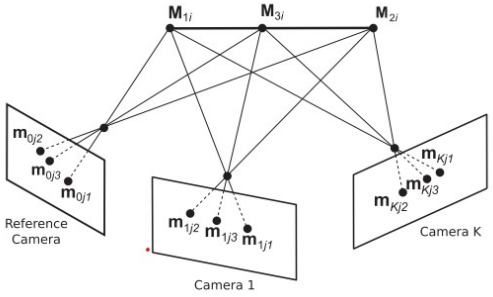

Figure 1.3: Calibration scheme for a set of n cameras and m points

The solution is obtained using linear methods that give an initial estimation of the parameters of the camera and then nonlinear methods based on the criterion of maximum likelihood is defined to refine the initial parameters of the camera [18]

Levente Hajder et al. [13] proposes a method of using 3D spherical calibration object for calibrating the cameras, sphere centres are used as image points for the calibration. Chessboards and other planar calibration targets are frequently employed for this camera calibration [14] [15], but recently spherical calibration objects [16] have also been proposed. The core premise is that depth sensors can reliably and automatically detect spheres. If at

least four sphere centres are located, extrinsic parameters can then be estimated via point-set registration techniques [17]. Due to the sparseness of the point cloud, as determined by a depth camera, planar target identification is unfortunately erroneous [14].

2 Preliminaries

2.1 Notation

The 2D image point coordinates is denoted by $m = [u, v]^T$. The 3D object point coordinates is denoted by $M = [X, Y, Z]^T$. The augmented vector of the image point and the object point are denoted as $\tilde{m} = [u, v, 1]^T$, $\tilde{M} = [X, Y, Z, 1]^T$ respectively. For the pinhole type of camera model, the relationship between 3D point *M* and its image projection *m* (perspective projection) is given by,

$$s\tilde{m} = K[Rt]\tilde{M} \tag{2.1}$$

where,

$$K = \begin{bmatrix} fx & \lambda & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix}$$

where, f is focal length, *c* is principal point, $(\lambda) is skew factor.$

[R, t] is called the *extrinsic parameters*, it is the rotation and translation of the camera reference coordinate system with respect to the world reference coordinate system.

K is called the *intrinsic parameter*, that link the pixel coordinates of the image point with the corresponding coordinates in the camera reference system.

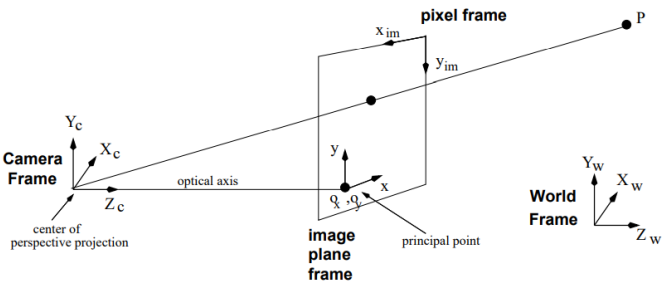

Figure 2.1: Coordinate system

Fig. 2.1 shows the different coordinates systems of the world frame *W*, image frame *im* and camera frame *C* in a different coordinate axes representation.

To study the relationship between the coordinate axes transformation between the object and the image space, the camera parameters has to be derived.

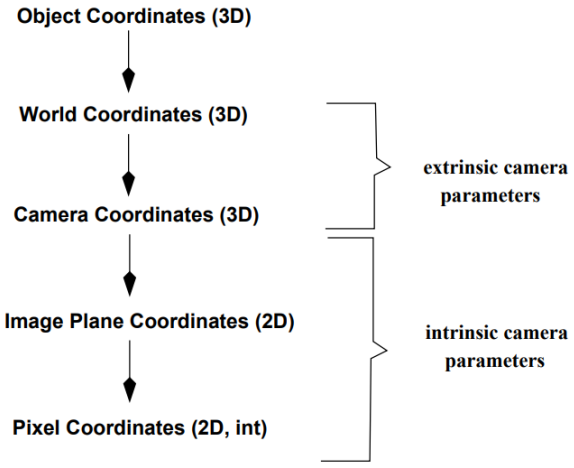

Figure 2.2: Coordinate system transformation

Fig. 2.2 gives the relationship between different coordinate systems of the object and the pixel points. Thus to understand the relationship between the object and image coordinates it is necessary to obtain the extrinsic and intrinsic camera parameters. Thus to obtain the camera parameters, camera calibration is the technique to be performed.

2.2 Stereo geometry

2.2.1 Standard setup

The stereoscopic standard setup consists of two cameras called left and right cameras separated at a distance of *b* baseline and positioned at projection centers *Cl, Cr*. In the standard setup the the optical axes of the camera are parallel to each other where there is no rotation of the image planes is observed. The focal length of both the cameras are taken to be equal $f_l = f_r$.

Fig. 2.3 shows the the standard canonical stereo setup. From the Fig. 2.3, the depth Z of the world point P is calculated from the disparity, d *is the difference between the left and right image coordinates.*

The depth is determined from d, f, b *parameters which is described in the below equation,*

$$[\ Z = \frac{bf}{d} \tag{2.2}$$

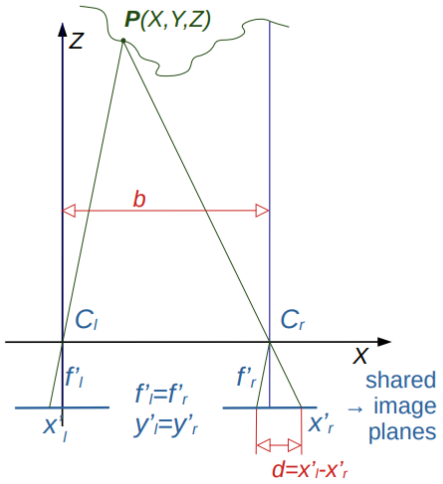

Figure 2.3: Standard canonical stereo setup

2.2.2 General setup

In General setup, the left and right cameras are rotated relative to each other by the Rotation matrix *R*. From the triangulation of rays projected from the object points to the image points, *epipolar plane*, thus the *epipolar line* is determined.

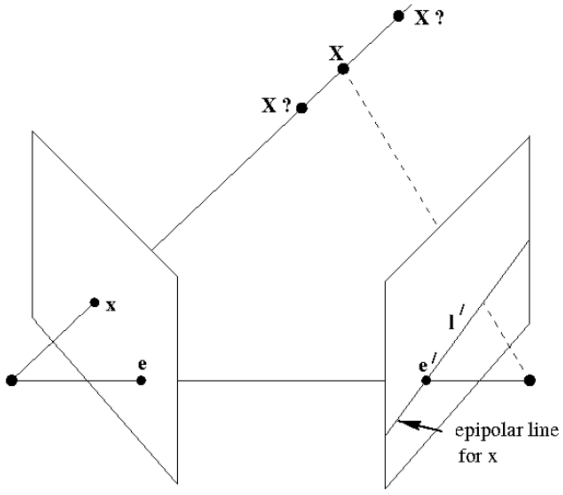

Figure 2.4: General canonical stereo setup

2.2.3 Basic equations

The problem of uncalibrated stereo camera is system is solved by applying the following procedure:

1. Initial estimation of intrinsic parameters using closed form solution.
2. Epipolar geometry relation between right and left camera images.
3. Finding the dense correspondences by 1D search .
4. Estimating the fundamental matrix

5. Estimate the essential matrix.
6. Decompose the rotation and translation matrices
7. Using corresponding image pairs triangulate to find the depth

Epipolar geometry

From the principle of Epipolar geometry, the epipolar constraint is defined,

$$x_l(t\dot{X}x_l) = 0 \tag{2.3}$$

Thus the relation between the right and left image points are derived as,

$$\begin{bmatrix} x_l \\ y_l \\ z_l \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x_r \\ y_r \\ z_r \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Thus the essential matrix relation is given by,

$$\begin{bmatrix} x_l & y_l & z_l \end{bmatrix} \begin{bmatrix} e_{11} & e_{12} & e_{13} \\ e_{21} & e_{22} & e_{23} \\ e_{31} & e_{32} & e_{33} \end{bmatrix} \begin{bmatrix} x_r & y_r & z_r \end{bmatrix} = 0$$

Dense correspondences

By apply 1d search problem over the epipolar line the dense correspondences of the image points are derived

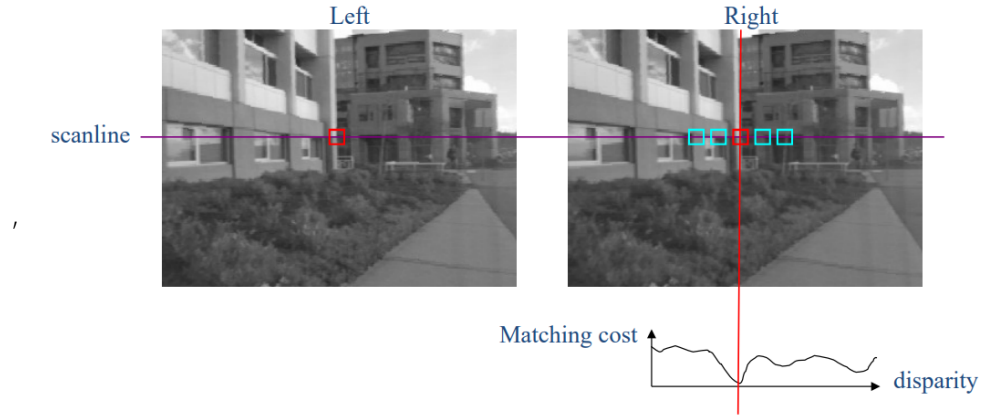

Figure 2.5: Dense correspondences

Estimating the fundamental matrix

From the set of corresponding features between the left and right images, the fundamental matrix relation is obtained as,

$$\begin{bmatrix} u_l & v_l & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} \begin{bmatrix} u_r \\ v_r \\ 1 \end{bmatrix} = 0$$

Thus forming the linear system of equations for the above matrix the following equation is obtained

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ x_mx'_m & x_my'_m & x_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} f11 \\ f21 \\ \dots \\ f33 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \dots \\ 0 \end{bmatrix}$$

Thus by applying least squares solution F matrix is derived.

Estimating the Essential matrix

Once F matrix is derived, essential matrix is obtained from the following relation,

$$E = K_l^T \dot{F} K_r \tag{2.4}$$

Then applying Singular value decomposition R and t are extracted from E

$$E = TxR \tag{2.5}$$

Computing the depth

From the relations,

$$\tilde{u_l} = P_l \dot{\tilde{x}_r} \tag{2.6}$$

$$\tilde{u_r} = M_i \dot{n} t \tilde{x_r} \tag{2.7}$$

From these imaging equations, depth is determined by least square solution using pseudo inverse,

$$x_r = (A^T A)\dot{}^{-}1 A^T b \tag{2.8}$$

Geometry constraint

In Chapter 4, Conceptual contribution, a reference bar of known length is used as the calibration object for calibration purpose. So it is necessary to define the constraint and no of equations required to get the solution

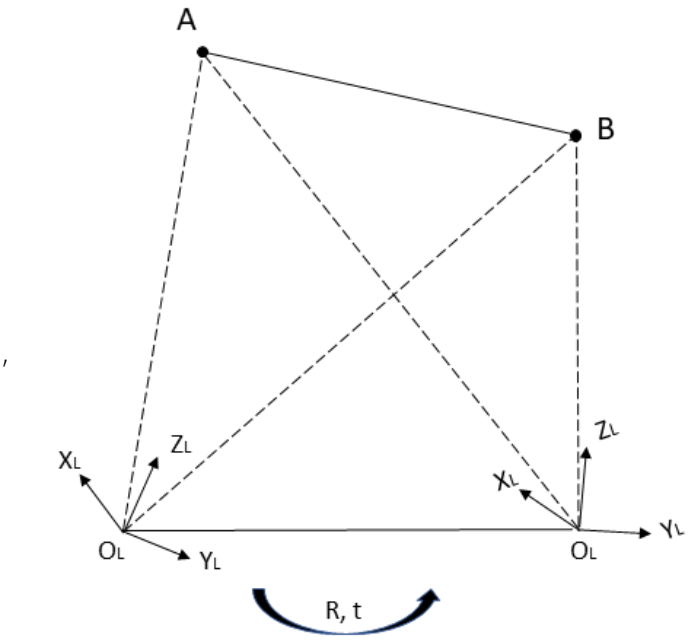

Figure 2.6: Geometry constraint

Fig. 2.6 shows the calibration geometry from two cameras *L, R* for a bar AB of length L. Thus the geometric constraint for the collinear feature points A,B ($A_j, B_j) denotes the location for the jth image pair$,

$$\|(A - B)\| = L \tag{2.9}$$

Equation system

Given N observations of the stick, we have five intrinsic parameters and 5N parameters for the point positions to estimate for each camera. Then we have 9 Rotation values. 3 translation unknowns, and 5 position unknowns so totally 17 unknowns. For two collinear points of known distances, the total no of unknowns is 17+5N unknowns for each camera, so for two camera totally 34+10N unknowns are to be determined. Each image point gives 2 equations, so for 2 image points 4 equations are obtained, for N images, from 2 cameras totally 16N equations are obtained.

Therefore n>=6 images are required to construct necessary equations that satisfy the unknowns.

3 Problem Statement

3.1 Problem Definition

The problem of Camera Calibration with a Reference bar is to compute the camera intrinsic and extrinsic parameters based on a number of points whose object points in the world coordinate system (Xw, Yw, Zw) are known and whose image coordinates (Xi, Yi) are measured. Thus, from the object and the image points, the camera parameters are initialized and optimised by reducing the reprojection error through nonlinear refinement based on the maximum likelihood criterion.

3.2 Task description

The goal of the project is to write a whole calibration function that fits in the framework of OpenCV. Calibration of several cameras in a large room by moving and tracking the reference bar of known length through object space.

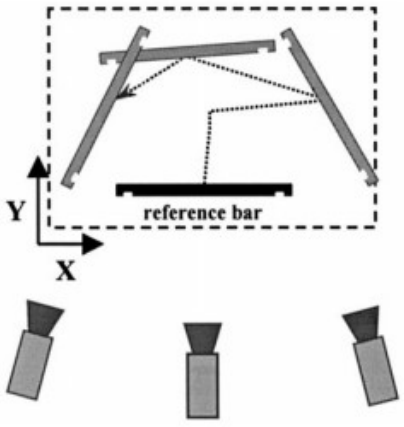

Figure 3.1: Configuration for Multicamera callibration

The first task of the project is to setup a system with at least two cameras to capture the videos of the bar. The second task is to capture the frames of the videos from both cameras on either synchronous or with precise time stamps. The third task is to detect the endings/markers of the reference bar. The fourth task is to do a create a bundle adjustment tool to calibrate both the cameras and to refine the camera parameters (intrinsic and extrinsic parameters)

1. Construct a stereo camera system for the calibration purpose.
2. Setup the system to capture synchronous images at precise time stamps for every nth second.
3. Take N images from both the cameras and save and label the images.
4. Build a reference bar(stick) of known length with the markers (spheres) attached on both ends.
5. Paint the markers (spheres) for the marker detection problem
6. Extract the pair of coordinates (sphere centres) from the masked AOI (Area of Interest) for both camera images.
7. Apply bundle adjustment technique to determine and optimize the camera parameters.
8. Compare the results with standard planar calibration patterns.

3.3 Methodology approach: Overview

This section briefly describes the methodology approached to solve the problem:

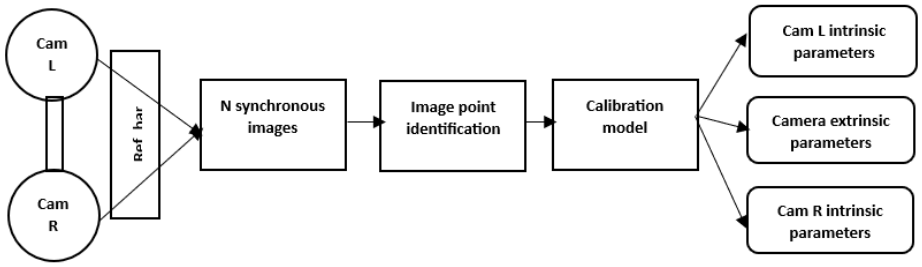

Figure 3.2: Flowchart for methadology approach

The methodology to address the Stereo camera calibration problem using a reference bar,

1. Feature matching of the marker endings from left and right camera images.
2. Initial estimation of camera parameters using the planar calibration object (checker-board) from closed form solution (analytical solution).
3. Estimation of Fundamental matrix.
4. Estimation of Essential matrix from Fundamental matrix
5. Perform Linear Triangulation using Chirality condition.
6. Non-linear triangulation for pose estimation using PnP.
7. Estimation of reprojection error (least squared error)
8. Levenberg- Marquardt- Bundle adjustment to optimize the camera parameters.
9. Comparison of camera parameters of the stereo system for same and different camera manufacturer devices.

3.4 Challenges Faced

Synchronization of Guppy F033C Cameras

The cameras of the MLCV Data lab [1] are used for capturing the synchronous images, but challenges were faced in the camera trigger process and was unable to configure the cameras. The two Guppy F033 cameras (85 fps, Pixel size: 9.9 µm × 9.9 µm) are connected to the computer and the API tool of Allied Vision Camera – Vimba SDK is used to configure the parameters of the camera. The source for the camera input is provided by the pulse

Figure 3.3: Guppy camera setup

Figure 3.4: Camera setup: front view

generator and the signal is observed using the oscilloscope.For triggering of both cameras to capture the synchronous images the setup is defined s follows:
Input: PC—<USB cable>—camera
Output: camera—<interface cable (Pin8='External GND' to GND
Pin4='Camera In 1' to 5V)>—Power supply 5V

Figure 3.5: Oscilloscope

Figure 3.6: Pulse generator

Master-Slave hardware triggering configuration:
The corresponding master output (one of the pins 1-3 with LineSource=ExposureActive) is connected to the input of the slave camera (pin 4). The same applies to the external GND of all cameras with each other (pin 8). For configuration, set TriggerMode=On for all cameras

and for the master camera in the DigitalIOControl section as follows (for example, for Line4):
LineSelector=Line4
LineSource=ExposureActive
The camera are configured for the creation of the hardware trigger signal but the due to the hardware failure of one of the input signal of the camera the trigger signal was unable to generate and the cameras are unable to configure.

Software trigger of the cameras were tried for the exposure signal, but it was not arrived. Hence as an alternate for this, two web cameras were connected and triggered to obtain the synchronous images.

Marker detection for Cube endings

Initially cube marker objects are used for the marker detection of the reference problem, but challenges were faced in the detection of the marker detection in identifying the cube centres.

7cmX7cmx7cm cube Polystyrene is fabricated for the marker ending. Patterns are marked for the feature detection. To detect the features, the edges of the cube are marked and patterns like chess board patterns were used for the feature detection.

To detect the cube centre the marker is pre-processed for morphological operations as follows:

1. Canny-edge detection algorithm is applied to the Gaussian blurred image to feature the edges of the cube using the method of L2 gradient method.canny = cv2.Canny(blurred, 50, 200)
2. Hough transform is used to find and draw the lines on the edges of the cube. lines = cv2.HoughLinesP(canny, 1, np.pi/180, 100, minLineLength=100, maxLineGap=10).
3. LSD algorithm is used to detect the edges using the Line segment detection algorithm. lsd = cv2.createLineSegmentDetector(0) lines, ,,$=lsd.detect(blur)$.

But due to the insufficient feature point detection unable to locate the centre of the cube for the marker detection problem, so as an alternate for this sphere (7cm) is used as a marker ending of the reference bar in this project.

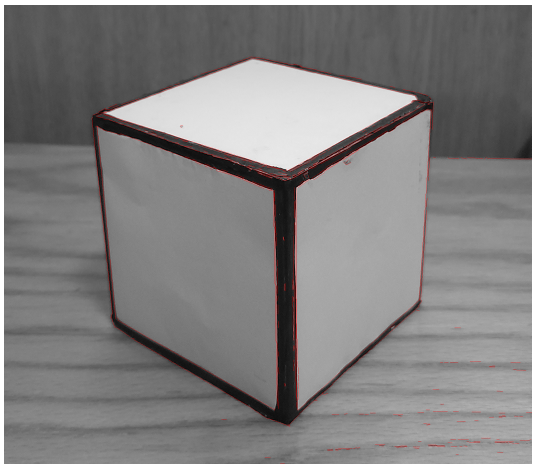

Figure 3.7: Line segment detector

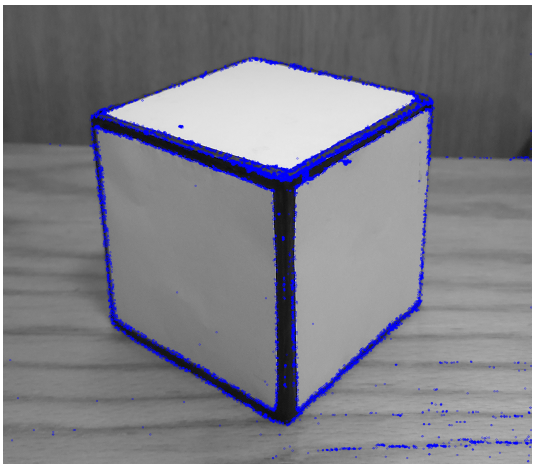

Figure 3.8: Canny edge detection

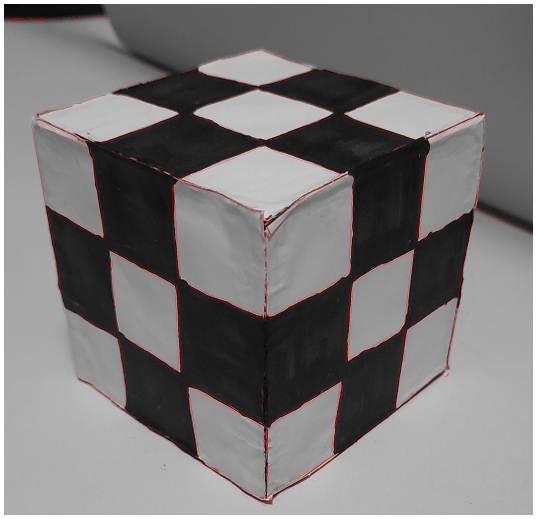

Figure 3.9: Pattern structure: edge detection

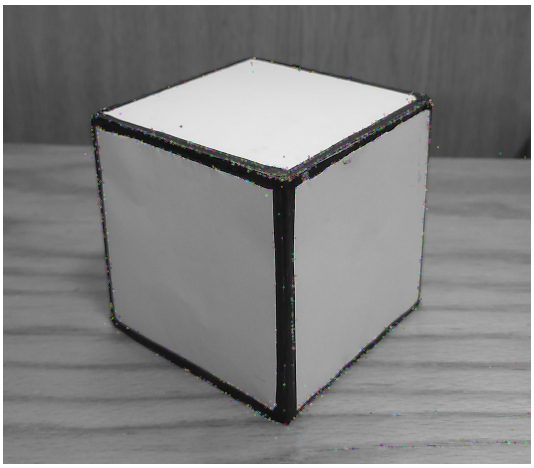

Figure 3.10: Edge detection: SIFT keypoints

4 Conceptual Contribution

The approach to calibrate the stereo camera system using the reference bar marker detection problem is described briefly in the flow chart below:

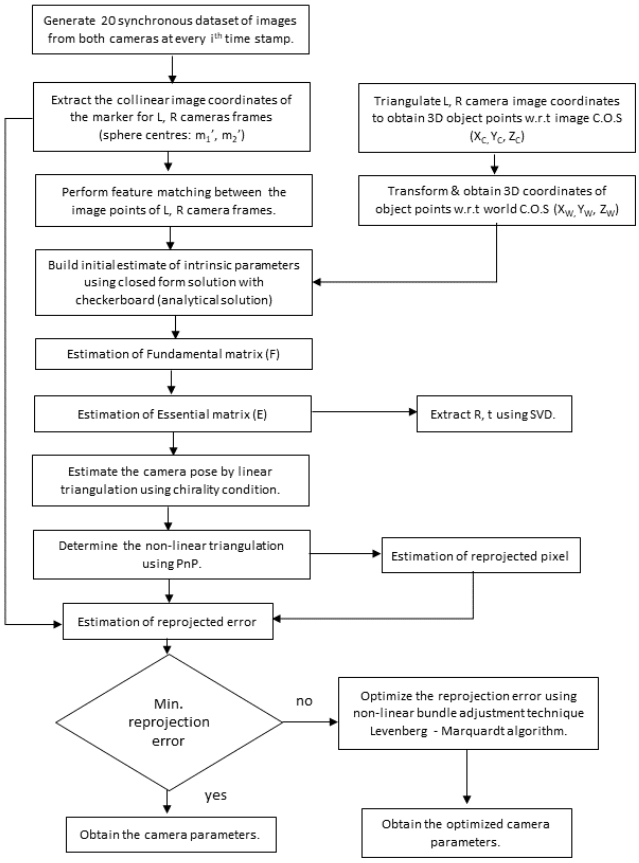

Figure 4.1: Stereo-camera calibration approach flowchart

4.1 Experimental setup

This method of stereo camera calibration approach uses two pinhole-cameras, whose camera intrinsic and extrinsic parameters has to be calibrated.

The approach was studied, and results were compared for both same and different camera device pair. The camera pair is labelled as left camera (L) and right camera (R) devices. In same camera system both L and R camera devices are the same manufacturer camera devices, while in the different camera system L and R camera device are different manufacturer camera devices.

In this experimental study Trust 17318 Cuby webcam and Dicota Webcam PRO Plus Full HD, D31841 camera devices were used for the calibration purpose.

Specifications	Trust 17318 Cuby webcam	Dicota Webcam PRO Plus Full HD, D31841
Resolution	HD 720p (1280x720 pixels)	Full HD 1080p (1920x1080 pixels)
Frame rate	30 frames per second	30 frames per second
Field of view	60-degree diagonal	Full 60-degree diagonal
Lens	Fixed focus lens	Fixed focus lens
Sensor (WxH)	3.0mm x 1.7mm	3.0mm x 1.7mm

Table 4.1: Camera specifications comparison

In same camera system, Trust 17318 Cuby webcam is used for both Left and Right cameras Whereas, in different camera system, Dicota Webcam is used for Left camera and Trust 17318 Cuby webcam is used for the Right camera and both the cameras are connected with USB 2.0 connectivity to the system for controlling the system. The two cameras are placed separated by the base width distance (b) of 70mm

Figure 4.2: Same camera system

Figure 4.3: Different camera system

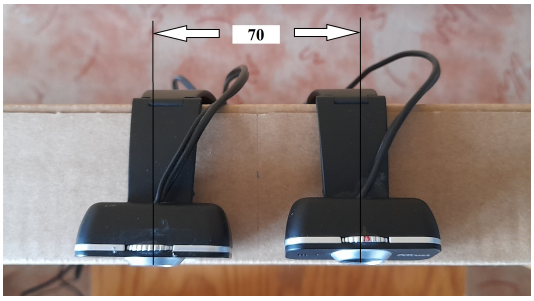

Figure 4.4: Baseline width

4.2 Capturing of synchronous images

For the calibration approach it is necessary to capture synchronous images at precise time stamps from both the cameras. OpenCV commands were used to capture the images at precise time stamps.

After initializing both the cameras, frame rate of the cameras is to be obtained using the 'get()' function and the 'CAP_PROP_FPS' property to calculate the time interval between frames in seconds for both the cameras. The current time of the frames are obtained using the 'time()' function from the 'time' module. Then the timestamps for the frames are calculated based on the time interval and start time with the duration for the capturing images.

The function is looped through the timestamps to capture frames from both cameras at the corresponding timestamp using the 'set()' function and the 'CAP_PROP_POS_MSEC' property to set the position of the video stream in milliseconds.

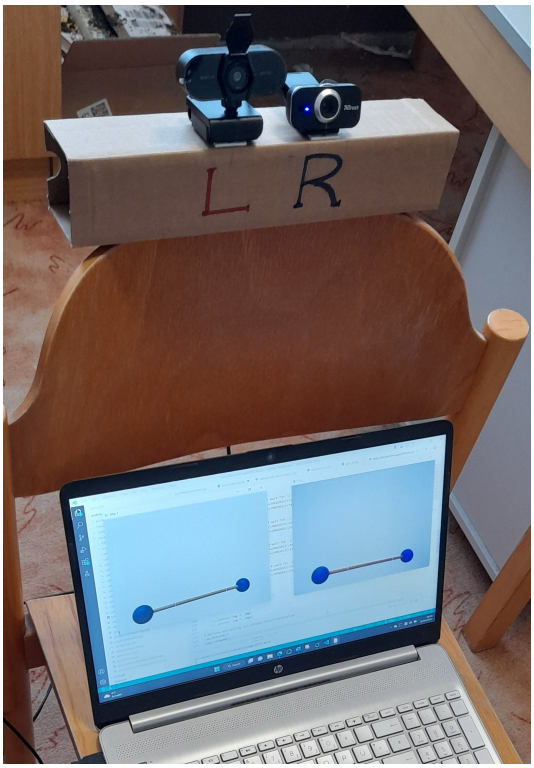

Figure 4.5: Synchronous image capturing

Once the frames at precise timestamps are captured, both the frames are saved for the image points detection. The timestamp precision of the frames is limited by the cameras frame rate.

The algorithm to capture the synchronous images at precise time stamps is defined below:

Algorithmus 1 : Capturing synchronous images at precise timestamp

```
Initialize camera 1
Initialize camera 2
Get the frame rate of camera 1 and camera 2
Calculate the time interval between frames
Get the current time
Calculate the timestamps for the frames based on the time interval and the start time
for each timestamp do
    Set the positions of the video streams to the corresponding timestamp
    Read frames from camera 1 and camera 2
    if frames were successfully captured then
        Save the frames with the corresponding timestamp
    end
    else
        Break out of the loop
    end
end
Release camera 1
Release camera 2
```

4.3 Calibration Object

This paper defines the study of two different calibration objects used for the experimentation. A 2D planar calibration object – checkerboard and 3D calibration object - reference bar of known length is used to obtain the image points for the calibration purpose.

The following section briefly describes the calibration methodology used for both calibration objects.

4.3.1 2D planar calibration object – checkerboard

A 7 x 7 squares checkerboard pattern of square size 22mm is used for the calibration purpose. The calibration procedure is as follows:

1. Generate dataset of images by using the Calibration Pattern in different views
2. Get the object points of the object from the pattern dimensions and each square dimensions.
3. Define a variable for the intrinsic parameters by defining focal length (fx, fy), principal point (cx, cy), distortion coefficients (k1, k2), scale factor (s) parameters.
4. Get the image points of the chess board pattern, from the corners using the function ‘cv2.findChessboardCorners’. Then redefine the corners using the function ‘cv2.cornerSubPix’. Append all the corner points to the image points variable.
5. Display the corners identified from 1. On the image using the function ‘cv2.drawChessboardCorners’.
6. Find homography H between world plane Z=0 and image plane

7. Using this homography and the image of absolute conic, solve equations to estimate the image (omega) of the absolute conic.
8. Estimate the initial estimate for intrinsic parameters (K) using closed form solution.
9. Build an initial estimate of R and t matrices using K and H. Condition matrix R to be orthonormal. Convert the matrix R to a vector Rvec using Rodrigues formula.
10. Optimize the geometric error between the actual corners and projected coordinates of world coordinates in the image plane using Levenberg - Marquardt algorithm.
11. Refine all intrinsic and extrinsic camera parameters by minimizing the maximum likelihood estimate error function.
12. Estimate the reprojection error for before and after optimization results.

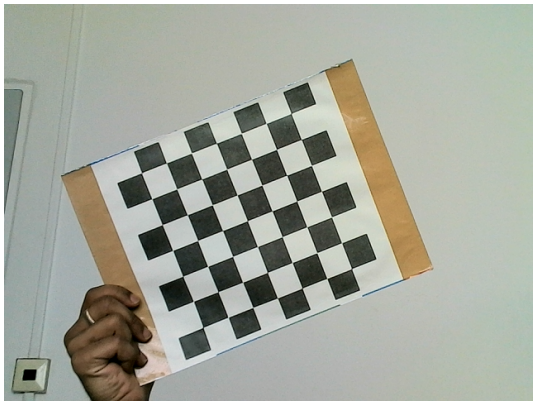

Figure 4.6: Left camera image

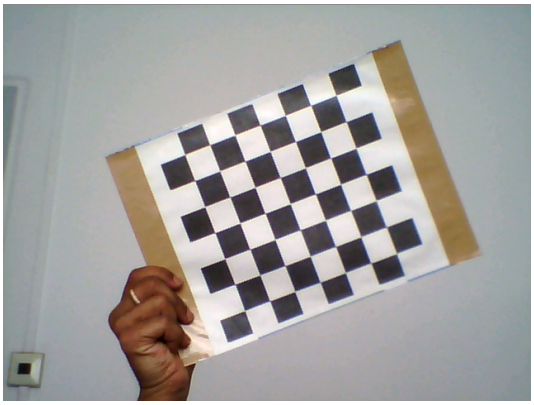

Figure 4.7: Right camera image

A dataset of 50 images for different object poses of the checkerboard is taken from each left and right cameras using both same and different camera system and the results were discussed.

4.3.2 3D calibration object - reference bar of known length

In this study, a 3D calibration object is used for the calibration purpose. A reference bar of known length is used as the calibration object. A 15mm diameter, 500mm length wooden stick is used as the reference bar. For the marker identification a 70mm sphere is connected at the ends of the stick to determine the point-to-point distance of the refence bar. The spheres are painted blue for the masker detection using HSV colour mask. Thus, by detecting the sphere centre using marker detection algorithm the reference bar length is obtained.

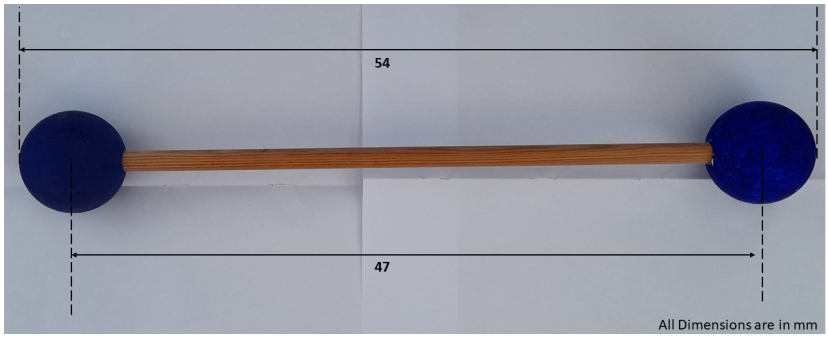

Figure 4.8: Synchronous image capturing

The calibration procedure is as follows:

1. Generate dataset of 20 synchronous images by using the
2. Calibration object in different orientations and positions.
3. Extract the image coordinates using the Marker detection algorithm.
4. Extract the object points using the triangulation.
5. Estimate Fundamental matrix.
6. Estimate Essential matrix from fundamental matrix.
7. Estimate the camera pose.
8. Perform linear triangulation.
9. Perform nonlinear triangulation to get the 3D object points.
10. Calculate the mean error and reprojection error.
11. Apply bundle adjustment: Levenberg-Marquardt Optimization

From the 'Algorithm 1: Capturing synchronous images at precise timestamp', 20 synchronous images are taken for every 5th second from the left and right cameras and for same and different cameras. All the images are stored separately with unique file name description.

4.4 Calibration Methodology

4.4.1 Masker detection: Extraction image coordinates

In this section, the image coordinates of the reference points are derived by applying the marker detection algorithm for both left and right camera images.

The image is masked for the Area of interest using the HSV colour detection. The image is converted from BGR to HSV colour space, upper and lower HSV ranges are defined to apply the colour mask. So, this range masks the image and detects the marker's area.

Morphological operations like dilate and closing operations are performed to remove any noises in the image for the masking operation.

Once the image is masked for the region of interest, the centre of the sphere is obtained by identifying the contours in the image. The masked images have the contours with the blue scale range mentioned in the HSV range scale.

From all the contours to discriminate the sphere contours, area condition is used. From the principle of studying the sphere areas if placed at a distance from 100-500cm from the camera's position, the area of the sphere appears to be grater than 1000. With this condition, contours with area > 1000 are filtered.

Then to identify the centres, moments of the contour is calculated, which gives the cx and cy values of the sphere centres. Then to append the sphere centres another condition of (cx-cy) <500 is used to identify both sphere centres. Thus, the sphere centre image coordinates for left right images are obtained.

Algorithmus 2 : Masker detection: Extracting image coordinates

```
input synchronous image: .pngfile
hsv.img ← convert img to HSV color space
upper_HSV ← [130, 255, 255]
lower_HSV ← [90, 70, 0]
mask ← apply color mask to hsv.img with upper_HSV and lower_HSV
kernel ← create 5 × 5 kernel with all ones
dilate ← apply dilation operation to mask with kernel
closing ← apply morphological closing operation to dilate with kernel
contours ← find contours in closing
centers := empty list [ ]
for contour in contours do
    area ← calculate area of contour
    if area > 1000 then
        draw contour on mask with [0, 255, 0]
        M ← calculate moments of contour
        cx ← intM['m10']/M['m00']
        cy ← intM['m01']/M['m00']
        if (cx − cy) < 500 then
            append (cx, cy) to centers
            draw circle on img at center (cx, cy) with color [0, 0, 255]
output Marker detection image coordinates: C1,C2
```

4.4.2 Extracting object points : left-right image triangulation

To measure the coordinates of object points placed in the space with respect to the world coordinate system placed in the left camera, the following steps are used:

1. Define the origin and orientation of the world coordinate system. Typically, the origin is chosen to be the position of the left camera and the orientation is such that the x-axis points to the right, the y-axis points upwards, and the z-axis points away from the camera.
2. Identify the object points in your left camera image and right camera image and extract their pixel coordinates.
3. Use the function cv2.triangulatePoints() to triangulate the object points in 3D space. This function takes the projection matrices of the left and right cameras and the pixel coordinates of the object points as inputs and returns their 3D coordinates in homogeneous coordinates.
4. To transform the 3D coordinates of the object points from the camera coordinate system to the world coordinate system, the extrinsic parameters of the left camera, which describe the translation and rotation of the camera with respect to the world coordinate system is used. of the left camera using the function cv2.decomposeProjectionMatrix() from OpenCV.
5. Remove the homogeneous coordinate by dividing the 3D coordinates by the last element.

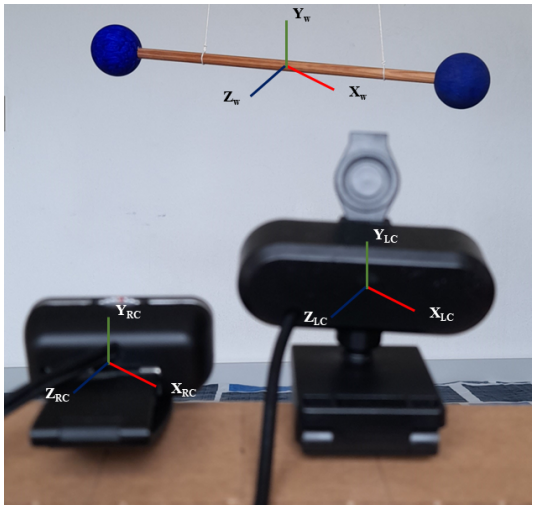

Figure 4.9: World coordinate system

Figure 4.10: Camera coordinate system

Algorithmus 3 : Extracting object points : left-right image triangulation

1 *left_img_pts* ← [C1x, C1y], [C2x, C2y]
2 *right_img_pts* ← [C1x, C1y], [C2x, C2y]
3 Triangulate the object points in 3D space
4 proj_matrix_left = $\begin{bmatrix} fx & 0 & cx & 0 \\ 0 & fy & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
5 proj_matrix_right = $\begin{bmatrix} fx & 0 & cx & tx \\ 0 & fy & cy & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$
6 obj_pts_3d_homogeneous = triangulate[proj_matrix_left, proj_matrix_right, left_img_pts, right_img_pts]
7 $translation\_left, rotation\_left = decomposeProjectionMatrix(proj_matrix_left)$
8 M = concatenate((rotation_left, translation_left.reshape(-1, 1)), axis=1)
9 obj_pts_3d_world_homogeneous = dot(M.T, obj_pts_3d_homogeneous) + translation_left.reshape(-1, 1)
10 obj_pts_3d_world = obj_pts_3d_world_homogeneous[:3] / obj_pts_3d_world_homogeneous[3]
11 output$Objectcoordinatesw.r.t.worldC.O.S$: *X, Y, Z*

4.4.3 Estimate the Fundamental Matrix

From the left and right image points matching correspondences, the fundamental matrix is calculated by forming the linear equation as below:

$$\begin{bmatrix} x_i' & y_i' & 1 \end{bmatrix} \begin{bmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{bmatrix} = \begin{bmatrix} x_i & y_i & 1 \end{bmatrix}$$

The linear system of equations is defined below:

$$\begin{bmatrix} x_1x'_1 & x_1y'_1 & x_1 & y_1y'_1 & y_1 & x'_1 & y'_1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ x_mx'_m & x_my'_m & x_m & y_my'_m & y_m & x'_m & y'_m & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{1,n+1} \\ a_{2,n+1} \\ \dots \\ a_{n,n+1} \end{bmatrix}$$

By applying Singular Value Decomposition (SV, we derive the fundamental matrix (f)D) to the above equation.

Algorithmus 4 : Estimate_F_matrix

Input : img1_pts, img2_pts

Output : F

normalize points
$x1$ = img1_pts[:,0]
$y1$ = img1_pts[:,1]
$x1dash$ = img2_pts[:,0]
$y1dash$ = img2_pts[:,1]
A = np.zeros((len(x1),9))
for *i in range(len(x1))* do
    $A[i]$ = np.array([$x1dash[i] * x1[i]$, $x1dash[i] * y1[i]$, $x1dash[i]$, $y1dash[i] * x1[i]$, $y1dash[i] * y1[i]$, $y1dash[i]$, $x1[i]$, $y1[i]$, 1])
end
$U, E, V =$ SVD(A)
F_est = $V[-1, :]$
F_est = $F\_est.reshape(3, 3)$
$ua, sa, va =$ SVD(F_est)
sa = diag(sa)
$sa[2, 2]$ = 0
$F = \text{dot}(ua, \text{dot}(sa, va))$
F = F / F[2,2]

4.4.4 Estimate the Essential Matrix

Once the camera intrinsic parameters (K) and Fundamental matrix (F) are derived then the Essential matrix can be estimated by the equation:

$$Eest = K^T FK \tag{4.1}$$

By doing the SVD to Eest, Essential matrix E is calculated. From essential matrix pose can be estimated, where the R and t are extracted from E using SVD.

Algorithmus 5 : Estimate E_Matrix from F_Matrix and K

Input : K, F

Output : E

$E_{est} \leftarrow K^T \cdot F \cdot K$

$U, S, V \leftarrow \text{SVD}(E_{est})$

$S \leftarrow \text{diag}(S)$

$S_{0,0}, S_{1,1}, S_{2,2} \leftarrow 1, 1, 0$

$E \leftarrow U \cdot S \cdot V$

4.4.5 Estimate the camera pose

Essential matrix composes of Rotation and translation matrices of the camera. Decomposition of the E matrix using singular value decomposition gives the camera pose configurations.

$$C1 = U(:,3) and R1 = UVW^T \tag{4.2}$$

$$C2 = -U(:,3) and R2 = UVW^T \tag{4.3}$$

$$C3 = U(:,3) and R3 = UV^TW^T \tag{4.4}$$

$$C4 = -U(:,3) and R4 = UV^TW^T \tag{4.5}$$

Algorithmus 6 : Estimating the camera pose

Input : E

Output : R, T

$U, S, V \leftarrow \text{SVD}(E)$

$W \leftarrow \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

$R_1 \leftarrow U \cdot W \cdot V$

$R_2 \leftarrow U \cdot W \cdot V$

$R_3 \leftarrow U \cdot W^T \cdot V$

$R_4 \leftarrow U \cdot W^T \cdot V$

$T_1 \leftarrow U[:, 2]$

$T_2 \leftarrow -U[:, 2]$

$T_3 \leftarrow U[:, 2]$

$T_4 \leftarrow -U[:, 2]$

$R \leftarrow [R_1, R_2, R_3, R_4]$

$T \leftarrow [T_1, T_2, T_3, T_4]$

for $i \leftarrow 0$ to 3 do

    if $\det(R[i]) < 0$ then

        $R[i] \leftarrow -R[i]$

        $T[i] \leftarrow -T[i]$

4.4.6 Triangulation of image points

From camera intrinsic parameter (k) and extrinsic parameters (R, T) the projection matrix is calculated and then homogenous coordinates for projection matrix is calculated. From homogeneous projection matrix and image points, the triangulating rays are directed to the intersect in the world coordinate system which gives the object space 3d coordinates.

Algorithmus 7 : point_triangulation(k, $pt1$, $pt2$, $R1$, $T1$, $R2$, $T2$)

Input : Camera intrinsics matrix k, image points $pt1$ and $pt2$, rotation matrices $R1$, $R2$, and translation vectors $T1$, $T2$

Output : 3D points of the object

Initialize an empty list $points\_3d$
Create a 3x3 identity matrix I
Reshape $T1$ and $T2$ to a 3x1 matrix
Calculate projection matrices $P1$ and $P2$ using k, $R1$, $T1$, and $R2$, $T2$ respectively
Create a homogeneous coordinate system for image points xy and xy_cap by concatenating ones to $pt1$ and $pt2$ matrices
for i *in range* $(0, length(xy))$ **do**
    Initialize an empty list A
    x = $xy[i][0]$, y = $xy[i][1]$,
    x_cap = $xy\_cap[i][0]$, y_cap = $xy\_cap[i][1]$
    Append $(y * p3 - p2)$ to A
    Append $(x * p3 - p1)$ to A
    Append $(y\_cap * p3\_cap - p2\_cap)$ to A
    Append $(x\_cap * p3\_cap - p1\_cap)$ to A
    Create a 4x4 array A from list A
    Compute the Singular Value Decomposition (SVD) of A to obtain u, s, and v
    Extract the last row of v to obtain $x\_$
    Normalize $x\_$ by dividing by its last element
    Append $x\_$ to $points\_3d$
end
Return the $points\_3d$ array

4.4.7 Linear triangulation

From n 3D points, 2D correspondences (x) in the image, intrinsic camera parameters (K) the 6-DOF camera pose is determined using linear least squares. The camera pose provides the three parameters of the rotation (roll, pitch, yaw) and a 3dimensional translation vector with respect to the world coordinate system.

The pose estimation problem is known as Perspective-n-Point (PnP). From normalized image coordinates (u,v) , corresponding homogeneous point in world coordinate system (X_tilda), $the Projection matrix is calculated. This can be written in the following equation$,

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} P1 \\ P2 \\ P3 \end{bmatrix} \tilde{X}$$

The camera projection matrix P can be used to determine the Rotation matrix R and Translational vector t. By applying singular value decomposition to projection matrix R,t is calculated.

Algorithmus 8 : linear_triangulation(R_Set, T_Set, $pt1$, $pt2$, k)

Input : Rotation matrices R_Set, translation vectors T_Set, image points $pt1$ and $pt2$, camera intrinsics matrix k

Output : 3D point set of the object

1 Create a 3x3 identity matrix $R1\_$ and a 3x1 zero matrix $T1\_$
2 Initialize an empty list $points\_3d\_set$
3 for i *in range* $(0, length(R\_Set))$ do
4 | Compute the 3D points of the object using $R\_Set[i]$, $T\_Set[i]$, $pt1$, $pt2$, k and $R1\_$, $T1\_$ as inputs and append the points to $points\_3d\_set$
5 end
6 Return the $points\_3d\_set$ array

4.4.8 Nonlinear optimization

Linear triangulation is obtained by the minimization of algebraic distance which is physically not meaningful. So, to refine the solution of camera parameters, nonlinear optimization through maximum likelihood inference is used to optimize the parameters.

From the projection matrix, and the image points the reconstructed 3d point new dataset is calculated using the least square optimization with the initial guess for the loss function.

Algorithmus 9 : Non-linear triangulation

Input : Rotation matrices R_1, R_2, translation vectors T_1, T_2, 2D image points $pt1$, $pt2$, 3D object points X, camera intrinsic matrix K, and number of iterations k

Output : Reconstructed 3D object points X

$I \leftarrow$ identity matrix of size 3×3
$P_1 \leftarrow K \cdot [R_1| - T_1]$
$P_2 \leftarrow K \cdot [R_2| - T_2]$
$points3D\_new\_set \leftarrow$ empty list
for $i \leftarrow 1$ **to** $len(X)$ **do**
    $opt \leftarrow$ least squares optimization of $loss$ function with initial guess $X[i]$ and arguments $pt1[i]$, $pt2[i]$, P_1, P_2
    $points3D\_new \leftarrow$ optimized 3D point
    append $points3D\_new$ to $points3D\_new\_set$
end
return $points3D\_new\_set$

For the loss function, the mean error will be calculated. The reprojection error is calculated from the image points (u,v), Projection matrices (P1, P2, P3) and the homogeneous object 3d point coordinates (Xtilda) which is expressed in the below equation.

$$e = \left(u - \frac{P_1^T \tilde{X}}{P_3^T \tilde{X}}\right)^2 + \left(v - \frac{P_2^T \tilde{X}}{P_3^T \tilde{X}}\right)^2$$

Algorithmus 10 : Calculate the mean error of the 3D points

Input : $R1, T1, R2, T2, pt1, pt2, X, k$

Output : e

$R1 \leftarrow$ reshape$(R1, (3, 3))$
$T1 \leftarrow$ reshape$(T1, (3, 1))$
$R2 \leftarrow$ reshape$(R2, (3, 3))$
$T2 \leftarrow$ reshape$(T2, (3, 1))$
$I \leftarrow$ identity(3)
Calculate projection matrices
$P1 \leftarrow k \times R1 \times$ hstack$(I, -T1)$
$P2 \leftarrow k \times R2 \times$ hstack$(I, -T2)$
$e \leftarrow$ []
for $i \leftarrow 1$ *to* *len*(X) **do**
$error \leftarrow$ loss$(X[i], pt1[i], pt2[i], P1, P2)$
e.append$(error)$
return mean(e)

Thus, to determine the loss function for n images and m points of the correspondences, the maximum likelihood estimate is used to minimize the loss function, the eqution is given by,

$$\sum_{i=1}^{n}\sum_{j=1}^{m}\|(m_{ij}-\hat{m}(A,R_i,t_i,M_j))\|^2 \tag{4.7}$$

is the is the projection of the point Mj in the image I, mij is the image coordinates. Thus, the error is calculated for the image and projected image points using the L2 normalization.

$$e1 = \left\|(u_1 - u_1')\right\|^2 + \left\|(v_1 - v1')\right\|^2 \tag{4.8}$$

$$e2 = \left\|(u_2 - u_2')\right\|^2 + \left\|(v_2 - v2')\right\|^2 \tag{4.9}$$

Then error average is obtained by dividing the appended error to the length of the image points and from it the reprojected points is calculated by takin the square root of the error average.

Algorithmus 11 : Reprojection error loss function

Input : 3D point X, image point (u_1, v_1) in camera 1, image point (u_2, v_2) in camera 2, projection matrices P_1 and P_2

Output : Reprojection error

`#Reshape projection matrices to` 3×4

1 $p_{11}, p_{12}, p_{13} \leftarrow P_1$

2 $p_{21}, p_{22}, p_{23} \leftarrow P_2$

3 $p_{11}, p_{12}, p_{13} \leftarrow$ reshape(p_{11}, p_{12}, p_{13})

4 $p_{21}, p_{22}, p_{23} \leftarrow$ reshape(p_{21}, p_{22}, p_{23})

`#Calculate the image points in camera 1`

5 $u_1' \leftarrow \frac{p_{11}X}{p_{13}X}$

6 $v_1' \leftarrow \frac{p_{12}X}{p_{13}X}$

`#Calculate the image points in camera 2`

7 $u_2' \leftarrow \frac{p_{21}X}{p_{23}X}$

8 $v_2' \leftarrow \frac{p_{22}X}{p_{23}X}$

`#Calculate the reprojection error`

9 error$_1$,e1 = $\|(u_1 - u_1')\|^2$ + $\|(v_1 - v1')\|^2$

10 error$_2$,e2 = $\|(u_2 - u_2')\|^2$ + $\|(v_2 - v2')\|^2$

11 $total\_error \leftarrow error_1 + error_2$

12 lossFunc, $error\_function = error\_mat.append[(total\_error)/(len(imagepoint)]$

13 $$error\_average = \frac{\sum_{i=0}^{n}\sum_{j=0}^{m} error\_mat[i][j]}{(len(imagepoint) * X)}$$

14 $error\_reprojection \leftarrow \sqrt{error\_average}$

15 return $error\_reprojection$

4.4.9 Bundle Adjustment : Levenberg-Marquardt Optimization

In this final step of camera calibration, refining of the camera pose is done using the method of Bundle adjustment by refining the camera poses and 3D points by minimizing the reprojection error.

This nonlinear technique uses the lm method, for this rotation matrix R is represented as 4-dimensional quaternion q. To refine the estimation of the camera pose, nonlinear least squares solver from scipy library is used to minimize the reprojection error.

Thus, from the nonlinear refinement, the optimized camera pose and 3d object point coordinates with respect to the world coordinate system.

Algorithmus 12 : Bundle Adjustment : Levenberg-Marquardt Optimization

Input : pose_set, X_world_all, map_2d_3d, K
Output : pose_set_opt, X_world_all_opt

$n\_cam \leftarrow$ number of cameras
$n\_3d \leftarrow$ number of 3D points
$indices \leftarrow$ list of indices of 3D points
$pts\_2d \leftarrow$ 2D points of all cameras
$indices\_cam \leftarrow$ list of camera indices
$x0 \leftarrow$ initial estimate of parameters
$A \leftarrow$ sparse matrix with sparsity pattern defined by $indices$ and $indices\_cam$
$result \leftarrow$ least_squares(fun=loss, x0=x0, jac_sparsity=A, verbose=2, x_scale='jac', ftol=1e-4, method='trf', args=(n_cam, n_3d, indices, pts_2d, indices_cam, K))
$param\_cam \leftarrow$ camera parameters from $result$
$X\_world\_all\_opt \leftarrow$ optimized 3D points from $result$
$pose\_set\_opt \leftarrow$ dictionary of optimized camera poses
for *each* cp *in* $param\_cam$ do
    $R \leftarrow$ rotation matrix from quaternion in $cp[:4]$
    $C \leftarrow$ translation vector from $cp[4:]$
    append (R, C) to $pose\_set\_opt$
return $pose\_set\_opt$, $X\_world\_all\_opt$

The method of camera calibration defines a initial guess of intrinsic parameters using closed form solution, then the parameters are optimized using the non linear optimization of camera intrinsic parameters through estimation of Fundamental matrix, essential matrix and triangulation of image points which gives the reprojection error. And finally, the nonlinear optimization of the reprojection error using Levenberg-Marquardt Optimization gives the optimized world points and camera pose.

5 Emperical Contribution

The cameras of the MLCV Data lab [1] are used for capturing the synchronous images, but challenges were faced in the camera trigger process and was unable to configure the cameras. The two Guppy F033 cameras (85 fps, Pixel size: 9.9 µm × 9.9 µm) are connected to the computer and the API tool of Allied Vision Camera – Vimba SDK is used to configure the parameters of the camera. 5.4

Figure 5.1: Quadric error metric simplification applied to the Stanford bunny

[3]

[1]

(a) Caption Figure 1

(b) Caption of Figure 2 which should be so long to se if the two captions are not equal in length

Figure 5.3: first figure

Figure 5.4: second figure

6 Conclusion and Further Work

7 Acknowledgement

This project work was supported by the chair of *Machine Learning for Computer Vision* chair at the Institute of Artificial Intelligence of the Faculty of Computer Science of TU Dresden. The author gratefully acknowledges all the support and guidance provided by Prof. Dr. Bjoern Andres and Mr. Holger Heidrich for the supervision throughout the course of this work. The author also gratefully acknowledges Ms. Sylvia Wuench for supporting in the administrative activities and for providing the access to work in the MLCV Data lab for the project.

List of Figures

List of Tables

A Appendix I

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

B Appendix II

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language. Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Bibliography

[1] Reimar K Lenz and Roger Y Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-d machine vision metrology. *IEEE Transactions on pattern analysis and machine intelligence*, 10(5):713–720, 1988.