**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.

Diagram

Description automatically generated with medium confidenceThe 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)

Task definition:

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:

**Cam L intrinsic parameters**

**Cam L**

**Ref .bar**

**Camera extrinsic parameters**

**Calibration model**

**Image point identification**

**N synchronous images**

**Cam R**

**Cam R intrinsic parameters**

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

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

**3.4. Challenges Faced:**

**3.4.1. 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.

A close-up of a camera

Description automatically generated with low confidenceA picture containing floor, indoor

Description automatically generated

The source for the camera input is provided by the pulse 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

A picture containing text, oven, appliance, kitchen appliance

Description automatically generatedA picture containing text, indoor

Description automatically generated

**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.

**3.4.2. 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:

* **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)

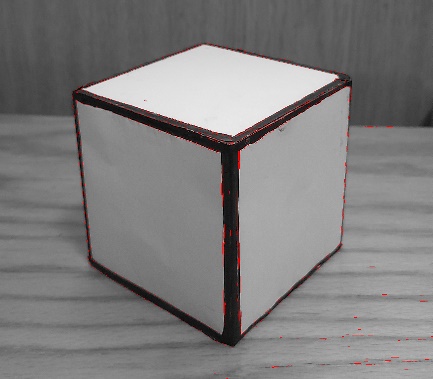
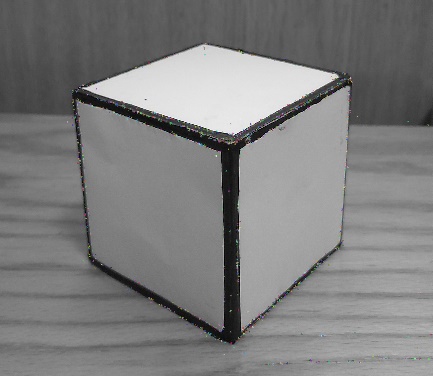
* **Hough transform** is used to find and draw the lines on the edges of the cube

lines = cv2.HoughLinesP(canny, 1, np.pi/180, 100, minLineLength=100, maxLineGap=10)

* **LSD algorithm** is used to detect the edges using the Line segment detection algorithm.

lsd = cv2.createLineSegmentDetector(0)

lines, \_, \_, \_ = lsd.detect(blur)



A rubik's cube on a white surface

Description automatically generatedA picture containing cake, indoor

Description automatically generated

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 endings of the reference bar in this project.