Online Automated Multi-Camera Calibration in Real Time

Team G:
Huan-Yang Chang (Peter)
Man-Ning Chen (Mandy)
Sambuddha Sarkar (Sam)
Siddharth Raina (Sid)
Yiqing Cai (Cece)

Date: 5/12/2017

oculus

facebook

THE ROBOTICS INSTITUTE

Carnegie Mellon University

Abstract

This report is a comprehensive summary of the work completed by Team G (Excalibr) on their project. The report is begins with a description of the problem being tackled, then it moves on to the system level requirements specific to the problem being solved. This is followed by the system architecture where the nuances of the system are highlighted. The successive pages discuss about the current status of the work and the project management techniques being used for this project.

This project is being supervised and sponsored by the ORP (Oculus Research Pittsburgh) and the requirements generated adhere to the demands of our sponsor.

Table of Contents

1. Project description

Oculus Research, Pittsburgh has constructed a multi-sensor capture system consisting of a large number of cameras and microphones to perform motion tracking and 3D reconstruction with unprecedented accuracy. A critical component of achieving highly accurate 3D Reconstruction and motion tracking results is accurate sensor calibration, which is the focus of our project. The calibration process employs three methods, namely:

- Sensor Noise Calibration
- Color Calibration
- Geometric Calibration

Our team, in collaboration with Oculus VR envisioned a system shown in Figure 1. We use a ABB robotic arm to maneuver a specially designed 3D calibration target in the capture space in order to capture images covering most of the field of view of all the cameras. Our path planning algorithm ensures that we are able to achieve a coverage of at least 85% in the field of view of each camera while selecting the minimum number of points the ABB robot arm would need to cover. After collecting the images of the calibration target at these selected points, we perform the sensor noise calibration and the color calibration. In sensor noise calibration, we remove the fixed pattern noise from the images using dark frame subtraction and gain normalization. In color calibration, we use a standard ColorChecker chart to adjust the RGB gain channels to obtain the true color of the images. These steps are crucial for accurate 3D reconstruction. The final step is applying the geometric calibration algorithm on the images post sensor noise and color calibration. The average calibration accuracy (reprojection error) achieved by our team was less than 0.2 pixels.

Figure 1 High-level system visualization

2. Use Case

Date: May 7
Year: 2017
Location: Mesla HQ
Subject: Mesla crash while operating on Autopilot.

Bob is at the helm of the camera calibration team at Mesla, he has been leading this department since it's inception. Mesla's managerial heads and legal department are breathing down his neck. It isn't getting any easier for Bob at all. It is a long day ahead for him.

The death of a 40-year old Joshua Brown has shaken up the whole autonomous of Mesla. Everyone is under fire, especially the teams working with sensor fusion as the algorithm and systems division has been cleared of any wrongdoing. With the recent rise in accidents relating to Autopilot, the pressure on the calibration team for the designing and operating fail-safe sensor-fusion arrays is more than ever. Calibration is a crucial component for any application which involves an engineering system operating on sensor-fusion. What can get Bob out of this snafu?

Well first and foremost, he has to make sure that all the vision based systems in the vehicles, specifically the cameras are always calibrated. That's a bummer, he can't have the cars being dragged into the service center after every Autopilot operated instance. That would just be impractical and plain suicidal for an automotive company. Disappointed with himself, Bob goes to the bar to get a pint and calm his nerves.

At the bar counter, Bob runs into an old friend of his - Shake. Bob cannot contain himself and spills his guts out to Shake. Shake hears Bob's blabbering for an hour and so and realizes that he has the perfect solution for Bob! He can't stand seeing his friend under so much stress and decides to take him to his lab down at Cranberry-Lemon University.

Location: Cranberry-Lemon University & Octopus VR technologies collaboration lab.

For 1 year Shake and his team have been working on this revolutionary idea of automating camera calibration and very excited to reveal his system to his dear friend Bob. Shake goes on to show his system:

Excalibr is a multi-camera calibration system, all contained in one box! The beauty of this system is that this calibration box can be stored in the trunk of a car and can calibrate as many cameras as they want. But is this reality?! Yes, it is. The Excalibr is a self-contained calibration setup. There a proprietary mechanisms and software algorithms by which a multitude of self-changing calibration targets can be used to calibrate camera systems remotely. The Excalibr calibration box is a hermetically sealed setup where a mobile robotic mechanism moves the calibration targets around and optical fibers transmit the light from the box in the trunk to the different cameras on the autonomous cars internally. This ensures that the not only can the cameras be checked for calibration status when the vehicle is at rest or human driving mode but also calibrated in real-time if necessary. Shake's idea will enable automatic real-time calibration of cameras throughout the vehicle, irrespective of the location of the camera. The calibration box aforementioned has been visualized below in Figure 2.1, where the optical fibers transmitting the

light from the box to the camera lenses are color coded as orange. The box communicates with an onboard computer to run the calibration algorithms on the images.

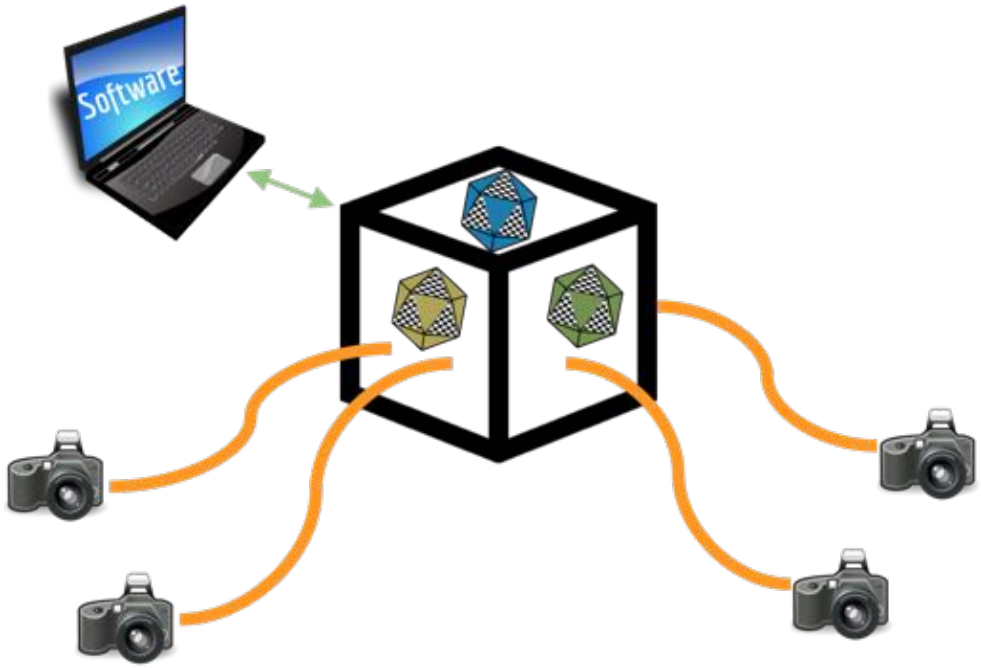


Figure 2.1 Excalibr Calibration Box Setup with movable reconfigurable calibration target.

Bob is flabbergasted after witnessing what Shake has been working on, it is the perfect solution to his problem! Not only he can ensure that his vision systems on Mesla cars work but also fix the calibration on them if necessary, all while the vehicle is operational! Shake's Excalibr technology will not only prevent countless accidents in the near future but also save many lives in the days to come! A victory for Shake and his system- Excalibr.

3. System-Level Requirements

3.1 Functional requirement

3.1.1 Functional requirements in Fall

M.F.1: Operate Autonomously
M.F.2: Fabricate calibration target
M.F.3: Control and Manipulate the calibration target by robot arm
M.F.4: Take high-resolution, stable and clear pictures of calibration target
M.F.5: Implement geometry camera calibration algorithms on RGB cameras
M.F.6: Implement photometric calibration and generate camera response function curve for GRB cameras
M.F.7: Implement sensor noise correction on RGB cameras
M.F.8: Build the calibration pipeline for multiple cameras

3.1.2 Functional requirements in Spring

M.F.1: Operate Autonomously
M.F.2: Fabricate calibration target

M.F.3: Control and Manipulate the calibration target by robot arm
M.F.4: Take high-resolution, stable and clear pictures of calibration target
M.F.5: Implement color calibration on RGB cameras
M.F.6: Validation of geometric calibration algorithm in virtual environment
M.F.7: Build the calibration pipeline for multi cameras

3.2 Performance requirements

3.2.1 Performance requirements in Fall

M.P.1: Manipulate the robot with 100 micrometers accuracy
M.P.2: Take pictures with multiple RGB cameras more than 10MP at 30fps
M.P.3: Complete one geometry calibration in at most 8 hours
M.P.4: The sensor noise correction algorithm must reduce the variance of the flat-field image for 90% or more.
M.P.6: The reprojection error of the geometry calibration result should be less than 1 pixel.

3.2.2 Performance requirements in Spring

M.P.1: Two-click Operation
M.P.2: Fabricate the target with 50 micrometers tolerance
M.P.3: Complete one geometry calibration in at most 8 hours
M.P.4: Independent color calibration : 60% color patch detection accuracy - Be able to find all color patch values of at least 60% of input images.
M.P.5: Avoid collisions - keep a distance of 0.3m away from the dome extremities and sensors
M.P.7: Path calculating algorithms complete within 8 hours
M.P.8: Build calibration pipeline for 3 RGB cameras
M.P.9: The reprojection error of the virtual geometry calibration result should be less than 0.1 pixels.
M.P.10: The reprojection error of the geometry calibration result should be less than 0.2 pixels.

3.2.3 Desired Performance requirements

D.P.1: Build calibration pipeline for 100 RGB cameras

3.3 Non-Functional requirements

3.3.1 Mandatory non-functional requirements

M.N.1: Complete the project by May 2017.
M.N.2: Keep budget within $5,000
M.N.3: Make the system user-friendly.

3.3.2 Desired non-functional requirements

D.N.1: Create a user-friendly GUI or physical button

4. Functional Architecture

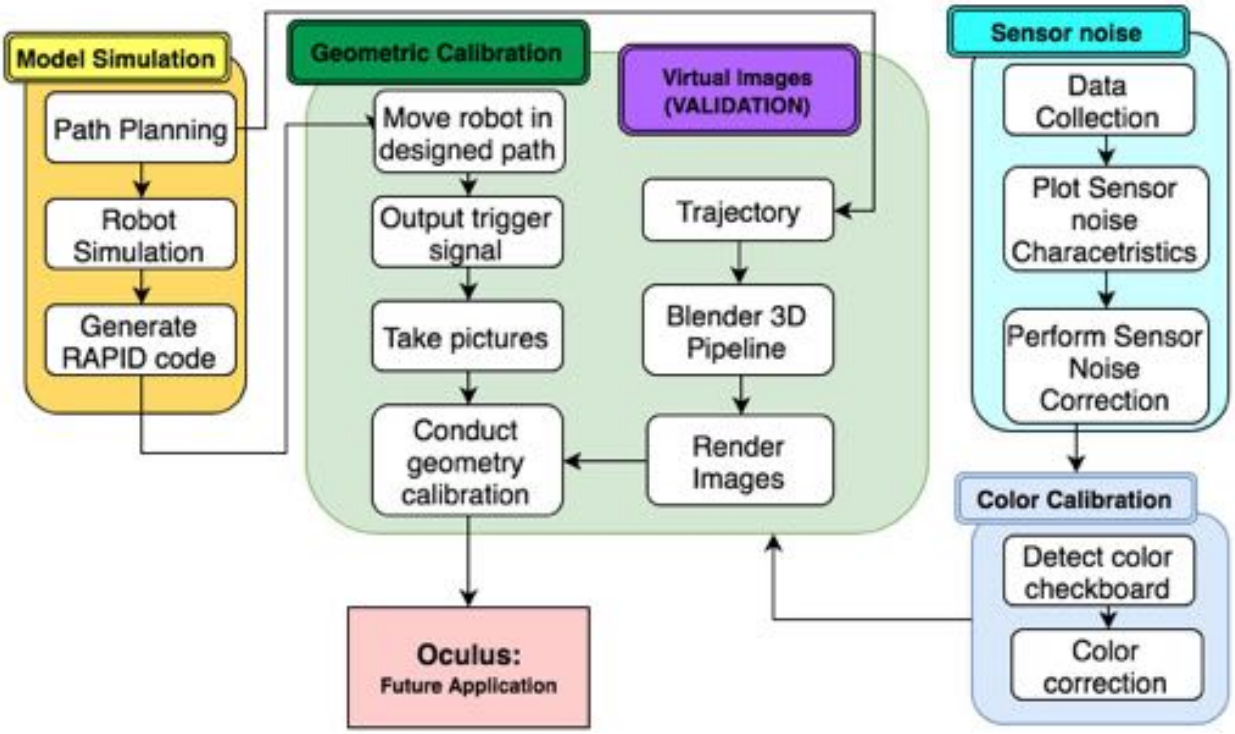


Figure 4.1 Functional Architecture - Complete system

The functional architecture (Figure 4.1) can be divided into the following sub-systems broadly: Robot simulation, Geometry calibration, Virtual Image Generation, Sensor noise calibration and Color calibration.

Generally speaking, when we push the start button, the whole system will begin by checking the safety situation and present accuracy, if the system needs calibration, we will carry out the four steps of calibration procedure successively.

4.1 Model simulation

In the model simulation, we are working on ABB robot arm model simulation and path optimization.

4.1.1 Robot model simulation

In the Robot model simulation, we would simulate the working environment of the ABB robot arm and the robot movement. The most important concern here is using simulation system to test the different type of robot movement and make sure all the positions along the robot trajectory are safe and valid. In the simulation, we would check the joint limit of the ABB robot arm and whether there would be collisions. It could save us tons of time to work on the real robot.

4.1.2 Path Planning

In the Path Planning, we tried to simulate the field of view (FOV) of a hundred cameras in the environment based on the requirements of the input to the geometric calibration algorithm, the depth of field, the camera focus plane and the working range of the ABB robotic arm to optimize the robot motion plan. Our objective was to generate the path with a high coverage of the field of view for each camera and to reduce the duplicate positions to make the path as efficient as possible.

4.2 Geometric calibration subsystem

The task of geometric calibration can be divided into 3 parts: the manipulation and control part, the image capturing part and the data processing and calibration part.

4.2.1 Virtual Image Generation

This is a virtual image data set generation system to aide in validation of the geometric calibration algorithm for a 3D calibration target. The pipeline loads in camera data (intrinsic & extrinsic), object data (calibration target), required configuration and the number of the cameras and the render settings. It spews out Images and other mesh data using the cycles render engine and bpy module.

4.2.2 Manipulation and control

From the access terminal, we give instructions to control the ABB Robot Arm which has the 3D calibration target and sensors mounted on it. The ABB robot arm will then move the calibration target in a pre-designed trajectory according to the path design algorithm, and the sensors on it will collect position data. The system will send back the encoded positional data back to the computer and used for the following calibration procedure.

4.2.3 Image capturing

Also from the access terminal, we give instructions to the triggering mechanism. Under that mechanism, the 12 MP @ 73 fps canon cameras will be triggered to capture images. For the discrete trigger, the ABB robotic arm will move in a discrete pattern with pause to avoid blurring, and the cameras will be triggered to capture images during the pause. Then this will be comparatively slower, but we will be able to get accurate images for the calibration process. When finishing capturing, all the images will be sent back to the computer and used for the following calibration procedure.

4.2.4 Data processing and calibration

During this part, we will use the captured image data and the position data of the calibration target to do the geometry calibration. We will run the geometry calibration algorithm to do the calibration work for multi cameras, and then the parameters of cameras and the positional and rotational data will be stored in a certain unit.

4.3 Sensor noise Calibration

In the sensor noise calibration, we would focus on four type of noise: shot noise, read noise, pattern noise, and thermal noise. In order to minimize these four types of noise influence in the camera for better camera calibration, we would design a noise collection and remove process.

With that process, we could systematically record the noise parameter for each camera and conducted the calibration to remove them.

4.4 Color calibration

Color calibration is to measure and adjust the color response of a device (input or output) to a known state. We used an X-Rite ColorChecker Classic Card as our ground truths. We would try to find the fittest model that could best convert the image color to ground truth color.

5. System-Level trade studies

5.1 Trade study on size of calibration target

For geometric calibration, the size and position of the calibration target is an important factor. This would directly affect the intrinsic and extrinsic parameters of the camera during calibration. The following table summarizes the design choices, first design (D1) uses a small checkerboard (length of each face < 3cm), Design 2 (D2) using medium check board (3cm<length in each face <20cm), Design 3 (D3) using medium check board (20cm<length in each face)

Table 5.1, Scoring of three target sizes

	Weight	D1: Small < 3cm	D2: Medium 3cm<X<20cm	D3: Large >20cm
Manufacture	5	5	3	1
Surface Printing	10	10	10	5
Safety	10	7	7	7
Automation	20	15	15	15
Not Complexity	10	10	10	10
Durability	15	10	10	10
Time saving	15	5	13	15
Accuracy	15	10	15	10
Total		72	83	73

The selection criteria of the system for trade study is manufacturing ease, surface printing, safety, automation, simplicity, durability, less time for fabrication and accuracy. D1 relatively scores high in manufacturing and surface printing, because the target is small enough we can use the commercial equipment to make it. D1 scores relatively low at time-saving because we have more steps that are needed to cover the whole target space. D2 is relatively high at accuracy because the sponsor would want to capture the face in detail. Hence, using a calibration target of similar size would lead to a better result. D3 manufacturing saves a lot of time, but in accuracy and manufacturing the D3 is relatively low because the target is harder to manufacture and accuracy is relatively hard to achieve. Hence we choose D2.

5.2 Trade study in configuration the calibration target

In our calibration system, we have to calibrate the different characteristics of various cameras, environment and microphones. Design 4 (D4) combines all subsystem level calibration targets together into a single calibration target and Design 5(D5) separates the subsystem-level calibration targets to different parts.

Table 5.2, Scoring of both target configurations

	Weight	D4: Combined calibration Target	D5: Separated Calibration Target
Manufacturing Ease	5	1	3
Safety	10	7	7
Automation	20	20	15
Simplicity	10	1	5
Durability	15	5	13
Time related efficiency	15	15	10
Accuracy	15	10	13
Total	90	59	66

Design 4 is relative high in automation and time saving, because the robot can keep performing different calibrations at the same time which can reduce the time it takes to replace the calibration target every time for each calibration procedure. On the other side, Design 4 is harder to manufacture and maintain the positional accuracy at the same time. Design 5 has higher durability and accuracy, because we can easily to check the accuracy of the subsystem-level calibration targets and make each more durable. Hence we chose D5 shown in Figure 5.2.1.

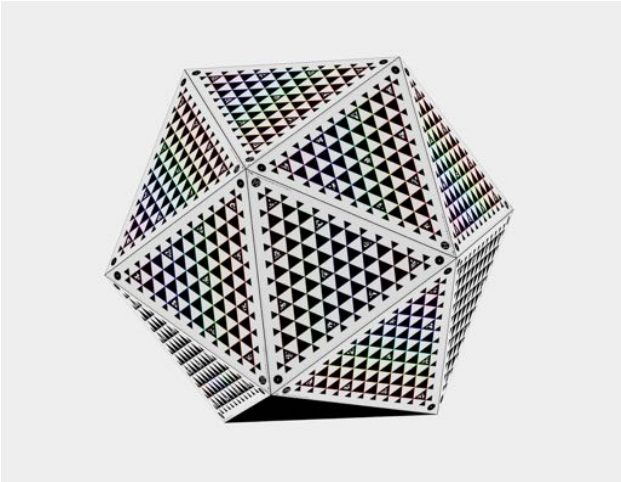


Figure 5.2.1, Final design chosen

5.3 CMOS Sensor Characterization in Cameras: Absolute Lumen Calibration

Problem: Obtain the absolute value of light intensity and calibrate the CMOS sensor output of the camera to match the said absolute value of light intensity.

There are two ways of approaching this problem:

1 **Method I**: Get the value of the intensity of light of the surface using Photometers/Lux meters/Radiometers[2].

2 **Method II**: Use a standardized light source with controllable[1] wavelength and intensity.

A comparative overview of the two stated methods have been given below in Table 5.3, ***each advantage is given an unweighted score of 1***:

Table 5.3, Scoring of both methods of calibration

	Method I	**Method II**
Principle of Operation	Uses a transducer to convert light intensity to digital signal.	Uses a transducer to convert digital signals into light waves.
Sensor/Transducer	Silicon(doped) Photodiode	Silicon(doped) LED / Tungsten Filament
Pay-off Table		
Cost	$ - Cheap	$$$ - Expensive
Luminous efficiency error	9% - High	0.001% - Low
Dependence on ambient light	In-effective/false positives under fluorescent lighting	Independent of ambient lighting
Response time	5 s	0.500 s
Characteristics of oblique incidence/ Luminance Spatial uniformity	**Incidence** 10° ±1.5% 30° ±3% 60° ±10% 80° ±30%	**Spatial Uniformity** >94% over 360º x 200º field of view
Spectral range	Lux meter: 1 Photometer: 850 nm to 940 nm	Visible, 850 nm to 940 nm
Spectral mismatch	1%	>0.00001%
Luminescence range	0.0 to 999 cd/m^2	10 to 700 cd/m^2
Typical application	Lux meter: Ambient light Photometer/Radiometer: Color of surface.	Calibration of Lux meters, Photometers, Radiometers, Cameras & other optical equipment.
Operational features	-Comparatively less stable output -Needs regular calibration -Integration with desktop on select models.	-Precise control -Integration with desktop: easy -Long life -Stable output
Total score	**2/12**	**7/12**

Result: Method II is the most desirable way to go about solving the problem at hand.

Recommendations:

1. **Gamma Scientific[4]:** Show in Figure 5.3.1.

Figure 5.3.1

2. **Labsphere[5]:** Shown in Figure 5.3.2.

Figure 5.3.2

LED light source is preferred over tungsten filament based source as it has the below stated superiority:

1. Life
2. Heat/IR – minimum
3. Multitude of Wavelengths generated
4. Precise selection of Wavelength & Intensity

6. Cyber-physical Architecture

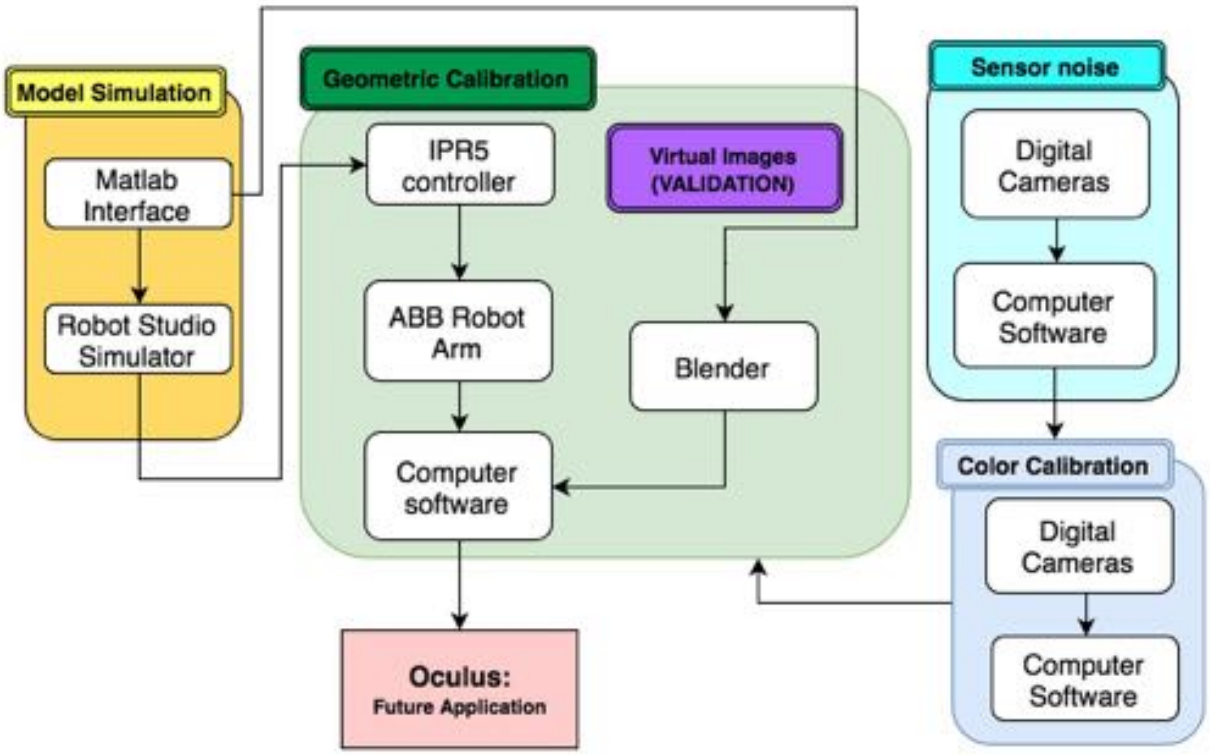


Figure 6.1 Cyber-physical Architecture - Complete system

The subsystem level implementation of the cyber-physical architecture (Figure 6.1) has been elaborated below.

6.1 ABB Robot Arm

The ABB robot arm is one of the major components to be used in this system. The robot arm would be controlled by the IPR5 controller and be equipped with position sensors which would send the position of the end effector to the access terminal/computer. The movement of the ABB robot arm can be simulated in RobotStudio to check collisions and joint limits to make sure the efficient and safe movements.

6.2 3D Calibration Target

The 3D calibration target would be used for the geometric calibration. Each surface of the target will be identified by the specific april tag. The target will be mounted on the ABB robot arm and the motion of this target would be controlled by the robot arm, which would send the position of the target back to the computer to trigger the cameras to capture images.

6.3 Digital Cameras

Digital cameras are the test equipments we are using. They would be used to capture original images and passed through all the different calibration process including geometry calibration, sensor noise correction and color calibration.

6.4 IPR5 Controller
The IPR5 Controller is used to control the ABB robot arm. After we complete the simulation in RobotStudio and generate the valid and optimized path, we will pass the path to the controller and make the ABB robot arm go through the pre-designed trajectory following a specified type of movement.

6.5 RobotStudio Simulation
The RobotStudio simulation system is used to test the different type of robot movement and make sure all the positions along the robot trajectory are safe and valid. Matlab Interface can be used to generate the RAPID code which can be used to simulate the system. In the simulation, we would check the joint limit of the ABB robot arm and whether there would be collisions. It could save us tons of time to work on the real robot and it is perfectly interfaced with the real ABB robot arm.

6.6 Synthetic Image Generation
The virtual environment is being modeled in an open source platform: Blender 3D 7.68a. It is a Maya based platform and is programmable by Python 3 and CPP-17. Custom scripts using the aforementioned languages can be utilized to manipulate the data blocks inside Blender 3D and run different calibration tests in different virtual scenes.

7. System Description and Evaluation

7.1 Sensor Noise Measurement
This subsystem deals with the measurement and if possible correction of the different noise types which can corrupt the image quality.

7.1.1 Subsystem Description
There are four major types of noise which affect the quality of the image captured from the camera sensor:

SHOT NOISE: When a camera is exposed to light, a stream of photons falls onto the sensor element of the camera. The flux of this incoming photon stream is not uniform. The fluctuations of this flux of photons around the average value of the photon flux constitutes the Shot Noise.

READ NOISE: After a sensor element is exposed to a stream of photons, it accumulates photoelectrons. These photoelectrons get converted to voltage signals. The digitization of signal introduces noise in the image which is proportional to the photon count and constitutes the read noise of the sensor.

DARK/THERMAL NOISE: Due to thermal excitation, electrons are emitted which are indistinguishable from the photoelectrons. This induces an additional signal to the sensor element which is reflected in the output voltage. This noise can be eliminated by subtracting a

dark frame from the image captured. Thermal noise increases with increase in the exposure time and vice versa.

PIXEL RESPONSE NON UNIFORMITY: Pixels in a camera sensor have a different ability to capture and count photons which incident on their respective sensor elements. Due to this difference, each pixel is associated with a gain parameter which is a measure of its efficiency in capturing the amount of light falling on the respective sensor element. PRNU increases with the exposure time.

The effect of the different noise types can be seen from the figure below:

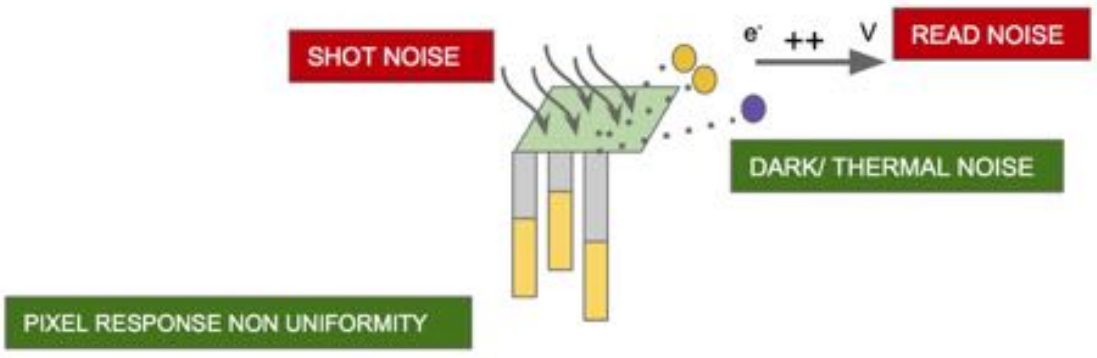


Figure 7.1.1 Sensor Noise Visualization

7.1.2 Subsystem Analysis

The EVT camera for which the different types of noise are being measured allows us to continually vary the exposure of the camera while capturing multiple images for noise measurement and noise removal. This subsection describes the experiments which were carried out to obtain the noise measure for different cameras and noise types.

MEASUREMENT OF READ AND SHOT NOISE: Two consecutive images of a uniform light source are taken at the same exposure and are subtracted to cancel out the effects of thermal noise and PRNU. The standard deviation of the noise (standard deviation of all the pixel values) from the resulting image is then calculated and divided by $\sqrt{2}$, as independent noise sources adds up in quadrature. The governing equations are explained below:

$$\sigma_{(N1)}^2 + \sigma_{(N2)}^2 = \sigma_{(Ntotal)}^2$$
$$\sigma_{(N1)} \approx \sigma_{(N2)}$$
$$\sqrt{\sigma_{(N1)}} = \sqrt{(\sigma_{(Ntotal)}/2)}$$

This would give us a datapoint $\{E_i,\sigma_{(Ni)}\}$ and the steps can be repeated for a set of exposure values. An illustrative visualization for the measurement of the shot noise and read noise can be seen from the figure below:

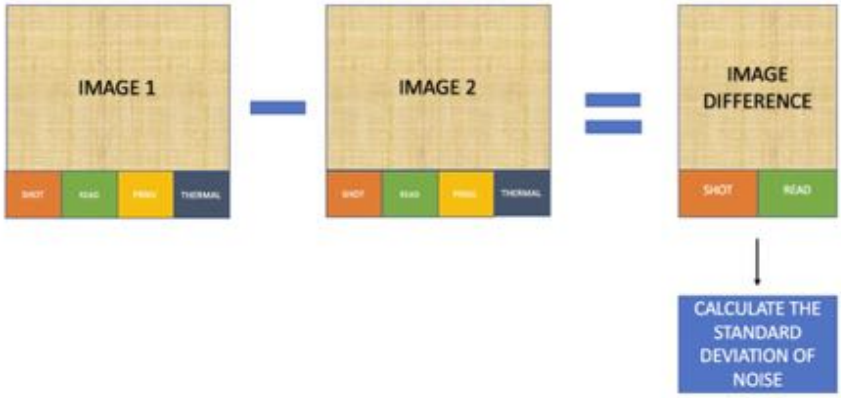


Figure 7.1.2a Read and Shot Noise Measurement

MEASUREMENT OF DARK/ THERMAL NOISE: The thermal noise can be measured by a dark frame subtraction from the image frame at a given exposure value. This can be done by collecting dark frame images (images with the camera lens cap on) for the exposure values we require to perform dark frame subtraction. An illustrative visualization for the measurement of thermal noise can be seen from the figure below:

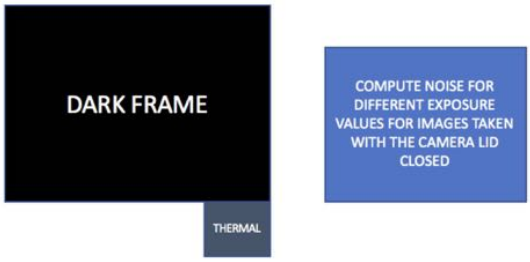


Figure 7.1.2b Thermal Noise Measurement

MEASUREMENT OF PIXEL RESPONSE NON UNIFORMITY: Flat images for the set of exposure values same as those for the read and shot noise measurement were captured. For each exposure value E_i, the following steps were followed:

1. The dark noise (calculated in the previous section) was subtracted from each image.
2. The read and shot noise was subtracted from the image:

$$\sigma_{(Ntotal)}^2 = \sigma_{(Nread)}^2 + \sigma_{(Nshot)}^2 + \sigma_{(NPRNU)}^2$$
$$\sigma_{(NPRNU)} = \sigma_{(Ntotal)}^2 - (\sigma_{(Nread)}^2 + \sigma_{(Nshot)}^2)$$

3. This would give us a datapoint $\{E_i,\sigma_{(NPRNUi)}\}$. This step was repeated for multiple exposures to analyse the noise plot. An illustrative visualization for the measurement of thermal noise can be seen from the figure below:

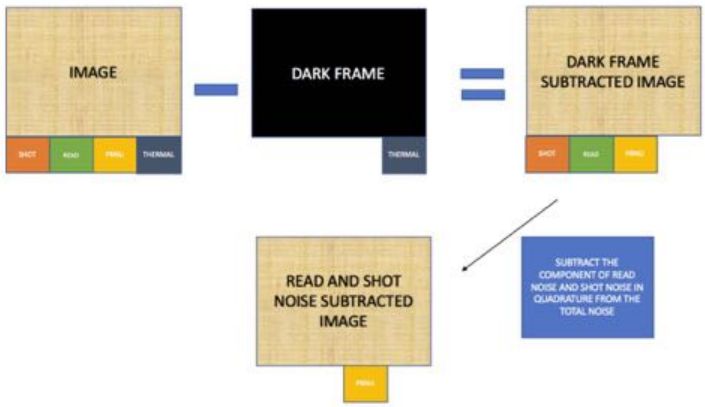


Figure 7.1.2c Pixel Response Non Uniformity Measurement

7.1.3 Subsystem Evaluation

Based on the experiments described in the subsection 7.1.2, The following plots were obtained which describe the noise characteristics of each different noise type:

READ AND SHOT NOISE:

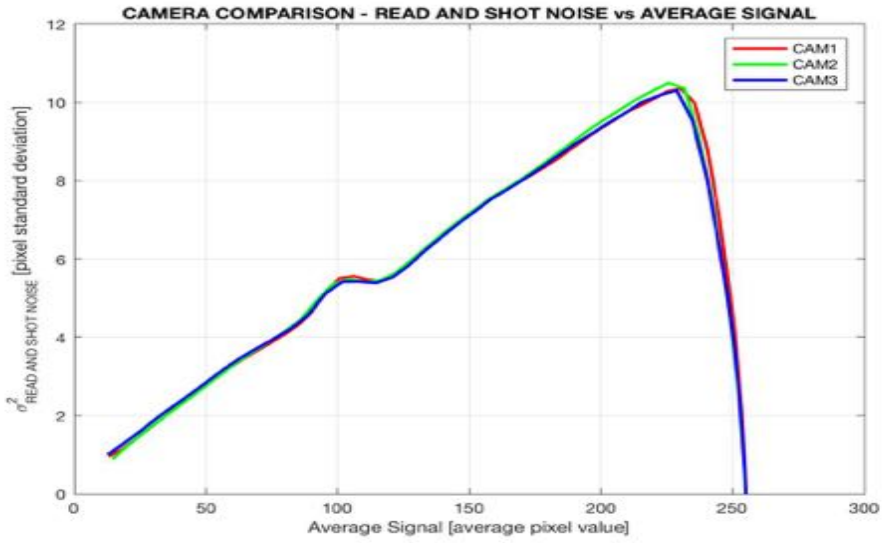


Figure 7.1.3a Read and Shot Noise Plot

DARK/THERMAL NOISE:

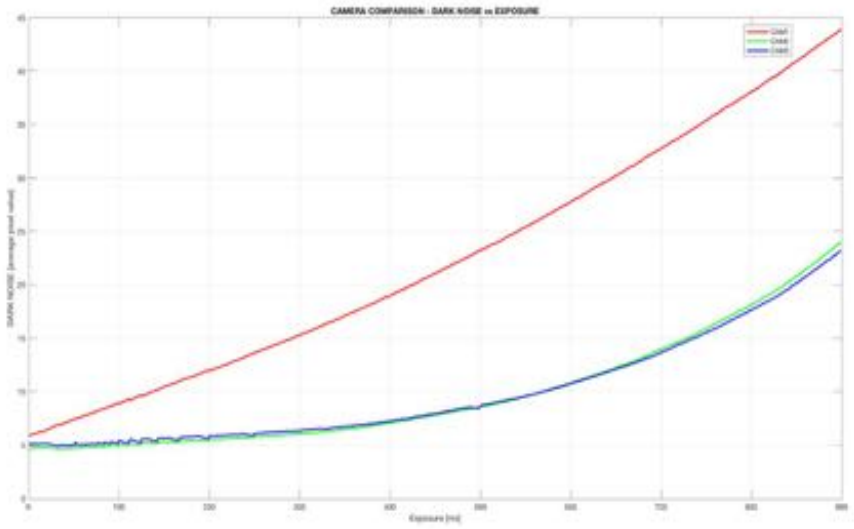


Figure 7.1.3b Dark Noise Plot

PIXEL RESPONSE NON UNIFORMITY:

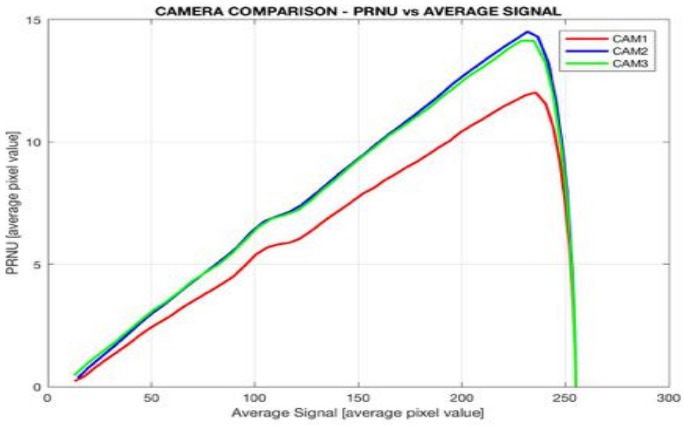


Figure 7.1.3c Pixel Response Non Uniformity Plot

The obtained plots are as expected by the experiments suggested in section 7.1.2. For the plots of the read and shot noise: The square of the noise is roughly a linear function of the average signal value. This follows from the read and shot noise equations in section 7.1.2. Similarly, the Pixel Response Non Uniformity is a linear function of the average signal. This too is in-line with the proposed model and dark noise is a curve monotonously increasing with the exposure value as

expected. The PRNU and dark noise correction was done by dark frame subtraction and gain normalization. The results obtained can be seen from the figures below:

Figure 7.1.3d Sensor Noise Correction on a Test Image

7.1.4 Strengths and Weaknesses

STRENGTHS:

1. The sensor noise calibration subsystem is an essential subsystem which can remove unwanted noise from the image which is a crucial step for any useful application, such as 3D reconstruction, where we don not want the unwanted noisy components to reflect on the image reconstruction.

2. Additionally, from the experiments performed in the section 7.1.2 and the results obtained in section 7.1.3, we can estimate the best range of operation for a given camera, which can help us to decide the camera needed for a given application under given environmental conditions (such as lighting and camera exposure).

WEAKNESSES:

1. As reflected from the reprojection errors in the final experiments for geometric calibration with and without sensor noise removal, sensor noise removal is not a required step for the camera calibration and can be omitted from the pipeline if calibration is the only end goal of the system.

2. Sensor noise calibration requires each camera to be separately calibrated outside the environment of application. This would involve disassembling the cameras and calibrating them. From the standpoint of a real time calibration system, this poses a serious disadvantage and with the current architecture, it is not possible for the entire system to operate in real time as an end to end system.

7.2 Color Calibration

7.2.1 System/subsystem descriptions/depictions

Because of various lighting conditions and the difference between cameras, the object color deviates from its true color in photos. In spite of outside conditions, we would like to recover the object colors in reconstruction. We use the X-rite ColorChecker manufactured with known color ground truth. Our system is capable of automatic ColorChecker detection and segmentation. It then maps the color values extracted from images to ground truth values.

7.2.2 Automatic detection and segmentation algorithm

CCFind is implemented to obtain color values recorded in the images. The process is shown as below

Fig 7.2.2. The CCFind pipeline

After we get the center of each color patch of the colorchecker, we can get the recorded values of each color patch.

7.2.3 Color mapping model

In order to find the best mapping model, a variety of models along with image preprocess were applied and tested. The best mapping model among them is a polynomial model combined with white balanced preprocessing.

Polynomial Model:

$$
\begin{aligned}
CalibratedR &= \alpha_0 + \alpha_1 R + \alpha_2 G + \alpha_3 B + \alpha_4 R^2 + \alpha_5 G^2 + \alpha_6 B^2 \\
CalibratedG &= \beta_0 + \beta_1 R + \beta_2 G + \beta_3 B + \beta_4 R^2 + \beta_5 G^2 + \beta_6 B^2 \\
CalibratedB &= \gamma_0 + \gamma_1 R + \gamma_2 G + \gamma_3 B + \gamma_4 R^2 + \gamma_5 G^2 + \gamma_6 B^2
\end{aligned}
$$

7.2.4 Testing and Analysis

The approach we used to decide the best model is comparing the mapped image color patches with the ground truth chart shown in Fig 7.2.4a

No.	Number	sRGB R	G	B	CIE L*a*b* L*	a*	b*	Munsell Notation Hue Value / Chroma
1.	dark skin	115	82	68	37.986	13.555	14.059	3 YR 3.7 / 3.2
2.	light skin	194	150	130	65.711	18.13	17.81	2.2 YR 6.47 / 4.1
3.	blue sky	98	122	157	49.927	-4.88	-21.925	4.3 PB 4.95 / 5.5
4.	foliage	87	108	67	43.139	-13.095	21.905	6.7 GY 4.2 / 4.1
5.	blue flower	133	128	177	55.112	8.844	-25.399	9.7 PB 5.47 / 6.7
6.	bluish green	103	189	170	70.719	-33.397	-0.199	2.5 BG 7 / 6
7.	orange	214	126	44	62.661	36.067	57.096	5 YR 6 / 11
8.	purplish blue	80	91	166	40.02	10.41	-45.964	7.5 PB 4 / 10.7
9.	moderate red	193	90	99	51.124	48.239	16.248	2.5 R 5 / 10
10.	purple	94	60	108	30.325	22.976	-21.587	5 P 3 / 7
11.	yellow green	157	188	64	72.532	-23.709	57.255	5 GY 7.1 / 9.1
12.	orange yellow	224	163	46	71.941	19.363	67.857	10 YR 7 / 10.5
13.	blue	56	61	150	28.778	14.179	-50.297	7.5 PB 2.9 / 12.7
14.	green	70	148	73	55.261	-38.342	31.37	0.25 G 5.4 / 8.65
15.	red	175	54	60	42.101	53.378	28.19	5 R 4 / 12
16.	yellow	231	199	31	81.733	4.039	79.819	5 Y 8 / 11.1
17.	magenta	187	86	149	51.935	49.986	-14.574	2.5 RP 5 / 12
18.	cyan	8	133	161	51.038	-28.631	-28.638	5 B 5 / 8
19.	white (.05*)	243	243	242	96.539	-0.425	1.186	N 9.5 /
20.	neutral 8 (.23*)	200	200	200	81.257	-0.638	-0.335	N 8 /
21.	neutral 6.5 (.44*)	160	160	160	66.766	-0.734	-0.504	N 6.5 /
22.	neutral 5 (.70*)	122	122	121	50.867	-0.153	-0.27	N 5 /
23.	neutral 3.5 (.1.05*)	85	85	85	35.656	-0.421	-1.231	N 3.5 /
24.	black (1.50*)	52	52	52	20.461	-0.079	-0.973	N 2 /

Cie L*a*b* values use Illuminant D50 2 degree observer sRGB values for Illuminate D65.

Fig 7.2.4a Color Ground Truth

By comparing the errors in each color channel of each color patch, we can find the best model.

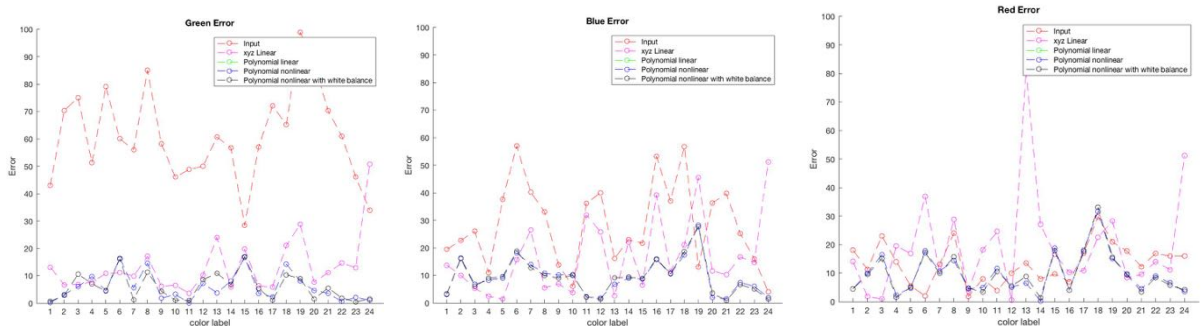


Fig 7.2.4b Error Analysis

Fig 7.2.4c Result Analysis

7.2.5 SVE performance evaluation

The SVE performance requirement of color calibration is achieving at least 60% of ColorChecker detection and segmentation rate. This requirement is successfully completed by 86% of the result detection and segmentation rate.

7.2.6 Strong/weak points

The strong points of the subsystem are that it is automatic and since CCFind is using the geometry features to detect the target. The algorithm is more robust, especially against special lighting conditions. In addition, the polynomial model performs better than the most linear model in previous studies. However, if the lighting condition is too bad making some pixels saturated in the image (over 255), the lost information can not be recovered by our method.

7.3 Robot simulation and path planning

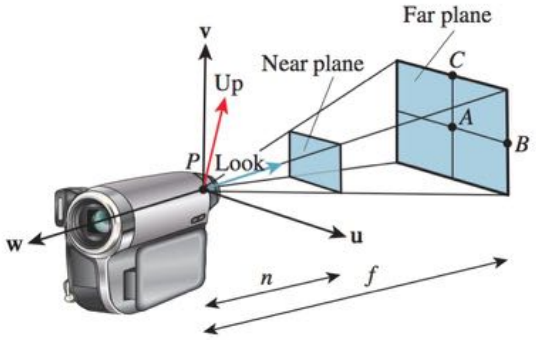
7.3.1 System description

For the Path Planning, we tried to simulate the field of view (FOV) of a hundred cameras in the environment based on the requirements of the input to the geometric calibration algorithm, the depth of field, the camera focus plane and the working range of the ABB robotic arm to optimize the robot motion plan. Our objective was to generate the path with a high coverage of

the field of view for each camera and to reduce the duplicate positions to make the path as efficient as possible.
Then, after we get the design trajectory, we would import it to RobotStudio, the ABB robot simulator, to confirm that the path was collision-free and reachable for the robot arm. We could also evaluated the speed of the robot for the total time cost and the smoothness of the robot arm.

7.3.2 Camera Model Building

The camera parameters consist of the intrinsic matrix (focal length and principal point), and extrinsic matrix (rotational matrix and translation vector). First, we need to specify the camera parameters given the camera positions and the looking direction.



$$\mathbf{w} = \frac{-\mathbf{look}}{\|\mathbf{look}\|} = \mathcal{S}(\mathbf{look}).$$

$$\bar{\mathbf{v}} = \mathbf{vup} - (\mathbf{vup} \cdot \mathbf{w})\mathbf{w}$$

$$\mathbf{v} = \frac{\bar{\mathbf{v}}}{\|\bar{\mathbf{v}}\|} = \mathcal{S}(\bar{\mathbf{v}}).$$

$$\mathbf{u} = \mathbf{v} \times \mathbf{w}.$$

Figure 7.3.2 Camera models

For the intrinsic matrix, all the parameters are in the unit of pixels, so we need to convert the unit of focal length and principal point into pixels. The principal point is the center of the image plane and we regard the left top corner as the origin of the image plane.For the extrinsic matrix, we need to compute the rotation and translation between the world frame and the camera frame.

7.3.3 Path Generation and Optimization

The ABB robot arm has working constraints and safety zone, and the calibration target must be within the depth of fields of the cameras so the original path is generated based on it.

Initially, we sampled 2000 points uniformly along the original path, which is able to cover all the possible positions for the calibration target which is mounted on the ABB Robot Arm. In order to record projections at each 3D position for all the cameras, we created a cell array which has 140 structs and each of them contains a matrix which is the same size of the image plane. The matrix is initially all zeros. Once got a projection, all the pixels in the covered area plus one. When all the projections are generated, for each matrix, the larger the pixel value, the more overlaps the area has.

For evaluating the quality of projections, we develop a score function. When the radius of projection is within 500 ~ 1500 pixels, the projection is valid, and when the radius equals to 800 pixels, the score equals to 1. From 500 ~ 800 pixels, the score is uniformly mapped into 0 ~ 1, the bigger the projection, the better it is. From 800 ~ 1500 pixels, the score is uniformly mapped

into 0 ~ 1, the smaller the projection, the better it is. Besides, we also take into consideration the spatial information of the projection. If the center of the projection is out of the range of the camera FOV, we also set the score to be zero.

Then we calculate the score for all the possible 3D positions for all the camera FOVs. For example, if we have 2000 possible positions for calibration target and 140 cameras, the score matrix would be the size of 2000*140, each element in the matrix is the evaluation for a single projection at a specific position onto a specific camera FOV. Then we sum the score matrix along the second axis to obtain a vector of size 2000*1, each element represents the total score for a specific position on all camera FOVs. With this score, we can select positions with higher projection quality.

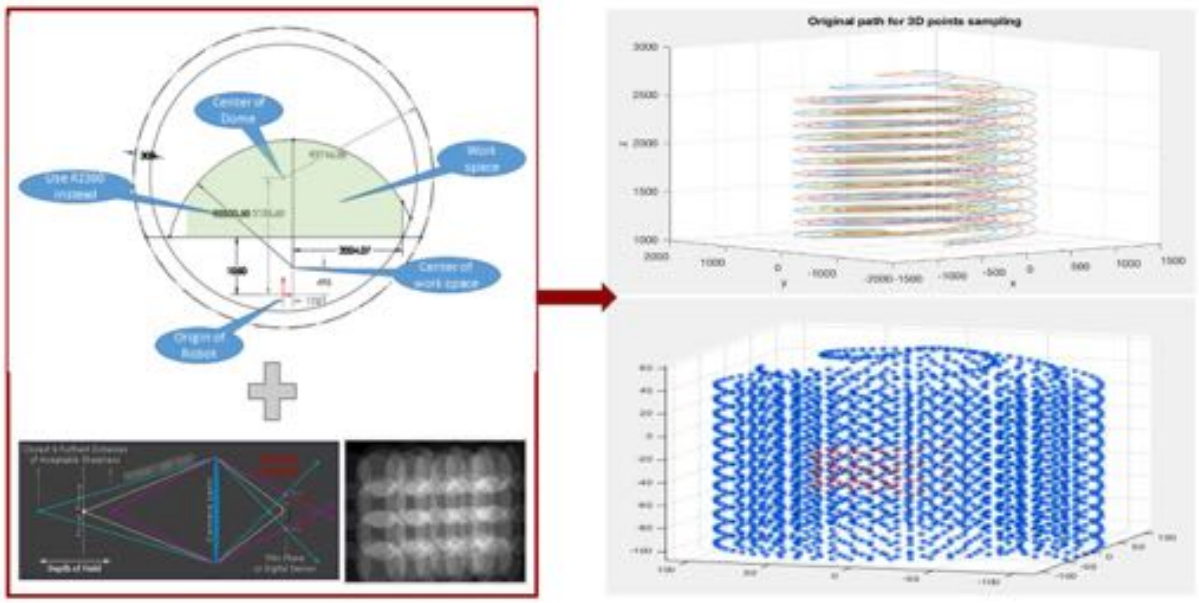


Figure 7.3.3 Path generation based on the DOF and working constraints

7.3.4 Robot simulation

In order to make our simulation real as possible, we not only used the blueprint of the dome and the robot but also made several real measurements from the existing construction spot. This data could help us to create the real situation in the simulation environment which would eliminate the afterward effort to test the path(Fig 7.3.4 left up). The calibration target was also been created in the simulation environment. After we get the model of the environment, we could design the safety work space for the robot arm.

According to the safety requirement in ANSI-RIA_R15.06, we have to set two type of the safe border for the robot arm, one was for programming and the other was the physical border. In our implementation, we set 50cm from the dome as the programming border for the robot arm, and use the dome itself as our physical border. In the robot operation, people was not allowed to walk in the dome, even the robot was in the standby mode(Fig 7.3.4 right up).

The programming work space was basing on the assumption that the dome was a sphere, but the floor and some extra part like a camera were not been considered in that assumption.

Therefore the collision checking still needed to apply. It allowed us to add the extra parts in the environment and didn't have to change the original path design flow if there is no collision. The threshold of the near object and it would indicate the close part by changing the color to yellow. If the collision was detected, the color in both connected parts would change to red to warn the users. In our final implementation, we allowed some robot motion to get closer than the threshold but didn't hit to make the robot path more smooth.

In the RobotStudio, there were several movement type could be used, like move in circle, ,move in linear, ,move in joint rotation. In our implementation, we used move in joint rotation(moveJ). The inverse kinematic of the ABB robot arm was the important part of creating the valid robot joint movement. When using the moveJ, the system would interpolate between two points with the joint rotation, rather than the points itself. Using moveJ could make the robot move smoothly and reduce the sudden movements. In our case, the path how robot move from a point to another was not important, because we wouldn't took a picture when robot moved.

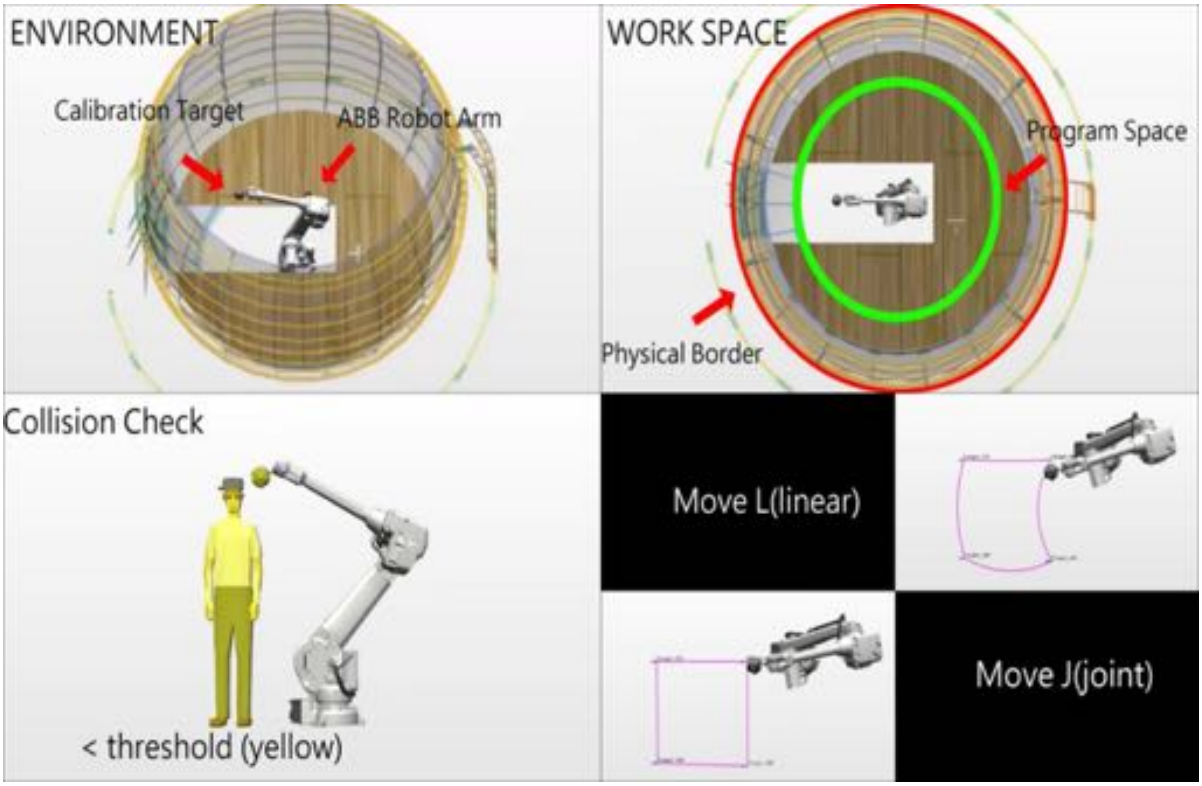


Figure 7.3.4, Simulation parts(left up: Environment CAD model; right up: safe work range; left down: collision check; right down: movement type)

7.3.5 SVE performance evaluation

We projected all the detected corners on the collected images back to the image plane and calculate the coverage of FOV of the 3 cameras we set up for the spring validation experiment, the coverage of the projection of the calibration target is 94.85%, 98.50% and 89.09% respectively.

For the robot arm control part, there was no any collision in the test, and the prepare time for the robot was less than 5 minute that we promised.

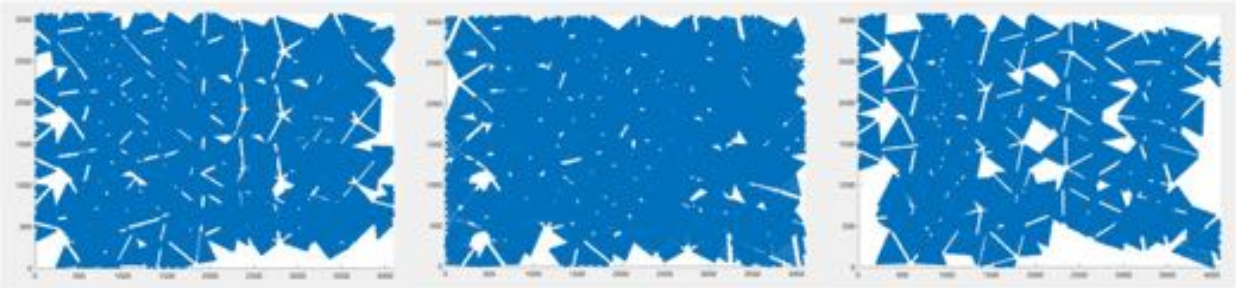


Figure 7.3.5, Detected corners projected back to the image plane

7.3.6 Strong/Weak points

Strong points: The planning algorithm runs very efficiently and the simulation model could react like the real dome.

Weak points: we still need to test the robustness of the path planning algorithms on systems with hundreds of cameras. For the simulation side, the camera simulation was still missing in the RobotStudio. If we could make that the path validation could be iteratively improved much easily.

7.4 Synthetic Image Generation for Validation

7.4.1 System description

It is very important to validate that the geometric calibration algorithm is working as it should, this is where synthetic data comes in. Software engineers need to test their algorithms performance and function during the development process as well as to evaluate results at the time of deployment. Blender 3D provides a good platform to generate great synthetic data to test the actual functioning of the geometric calibration algorithm. The synthetic data generation pipeline (Figure 7.4.1) loads in camera data (intrinsic & extrinsic), object data (calibration target), required configuration and the number of the cameras and the render settings. It spews out Images and other mesh data using the cycles render engine and bpy (blender-python module) module.

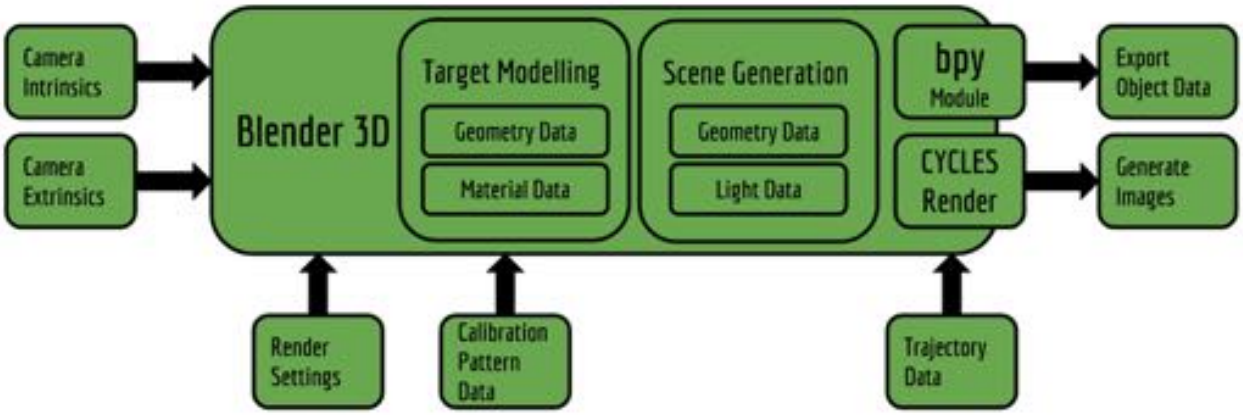


Figure 7.4.1, Blender rendering pipeline

7.4.2 Modelling

The modelling of the virtual environment entails many subsequent modelling and programming stages as listed out below.

Calibration Target Mesh Model: Icosahedron

Using the model editor, the mesh model of the Icosahedron was created in Blender 3D. The mesh model of the required dimension has been shown if Fig. 7.2.2a; This mesh model was then covered with faces created from the vertices of the model. The convergence was parabolic so the model has sharp edges as it would have in the real world and not razor sharp as a computer generated model generally have. The surfaced model is shown in Fig. 7.2.2b.

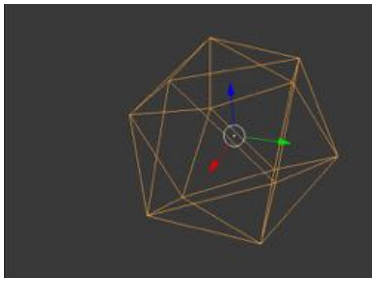


Fig. 7.4.2a, Mesh Model of the Icosahedron

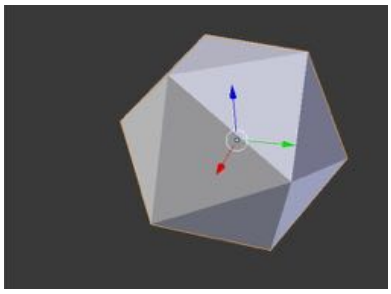


Fig. 7.4.2b, Surfaced Model of the Icosahedron

Calibration Target Mesh Texture: UV Unwrapping

The target pattern has to be imprinted onto the 3D model of the calibration target. This is achieved using a feature known as UV unwrapping in 3D modeling. Here we split the image into an unwrap pattern (Fig 7.4.2c) and this unwrapped pattern is mapped onto the 3D object in the environment (Fig. 7.4.2d). The orientation of the faces on the calibration target is very specific and this has to mapped exactly to the designated vertices of the Icosahedron. The pattern has to digitally mapped with sub-pixel precision. Paper bump map has been applied through the surface of the target.

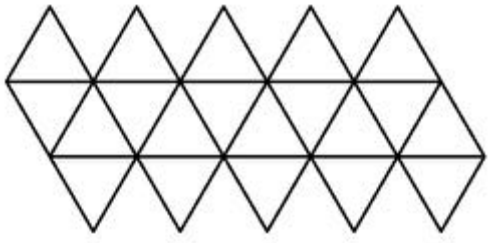


Fig. 7.4.2c, Unwrapping Style.

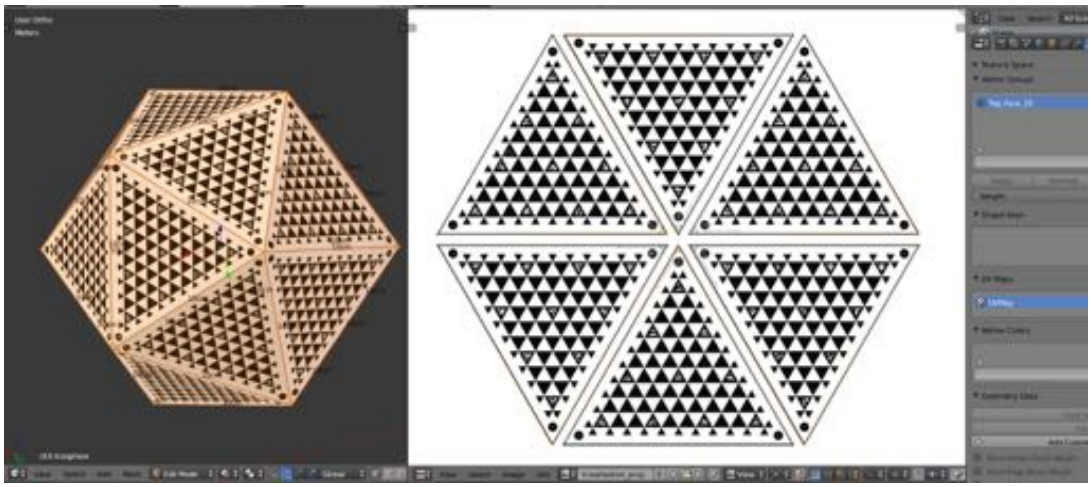


Fig. 7.4.2d, UV Mapping of the pattern layout onto the calibration target.

Visualizing the planes of the patterns

The process of marking the geometric locations of the face patterns involves the assignment of a local reference frame on the calibration target (any one vertex of the target), then assigning each face of the target a local reference frame and finally relating all these frames to the world frame. The reference frame for each individual face is set according to some parameters relating to the geometry of the calibration target itself. The Z axis is aligned along the face normal, the X-Y axes are chosen such that each face's geometric pattern can be described using just on a single description file. Now in figure 7.4.2e one can visualize the reference frames of the faces as well as the calibration target with the script that I have written. This script stores the required data for exporting inside the blender environment and its data blocks, this means that everything is contained inside a very small sized blender file (.blend extension) and can be generated on demand on any computer or terminal.

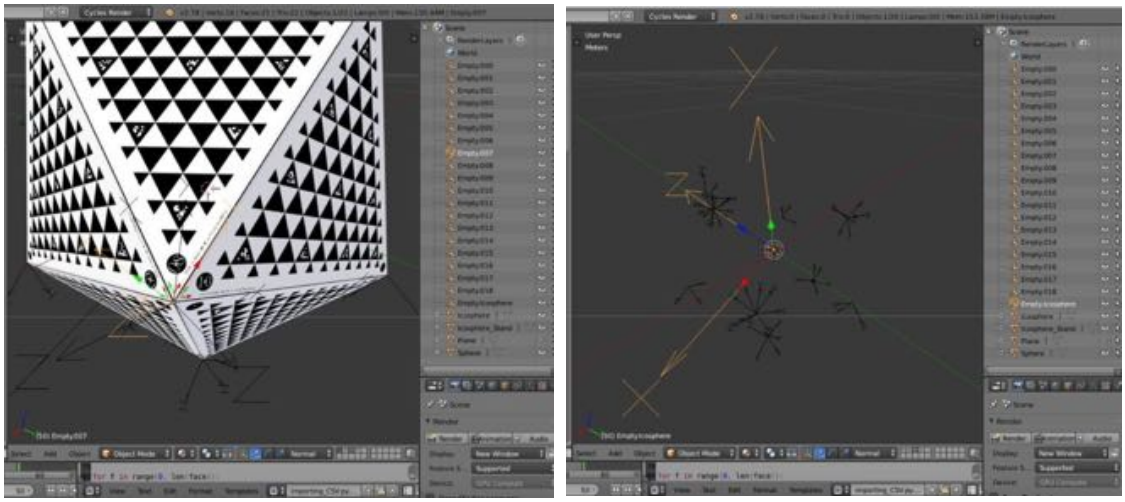


Fig. 7.4.2e, Mappings on Calibration Target Visualized (with & without target)

Importing data in Blender (Camera, Trajectory, Object)
The Camera Data and Trajectory Data for the objects are imported into blender using a predefined CSV as visualized in Figure 7.4.2f.

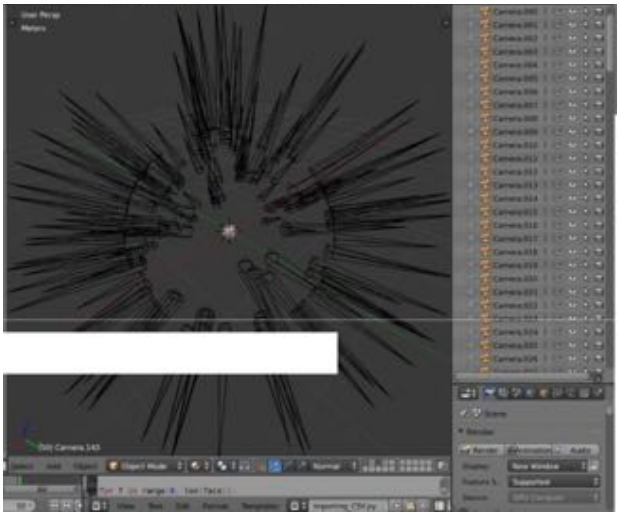


Fig. 7.4.2f, Camera Visualizations.

Lighting of Environment
The lighting for now has been achieved using a simple "SUN" model whereby the whole environment is bathed in uniform light with no specularity. (specular: the property which dictates the sheen/shine factor of a surface).

7.4.3 Simulation

Automated Image Generation
The rendering for all the cameras can be pushed into the render stack by using a click of a button. The output is shown via a diagnostic snippet we wrote to monitor the status of the render pipeline. This diagnostic tool can be activated in the blender environment on MAC/Windows/Linux with ease is compatible with Terminal/Command Promt.

Final Results
The final quality of a sample render is shown in Figure 7.4.3.a and 7.4.3b

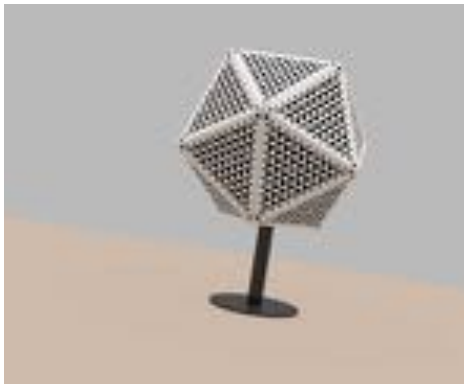


Figure 7.4.3a, Rendered Target

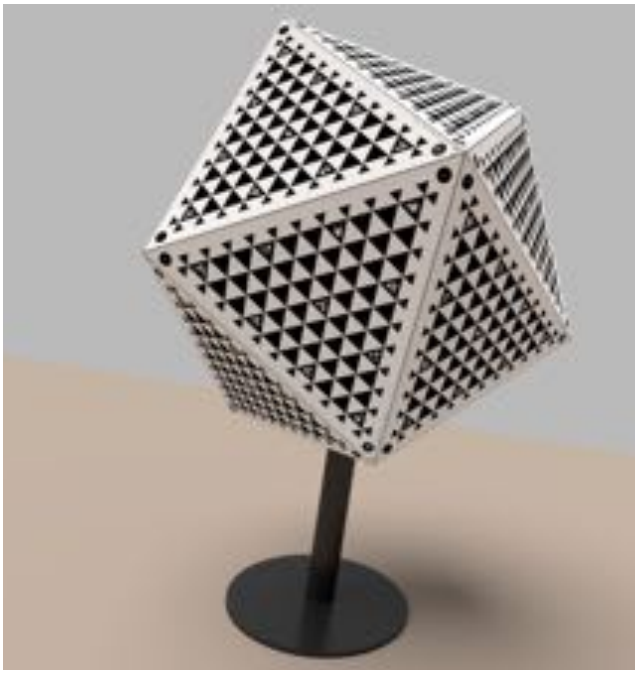


Figure 7.4.3b, Rendered Target

7.4.4 SVE performance evaluation

A reprojection error of less than 0.1 pixel was estimated for the geometric calibration result on synthetic images and that the final results were better than that (0.088743) as shown in Fig. 7.4.5.

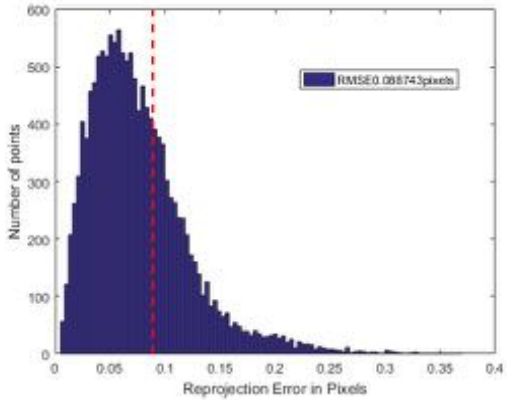


Figure 7.4.4, Reprojection error for geometric calibration on synthetic images.

7.4.5 Strong/weak points

The most useful aspect of synthetic images is the ease of scene generation with the help of python scripts with a build time of couple of days.

One of the major hurdles of generating synthetic images is the render time. A 32GB NVIDIA Titan X graphics card running 16 threads still rendered images with an average render time of 1 minute, which equals 20 days for 144 cameras which is too long. Hence native graphic render farms at Oculus were used to render the images.

7.5 Geometric Calibration

7.5.1 System description

In the geometric calibration, we use the three cameras and one ABB robot arm setup to conduct our experiments(Fig 7.5.1). We would use the images that captured by the camera-robot system to conduct the geometric calibration. We wanted to test two things, one is we could get the high-quality images from path planning part, and the other is our algorithm could deal with these image in an efficient method and provide the good accuracy.

Figure 7.5.1 Camera and robot arm setup

7.5.2 SVE performance evaluation

After the image acquisition, we got 80 raw images per camera to conduct the camera calibration. The process time of the camera calibration was quite efficient and the RMSE reprojection error was 0.1698 pixel (Fig 7.5.2) which was meet our SVE requirement that the reprojection error should less than 0.2 pixel.

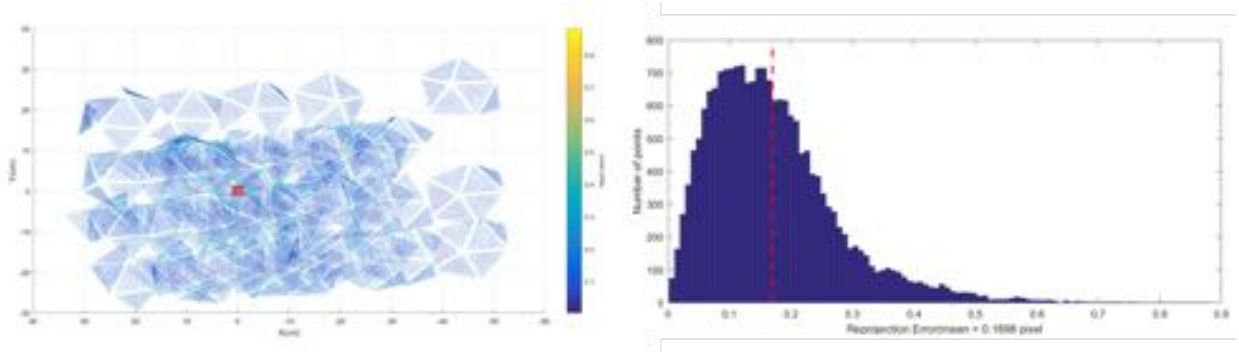


Figure 7.5.2 Geometric calibration result(left: reconstruction result; right: reprojection error)

We also compare the denoise image and raw images in the geometric calibration. The result was quite similar, so in the geometric calibration, we could directly use the raw image and saved time doing denoising.

7.5.3 Strong/weak points

Strong: our system meet the quantity requirements in the efficient way.
Weak: We didn't work on the hundred cameras' data and that could be a challenge for computational efficiency and hard drive storage limitation.

8. Project Management

8.1 Schedule

Table 8.1 Team G Spring schedule

	Excalibr Spring Schedule						
	Sensor Noise	Color Calibration	Geometry Calibration	ABB Robot Arm model	Real Robot Arm	Path Optimization	3D Scene: BLENDER
16-Jan	Sensor Noise Documentation	Color checker detection and segmentation		RobotStudio Model		Building 200 camera parameter model	Generate Images
23-Jan	Sensor Noise Documentation	Color checker detection and segmentation		RobotStudio Model		Building 200 camera parameter models	3D Mesh Model of Camera (EVT)
30-Jan		Color checker detection and segmentation				Generating original path based on motion constraints	Texturing of Camera and Compositing
6-Feb	Concrete plan	Color mapping function+ Color checker detection function robust check		Creating a generating RAPID code process	Delay- Oculus not ready	Generating projection on all camera FOVs	Capture Multiple Images from 2 Cameras
13-Feb	Detailed Plan:	Color mapping function:linear + poly		Creating a generating RAPID code process: show robot position and trigger at the position	Delay- Oculus not ready	Dealing with totally overlapped projections and unnecessary points	Set up scene for 100 Cameras
20-Feb	Integrating Sphere : Read Noise Calculation			Creating a generating RAPID code process	Delay- Oculus not ready	Testing path planning code on randomly selected cameras	Set up scene for 200 Cameras
27-Feb	Integrating Sphere : Pattern Noise Calculation	Color mapping function	Testing 3D geometry calibration algorithm	Creating a generating RAPID code process	Learning how to use control ABB robot arm	Developing evaluation function for projections of calibration target	Set up scene for 200 Cameras

6-Mar	Integrating Sphere : PRNU Noise Calculation	Color mapping function	Testing 3D geometry calibration algorithm		Learning how to use control ABB robot arm	Integrating evaluation function into path planning	Set up scene for 200 Cameras
13-Mar	Spring Break						
20-Mar	Integrating Sphere : PRNU Noise Correction	Color mapping function	Testing 3D geometry calibration algorithm		Learning how to use control ABB robot arm	Integrating evaluation function into path planning	Build 3D model for Dome
27-Mar	Integrating Sphere : PRNU Noise Correction	Real environment experiment(no robot)	Testing 3D geometry calibration algorithm	Integration the geometric calibration system		Testing path planning code on randomly selected cameras	Build 3D model for Dome
3-Apr	Integrating Sphere : Thermal Noise Calculation			Integration the geometric calibration system		Testing path planning code on all cameras	Build 3D model for Dome
10-Apr	system Integration	Real environment experiment			ABB robot arm trajectory control	Revising path planning code and testing on all cameras	Texturing and Compositing of Scene
17-Apr	system Integration	Real environment experiment		Testing geometry calibration		Revising path planning code and testing on all cameras	Texturing and Compositing of Scene
24-Apr	system Integration	Real environment experiment		Testing geometry calibration		Testing on real ABB robot (Dependency : Robot is ready)	Texturing and Compositing of Scene
1-May	system Integration		camera calibration validation	Testing geometry calibration			Texturing and Compositing of Scene
8-May							Generate Images

In this semester, we followed this schedule to work on. By doing work dependency and length estimation, we found out that the critical path of our project was the final integration. Because at that time, the dome in Oculus was not functional and even the robot arm was just

installed. Hence, we not only kept tracking the construction progress of the dome but also managed our work to fit it. For example, we started the simulated the robot arm instead of controlling the robot arm; we postponed the geometric calibration until we could get the access to the camera and the abb robot arm. This strategy was proven good for our case and made us to finish part of task in time and not stop due to the dependency.
For each subsystem schedule, basing on last semester experience, we made a huge change about how project management worked in this semester. In the first beginning, we set the several monthly goals for each subsystem, and discussed with the sponsor for goal validation. Then we individually did work breakdown for each week. However, due to the uncertainty of actual work and the change of our project domain, we would meet with sponsor each week and discuss about the difficulties on the current work, and then made some adjustment for the next week work. Then for each month work checking, we would check the current progress for each subsystem. If there was a delay or some unsolvable problem, we would discuss with sponsor to seek help or do descope the problem.

8.2 Budget

As our project was sponsored by Oculus Research, Pittsburgh, almost all of the equipments and components were purchased by them. A detailed description of the expenses have been mentioned in the table below:

Table 8.2 Budget list

	Budget List					
Sl. No	**Item**	**Quantity**	**Unit(s)**	**Cost per unit (USD)**	**Cost (USD)**	**Purchaser**
1	AEROTECH PRO225SL	1	set	20,000	20,000	Oculus
2	AEROTECH PRO115SL	2	set	15,000	30,000	Oculus
3	AEROTECH A3200 Controller	3	set	1,000	3000	Oculus
4	ABB Robot arm	1	set	150,000	150,000	Oculus
5	Emergent:HR-12000 with lens	3	set	5,400	16,200	Oculus
6	Desktop PC	2	set	1,500	3,000	Oculus
7	3D printing material(PLA)	1.5	kg	47.47	71.2	Oculus
8	Cable carriers	12	ft	19.63	236	Oculus
9	Integrating sphere	1	set	25999.00	25,999	Oculus
Total cost (USD)					**24,8505.8**	
Amount spent from Team G budget (USD)					**0**	

8.3 Risk Management

The major Risks faced by us have been mentioned below:
1. Construction Delay
2. Robot arm malfunction
3. Camera malfunction
4. Integration failure

To assess each of the risks during our project, we have developed a rubric that defines our risk likelihood and consequences. This can be visualized from Table 8.1.

Table 8.1 Risk likelihood legend & Risk consequence legend

1	Highly unlikely
2	Possible to occur
3	Likely to occur
4	Expected to occur
5	Estimated to occur

1	Time delay < 3 days
2	3 days <Time delay < 1 week
3	1 week <Time delay < 2 weeks
4	2 weeks < Time delay < 1 month
5	Time delay > 1 month

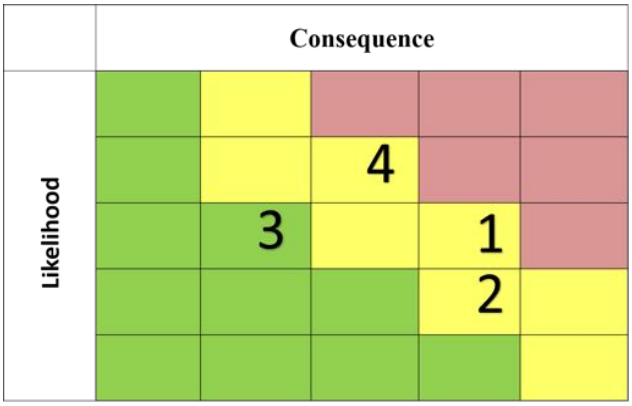
The table below shows the risk level definition

Table 8.2 Total risk level definition

Low risks	1-5	Acceptable risks
Medium risks	6-10	Risk tracking is required
High risks	10+	Mitigation strategies are mandatory

We have managed to mitigate all of our risks throughout our project. The Likelihood - Consequence Matrix is visualized in the table below:

Table 8.3 Risk Matrix



In the beginning of the project, the hardware, robot arm and camera were always one of the biggest issue of our project. Because we were not the expert of these equipment, hence we would unintentionally make small mistake when operation. However, the accessible quantity of the equipment was limited and it would take a long time and extra cost to buy a new one. In order to prevent that happens, we would read the manual and consult the sponsor before we conduct any operation and for each advanced operation, we would start some checking test to prevent any hazard happen.

In the second semester, one of the biggest problem for our project was the construction progress of the dome. In the end of the February, the robot arm and the dome were still in construction. At that time, we discussed with sponsor that if the construction was not ready after the spring break, we need to change our goal from the dome to another space which had camera rather no robot arm. Fortunately, just before the spring we were noted by the sponsor that we could start use the robot arm, which much reduce our risk about the construction delay. On the other hand, the camera still couldn't mount on the dome in time, so we changed our scope from 20 cameras on the dome to 3 cameras on the tripod. By changing the final goal, we were able to start integration our robot system as soon as possible without waiting. In the end, the enough preparing time let us could solve some hidden problems.

Last, the integration part in our system was quite straightforward but it depend on the subsystems worked correctly. Like we mentioned before, the rich left time for testing and integration gave us to opportunity to fix it and also did the fine tune.

9. Conclusion

In this day and age of automation, camera calibration still relies heavily on human labour and intervention. Calibration is a crucial component for any application which involves an engineering system operating on sensor-fusion. An automated calibration system would be highly beneficial to all such systems. With Oculus behind us, we designed a turnkey solution for

automated calibration of multiple cameras in a constrained environment with minimal human intervention. This system exceeds the accuracy, reliability and precision of the existing methods employed in camera calibration.

9.1 Learning

For project management, the dynamic schedule system was working quite well for the project like us with several subsystems, because for each subsystem, there were different developing curve. Hence, the flexibility of the schedule and ability to reschedule was important for actual project managing.

9.2 Future Work

As mentioned in section 7.1.4, Sensor noise calibration requires a standalone calibration for each camera. This would involve disassembling the cameras and with the current architecture, it is not possible for the entire system to operate in real time as an end to end fully automatic system. The work done so far can be improved by designing a system which could enable real time sensor noise calibration. A possible mechanism which can be explored can be augmenting the calibration target with a uniform light panel which can make it possible to collect flat images required for real time noise calibration.

10. References

1. http://ericfossum.com/Publications/Papers/1999%20Program%20Test%20Methodologies%20for%20Digital%20Camera%20on%20a%20Chip%20Image%20Sensors.pdf
2. http://sensing.konicaminolta.us/2013/10/measuring-light-intensity-using-a-lux-meter/
3. http://photo.net/learn/dark_noise/
4. http://www.gamma-sci.com/products/light_sources/
5. https://www.labsphere.com/labsphere-products-solutions/imager-sensor-calibration/
6. http://theory.uchicago.edu/~ejm/pix/20d/tests/noise/