Smart Displays and Projectors with Embedded Visible Light Communication (VLC)

ECE FYP FINAL REPORT

PROJECT NAME: SMART DISPLAYS AND PROJECTORS WITH EMBEDDED VISIBLE LIGHT COMMUNICATION (VLC)

Project ID: PY03a-20

Supervisor: Professor Yue Patrick

Author: CHEN, Runzhou

Student ID: 20492340

Date: 24 March. 2021

TABLE OF CONTENTS

Abstract

In view of the latest development of smart indoor services, a visible light positioning (VLP) algorithm is proposed in this project to provide the foundation for a smart display and projector system. The VLP algorithm mainly applies geometry methods to first calculate the angles of the camera and the user's location based on a single input image that contains a simple object quadrilateral. Afterwards the calculation is extended to the quadrilateral with a rolling shutter pattern that corresponds to visible light communication and other shapes. The algorithm is implemented in Matlab and then converted to C++. The test results show that the algorithm can estimate the camera angle and location accurately and the work can be further expanded to indoor robot positioning and other relevant applications.

Section 1. Introduction

1.1 Background and Problem Statement

Recent years have witnessed rapid social-economic development as enormous buildings like airports, universities, and shopping malls are built in most of the major cities. Therefore, the accessibility of feasible intelligent indoor services is becoming essential to satisfy people in daily life as they are spending more and more time indoors. Comprehensive indoor services based on radio frequency technology are a consequence of the booming demand for quick communication between people and the information system of those buildings.

Figure 1. The Gobo projector used in the project

As implied by its name, a well-designed indoor system ought to render services such as information delivery, indoor positioning, or other novel functionalities based on users' locations. The existing indoor system, though, fails to suffice for the increasing demand of service capacity since precise indoor localization is formidable to accomplish simply using the Global Positioning System (GPS) [1]. Besides accuracy, the high cost and large energy consumption of the current products might also impair the prevalence of indoor services. Notwithstanding, these problems can be managed by the smart projector system based on visible light positioning (VLP) presented by the research group led by Prof. Patrick Yue [2]. Using visible light communication technology, LEDs serve as an information system and the

building's illumination concurrently and provides a variety of indoor services in real-time, an example of the projector used in the project is shown in Figure 1.

Visible light communication (VLC) is a fashionable broadcast communication technology that employs LED light modulation to deliver information. The light from the LED transmitter will be received by a CMOS camera image sensor or a photodetector to extract the information. Due to the highly directional nature of light, the information transmitted can be acquired directly by aiming the camera or sensor at the LED transmitter. The above idea can be extended to any device that contains a light source. As LEDs are going to be the ubiquitous illumination source for indoor circumstances and it has advantages of better energy efficiency, long life span and the ability of instantaneous On/Off [3], it provides the foundation of the implementation of the smart indoor service system.

Modern LCD displays and projectors both use LED light sources for display and projection for their higher energy efficiency, improved contrast ratio, and enhanced display quality [2]. Therefore, in the smart projector system, the LED backlight of the LCD display or the light source of the Gobo projector is modulated to broadcast information using VLC. In order to cooperate with the projector, my work will render algorithms based on single-luminaire visible light positioning (VLP), which is an important part of the prototype of the smart projector system. The VLP-based algorithm is capable to mitigate the current shortcomings of mainstream broadcasting technology and provide foundations for the promising indoor services that will be developed in the future.

1.2 Motivation

Nowadays, large buildings such as shopping centers are attaching more importance to diversity and intelligence. On one hand, the customers desire to take advantage of the smart indoor system for more information and better service. On the other hand, the building managers need reliable infrastructures to implement this kind of service and provide people with an enjoyable experience.

The gap between demand and supply of feasible technical support to the indoor service is the motivation of this project. By filling such a gap, the smart projector system based on the VLP algorithm is believed to make a significant contribution to the enhancement of the present situation of the indoor system. On a lower level, the smart projector embedded with VLC will bring opportunities that users can interact with the building and different indoor services can be implemented as the user’s location is estimated. On a higher level, since the VLC projection can be decoded by a headset or other device, the product may also lay the foundation for the advancement of indoor entertainment such as VR or even the extensive adoption of smart buildings.

1.3 Literature Review of Existing Solutions

Nowadays, there are diverse indoor services technologies employed by large buildings in order to build an association between users and their information systems. These strategies intend to provide users with distinct indoor services based on their indoor position. Currently, three kinds of technologies employed by buildings for indoor services are Wi-Fi-based, Bluetooth-based, and VLC-based.

As the dominant networking transmission system, Wireless fidelity (Wi-Fi) is established in every big building. Wi-Fi-based indoor service is adopted by more indoor firms today as a result of the maturity of Wireless Local Area Network technology. The user can log on to websites for valuable information about the shops inside a shopping center, and data analysis will be applied to render more desirable services [4]. Nevertheless, as Wi-Fi demands the user to seek their information manually, the indoor service based on it is not intelligent enough. Such kind of passive working mode would become troublesome when too many details are needed to be looked up. Also, in some special places like hospitals or airports where the radiofrequency waves are transmitted for the purpose of communication or diagnosing, the Wi-Fi-based system would interrupt their systems and cause troubles [5]. Thus, this approach not appropriate.

In order to mitigate the above problems, indoor service based on Bluetooth is admired by some researchers. As mentioned by Ehud Mendelson [6], the Bluetooth-based system utilizes machine learning to infer the user's preference on account of the information collected from the Bluetooth devices deployed. When the user's phone detects the signal from multiple Bluetooth devices, trilateration, which is an energy-range-based mathematical approach, will be used to estimate his current location and deliver the service accordingly [7]. Compared with the Wi-Fi-based method, there is no need for the user to keep looking for the information in distinctive places because it will be shown on the phone automatically when nearing them. However, according to Hao Xia [7], the Bluetooth-based indoor positioning can have an error of approximately 2.4 meters due to the signal power oscillation, which makes Bluetooth not precise enough for a high-performed indoor system. Furthermore, Bluetooth devices usually have high infrastructural cost as well as energy consumption, which is another troubling factor for the implementation of a feasible indoor service. Presumably, a more appropriate technique is coveted.

Compared to traditional radio frequency communication, VLC has certain advantages that make it suitable for various technical applications, which are the motivations of our work. First, the accessibility of the spectrum (usually 380~750THz) makes it possible for low-cost broadband communication to mitigate the problem of congestion inside the spectrum which is manifest in the 2.4 GHz band reserved for civilian (GSM), industrial, and medical (ISM) [8].

Secondly, due to the simplicity of VLC signal transmission and the dispensability of IQ modulation compared to classic RF communication, it does not demand advanced algorithms or frequency mixers to correct impairments in RF signal such as IQ imbalance and phase noise that arises from the sharp, short term, random phase fluctuations that occur in a signal [9]. Lastly, the VLC's location-related communication capability perfectly satisfies the requirements from the ubiquitous location-based service: discount or advertisement information in a shop can be accurately delivered based on the users' locations [10].

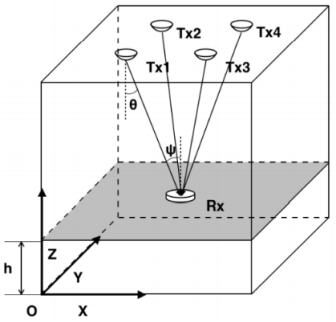


Figure 2. A demonstration of VLP based on multiple sources

At the receiver end, when the sensor receives the light signal from the projector, the system needs to calculate the position of the mobile or headset accordingly. Visible light positioning (VLP) is introduced in many previous works and usually it takes more than three LEDs, an example is shown in Figure 2. The basic idea is to apply trilateration by using visible LED as anchors. Each LED, besides its primary illumination part, is used as a location mark as well. It distributes the location of the bulb and the duty cycle to expedite localization of the receiver side as location beacons carrying information, via the light carrier [11]. A receiver (for example, a headset) utilizes a light sensor to recover the location data and then calculate the received signal strengths (RSS) from multiple bulbs and then calculate the distance to every LED light source via the optical channel. Finally, the receiver determines its position according to the recovered data and distance measured to all beacons [11].

To better approximate the situation of using a projector that will produce only one luminary, we will discuss the single-LED VLP algorithm that combines the geometric features and IMU readings to deal with the IMU data error [2]. The basic theory is to construct VLP based on a single LED by integrating the projective geometry and IMU measurements. While the IMU sensor measurements can offer angle information for localization, the data is not precise enough, and calibration is needed to address that. As we know, abundant information of orientation is contained in the projective geometry. For instance, when casting a circular LED luminaire on the wall, the high location angle of the observation along the horizontal axis will lead to the flat shape of the ellipse we see from the camera image [2]. Though the orientation cannot be derived straight from the projective features, it is still possible to employ the projective geometry for the calibration of the IMU data.

This proposed VLP method in [2] has, however, unavoidable flaws. The error of localization mainly originates in the image processing steps which include the geometric features and extraction ellipse fitting. Besides, the weakness of perspective projection assumptions and the IMU data of angles will lead to problems as well. Furthermore, since the projection of the transmitter is in square shape instead of a circular, the projective geometry for the calibration of angle is different than in our project, but the basic idea and calculation of our proposed method will be similar to this method.

1.4 Objectives

1.4.1 Main Objective

In view of the above factors, a receiver algorithm based on visible light positioning (VLP) for the smart projector system will be developed in this project to estimate the indoor location of the user based on a single luminary, which will lay the foundation for the further development of smart indoor services. Afterwards, tests will be conducted to estimate the error rate of the camera angle and location calculation.

The intention of the existing smart projector system is to provide a cheap and cost-effective way of transforming existing display devices into smart VLC embedded devices which users can interact with using their smartphones. The system involves two parts: the transformation of the transmitter hardware by VLC modulator into the backlight display driver circuitry of the display/projector and a receiver algorithm based on VLP to receive and demodulate those signals. Our project will focus on the implementation of the second part.

1.4.2 Objective Statements

To achieve the ultimate goal of indoor positioning based on visible light, our project will follow a series of stages.

Objective Statement 1: to locate the four corners of the quadrilateral luminary from the input image based on a number of algorithms and functions.

Objective Statement 2: to find the plane of the quadrilateral in the previous statement and to calculate the roll and pitch angle and then the indoor location of the user.

Objective Statement 3: to expand the angle and location calculation to the case of the rolling shutter pattern image.

Objective Statement 4: to develop the method that performs similar indoor-positioning processes based on the input images that contain other forms of LED (e.g., tubes) and test its feasibility.

Objective Statement 5: to convert the algorithm from a Matlab program to a C++ program based on OpenCV library and specific VLC recognition algorithm and compare the accuracy between these two.

1.5 ECE Knowledge

This project is about visible light communication (VLC), which is basically a way of information transformation. In order to understand how the information is encoded and transmitted, knowledge from ELEC2100 Signal and Systems is essential, including signal modulation and demodulation. Since the biggest part of this project is based on Matlab for image processing, the skills from ELEC4130 Digital Image Processing will be important to construct the code, for example, the way of image reading and restoration, colour image processing, image compression, image segmentation, image representation and description, and also the usage of OpenCV library. Meanwhile, light communication also requires knowledge about physical optics including some geometry calculations, so the course ELEC4610 Engineering Optics is also relevant to this project.

Section 2 METHODOLOGY - PROGRESS

2.1 Overview of the Main Objective

2.1.1 Main Objective Diagram

The universal VLC-based smart projector system that is the basis of this project is demonstrated in Figure 3. The block diagram includes two parts, the front one is the transmitter system by which Li-QR is broadcasted using an LCD display and the latter one is the receiver that implements visible light positioning (VLP). For the transmitter part, a VLC modulation circuit is inserted after the LED driver to modulate the light of the projector. The Li-QR controller contains an MCU-based controller that can create binary sequences for each unique Li-QR ID. It is required to identify the LED driver circuit on the board and retrofit a VLC modulator device to enable modulation on the display. Our project will design the receiver part (in blue colour) that enables VLP by means of a series of geometrical algorithms.

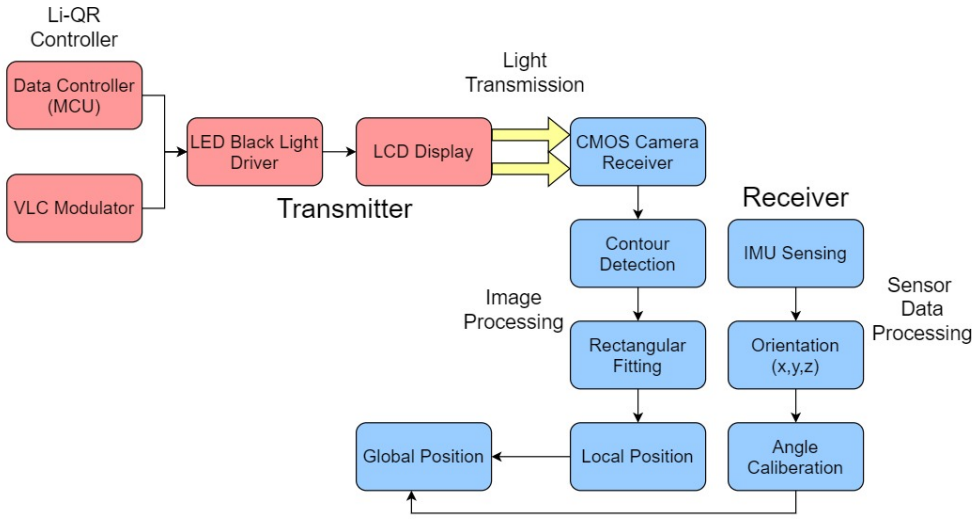


Figure 3. The flowchart of the smart system

A CMOS camera inside the smartphone on the receiver side captures the projected image. By applying image processing, the captured image which includes ID transmitted in the form of a rolling shutter pattern can be decoded. Then, the image will be handled by the algorithm which first detects the area of interest (AOI) and afterwards determines the threshold between different stripes in order to do the translation from the image into a binary sequence of zeros and ones. Then, the produced binary stream will be matched with a data-frame for identification of payload, preamble, and error detecting sequence. Finally, the results will be compared to the URI and then transmitted to the application layer in which the corresponding information is loaded based on the link.

In order to calculate the position of the user's device, we implement VLP based on a single rectangularly luminary. As the projector casts light on a wall, the angle, and the distance of the device with respect to the projection can be measured by integrating image processing and IMU data. For example, a universal rectangular LED luminary will often be projected to a trapezoid on the wall. For extraction of the trapezoid, we can apply Gaussian blur and the Sobel-operator for contour detection [11]. Afterwards, we perform the VLP algorithm including localization based on the geometric value from the image and angle calibration based on IMU sensor data (pitch, roll and azimuth). Combining the results, we can get the global location of the device. The mathematics work such as matrix multiplication and trigonometric calculation will be done in Matlab and converted to C++.

A demonstration of the expected smart projector system is shown in Figure 4. The projector will cast light on the wall and our receiver app will assist the headset to locate the user and estimate the angles. Then the system controller can provide services accordingly. Once the modulation function is implemented, we will use a receiver application with a VLP algorithm

that can run on a smartphone and receive data from the modulated device and determine the position of the device in the room. Overall, our project will help render a smart display and detection system that is able to achieve high-performance indoor data transmission and positioning accuracy using visible light.

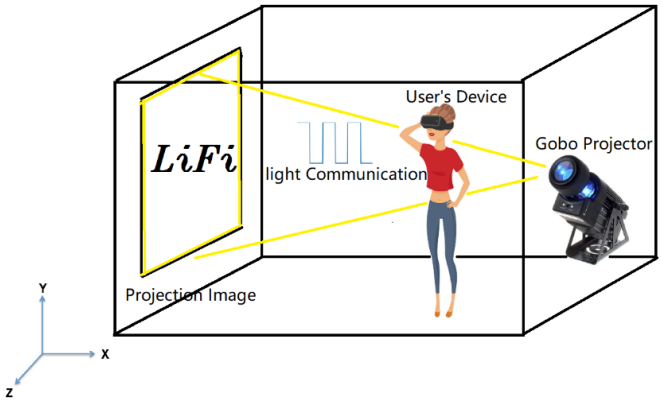


Figure 4. A demonstration of the usage of the smart projector system

2.1.2 Components List

Table 1. List of Specifications

Items*	Specifications/Model
Camera Roll Angle	Can be obtained from the calculation or sensor data, in degrees.
Camera Pitch Angle	Can be obtained from the calculation or sensor data, in degrees.
Camera Location	Can be obtained from the calculation, in centimeter.

2.2 Objective Statements

2.2.1 Objective Statement 1 - to locate the four corners of the quadrilateral luminary from the input image based on a number of algorithms and functions

2.2.1.1 Objective Statement System Block Diagram

Figure 5 is a flow chart of the first objective statement. The proposed algorithm finds the corner of the luminary from the input image with the following steps: first, it performs Harris corner detection, and then edge detection by the Canny operator. Then it does Hough line detection based on the edges. Then the program can determine if the trapezoid is vertical or horizontal and pick 4 point-of-interests for perspective correction. For this further operation,

the 8 points of interest (4 before correction, 4 after correction) will be the input of the homogenous matrix and calculate the 8 parameters. After that, the correction is performed by Matlab.

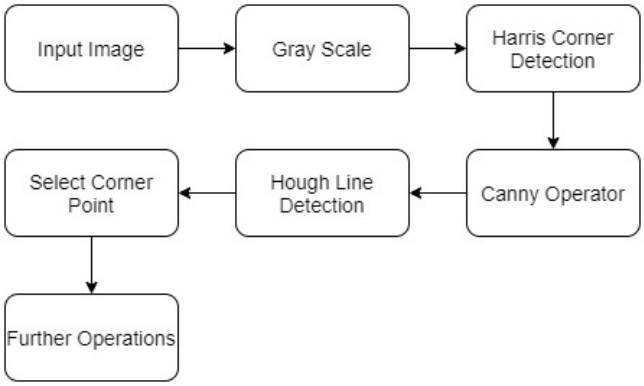


Figure 5. The flowchart of objective statement 1

Both Harris corner detection and Hough line detection are based on pixel calculation and can extract the shape information from the image. The Canny Operator is a function to estimate the edges of the object inside an image and we use it to assist the Hough line detection and to increase its accuracy. During the process of corner point selection, we will compare the results of both algorithms so that the points that match both must be a point of interest (POI). To test the performance of POI selection, I will do perspective correction and see the results.

2.2.1.2 Specific Task 1 – Harris Corner Detection

Harris detection is used to find corners in a picture. The basic idea of the algorithm is to use a fixed window to slide on the image in any direction and compare the gray degree of change of the pixels in the window before and after the move. If there is a large gray change in any direction of the slide, then we can consider that there are corner points in the window. This differentiates the gray-scale gradient in both x and y-direction. The key matrix M is constructed as

$$M = \sum_{x,y} W(x,y) \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix},$$

where W(x,y) is the window function that controls the input pixels and the 2-by-2 matrix represents the products of components of light intensity gradients in the x and y-direction.

The corner usually has a large derivate in two directions compared to the line or flat plane. By determining the eigenvalue of M (λ_1 and λ_2), we can estimate the response of each pixel by the following formula using determinant ($\lambda_1 * \lambda_2$) and the trace of M ($\lambda_1 + \lambda_2$). In the following

equation, we use R to determine if there exists a corner. When R is larger than a threshold, we say that it is a corner and record its location [12]:

$$R = det(M) - k \cdot (trace(M))^2 .$$

To implement the Harris corner detection, we conducted the following process in Matlab: 1. Calculate the partial derivatives I_x and I_y in the X and Y directions of the input image. 2. Apply a Gaussian low pass filter and calculate I_{xy}, I_x^2, I_y^2. 3. Define the value of k, threshold and R. 4. Go over each pixel and compare its R-value to the threshold. 5. Mark those pixels with a large R-value and output their coordinates. By setting different thresholds, we can control the number of corners detected. In Figure 6, the detected corners are marked in red crosses. The results demonstrate that the Matlab program can successfully find the apparent corners of the object in an image.

(a) (b)

Figure 6. The input (a) and output (b) image to the Harris corner detection

2.2.1.3 Specific Task 2 – Hough Line Detection

Hough Transformation is the main method to find lines in an image. The reason for using Hough Transformation is that it can provide details of lines such as the start and end position of each line, angle theta and length r in polar coordinates. Using polar coordinates, we transform each point (x, y) to a curve and turn line detection in the Euclidean space into finding the greatest number of sinusoids passing through the point (r, θ) in the polar parameter space [13]. Canny edge detection is required to transform the grayscale to binary. Figure 7 is the output of our Hough-detection program implemented in Matlab. The upper left image is the grayscale of the input image and the green lines in the lower right image are the edges detected from the input. This means our program can automatically select a number of edges in the image according to the threshold.

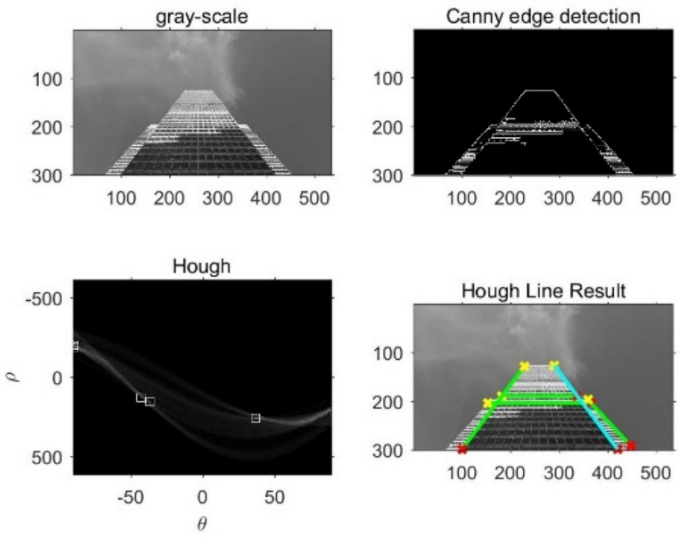


Figure 7. The input and output image to the Hough Line detection

2.2.1.4 Specific Task 3 - Perspective Transformation and homogenous coordinates

In order to test the efficiency of the POI selection in the previous steps, I used the POIs to perform perspective transformation for the input image. A Perspective Transformation is a projection of an image onto a new visual plane. It is known that both the scaling and the rotation transformation can be expressed as cascading of matrix multiplication. However, translation is not a form of matrix multiplication, but matrix addition. The use of homogenous coordinates changes the image from 2D to 3D coordinates that allow the combination of translation with rotation and scaling. Following is an example of rotation, scaling and translation operation in homogenous coordinates:

$$\begin{bmatrix} x2 \\ y2 \\ 1 \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x1 \\ y1 \\ 1 \end{bmatrix}, \begin{bmatrix} x2 \\ y2 \\ 1 \end{bmatrix} = \begin{bmatrix} kx & 0 & 0 \\ 0 & ky & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x1 \\ y1 \\ 1 \end{bmatrix}, \begin{bmatrix} x2 \\ y2 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 0 & tx \\ 0 & 0 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x1 \\ y1 \\ 1 \end{bmatrix}$$

In homogenous coordinates, suppose we want to move a point (x, y, 1) to another location (X, Y, Z). It should multiply a 3-by-3 matrix A, where $a_{33} = 1$. And we also need to transform (X, Y, Z) to 2D (divide by Z). For a quadrilateral, we have eight variables to calculate to determine X/Z and Y/Z, four of which are before the transformation and the other four after the transformation. We can write the formula into matrix multiplication as follows:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} x & y & 1 & 0 & 0 & 0 & -xX' & -X'y \\ 0 & 0 & 0 & x & y & 1 & -xY' & -yY' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix} \begin{bmatrix} a_{11} \\ a_{12} \\ \vdots \\ a_{32} \end{bmatrix} = \begin{bmatrix} X' \\ Y' \\ \vdots \\ \vdots \end{bmatrix}$$

That is why eight points are required to solve the following polynomial. Afterwards, Matlab will correct the perspective automatically.

2.2.1.5 Specific task 4 -- Automatic perspective correction

In order to test the performance of the POIs selection, I did the automatic perspective correction for Figure 8. The Matlab program will find the corner points of the distorted window and convert it from a trapezoid back to a rectangle. The resulting image is shown below and we can see that the window's perspective is corrected, which means the steps listed above are executed without error.

Figure 8. The input and output image to the perspective correction

2.2.1.6 Objective Statement Evaluation

Since both the Harris corner detection and the Hough line detection cannot distinguish the object automatically, the influence from the background will play an important role in the algorithm accuracy. When there is too much disturbance in the background, the algorithm may have difficulty finding the correct corners of the object. So it is necessary to ensure that the input image has a clear background. In conclusion, the objective statement has been completed.

2.2.2 Objective Statement 2 - to find the plane of the quadrilateral and calculate the roll and pitch angle and the indoor location of the user

2.2.2.1 Objective Statement System Block Diagram

In the next step, I mainly focus on camera positioning from a single image of a rectangle. The task is basically determining the coordinates and the pitch and roll angle of a camera with the given picture of a square light projection. I did research on two different methods, the geometrical approach and the perspective-n-point (PnP) approach. The core of PnP is to calculate the projection relation between N feature points in the world and N image points in the image, so as to obtain the pose of the camera or object. Though PnP is precise and effective, it is time-consuming to implement it in MATLAB, so I chose the geometrical approach which is concise and straight-forward. The main steps are shown in Figure 9.

As presented in the flowchart, after the selection of POIs based on the last objective statement, we perform the geometry methods to find the four edges and the expanded quadrilateral for the input object to calculate the roll and pitch angle of the camera. Then, we can estimate the camera location based on the angles and the object's center point.

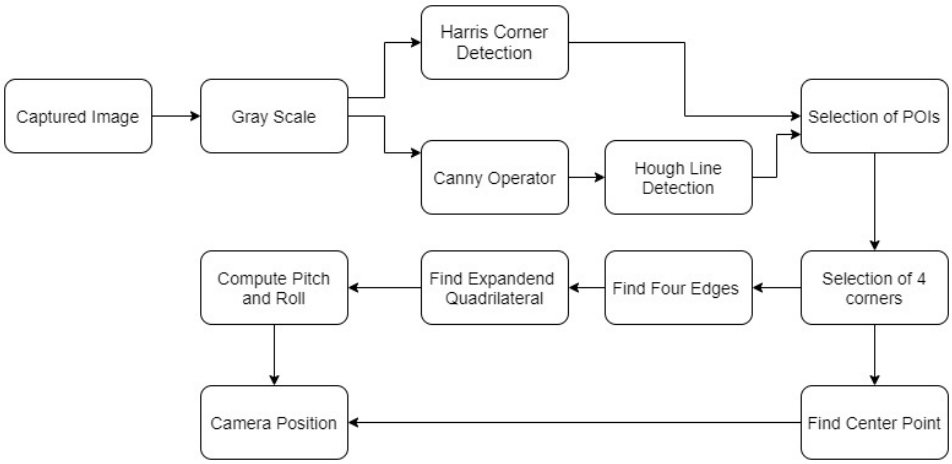


Figure 9. The flowchart of objective statement 2

2.2.2.2 Specific Task 1- Determine the Coordinates of the Projected Object by Internal Matrix

In a simplified pinhole model (shown in Figure 10), we can represent the projected object's 3D coordinate in pixels. The x and y coordinates are the location of each pixel in the image, while the z coordinate is the focal length (f) in terms of pixels. To estimate the camera-object distance and the scale, we need to make use of the internal matrix which converts the camera coordinates to the pixel coordinates.

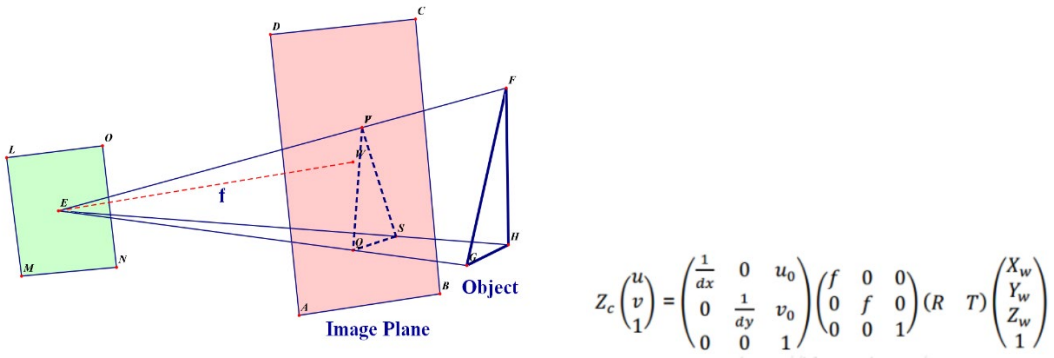


Figure10. A simplified camera pinhole model, where f means the focal length

The internal parameters of the camera are 1/dx, 1/dy, U0, V0, and f. U0 and V0 are the projected coordinates of the optical axis in the image pixel coordinates system, namely, the coordinates of the main point. 1/dx and 1/dy are the physical size of each pixel in the x-direction and the physical size of each pixel in the y-direction, respectively. However, as we don't know Z_c (z-value of the object in the camera's coordinates), we cannot divide the location in the image plane by the internal matrix to get the location in the camera's coordinates. Also, the image plane has its original point at the upper-left corner instead of the center so the angles calculated in the next step will not be correct. As a result, I represent each point of the projected object by the following with the unit of a pixel so that it is assumed that the camera is pointing to the center instead of a corner:

$$[x \quad y] \rightarrow [x - u0 \quad -y + u0 \quad f/dx]$$

.

2.2.2.3 Specific Task 2 - Camera Angle Calculation of a Center-Located Quadrilateral

When we have an image of a center-located quadrilateral, we first detect its four vertexes by the method described in Section 2.2.1 and then transform its coordinates to 3D by the above formula. The four vertexes will be labeled P_1, P_2, P_3, and P_4. Then we calculate the pitch and roll of the camera by the following geometrical method.

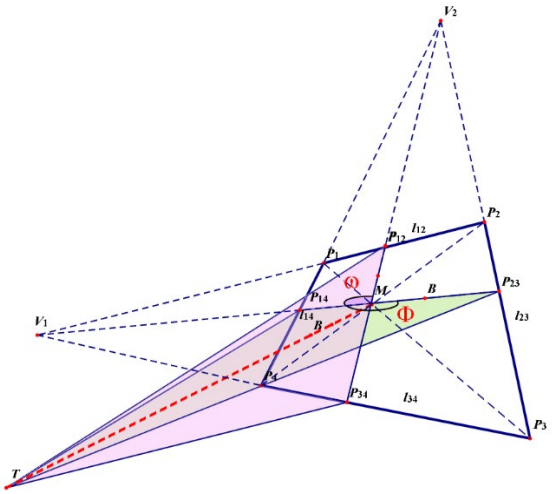


Figure11. A quadrilateral seen from camera at point t

In homogeneous coordinates, we can get a line by the cross product of two points and can get a point by the cross product of two lines. Based on this, we can calculate the expression of the four edges: l_{12}, l_{23}, l_{34}, and l_{14} by the cross product of related vertexes:

$$l_{ij} = P_i \times P_j.$$

Similarly, we can find the two diagonals, diagonal intersection point M, and two vanishing points V1 and V2 by the cross product of the corner points:

$$M = (P_1 \times P_3) \times (P_2 \times P_4)$$

$$V_1 = (P_1 \times P_2) \times (P_3 \times P_4)$$

$$V_2 = (P_1 \times P_4) \times (P_2 \times P_3).$$

Also, the four projected middle points of the four edges (P_{12}, P_{23}, P_{34}, P_{14}) can be calculated by the same method. Then, using the trigonometric approach described in [14] and [15], we can estimate the angles between the camera and the x, y-axis of the original object (ω, Φ shown in Figure 11):

$$\alpha = \arccos\left(\frac{P12 * M}{|P12||M|}\right)$$

$$\beta = arccos\left(\frac{P34*M}{|P34||M|}\right)$$

$$\gamma = arccos\left(\frac{P14*M}{|P14||M|}\right)$$

$$\delta = arccos\left(\frac{P23*M}{|P23||M|}\right)$$

$$\omega = arctan\left(\frac{2}{cot(\beta)-cot(\alpha)}\right)$$

$$\Phi = arctan\left(\frac{2}{cot(\delta)-cot(\gamma)}\right).$$

After we get ω and Φ, we can calculate the pitch and roll by subtracting them from 90 degrees, respectively. To prove the reliability of the method in camera angle calculation of center-located quadrilaterals, we present two pictures in Figure 12 as an example:

Figure 12 (a) (bottom left): Roll = 45.5822, Pitch = 0.7849

Figure 12 (b) (bottom right): Roll = 3.0733, Pitch = 63.4673

This proves that the overall result is reasonable by checking the distortion of the pictures.

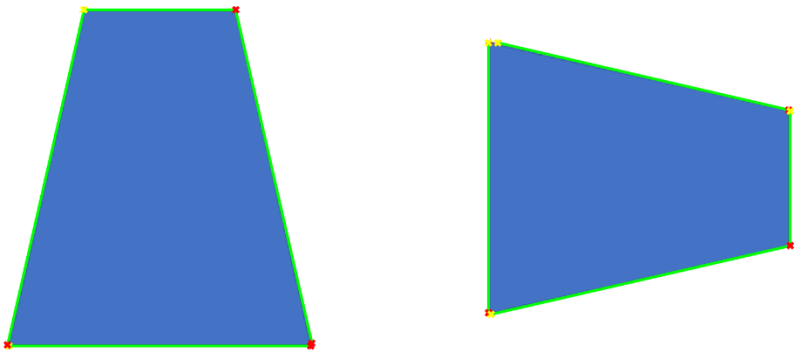


Figure 12. A horizontal (a) and a vertical trapezoid (b) example used for angle calculation

2.2.2.4 Specific Task 3 - Camera Angle Calculation of a Marginal Quadrilateral

From the method in Section 2.2.2.3, we know that the pitch and roll can be calculated by the geometrical method when the quadrilateral is located close to the center of the image. However, we cannot directly calculate the pitch and roll when the quadrilateral is away from the center because in this case, the angle between the camera and the quadrilateral center is not the same as the angle between the camera and the image plane. The following algorithm will deal with this case and render a reasonable result. The basic idea is to extend the quadrilateral to a larger one that is center-located and shares the same plane as the original one.

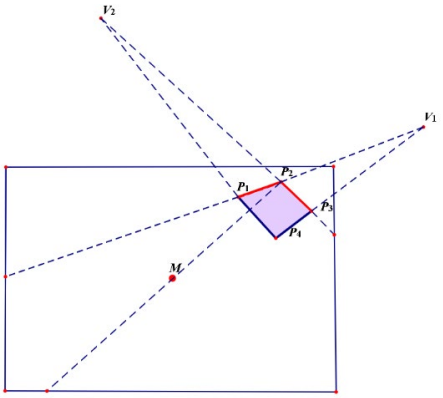


Figure 13. The rectangle boundary represents the image plane while the purple quadrilateral is the projected LED light

Step1: In Figure 13, a quadrilateral is near the upper-right corner of the image. To expand this quadrilateral in direction to the center of the image, we need to find the "corner vertex", that is the vertex having the smallest distance to the image corners. In this case, the vertex P_2 is closest to the upper-right image corner and has thus been selected to be the vertex of the expanded quadrilateral. Then, we also select the neighbor edges of this "corner vertex", which are the two red edges shown in Figure 13. We name line P_2P_3 as edge1 and line P_1P_2 as edge2.

Step2: we define the center of the camera image as M, and then calculate the vanishing points V_1 and V_2 by the method in section 2.2.2.3 (shown in Figure 11). If there are parallel lines, there will be infinite vanishing points with z-value equals to 0.

Step3: since the expanded quadrilateral will have its diagonal intersection at M, we know that it must have a vertex on the line that passes through P_2 and M. We call that vertex a "symetrical_point", shown in Figure 14. Then we connect the "symetrical_point" with V_1, V_2 to create the other two edges: edge4 and edge3 shown in Figure 14.

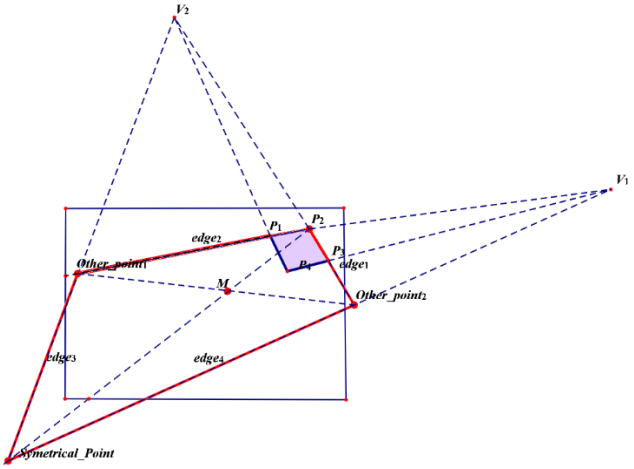


Figure 14. An expanded quadrilateral that will be used for angle calculation, represented in red

Step4: the other two vertexes of the expanded quadrilateral can be gained by intersecting edge2 with edge3 and edge1 with edge4. We name that two vertexes “other_point1” and “other_point2”. Now we have all the four vertexes of the quadrilateral, and we know that point M is the diagonal intersection of the quadrilateral, so the “other_point1”, “other_point2” and point M must be collinear.

Step5: based on these facts, we can start the calculation of the expanded quadrilateral’s location. Since the coordinates of the symetrical_point is unknown, we set it to be X_1= [a, b, f/dx] with two variables a and b. Then edge3, edge4, other_point1 and other_point2 can be calculated by

$$edge3 = X1 \times V2$$

$$edge4 = X1 \times V1$$

$$other\_point1 = edge2 \times edge3$$

$$other\_point2 = edge1 \times edge4$$

There are two requirements to be satisfied, one is that “corner_point”, “symetrical_point” and point M must be collinear, the other is that “other_point1”, “other_point2” and point M must be collinear. When three points are collinear, the dot product of one point with the cross product of the other points must be 0. Accordingly, we can write down two equations for these two requirements (shown below), and since there are two variables, we know that they can be solved.

$$Equation1: X_1 * (corner\_point \times M) = 0$$

$$Equation2: M * (other\_point1 \times other\_point2) = 0$$

Step6: In order to solve X_1 (which is equal to [a, b, fu]), we define M_1 as the cross product of corner_point and M, and then convert the above equations to one matrix multiplication equation f:

$$\mathbf{v2 = \{v20, v21, v22\};}$$
$$\mathbf{v1 = \{v10, v11, v12\};}$$
$$\mathbf{edge1 = \{e10, e11, e12\};}$$
$$\mathbf{edge2 = \{e20, e21, e22\};}$$
$$\mathbf{m0 = \{m00, m01, m02\};}$$
$$\mathbf{m1 = \{m10, m11, m12\};}$$
$$\mathbf{x1} = \left\{a, -\frac{a\,\mathbf{m10} + \mathbf{fu\,m12}}{\mathbf{m11}}, \mathbf{fu}\right\};$$
$$f = ((\mathbf{v2} \times \mathbf{x1}) \times \mathbf{e2}) \times ((\mathbf{v1} \times \mathbf{x1}) \times \mathbf{e1}).\mathbf{m0};$$

By simplifying f, which is a series of matrix multiplication, we get a quadratic equation with one unknown variable a:

$$Aa^2 + Ba + C = 0$$

The details of the expression of A, B and C are shown in Appendix D, in forms of multiplications of a number of known values. After we have all the three coefficients, we can calculate the value of a simply by

$$a = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A}.$$

By solving the above equations we can get the coordinates of the symetrical_point M and then the coordinates of the two "other_point".

Step7: Up to now we have gained all four vertexes of the expanded quadrilateral and the four edges (shown in red color in the above picture). Since the quadrilateral has its diagonal intersection at M, the center of the image, we can perform the same geometric approach described in section2 to calculate the pitch and roll of the camera. Since the expanded quadrilateral and the original quadrilateral have the same vanish points, they share the same plane, and thus the resultant pitch and roll will also be angles to the original quadrilateral. Then the angles of the camera are obtained.

2.2.2.5 Specific Task 4 - Camera Positioning

Based on the roll and pitch angle calculated in previous steps, the camera's location can now be solved. Since it is assumed that the center of the LED projection is the original point (0,0,0) in world coordinates, we only need to find the location of the LED projection's center in the camera's coordinates and take opposite numbers to get the camera's location. In order to do that, the following steps are necessary:

Step1: We need to find the LED projection's center in pixel coordinates. This can be achieved by the cross-product method mentioned in previous sections: $M_P = P1 \times P2 \times P3 \times P4$.

Step2: In order to transform the LED projection's pixel coordinates into the camera coordinates, we apply the pinhole model and have the equation $M_W = M_P \times d \,/\, |M_P|$. Here, d means the distance between the camera and the LED projection. Since the distance is supposed to be much larger than the side length of the projection, we assume that the distances from the camera to each point inside the projection are equal. The distance can be calculated by the following equation:

$$d = \frac{object\_width \times fu}{pixel\_width}.$$

The *object_width* is the side length of LED projection in the real world, which is 20 cm in our experiment. The factor *fu* is the focal length in pixels and the *pixel_width* is the LED projection's size in the image, it is calculated by taking the average of the four sides' lengths in pixels. After obtaining the distance, we can calculate the LED projection's world coordinates M_W.

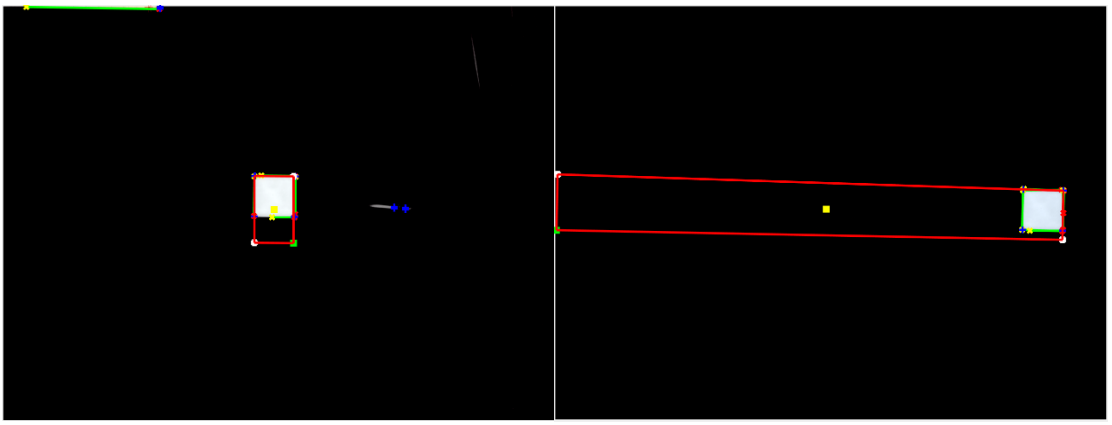
Step3: since the LED projection's world coordinates M_W is the location after rotation, we can calculate the original LED location M_W' by multiplying M_W is rotation matrices of pitch and roll. The formula is shown below. Then we take the reverse values and get the camera location.

$$M'_W = M_W \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(roll) & -\sin(roll) \\ 0 & \sin(roll) & \cos(roll) \end{bmatrix} * \begin{bmatrix} \cos(pitch) & 0 & \sin(pitch) \\ 0 & 1 & 0 \\ -\sin(pitch) & 0 & \cos(pitch) \end{bmatrix}$$

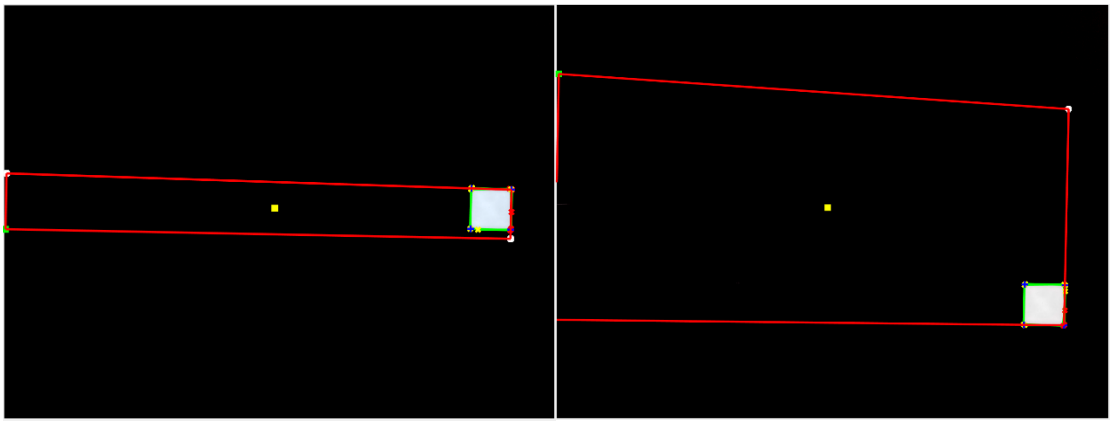
2.2.2.6 Specific Task 5 Accuray testing for angles and location calculation

To test the accuracy of the algorithm, we perform our approach on six pictures of projected LED light whose original edge length is 20 cm. Figure 15 shows examples of the original quadrilaterals (green contour) and their expanded quadrilaterals (red contour). The yellow dot in the middle is point M with coordinates [U_0, V_0, f/dx]. From those examples we can see that the original quadrilaterals are properly expanded into quadrilaterals that have a diagonal intersection at point M. In this case, we have got the plane in which the object quadrilateral stays and we only need to calculate the angles (pitch and roll) of the camera with respect to this plane and the result will be the same for that quadrilateral.

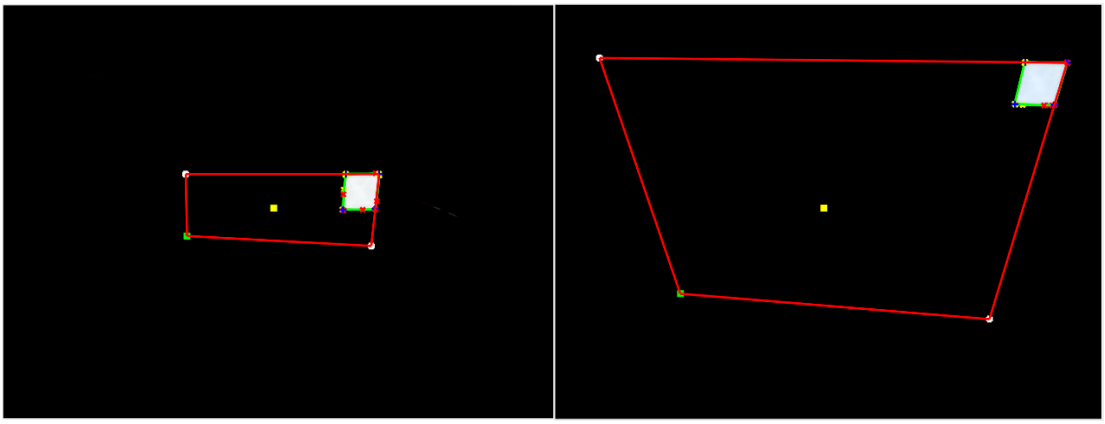
The evaluation results are shown in Table 2 and 3, which compare the generated and given angles and locations. The error rates are also calculated and presented.



(a) (b)



(c) (d)



(e) (f)

Figure 15. (a)-(f). Six examples of LED projections with expanded quadrilaterals generated by the Matlab program

The comparison of the Matlab-generated roll and pitch and the measured roll and pitch from the experiments is shown in Table 2, the unit is in degree. When the original square is close to the middle of the image (e.g. Image 1), the error is very small. However, when the square is moved to the edge of the image (e.g. Image 2,3), there will be a deviation around 5 –6 degrees. One possible reason for this kind of error is the lens edge distortion. In order to get rid of the lens edge distortion, we need one extra matrix that will compensate for this effect.

Image Number	Generated Roll	Measured Roll	Generated Pitch	Measured Pitch
1	0	0	0	0
2	0	0	6	0
3	0	0	6	0
4	35.8	30	6.9	0
5	0	0	32.2	30
6	15.8	15	18.5	15

Table 2. Generated roll, pitch and the measured roll, pitch

Then we test the accuracy of the camera location calculation mentioned in Section 2.4. In Table2, we list the results and the error rates of calculation. The error rate is defined below. Figure 16 shows a Matlab-generated position transformation of Image5 in which the blue vector represents the camera location before multiplying the rotation matrices and the red vector represents the camera location after multiplying the rotation matrices.

Error Rate = $$\frac{\sqrt{\Delta X_W{}^2 + \Delta Y_W{}^2 + \Delta Z_W{}^2}}{\sqrt{X_W{}^2 + Y_W{}^2 + Z_W{}^2}} * 100$$

Image Number	Generated Position (cm)	Measured Position (cm)	Error Rate (%)
1	[6.5 0.2 206.9]	[0 0 200]	4.74
2	[0.8 114.6 173.1]	[0 100 200]	13.69
3	[42.5 112.1 170.0]	[48 100 200]	14.34
4	[58.5 107.9 154.5]	[48 100 200]	20.71
5	[48.9 118.3 206.7]	[48 100 200]	8.53
6	[56.2 121.5 204.5]	[48 100 200]	10.25

Table 3. Generated position and the measured position of the camera, and error rates

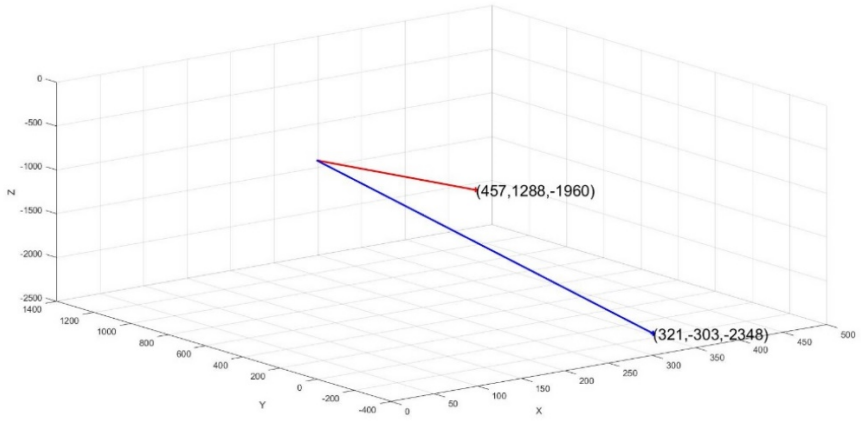


Figure 16. The camera location before (red) and after (blue) multiplying the rotation matrices

2.2.2.7 Objective Statement Evaluation

Generally, the proposed algorithm can correctly calculate the pitch and roll of the camera with respect to the given picture. It can work well no matter where the original quadrilateral is. This approach provides the foundation of the next step in which we will calculate the global position of the camera.

There are some limitations to this approach. Firstly, the lens edge distortion will cause some error to the measured angles. Secondly, this approach works well when the original quadrilateral is placed in almost horizontal or vertical pose. If that is not the case, there will be some difficulties detecting the four vertexes of the quadrilateral. However, those cases are

rare and will not affect the overall result. Thus in conclusion, the objective statement has been completed.

2.2.3 Objective Statement 3 - to expand the angle and location calculation to the case of the rolling shutter pattern image

2.2.3.1 Objective Statement System Block Diagram

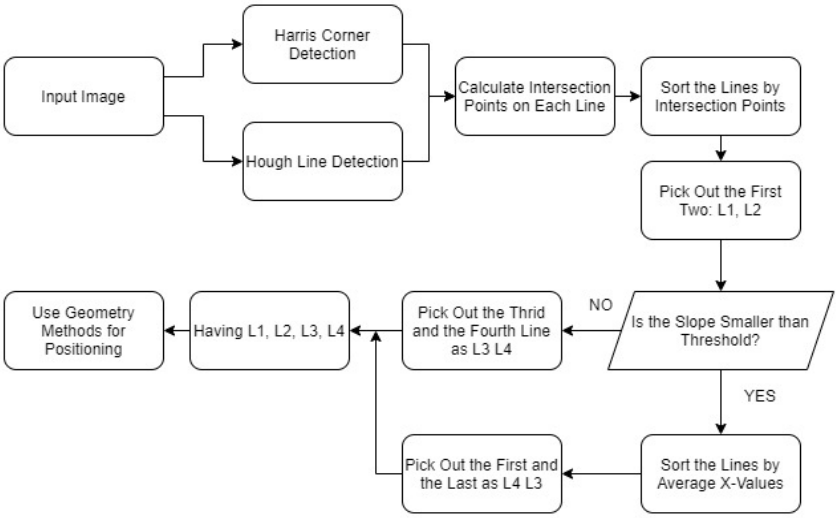


Figure 17. The flow chart of the proposed algorithm

Figure 17 presents the flowchart of the VLP algorithm for the rolling shutter pattern images. Based on the methods listed in Section 2.2.2, we can now find the camera angles and positions with a single input image of a quadrilateral. However, in visible light communication, the input image is usually in a rolling shutter pattern (like a barcode), and we need extra operations to locate the object and to achieve the same goals as before. Here, I propose a geometry-based method that first finds the upper and lower two edges of the rolling shutter pattern and then locates the four corners of the shape. Once we have the four corners, we can do the same geometry calculation to find the camera angles and location.

I use this geometry method because several other possible solutions (to be discussed in Section 2.2.3.2) were proved inefficient, and this algorithm is accurate and not too complicated. There are some limitations and considerations that will be mentioned below. Afterwards, we will test the error rates of the angles and location calculation and show the results.

2.2.3.2 Specific Task 1 – Using Gaussian blurring and interception method

First, I used Gaussian blurring to get rid of the stripe pattern, it can be seen in Figure 18 that though the stripes no longer bother, both the line detection (in green color) and the corner detection (in red color) perform badly and predicts a completely false shape. In this case, the algorithm cannot find the vertices accurately, so the next part of camera positioning will not work either.

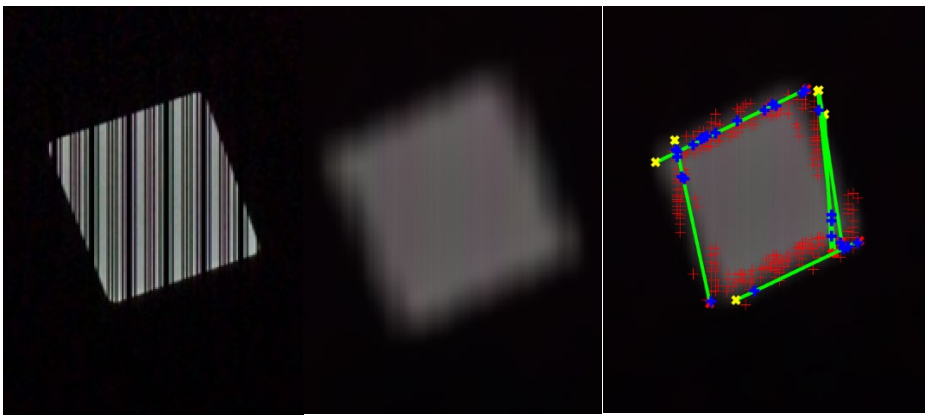


Figure 18. Testing Result of gaussian blurring

I also studied the method proposed in [16], and it seems not to work well in this situation, either. The method measures the interceptions of lines with the polygon and takes the limit to find the four vertexes. I started testing the program with the six images shown in Figure 19 that we used before (they are much simpler than the rolling shutter pattern).

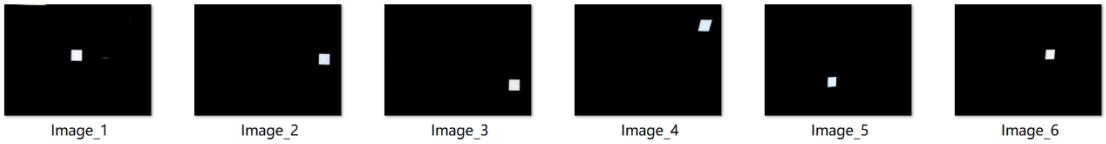


Figure 19. Six simple images for testing

However, five of them give no output and I think something is wrong with the line: [~,binmax] = maxk(H(:),k); since there isn't a 'maxk' function in my Matlab version, so I wrote a 'maxk' function myself to pick out the top k value in a list. But still, the program did not give any output. When I changed this line to binmax = max(H(:), k); the Image_4 gives an output shown in the left part in Figure 20, and others still not responding. After I applied Gaussian blurring, Image_4 gives an output shown in the right part in Figure 20, and no others did. Those there was one corner, the algorithm is far from being useful in our case.

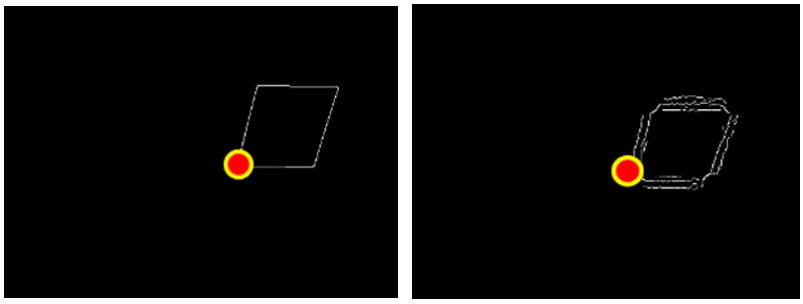


Figure 20. Testing Results without rolling shutter pattern

Then I tried it in the rolling shutter pattern. If I don't use Gaussian Blurring, none of the images works. If I use it, Image_3 will give an output shown similar to Figure 20 and no others responded. The accuracy is quite low and obviously, this method cannot be applied.

In conclusion, I think this method cannot work in our case due to its mechanism and any method aiming to find the vertexes directly may not work well for rolling shutter pattern, either. The reasons are as follows: first, if I did not use Gaussian blurring, there are cases that the pattern doesn't have four clear vertexes due to the dark strip near the edge. Any vertex detection method that cannot identify this corner will have a large error in its final output.

2.2.3.3 Specific Task 2 - Proposed Geometrical Solution

In view of the problems mentioned in Section 2.2.3.2, I decided that the new method should not use Gaussian blurring and should not find the four vertexes directly. So, I proposed a new algorithm that chooses to find the four edges instead. An example of the algorithm output is shown in Figure 21. First, I assume that among the four edges, there are two that are more horizontal (has a smaller slope) and two that are more vertical (larger slope). In Figure 21, the red lines mark the two with a smaller slope (L1, L2) and the green lines mark the two with a larger slope (L3, L4).

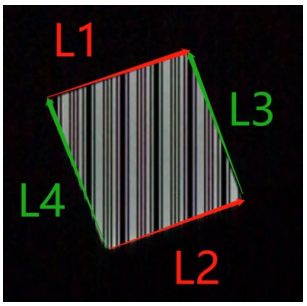


Figure 21. Rolling shutter pattern's four edges marked by the slope

We can see that L1 and L2 have more intersection points with the stripe than L3 and L4 do. This is the motivation for how I find the edges. I first use Harris corner detection to find all intersection points in the picture (marked in red on the left part of Figure 22).

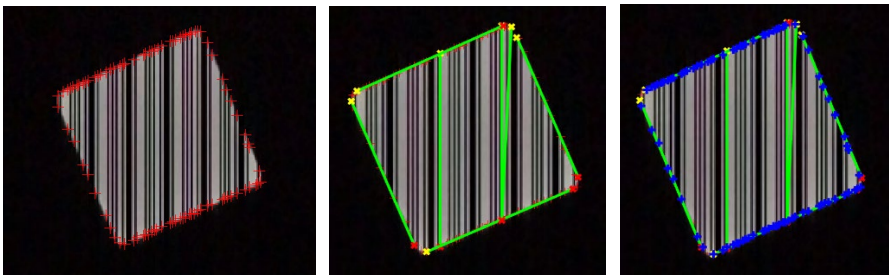


Figure 22. Rolling shutter pattern's four edges with points on them

Then I use Hough line detection to find all possible edges with a length of more than 110 (marked in green on the middle part of Figure 22). We can see that some stripes are regarded as edges. But according to the theory, the two more horizontal edges have the most interception points (L1, L2), the two more vertical edges come after (L3, L4), and the stripes should only have around the two interception points.

Therefore, we count how many interceptions point each edge has. This is simple. I just calculate the distance from each point to the edge and compare it to the length of the edge. When they are close enough, I mark them with a blue mark on the right part of Figure 22 and accumulate them. Then, I just need to count how many blue points are there on each green line.

If I sort the edges by their blue points, then the top four should be the edge of the rolling shutter pattern (the top two should be the more horizontal edges and the top 3 or 4 should be the more vertical edges). I mark them in order: the topmost is called L1, the right-most is called L3, the bottom-most is called L2 and the left-most is called L4, just like in Figure 21. Then I use homogeneous coordinates to calculate the cross product of these four lines and get the four vertexes. The remaining part is to calculate the camera position similar to what I did before.

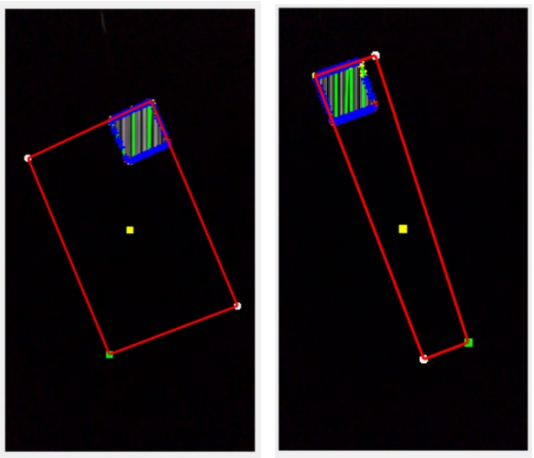


Figure 23. Testing result of slant quadrilateral

The result of the algorithm is shown in Figure 23. The vertexes can be accurately calculated by this method if the quadrilateral is placed in a slant, and also the camera position should have little error.

However, there is one particular case that this method may cause some errors. If the quadrilateral is placed with a small angle, like the one in Figure 24, then only two edges will have a number of intersection points (L1, L2), then it is hard to find the other two edges (L3, L4) since all the blue points are on L1 and L2.

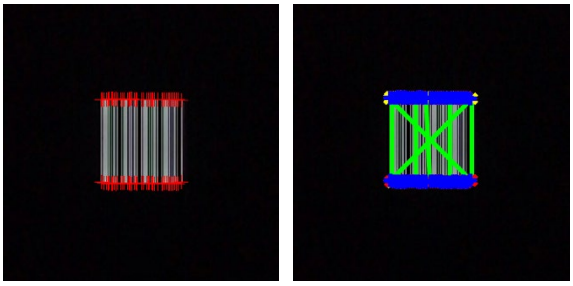


Figure 24. The special case when the quadrilateral is placed without angle

In order to solve this problem, I use the following approach. First, the algorithm will pick out L1 and L2 in the same way as before (sorting the lines according to the number of blue points). Then I measure the slope of these two lines. If they are too small, e.g., smaller than tangent (2.5), then it is the special case. I will then sort the rest of the lines according to their average x-value. The right-most line with the largest average x-value must be L3 and the left-most must be L4. With L3 and L4, we can perform the geometric method just like before. The testing result is shown in Figure 25, with a trustable outcome.

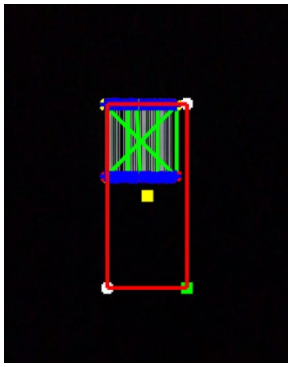


Figure 25. Testing result of the special case

2.2.3.4 Specific Task3 – Result testing

To estimate the accuracy of the proposed algorithm, I conduct tests with 35 images of a rolling shutter pattern. I used those images as the input of the program and collected the roll, pitch, distance and location (x and y value) from the output and compare them with the reference values to get the error range. An overview of the results is shown in Figure 26 and the CDF is shown in Figure 27. And the average errors are:

Error of Pitch: 7.0 degrees *Error of Roll: 4.8 degrees* *Error of X-value: 18.1 cm*

Error of Y-value: 23.0 cm *Error of Distance: 32.4 cm* *Error of 3D location: 43cm*

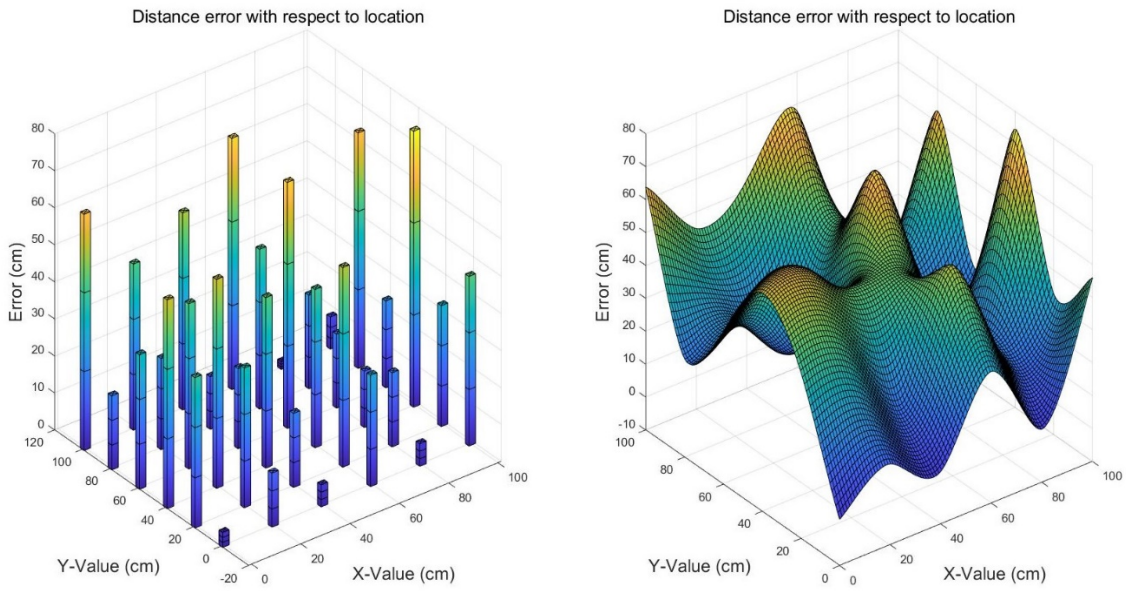
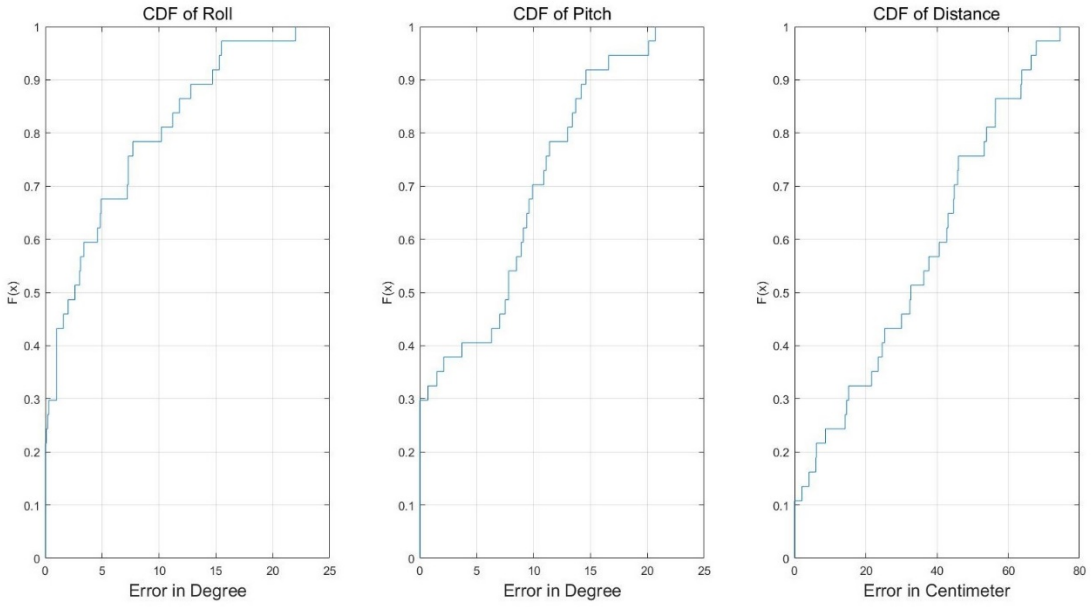


Figure 26. 3D plot of the testing results with error represented by height of each coordinate



(a) (b) (c)

Figure 27. Summary of CDF of the testing results of roll (a), pitch (b) and distance (c)

2.2.3.5 Objective Statement Evaluation

By analyzing the result, I found that several factors contribute to the large errors and the unstable performance of the algorithm. Those factors are discussed and demonstrated in Section 2.2.3.5.1 – 2.2.3.5.3.

2.2.3.5.1 Impact of Stripe Density

When the stripe density is low, there will be cases that a thick dark line appears at the edge and cut it shorter. The line detection will have some error, and from the red circle in Figure 28 we can see that the green line is not actually the edge of the pattern. The real edge is cut so short that the algorithm cannot detect it. Thus, the resultant pitch and roll will differ from the given value, and so will the location of the camera.

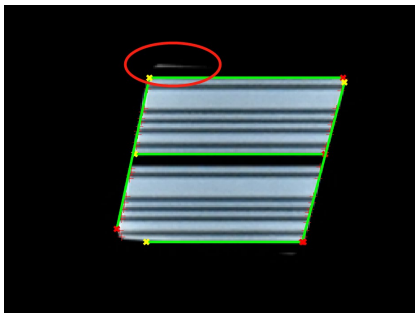


Figure 28. Impact of stripe density

2.2.3.5.2 Impact of Double Edges

Sometimes, when the edges of the pattern are not sharp enough, even though all four edges are recognized, there are cases that more than one line is detected for each edge, as shown in Figure 29. In this case, the algorithm cannot sort them according to the number of intersection points and there will be no result generated. This case is mainly due to resolutions.

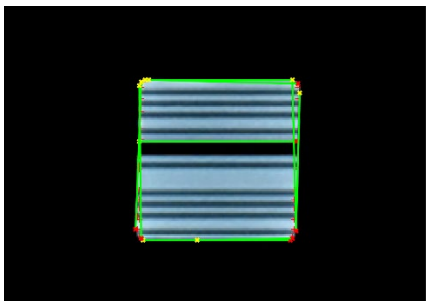


Figure 29. Impact of double edges

2.2.4.5.3 Impact of Vertical and Horizontal Image

In the test, I convert each image of both vertical and horizontal, and the results given from the algorithm are different even though all the parameters remain the same. This case is particularly obvious when the projection is not slanted (such as in Figure 29). For example, in Figure 30, I rotate the image by 90 degrees and perform the same algorithm using identical parameters. The resultant expanded shapes are not the same and the pitch-roll for these two are (25.2, 32.5) and (55.2, 33.8).

This phenomenon occurs in almost every image and I think the reason comes from the working principle of Hough line detection. Even if the input parameter doesn't change, the algorithm's sensitivity towards vertical and horizontal lines is not the same. This is the thing that needs to be aware of when applying this algorithm. But the testing result displayed high performance and accuracy when dealing with the rolling shutter pattern image and this corresponded to the objective statement. Therefore, in conclusion, the objective statement has been completed.

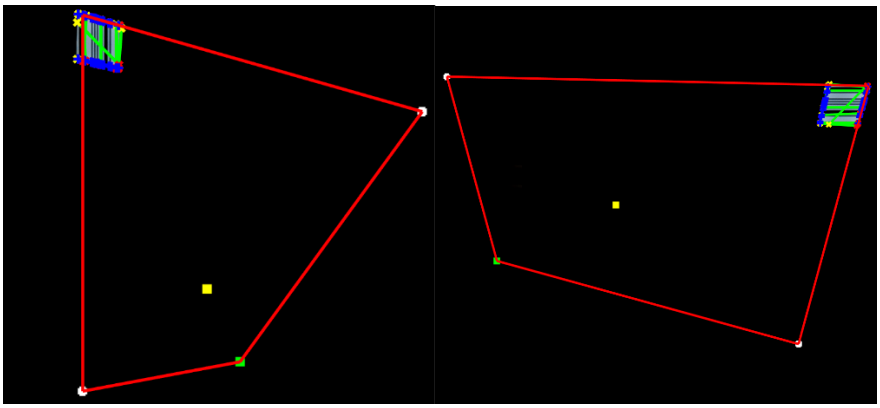


Figure 30. Impact of vertical and horizontal image

2.2.4 Objective Statement 4 - to develop a method that performs similar indoor-positioning processes based on images that contain other forms of LED

2.2.4.1 Objective Statement System Block Diagram

The flow chart of objective statement 4 is shown in Figure 31. When dealing with LEDs with shapes other than quadrilateral (e.g., tube), the following method can be used to find the camera position. One thing that is different from the previous geometric method is that, in this case, the camera angles (roll, pitch and azimuth) are given by the sensor data and there is no need to calculate them in the program. Then the goal of this method is to find the transition matrix and thus the location of the camera.

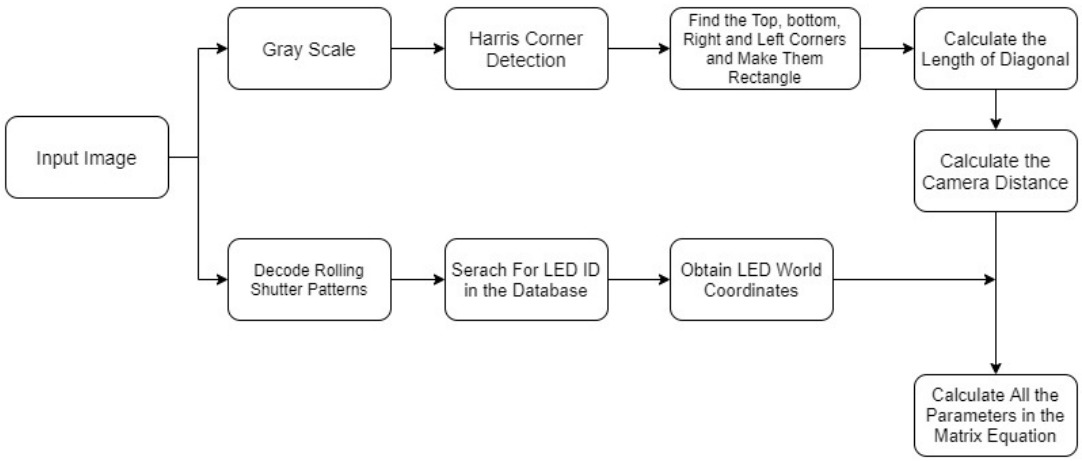


Figure 31. The flowchart of objective statement 4

In this method, we first convert the image to grayscale and use the Canny operator to find the edges of the LED. As the LED is in the shape of a tube (or narrow eclipse), we can simply regard it as a line and its length is the diagonal of the rectangle that can just encircle it. After this step, we can estimate the distance and calculate the camera location. In the previous stages I worked out a method that mainly deals with square-shaped LEDs, and other FYP students are working on circular LEDs. So, in this stage, I will mainly focus on tube LEDs and rectangular LEDs. Figure 32 shows LEDs in different shapes.

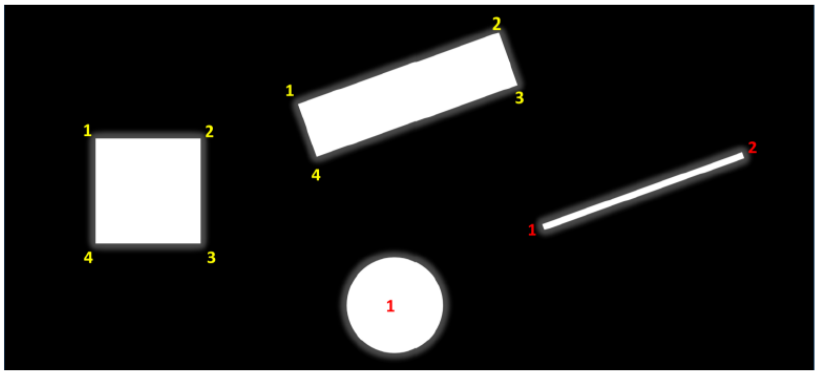


Figure 32. An example of the different forms of LED

2.2.4.2 Specific Task 1 – Find the rectangle the encloses the LED

To find the rectangle that encloses the LEDs, we use the Canny operator to pick out the points on the edge that are at the top, bottom, right and left. We can use their coordinates that form a rectangle and then get the length of their diagonal. The rectangle is the one that encloses the LED, and its diagonal length is the length of the LED.

2.2.4.3 Specific Task 2 – Calculation of the camera location

After we have the length of the diagonal (the length of the object) L_d, the original length of the LED L_0 and the camera focal length f, we can calculate the distance between the camera and the LED d based on the model mentioned in [17]:

$$d = L0 \times \frac{f}{Ld}.$$

Then we pick the center of the rectangle M_o and multiply it with the rotation matrix inside which the angles are reversed (similar to step 3 in Section 2.2.2.5). In this way, we get the original location of the LED center in the image plane M_o'. Then, in the camera coordinates, the LED center M_c can be expressed as

$$Mc = d \times \frac{Mo'}{|Mo'|}$$

From M_c, we can have the translation matrix since we know the related location between the camera and the LED. If we have the LED's center location in world coordinates M_w by checking

its light ID, then the camera location in the world coordinates C_w can be calculated by the following matrix equation:

$$Cw = Mw - Mc.$$

2.2.4.4 Objective Statement Evaluation

As the Matlab program developed in the project mainly focuses on the input images with quadrilateral objects, this part will render a theoretical solution for the objects with other shapes. The technical challenges may appear in the step of corner detection as the angle of the corners may be large and be regarded as curves. But the proposed method can handle this issue by adjusting the threshold of the Harris corner detection. Thus, the objective statement has been completed.

2.2.5 Objective Statement 5: to convert the algorithm from a Matlab program to a C++ program based on OpenCV library and specific VLC recognition algorithm and compare the accuracy between the two

2.2.5.1 Objective Statement System Block Diagram

Figure 33 shows the flowchart for objective statement 5. In this step, I will convert the Matlab program to C++ so that it can be implemented in various platforms or operating systems. Most of the Matlab code can be directly converted by replacing the Matlab data types with the corresponding OpenCV data types. However, there are Matlab functions that cannot be transformed into C++ codes. Therefore, some approaches were tried to complete the conversion. The two major ways will be explained in detail in the following.

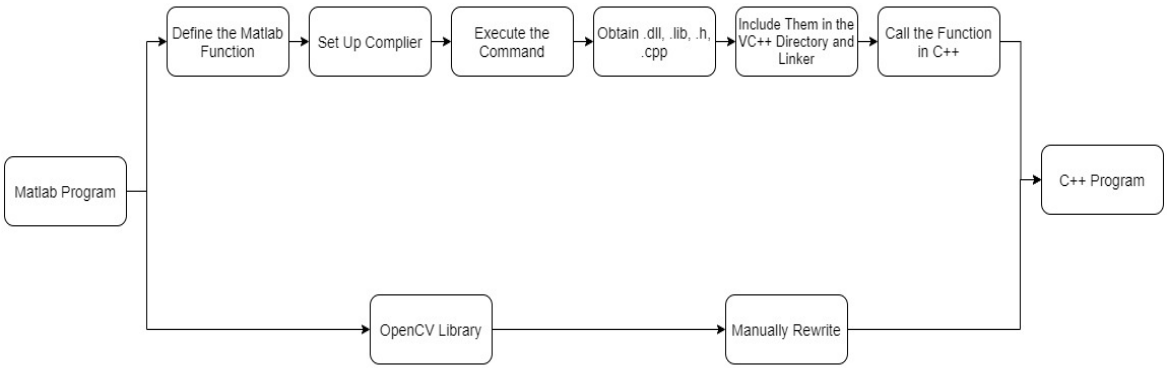


Figure 33. The flowchart for objective statement 5

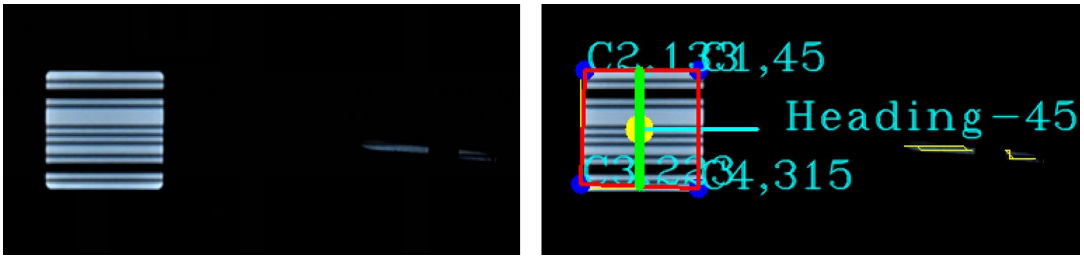
2.2.5.2 Specific Task 1 – Using Matlab coder, VS compiler, and Matlab deployment tool

We use the Matlab coder, VS compiler, and Matlab deployment tool for the conversion. These methods are similar and mainly convert a Matlab built-in function to an executable C++ project or code. They use different compilers to achieve the goal. The resultant files

include .h, .dll, .cpp and .lib files. However, the solve function in Matlab is not defined in the .m file and thus cannot be translated into C++. In order to simplify the task, the VLC recognition algorithm provided by our group is used for convenience.

2.2.5.3 Specific Task 2 – Using the VLC recognition algorithm provided by our group and manually rewrite the program

The VLC recognition algorithm provided by our group will automatically recognize the image object with VLC light ID and output the coordinates of its corners. An example input and output are shown in Figure 34. By applying the VLC recognition algorithm , my C++ program will only have to continue for the angle calculation and the parts like corner detection and selection can be omitted. However, the accuracy of the VLC recognition algorithm 's corner detection will be a disturbing factor, and in the next step, I will conduct tests and compare its results with those of the Matlab program.



(a) (b)

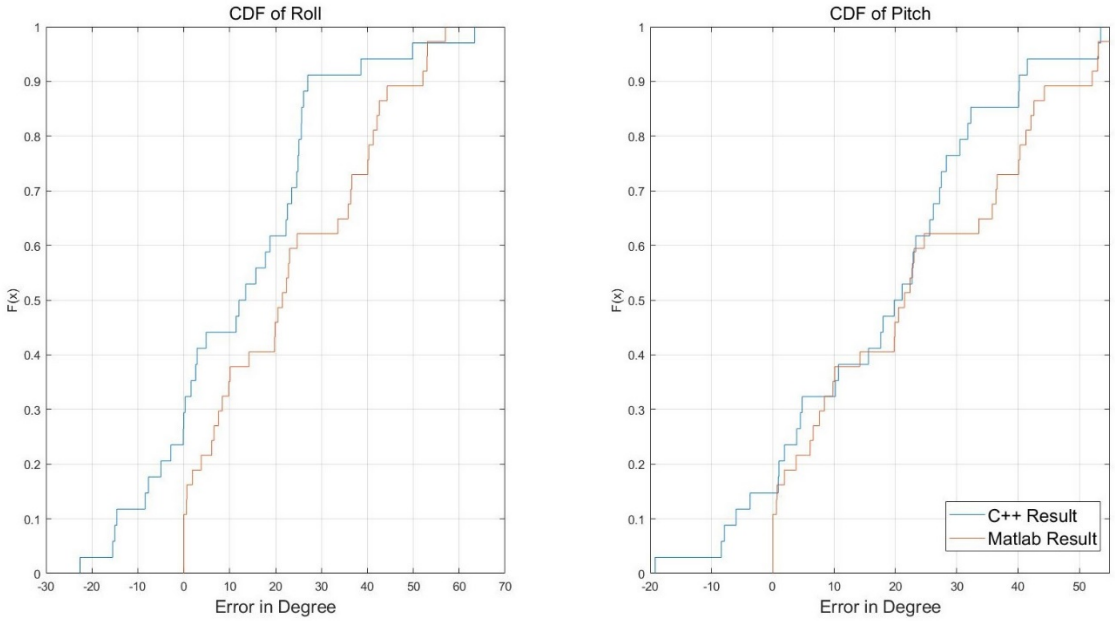
Figure 34. An example input (a) and output (b) image of the VLC recognition algorithm provided by our group

2.2.5.4 Specific Task 3 – Accuracy testing for angle calculation

To estimate the accuracy of the converted C++ program with VLC recognition algorithm in comparison with the Matlab program, I conduct a test with 35 images of rolling shutter pattern using the same method stated in Section 2.2.3.4. An overview of the results is shown in Figure 35. And the average errors are

Error of Pitch: 12.0 degrees *Error of Roll: 17.9 degrees*

The corner-calculation errors are around 2.4 times larger than that of the Matlab program, and the CDF plot in comparison is shown in Figure 35.



(a) (b)

Figure 35. The CDF comparison of the roll (a) and pitch (b) calculation between the C++ and the Matlab program

2.2.5.5 Objective Statement Evaluation

From Figure 35 we can see that the performance of the Matlab program is much better than the C++ one, with many fewer errors. The main technical challenge comes from the relatively low accuracy of the VLC recognition algorithm ’s corner detection function, which influences the steps of angle calculation. But in general, the converted C++ can work properly. Thus, the objective statement has been completed.

2.3. Main Objective Evaluation & Discussion

As stated in Section 1.4.1 the main objective of our project is to develop an algorithm on visible light positioning (VLP) algorithm for the smart projector system to estimate the indoor location of the user based on a single luminary. This receiver algorithm will be applied in the implementation of the smart display and projector, and to address the gap between demand and supply of feasible technical support. By filling such a gap, the smart projector system based on our VLP algorithm is believed to make a significant contribution to the enhancement of the present situation of the indoor system.

The main results from Section 2.2 demonstrate that the proposed VLP algorithm is able to provide indoor positioning with low error rates and simple demands, which successfully fulfils the main objective statement of designing a feasible receiver algorithm for the smart indoor service system.

Compared to the benchmark systems mentioned in the literature review, the VLP algorithm developed in this project has its advantages. In terms of the error range of location detection, the VLP is more precise than the Wi-Fi-based, and Bluetooth-based systems whose error can be as large as 2 meters. Moreover, as this algorithm requires only one luminary of positioning, the complexity is much lower than the traditional VLC based indoor positioning. It can be concluded that the proposed VLP algorithm can cover the shortages of the above systems and the major problems are solved.

In sum, the project outcome is satisfactory, and its importance comes from the potential applications in multiple circumstances. Since the VLC projection can be decoded by a headset or other device, the product may lay the foundation for the advancement of indoor entertainment such as VR or even the extensive adoption of smart buildings. In addition, this work could have been done more efficiently if the program had been directly developed in C++ and tested in the robot operating system (ROS) as it would have saved more time for algorithm development and made it easier to test the performance.

Section 3— Conclusion

In order to provide the foundation for the smart display and projector system, a receiver algorithm based on visible light positioning (VLP) for the smart projector system is developed to estimate the camera angle, distance and the indoor location of the user based on a single luminary. In this project, we first applied shape-detecting algorithms to locate the boundary of the object and then used self-developed mathematical methods to calculate camera angle and distance with respect to the image. Afterwards the user's location was estimated based on the previously obtained data. The VLP algorithm was also extended to shapes other than quadrilateral and the programming language C++. According to the test results, the VLP algorithm successfully achieves high-accuracy indoor positioning based on a single image and all the objective statements are completed.

In the future, this work can be further applied to other platforms or operating systems to serve as a foundation of light-based indoor positioning. For example, a robot arm may need real-time data about its angles and coordinates in all three dimensions and autonomous vehicles may require VLP for indoor navigation. To sum up, the VLP algorithm rendered in this project has a promising functionality in terms of accuracy and will have a wide range of potential usages in the area of smart indoor services.

REFERENCE

[1] P. Pivato, L. Palopoli, and D. Petri, “Accuracy of RSS-based centroid localization algorithms in an indoor environment,” *IEEE Trans. Instrum. Meas*, vol. 60, no. 10, pp. 3451–3460, Oct. 2011.

[2] B. Hussain, X. Li, C. Y. Lee, and C. P. Yue, "Smart LCD Displays with Modulated LED Backlights for Li-Fi Enabled Applications," in 2018 Joint Symposia on Optics, (Optical Society of America, 2018), paper 31aAJ4.

[3] L. Li, P. Hu, C. Peng, and F. Shen, “Epsilon: A visible light-based positioning system,” in the 11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14). 331-343, 2014.

[4] Y. Brian Bai, S. Wu, and G. Retscher, Eds., “A new method for improving Wi-Fi-based indoor positioning accuracy,” Journal of Location-Based Services, pp.135-147, Oct. 2014.

[5] A. Rommel, “Using light to transmit data –Visible Light Communication (VLC),” Jun. 2015. [Online], Available: https://www.fraunhofer.de/en/research/fieldsofresearch/communicationknowledge/broadband-communications/visible-light-communication.html [Accessed November 22, 2020].

[6] E. Mendelson and C. Springs, “Local Base Service Application Utilizing RF Bluetooth Beacons,” U.S. Patent 9.204.251 B1 Dec. 1, 2015.

[7] H. Xia, J. Zuo, and S. Liu, Eds., “Indoor Localization on Smartphones Using Built-In Sensors and Map Constraints,” IEEE Transactions on Instrumentation and Measurement, vol. 68, no. 4, Apr. 2019.

[8] A. Jovicic, J. Li and T. Richardson, "Visible light communication: opportunities, challenges and the path to market," *IEEE Communications Magazine*, vol. 51, no. 12, pp. 26-32, December 2013, DOI: 10.1109/MCOM.2013.6685754.

[9] A. Jovicic, J. Li and T. Richardson, "Visible light communication: opportunities, challenges and the path to market," in *IEEE Communications Magazine*, vol. 51, no. 12, pp. 26-32, December 2013, doi: 10.1109/MCOM.2013.6685754.

[10] J. Hao, Y. Yang and J. Luo, "CeilingCast: Energy efficient and location-bound broadcast through LED-camera communication," IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications, San Francisco, CA, 2016, pp. 1-9, doi: 10.1109/INFOCOM.2016.7524511.

[11] J. Hao, J. Chen and R. Wang, "Visible Light Positioning Using A Single LED Luminaire," in IEEE Photonics Journal, vol. 11, no. 5, pp. 1-13, Oct. 2019, Art no. 7905113, doi: 10.1109/JPHOT.2019.2930209.

[12] Hemalata Bhujle. 2014. An Improved Harris Corner Detector. In Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing (ICVGIP '14). Association for Computing Machinery, New York, NY, USA, Article 75, 1–8. DOI: https://doi.org/10.1145/2683483.2683558

[13] J. Illingworth, J. Kittler.1988. A survey of the hough transform. Computer Vision, Graphics, and Image Processing. Volume 44, Issue 1, Pages 87-116, ISSN 0734-189X, DOI: https://doi.org/10.1016/S0734-189X(88)80033-1.

[14] Estrems Amestoy, M.; de Francisco Ortiz, Ó. Global Positioning from a Single Image of a Rectangle in Conical Perspective. *Sensors* 2019, *19*, 5432.

[15] Wefelscheid, Cornelius, T. Wekel and O. Hellwich. “Monocular Rectangle Reconstruction - Based on Direct Linear Transformation.” VISAPP (2011).

[16]https://www.mathworks.com/matlabcentral/fileexchange/74181-find-vertices-in-image-of-convex-polygon

[17] W.-L. Du and X.-L. Tian, “An automatic image registration evaluation model on dense feature points by pinhole camera simulation," in 2017 IEEE International Conference on

Image Processing (ICIP). IEEE, 2017, pp. 2259{2263.

APPENDICES

Appendix A. Final Project Plan

A.1 Project Schedule—Gantt Charts

A.1.1 Include an overall Gantt chart for the entire project

Table 5. Gantt Chart for Project.

Objective Statements*	SEP	OCT	NOV	DEC	JAN	FEB	MAR	APR
locate the four corners of the object								
calculate the angles and location								
calculate the rolling shutter pattern								
expand to other forms of LEDs								
convert from matlab to C++								

A.1.2 Include an individual Gantt chart for each Objective Statement to show a breakdown of tasks and a detailed schedule of the work needed to be completed to realize the Objective Statement (use weeks for the time scale).

Table 6. Gantt chart for Objective Statement 1

Specific Tasks*	WK1	WK2	WK3	WK4	WK5	WK6	WK7	WK8	WK9	WK10	WK11	WK12
Harris Corner Detection												
Hough Line Detection												
Perspective Transform												

Table 7. Gantt chart for Objective Statement 2

Specific Tasks	WK9	WK10	WK11	WK12	WK13	WK14	WK15	WK16	WK17	WK18

Coordinate of the Projected Object									
Calculation of a Center-Located Quadrilateral									
Calculation of a Marginal Quadrilateral									
Camera Positioning									

Table 8. Gantt chart for Objective Statement 3

Specific Tasks*	WK17	WK18	WK19	WK20	WK21	WK22	WK23	WK24	WK25	WK26
Gaussian Blurring and Interception Method										
Proposed Geometrical Solution										

Table 9. Gantt chart for Objective Statement 4

Specific Tasks*	WK25	WK26	WK27	WK28	WK29	WK30	WK31	WK32	WK33	WK34
Find the rectangle that encircle the LED										

Calculation of the camera location								

Table 10. Gantt chart for Objective Statement 5

Specific Tasks*	WK29	WK30	WK31	WK32	WK33	WK34	WK35	WK36	WK37	WK38
Using Matlab coder and other tools										
Using the VLC recognition algorithm and mannually rewrite										
Accuracy testing										

A.2 Table of Responsibilities (for groups)

Table 11. Expected contributions of group members

Group Member	Objective Statement 1*	Objective Statement 2*	Objective Statement 3*	Objective Statement 4*
Chen Runzhou	All	All	All	All

A.3 Budget

Table 12. Budget

Items*	Cost
VLC Modulator	$250
Other Components	$550

TOTAL	$900

Appendices B. Meeting Time

Meeting 1

Date: 12/9/2020

Time: 14:30PM – 15:10 PM

Pattern: Online

Attendees: Chen Runzhou (student), Prof. Patrick Yue (supervisor), Babar Hussain (Ph.D.)

Progress Report (Report the progress made by Chen Runzhou since the beginning of the project.)

Chen Runzhou has finished 70% of the proposal report and done the background research for hardware part with 11 relevant papers.

Discussion Items:

Specify the objective of the project, what will be done in software and hardware part and what should be added to the proposal report.

Table 13. Action Items for Next Meeting

Action Item to be completed	By when	By whom
Specific Task 1 of Objective Statement 1	Oct 10th	Chen Runzhou
Specific Task 2 of Objective Statement 1	Oct 15th	Chen Runzhou
Specific Task 3 of Objective Statement 1	Oct 20th	Chen Runzhou

Meeting 2

Date: 27/10/2020

Time: 14:30PM – 15:20 PM

Pattern: Online

Attendees: Chen Runzhou (student), Prof. Patrick Yue (supervisor), Babar Hussain (Ph.D.)

Table 14. Action Items from Previous Meeting

Action Item to be completed	By when	By whom	Status
Specific Task 1 of Objective Statement 1	Oct 10th	Chen Runzhou	Completed
Specific Task 2 of Objective Statement 1	Oct 15th	Chen Runzhou	Completed
Specific Task 3 of Objective Statement 1	Oct 20nd	Chen Runzhou	completed

Table 15. Action Items for Next Meeting

Action Item to be completed	**By when**	**By whom**
Specific Task 1 of Objective Statement 3	Jan 10th	Chen Runzhou
Specific Task 2 of Objective Statement 3	Jan 15th	Chen Runzhou
Specific Task 3 of Objective Statement 3	Jan 20th	Chen Runzhou

Meeting 3

Date: 2/2/2021

Time: 14:30PM – 15:10 PM

Pattern: Online

Attendees: Chen Runzhou (student), Prof. Patrick Yue (supervisor), Babar Hussain (Ph.D.)

Table 16. Action Items from Previous Meeting

Action Item to be completed	By when	By whom	Status
Specific Task 1 of Objective Statement 3	Jan 10th	Chen Runzhou	Completed
Specific Task 2 of Objective Statement 3	Jan 15th	Chen Runzhou	Completed

Specific Task 3 of Objective Statement 3	Jan 20nd	Chen Runzhou	completed

Table 17. Action Items for Next Meeting

Action Item to be completed	By when	By whom
Specific Task 1,2 of Objective Statement 4	Feb 10th	Chen Runzhou
Specific Task 1,2,3 of Objective Statement 5	Feb 15th	Chen Runzhou

Meeting 4

Date: 18/3/2021

Time: 14:30PM – 15:10 PM

Pattern: Online

Attendees: Chen Runzhou (student), Prof. Patrick Yue (supervisor), Babar Hussain (Ph.D.)

Table 18. Action Items from Previous Meeting

Action Item to be completed	By when	By whom	Status
Specific Task 1,2 of Objective Statement 4	Feb 1st	Chen Runzhou	Completed
Specific Task 1,2,3 of Objective Statement 4	Mar 10th	Chen Runzhou	Completed

Appendix C. Deviation(s) from the proposal and supporting reason(s)

There is one main deviation from the proposal. Initially, the project was designed to include both the software part and the hardware part, as mentioned in the proposal report. However, due to the covid-19 situation and the quarantine policy in Mainland China and Hong Kong, I was not able to return to HKUST during year 4. As result, I did more coding work remotely and the focus of my project was on software development. This change is indicated in the progress report compared to the original plan in the proposal report.

Appendix D. Expression of the coefficient A, B and C

We express A, B and C using a number of known values for the convenience of calculation. All values used below are defined in step 6 of Section 2.2.2.4.

The second-order coefficient A = $\frac{e11e20m02m10^2v10v20}{m11^2} - \frac{e10e21m02m10^2v10v20}{m11^2} + \frac{e11e20m02m10v11v20}{m11} -$

$\frac{e10e21m02m10v11v20}{m11} - \frac{e11e20m00m10^2v12v20}{m11^2} - \frac{e11e21m01m10^2v12v20}{m11^2} - \frac{e12e21m02m10^2v12v20}{m11^2} +$

$\frac{e10e20m00m10v12v20}{m11} + \frac{e10e21m01m10v12v20}{m11} + \frac{e12e20m02m10v12v20}{m11} + \frac{e11e20m02m10v10v21}{m11} - \frac{e10e21m02m10v10v21}{m11} +$

$e11e20m02v11v21 - e10e21m02v11v21 + e10e20m00v12v21 + e10e21m01v12v21 +$

$e12e20m02v12v21 - \frac{e11e20m00m10v12v21}{m11} - \frac{e11e21m01m10v12v21}{m11} - \frac{e12e21m02m10v12v21}{m11} +$

$\frac{e10e21m00m10^2v10v22}{m11^2} + \frac{e11e21m01m10^2v10v22}{m11^2} + \frac{e11e22m02m10^2v10v22}{m11^2} - \frac{e10e20m00m10v10v22}{m11} -$

$\frac{e11e20m01m10v10v22}{m11} - \frac{e10e22m02m10v10v22}{m11} - e10e20m00v11v22 - e11e20m01v11v22 -$

$e10e22m02v11v22 + \frac{e10e21m00m10v11v22}{m11} + \frac{e11e21m01m10v11v22}{m11} + \frac{e11e22m02m10v11v22}{m11} -$

$e12e20m01v12v22 + e10e22m01v12v22 + \frac{e12e21m00m10^2v12v22}{m11^2} - \frac{e11e22m00m10^2v12v22}{m11^2} -$

$\frac{e12e20m00m10v12v22}{m11} + \frac{e10e22m00m10v12v22}{m11} + \frac{e12e21m01m10v12v22}{m11} - \frac{e11e22m01m10v12v22}{m11}$

The first-order coefficient B = $\frac{e11e20fum01m10v10v20}{m11} - \frac{e10e21fum01m10v10v20}{m11} - \frac{e12e20fum02m10v10v20}{m11} +$

$\frac{e10e22fum02m10v10v20}{m11} + \frac{2e11e20fum02m10m12v10v20}{m11^2} - \frac{2e10e21fum02m10m12v10v20}{m11^2} + e10e20fum00v11v20 +$

$e11e20fum01v11v20 + e10e22fum02v11v20 - \frac{e11e20fum00m10v11v20}{m11} - \frac{e11e21fum01m10v11v20}{m11} -$

$\frac{e12e21fum02m10v11v20}{m11} + \frac{e11e20fum02m12v11v20}{m11} - \frac{e10e21fum02m12v11v20}{m11} + e12e20fum01v12v20 -$

$e10e22fum01v12v20 + \frac{e12e20fum00m10v12v20}{m11} + \frac{e11e22fum01m10v12v20}{m11} + \frac{e12e22fum02m10v12v20}{m11} -$

$\frac{2e11e20fum00m10m12v12v20}{m11^2} - \frac{2e11e21fum01m10m12v12v20}{m11^2} - \frac{2e12e21fum02m10m12v12v20}{m11^2} + \frac{e10e20fum00m12v12v20}{m11} +$

$\frac{e10e21fum01m12v12v20}{m11} + \frac{e12e20fum02m12v12v20}{m11} - e10e20fum00v10v21 - e10e21fum01v10v21 -$

$e12e20fum02v10v21 + \frac{e10e21fum00m10v10v21}{m11} + \frac{e11e21fum01m10v10v21}{m11} + \frac{e11e22fum02m10v10v21}{m11} +$

$\frac{e11e20fum02m12v10v21}{m11} - \frac{e10e21fum02m12v10v21}{m11} - e11e20fum00v11v21 + e10e21fum00v11v21 -$

$e12e21fum02v11v21 + e11e22fum02v11v21 + e10e22fum00v12v21 + e12e21fum01v12v21 +$

$e12e22fum02v12v21 + \frac{e12e21fum00m10v12v21}{m11} - \frac{e11e22fum00m10v12v21}{m11} - \frac{e11e20fum00m12v12v21}{m11} -$

$\frac{e11e21fum01m12v12v21}{m11} - \frac{e12e21fum02m12v12v21}{m11} + e12e20fum01v10v22 - e10e22fum01v10v22 -$

$$\frac{e10e22fum00m10v10v22}{m11} - \frac{e12e21fum01m10v10v22}{m11} - \frac{e12e22fum02m10v10v22}{m11} + \frac{2e10e21fum00m10m12v10v22}{m11^2} +$$

$$\frac{2e11e21fum01m10m12v10v22}{m11^2} + \frac{2e11e22fum02m10m12v10v22}{m11^2} - \frac{e10e20fum00m12v10v22}{m11} - \frac{e11e20fum01m12v10v22}{m11} -$$

$$\frac{e10e22fum02m12v10v22}{m11} - e12e20fum00v11v22 - e11e22fum01v11v22 - e12e22fum02v11v22 +$$

$$\frac{e12e21fum00m10v11v22}{m11} - \frac{e11e22fum00m10v11v22}{m11} + \frac{e10e21fum00m12v11v22}{m11} + \frac{e11e21fum01m12v11v22}{m11} +$$

$$\frac{e11e22fum02m12v11v22}{m11} + \frac{2e12e21fum00m10m12v12v22}{m11^2} - \frac{2e11e22fum00m10m12v12v22}{m11^2} - \frac{e12e20fum00m12v12v22}{m11} +$$

$$\frac{e10e22fum00m12v12v22}{m11} + \frac{e12e21fum01m12v12v22}{m11} - \frac{e11e22fum01m12v12v22}{m11}$$

The constant C = $-e12e20fu^2m01v10v20 + e10e22fu^2m01v10v20 +$
$(e11e20fu^2m01m12v10v20)/m11 - (e10e21fu^2m01m12v10v20)/m11 -$
$(e12e20fu^2m02m12v10v20)/m11 + (e10e22fu^2m02m12v10v20)/m11 +$
$(e11e20fu^2m02m12^2v10v20)/m11^2 - (e10e21fu^2m02m12^2v10v20)/m11^2 + e12e20fu^2m00v11v20 +$
$e11e22fu^2m01v11v20 + e12e22fu^2m02v11v20 - (e11e20fu^2m00m12v11v20)/m11 -$
$(e11e21fu^2m01m12v11v20)/m11 - (e12e21fu^2m02m12v11v20)/m11 +$
$(e12e20fu^2m00m12v12v20)/m11 + (e11e22fu^2m01m12v12v20)/m11 +$
$(e12e22fu^2m02m12v12v20)/m11 - (e11e20fu^2m00m12^2v12v20)/m11^2 -$
$(e11e21fu^2m01m12^2v12v20)/m11^2 - (e12e21fu^2m02m12^2v12v20)/m11^2 - e10e22fu^2m00v10v21 -$
$e12e21fu^2m01v10v21 - e12e22fu^2m02v10v21 + (e10e21fu^2m00m12v10v21)/m11 +$
$(e11e21fu^2m01m12v10v21)/m11 + (e11e22fu^2m02m12v10v21)/m11 + e12e21fu^2m00v11v21 -$
$e11e22fu^2m00v11v21 + (e12e21fu^2m00m12v12v21)/m11 - (e11e22fu^2m00m12v12v21)/m11 -$
$(e10e22fu^2m00m12v10v22)/m11 - (e12e21fu^2m01m12v10v22)/m11 -$
$(e12e22fu^2m02m12v10v22)/m11 + (e10e21fu^2m00m12^2v10v22)/m11^2 +$
$(e11e21fu^2m01m12^2v10v22)/m11^2 + (e11e22fu^2m02m12^2v10v22)/m11^2 +$
$(e12e21fu^2m00m12v11v22)/m11 - (e11e22fu^2m00m12v11v22)/m11 +$
$(e12e21fu^2m00m12^2v12v22)/m11^2 - (e11e22fu^2m00m12^2v12v22)/m11^2$