NANYANG TECHNOLOGICAL UNIVERSITY

REPORT
ON
INDUSTRIAL ATTACHMENT
WITH
INSTITUTE OF INFOCOMM RESEARCH

PREPARED BY: KOA MING DI
075023E06
SCE

IA SUPERVISOR: DR SONG PENG

NTU SUPERVISOR: DR HENRY JOHAN

Abstract

This report describes the working experience during my industrial attachment with Institute of Infocomm Research. During the industrial attachment, I was tasked with the responsibility of developing projector-camera applications on the mobile platform.

Development on the mobile platform proved to be a challenging task. There were several features on the desktop platform that were not implemented onto the mobile platform. The project proved to be very code intensive as there were no available camera APIs compatible with Windows Mobile.

Throughout the development of the project, several optimized solutions were made to improve the performance of the application on the mobile platform. This includes finding the appropriate calibration technique, image warping technique, creating custom DirectShow filters and utilizing touchscreen gestures to increase the usability of the application.

Enhanced features, such as hand gesture recognition and object detection, of the application will be implemented in future work.

Acknowledgements

I would like to take this opportunity to express my gratitude to NTU and Institute of Infocomm Research(I^2R) for giving me the chance to fulfill my Industrial Attachment in I^2R.

I would like to express my thanks to the following people whom have helped me with my project during my attachment:

Dr. Song Peng, my IA supervisor, for guiding and providing me with suggestions when I was faced with problems.

Mr Koa Ming Hong, a Research Officer from IHPC, for providing me with various solutions regarding image processing and camera calibration.

Last but not least, I would like to thank my NTU Tutor, Asst Prof Henry Johan for his assistance in agreeing to co-supervise the continual of my IA project.

Table of Contents

REPORT ON INDUSTRIAL ATTACHMENT WITH INSTITUTE OF INFOCOMM RESEARCH ____ *i*

Abstract ____ *ii*

Acknowledgements ____ *iii*

Table of Contents ____ *iv*

List of Figures ____ *vi*

Chapter 1: Introduction ____ *1*

1.1 Objective ____ **1**

1.2 Company Profile – Institute of Infocomm Research (I2R) ____ **1**

Chapter 2: Project Specifications ____ *2*

2.1 Objective ____ **2**

2.2 Background ____ **2**

2.3 Project Requirements ____ **3**

2.4 Non-functional Requirements ____ **4**

2.5 Breakdown of Modules ____ **5**

2.5.1 Image Processing Module (C++) ____ 5

2.5.2 Camera Capture Module (C++) ____ 5

2.5.3 GUI Module (C#) ____ 6

2.6 Software and Equipments ____ **7**

Fig. 2.6.2 Samsung Omnia II i8000Chapter 3: Development of Image Processing Module ____ **8**

Chapter 3: Development of Image Processing Module ____ **9**

3.1 Camera Calibration ____ **9**

3.2 Calibration Techniques ____ **10**

3.3 Camera Projector Calibration ____ **12**

3.3.1 Calibration Approach ____ **12**

3.3.2 Frame Averaging ____ **16**

3.4 Automatic Keystone Correction ____ **17**

3.4.1 Implementation of Naïve Method ____ 18

3.5 Manual Keystone Correction ____ **21**

3.5.2 Implementation ____ 22

Using a look-up table ____ 22

3.6 Holes ____ **24**

Chapter 4: Camera Capture Module ____ *25*

4.1 Introduction to DirectShow ____ **26**

4.2 Application Model 26
4.3 Steps to get a Camera Preview 27
4.4 Preview Resolution Selection 28
4.5 Preview Rotation Problem 29
4.6 Solving Preview Rotation 31
4.6.1 Color Space Converter Filter 31
4.6.2 Rotation Transform Filter 32
4.7 Advance Camera Features – DirectShow 32
4.8 Advanced Camera Features – Samsung SDK library 34
4.9 Frame Grabber Filter 35
4.10 Conversion of RGB565 to RGB24 36
4.11 Final Filter Graph 37
Chapter 5: GUI Module *38*
5.1 Camera Application UI 38
5.2 Folder Browser UI 39
5.3 Polygon UI 40
5.4 Slideshow UI 43
Chapter 6: Conclusion *44*
Project Summary 44
IA Experience 45
References *47*

List of Figures

Figure 2.2.1	Setup of Sixth Sense	3
Figure 2.2.2	Hand Gestures Interaction	3
Figure 2.5.1	Decomposition of Application	5
Figure 2.6.1	3M's MPro120 Projector	8
Figure 2.6.2	Samsung Omnia II i8000	8
Figure 3.1	Pinhole Camera Model	9
Figure 3.2.1	A Calibration Object	10
Figure 3.2.2	Chessboard Calibration	11
Figure 3.3.1.1	Flowchart of Calibration Process	13
Figure 3.3.1.2	Captured Black Image	14
Figure 3.3.1.3	Captured White Image	14
Figure 3.3.1.4	Image Mask obtained after subtraction	14
Figure 3.3.1.5	captured chessboard pattern	15
Figure 3.3.1.6	chessboard after image mask	15
Figure 3.3.1.7	FindChessBoardCorners	15
Figure 3.3.1.8	Homography checking	15
Figure 3.3.2.1	Rainbow effect observed	16
Figure 3.3.2.2	Rainbow effect reduces after 5 frames were averaged	16
Figure 4.2.1	Interfaces exposed by Application	27
Figure 4.2.2	Flowchart of Filter Graph	28
Figure 4.4	Interfaces exposed by Application 2	29
Figure 4.5.1	Rotated preview	29
Figure 4.5.2	Physical Orientation	30
Figure 4.5.3	Shell Orientation	30
Figure 4.6.1	Filter Graph after adding rotation Transform Filter	32
Figure 4.7.1	VideoCaptureFilter's interfaces	33
Figure 4.7.2	Omnia II's compatibility with IAMVideoProcAmp	33
Figure 4.7.3	Omnia II's compatibility with IAMCameraControl	33
Figure 4.9.1	IBasicVideo captured image 1	35

Figure 4.9.2	IBasicVideo captured image 2	35
Figure 4.10.1	Bit representation of RGB565 to RGB24 conversion	36
Figure 4.11	Final Filter Graph's flowchart	37
Figure 5	Main form for application	38
Figure 5.1.1	Camera Application UI design	39
Figure 5.1.2	Camera Application UI design 2	39
Figure 5.2	Folder Browser UI	40
Figure 5.3.1	Polygon Form UI	41
Figure 5.3.2	Polygon Form UI 2	41
Figure 5.3.3	Polygon Form with warpped image	42
Figure 5.3.4	Projecting a undistorted image using Polygon UI	42
Figure 5.4	Keystone button being disabled	43

Chapter 1: Introduction

1.1 Objective

This report covers the project done by me during the period of 7th Jan 2010 to 28th May 2010. It includes the knowledge and experiences I learnt during the development of the project.

1.2 Company Profile – Institute of Infocomm Research (I2R)

I^2R is a member of the Agency for Science, Technology and Research (A*STAR). It is established in 2002 with a mission aiming to be the globally preferred source of innovations in 'Interactive Secured Information, Content and Services Anytime Anywhere', through research by passionate people dedicated to Singapore's economic success.

I^2R performs R&D in information, communications and media (ICM) technologies to develop holistic solutions across the ICM value chain. Their research capabilities are in information technology, wireless and optical communication networks, interactive and digital media, signal processing and computing.

Chapter 2: Project Specifications

2.1 Objective

The objective of this project was to research on the extensibility of the camera projector system on the mobile platform. General scope of the work includes developing a frame capturing framework and also developing basic applications to test the feasibility of such a system.

2.2 Background

Projectors are becoming more and more widely used as display devices because of its cost and mobility. A lot of interactive applications have been developed in the past using cameras and projectors on the PC platforms. The introduction of pico-projectors and camera phones would provide a good foundation for the development of a mobile camera-projector system. There have also been new projector-camera phones being released in the market that might further extend the feasibility of a mobile camera-projector system.

One of the more popular existing interactive mobile camera-projector systems is TED's Sixth Sense[1]. Sixth Sense is a product developed by students from the MIT Media Lab. It is a wearable computing system that turns any surface into an interactive system. This system creates a touchscreen operation without the actual touchscreen hardware. It consists of a webcam, a phone, a pico projector and a mirror. Users can interact with the system using hand gestures observed by the webcam.

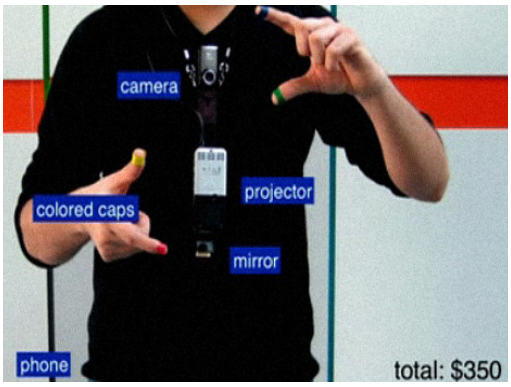


Fig. 2.2.1 Setup of Sixth Sense


Fig. 2.2.2 Hand Gestures Interaction

2.3 Project Requirements

Currently, the sixth sense appears to be an impressive system but also appears to be rather bulky. Given the current improvements in technology, such a mobile camera-projector system can be improved. Our project intends to remove the use of the webcam and fully make use of the phone's camera feature. Image processing will also be done on the phone instead of using a cloud server. This will ensure that our system can be used by non-internet users.

Our application intends to do automatic and manual keystone correction to properly adjust the projected image. The automatic correction involves frame capturing using the camera and also camera calibration to automatically adjust the projected image to make it look properly aligned in the user's point of view. The manual correction involves the user self-adjusting the locations of the corners of the projected image on the phone. It will be done synchronously with the user looking at the projected image on the screen.

2.4 Non-functional Requirements

Usability

The application developed must make use of the phone's touchscreen features in its UI, which will make the application easier to use.

Performance

Due to limitations of the computing resource and memory of the smartphones, the code must be optimized to reduce processing time and also to avoid memory leaks.

Platform compatibility

As there are too many variations of hardware used for mobile devices, it is not possible for every handphone to be compatible with the application. The application will be made at least to target high-end Samsung handphones.

2.5 Breakdown of Modules

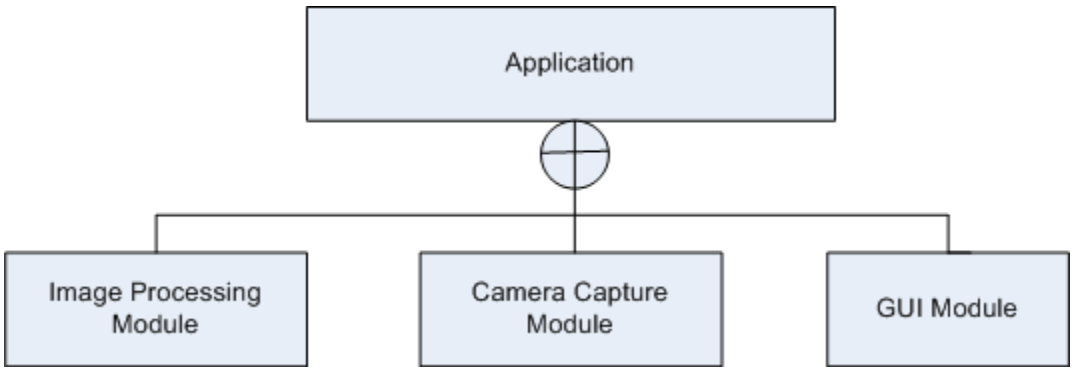


Fig. 2.5.1 Decomposition of Application

2.5.1 Image Processing Module (C++)

This module will import the Intel's image processing library OpenCV. It will contain image processing methods for camera calibration and also image warping functions for keystone correction.

2.5.2 Camera Capture Module (C++)

This module will make use of the DirectShow framework to capture camera images. This module will contain methods to start camera capturing and also methods for adjusting the camera's white balance, exposure and zoom through DirectShow. It also contains transform filters to grab and rotate camera frames, so that it will reduce the workload on the image processing module.

2.5.3 GUI Module (C#)

This module will be programmed in Managed C#. It will make use of the touchscreen features to ensure easy usability of the application. As most image processing and camera capturing features are not available in managed code. The GUI Module will also perform inter-operation calls to activate the C++ modules.

2.6 Software and Equipments

Intel Image Processing Library OpenCV 2.0.0

OpenCV is a computer vision library that contains various image processing functions. All functions in OpenCV are programmed to optimized CPU usage. However, the existing OpenCV library is not designed to work on the mobile platform. There is only 1 existing method of exporting OpenCV to the mobile platform and it can be found on Alex's Karpus' website: http://alexkarpus.com/opencv/

Camera libraries of OpenCV are not compatible with the mobile platform. Hence to access camera features, we will need to program using C++'s DirectShow framework.

Microsoft Visual Studios 2008

Microsoft Visual Studios is an Integrated Developer Environment (IDE) that reduces the complexity of programming in C++/C#. It provides a drag and drop plugin for easy development of GUI in C# language. This IDE is preferred over Microsoft Visual C++, as Microsoft Visual C++ does not support C# programming.

Windows Mobile SDK 6.5 Developer Environment and Tools

The Windows Mobile SDK provides libraries equivalent to the PC version for the Windows Mobile platform. It also contains emulator images for testing purposes without the phone. The 6.5 SDK also provides additional gesture APIs that is meant for creating touchscreen applications. Libraries for both unmanaged C++ and managed C# are available in this SDK.

Samsung Windows Mobile SDK 2.2

The Samsung Windows Mobile SDK offers APIs to use advanced features unique to Samsung phones. For this specific project, we will be using the advanced camera API in this SDK.

3M's MPro120 Pico Projector

The MPro120 is a miniature handheld projector that is capable of casting a big image from 8inches to 50 inches wide. It uses an energy efficient LED light source that can last up to 20,000 hours.

Fig. 2.6.1 3M's MPro120 Projector

Samsung Omnia II i8000

The Omnia II was chosen as the ideal phone for this project because it has TV-out connectivity. The Omnia II can use either the Windows Mobile 6.0 or 6.5 SDK. It also has a processing power of 800MHz and a 256MB RAM. It has good camera functions as well.

Fig. 2.6.2 Samsung Omnia II i8000

Chapter 3: Development of Image Processing Module

3.1 Camera Calibration

The purpose of geometric calibration is to find the mathematical relationship between 3D positions in the world co-ordinate frame and its projection on the 2D camera image plane.

Firstly, the pinhole camera model is used to model the mathematical relationship between 3D points in the camera co-ordinate frame and its projection onto the 2D camera image plane.

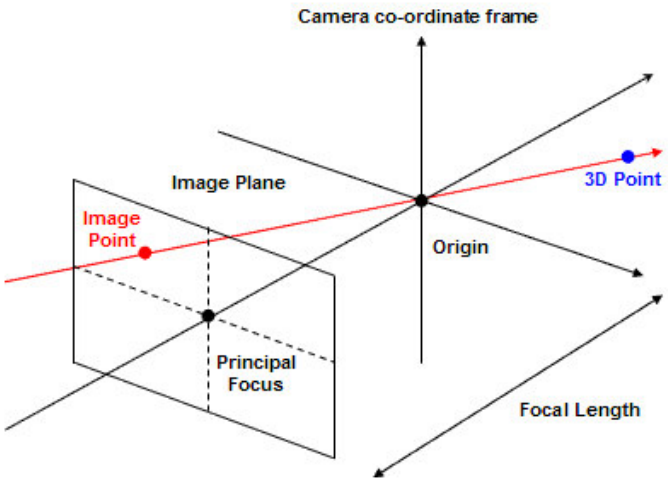


Fig. 3.1 Pinhole Camera Model

The mathematical relationship between the camera co-ordinate frame and image plane can be represented in matrix form. This matrix is known as a homography.

3.2 Calibration Techniques

All homography calculations used today require corresponding pairs of 2D points as input. Through different methods, 2D positions on the image plane and their corresponding 2D positions in the camera frames are located and paired together. These pairs of correspondences are then used with a calibration algorithm to compute an optimized solution.

There are 2 main methods used in generating the pairs of correspondences. The first method uses a specially constructed calibration object. The object has specially marked out points along the surface edges, each having a predetermined 3D position. From the image captured by the camera, each point marked out points in the image generates a corresponding pair of coordinates.

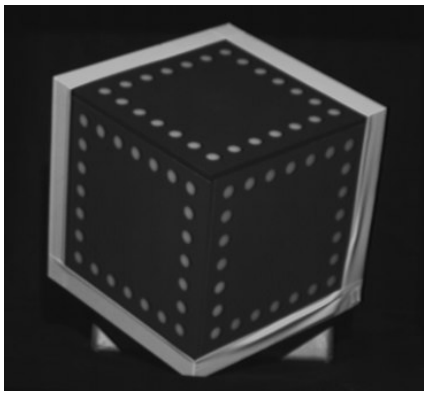


Fig. 3.2.1 A Calibration Object

The second method uses a plane with a checkerboard pattern to calibrate the camera. Either the checkerboard or the camera is moved to capture images of the checkerboard pattern from different angles. A corner detection algorithm is used on the images to

detect the internal corners of the checkerboard pattern. By fixing the world co-ordinate origin at one of the extreme corners of the board and assuming that the board is on the plane Z=0, a 3D position can be calculated for all corners on the board. Thus, every corner will generate a pair of correspondences for every view image.

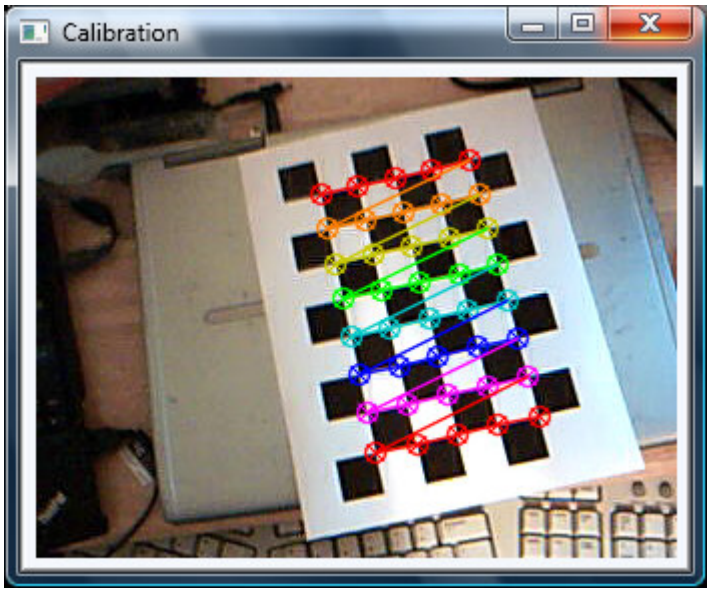


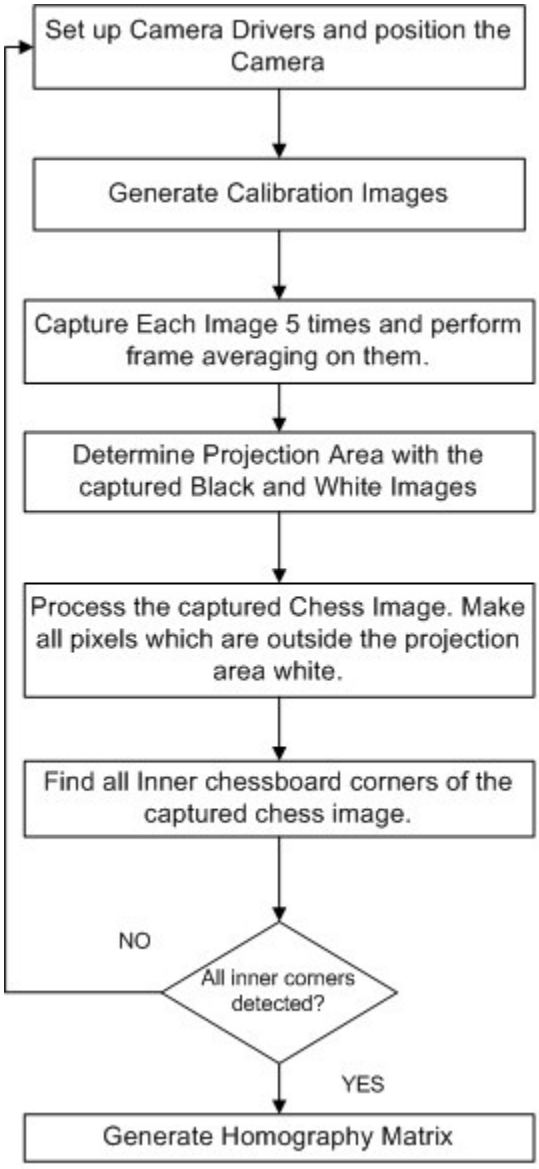
Fig. 3.2.2 Chessboard calibration

3.3 Camera Projector Calibration

The projector is used to project a pattern onto the display surface and the projection is captured using the phone's camera. Pairs of 2D and 2D correspondence can be formed and the homography (with respect to the camera) can be estimated using at least 4 pairs of correspondences. To further increase the accuracy of the homography, there is a need to generate more pairs of correspondences and to find the best fit homography among them. This method is only accurate on the calibration plane and it does not consider distortion parameters. Since we want to reduce the computation time for the mobile platform, we should ignore distortion parameters during the calibration phase.

3.3.1 Calibration Approach

For this project, I used a calibration process similar to a software known as LightDraw[2], a laser pointer based Camera-Projector system. Lightdraw displays four images (Black and White Image, followed by a chessboard and a crop chessboard image) sequentially on the screen for a second each. Each of the four images is captured by the camera and processed to capture lighting information about the set-up environment. To reduce the calibration time, I used only the 1st 3 calibration images.



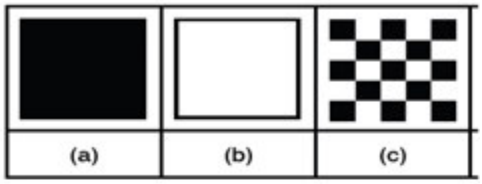


Fig 3.3.1.1 Flowchart of Calibration Process

Black and White Image

The black and white images are used to determine the projection area. Captured images are averaged and converted to grayscale. We can obtain the projection area by performing subtraction between the 2 images. Pixels that change by more than a fixed threshold are considered to be inside the projection area. Once a projection area has been obtained, an image mask which limits all processing within the masked region, can be created.

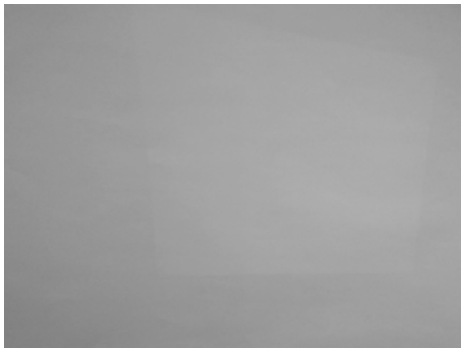


Fig. 3.3.1.2 Captured Black Image

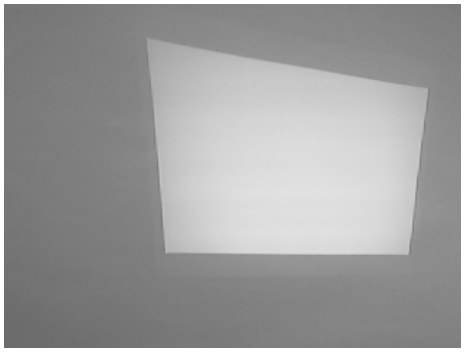


Fig. 3.3.1.3 Captured White Image

Fig. 3.3.1.4 Image Mask obtained after subtraction

Chessboard Pattern Image

The chessboard pattern image is used to generate a homography matrix. We project a 6x6 chessboard pattern on the screen. The captured image is first processed to convert pixels that are not within the projection area to white (Figure 3.3.1.6). The image is then passed to the OpenCV's FindChessboardCorners (Figure 3.3.1.7) to extract the internal corner co-ordinates. Given a 6x6 matrix, there will be 25 internal corners. Once all 25 corners are found, we can generate a 3x3 homography (camera to image, image to camera) matrix.

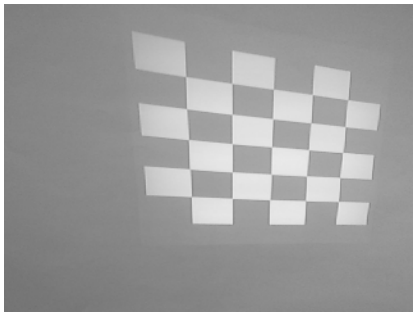


Fig. 3.3.1.5 captured chessboard pattern

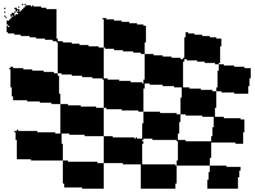


Fig 3.3.1.6 chessboard after image mask

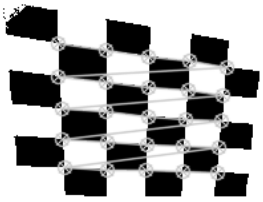


Figure 3.3.1.7 FindChessBoardCorners

After the calibration is completed, a test can be done to check if the homography is correct. We use the Camera to Image homography to perform a transform on a Figure.3.3.1.5. If the homography is correct, we should be able to obtain an image completely occupied by the 6x6 chessboard.

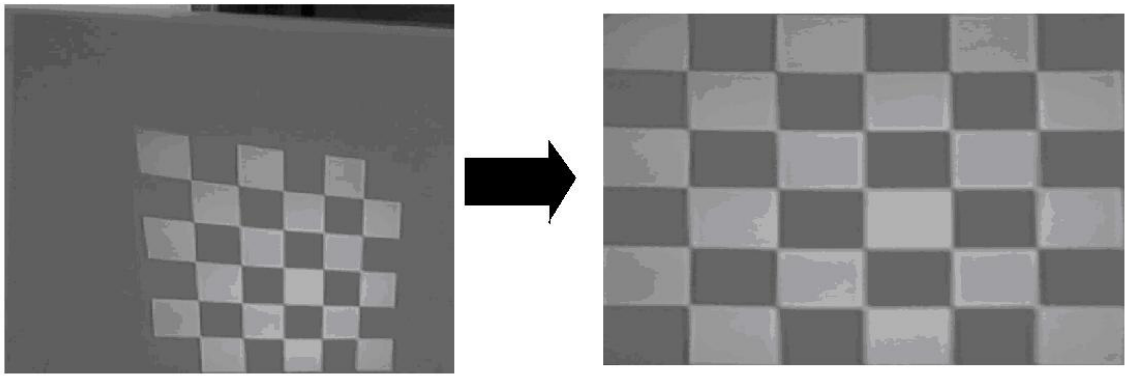


Fig. 3.3.1.8 Homography checking

3.3.2 Frame Averaging

Frame Averaging is a process of adding a few images together, and later dividing them to get the average pixel intensity for each pixel. Frame Averaging should not be excluded during camera calibration. This is mainly due to the fact that the phone's camera is unable to view the projector image clearly. The MPro120 is a LCoS (Liquid Crystal on Silicon) projector and it is known to exhibit an effect known as the rainbow effect. The low exposure time of the phone's camera combined with the low display frequency (29.97Hz) of the projector makes the 'rainbow' effect visible as seen in Figure 3.3.2.1.

Most pico projectors in the market are either DLP (Digital Light Processing) or LCoS. Both are known to exhibit the rainbow effect. Since there is no room for changing hardware, Frame Averaging is seen as one of the easier solutions to remove this problem.

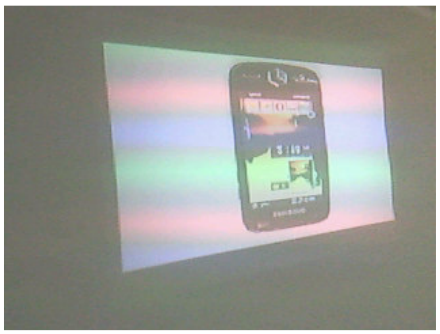


Fig. 3.3.2.1 Rainbow effect observed

After Frame Averaging, the 'rainbow' effect is reduced as seen in Figure 3.3.2.2

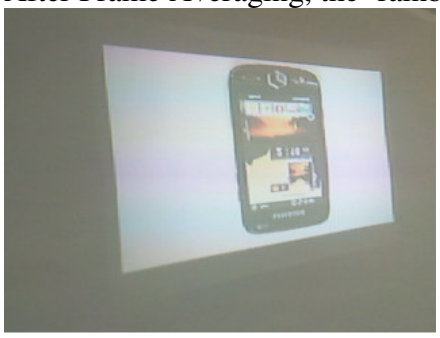


Fig. 3.3.2.2 Rainbow effect reduces after 5 frames were averaged.

3.4 Automatic Keystone Correction

In a research paper titled - Smarter Presentations: Exploiting Homography in camera-projector systems, which is written by Rahul Sukthankar[3], proposes 2 methods for keystone correction.

Naïve Method

The 1st method of keystone correction is called the "naïve" method. In the first step, the camera captures a projected image and determines the best rectangular region in which the contents of the image should appear. In the second step, back project the corners of this rectangle into projector coordinates. In the last step, we determine the projective transform that warps the slide to this desired quadrilateral. The naïve method will produce a rectangular image to the camera's point of view.

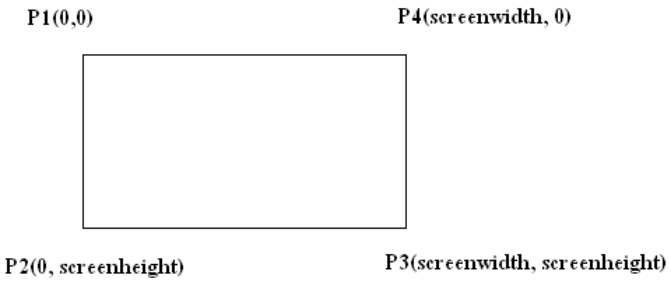
Proposed Method in the Paper

The proposed method does a image capturing to obtain the 4 corners of the projector screen. It later determines a model for the distortion between the projector and the projector screen. It later computes a pre-warp that maps the slide to a suitable target rectangle on the projection screen.

Since we want to exploit the portability of the mobile camera-projector system, we have to assume that the projector screen is not always readily available in this scenario. The targeted surfaces will usually be a table top or wall, hence it is not practical for us to do corner detections to obtain the edge of a table top or the edge of a wall.

3.4.1 Implementation of Naïve Method

1. Create a 3x4 Matrix to store the coordinates of the screen(phone) resolution.



$$\text{Corners} = \begin{bmatrix} 0 & 0 & screenwidth & screenwidth \\ 0 & screenheight & screenheight & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

2. Locate inner corners of the above matrix on camera coordinates.

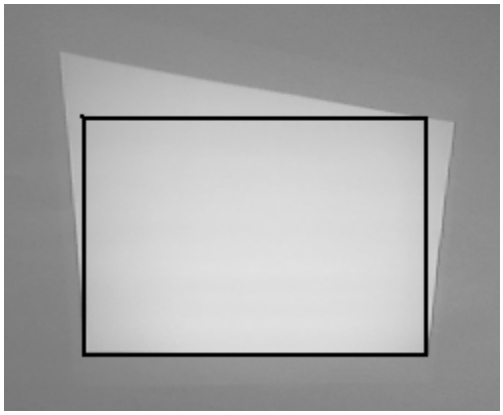
Inner Corners = Homography (screen to camera) x Corners

$$\begin{matrix} \textbf{\textit{P1}} & \textbf{\textit{P2}} & \textbf{\textit{P3}} & \textbf{\textit{P4}} \end{matrix}$$
$$\begin{bmatrix} X_0 & X_1 & X_2 & X_3 \\ Y_0 & Y_1 & Y_2 & Y_3 \\ Z_0 & Z_1 & Z_2 & Z_3 \end{bmatrix} = \begin{bmatrix} 3\,x\,3 \\ Matrix \end{bmatrix} \quad \text{X} \quad \begin{bmatrix} 3\,x\,4 \\ Matrix \end{bmatrix}$$

3. Normalize inner corners Matrix, set Z = 1.

$$\begin{matrix} \textbf{\textit{P1}} & \textbf{\textit{P2}} & \textbf{\textit{P3}} & \textbf{\textit{P4}} \end{matrix}$$
$$\text{Inner Corners} = \begin{bmatrix} \frac{X_0}{Z_0} & \frac{X_1}{Z_1} & \frac{X_2}{Z_2} & \frac{X_3}{Z_4} \\ \frac{Y_0}{Z_0} & \frac{Y_1}{Z_1} & \frac{Y_2}{Z_2} & \frac{Y_3}{Z_3} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

4. Calculate best fitting rectangle size based on inner corners.



The above rectangle, can be computed by the following algorithm below:

Min X_C = Max (X_0, X_1)
Max X_C = Min(X_2, X_3)
Min Y_C = Max(Y_0, Y_3)
Max Y_C = Min(Y_1, Y_2),
where X_0,Y_0 are coordinates of P1 in Inner Corners and X_1,Y_1 are coordinates of P2 in Inner Corners and so on.

The coordinates of the above rectangle can be represented by a Matrix <CamPoints>.

$$\text{CamPoints} = \begin{matrix} \textit{P1} & \textit{P2} & \textit{P3} & \textit{P4} \end{matrix} \begin{bmatrix} \textit{MinXc} & \textit{MinXc} & \textit{MaxXc} & \textit{MaxXc} \\ \textit{MinYc} & \textit{MaxYc} & \textit{MaxYc} & \textit{MinYc} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

5. Locate coordinates of corrected rectangle on screen(phone).

Screenpoints = Homography(Camera to Screen) x CamPoints

$$\begin{matrix} \textit{P1} & \textit{P2} & \textit{P3} & \textit{P4} \end{matrix} \begin{bmatrix} X_0 & X_1 & X_2 & X_3 \\ Y_0 & Y_1 & Y_2 & Y_3 \\ Z_0 & Z_1 & Z_2 & Z_3 \end{bmatrix} = \begin{bmatrix} 3\,x\,3 \\ \textit{Matrix} \\ \end{bmatrix} \quad \text{X} \quad \begin{bmatrix} 3\,x\,4 \\ \textit{Matrix} \\ \end{bmatrix}$$

6. Normalize Screenpoints

$$\text{Screenpoints} = \begin{bmatrix} \frac{X_0}{Z_0} & \frac{X_1}{Z_1} & \frac{X_2}{Z_2} & \frac{X_3}{Z_4} \\ \frac{Y_0}{Z_0} & \frac{Y_1}{Z_1} & \frac{Y_2}{Z_2} & \frac{Y_3}{Z_3} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

7. Compute Homography (Keystone)

Keystone x Inner Corners = Screenpoints

$$\begin{bmatrix} 3\,x\,3 \\ Matrix \end{bmatrix} \text{X} \begin{bmatrix} 3\,x\,4 \\ Matrix \end{bmatrix} = \begin{bmatrix} 3\,x\,4 \\ Matrix \end{bmatrix}$$