Abstract

In general, the whole project is about developing a system for playing a board game against a computer by using projected Augmented Reality. The key point of this project is that it utilises Augmented Reality techniques as well as traditional physical pieces of the game. It tries to blend the digital experience together with traditional game interactions. The game here in this project mainly refers to chess game, but in future it can be referred to many other similar board games. In this article, it will mainly focus on the projection part. The projector is supposed to show visual feedbacks onto the chessboard, so that users can get information such as game tips, scores during the game. More technically, the projector should be able to project graphic annotations onto the proper locations of the chessboard. Therefore, briefly, the aim of this project became to develop a automatic projector calibration system. In this project, there were several main contributions and achievements as following:

- In a word, I built a vision-based automatic calibration for projector, only adopting a normal webcam, a projector, computer and a chessboard.
- I tried to reconstruct the 3D positions of chessboard inner corners by using ray-plane intersection algorithm.
- I implemented homography to reconstruct the mappings between camera-captured image and projected image (screen shot on monitor).
- An evaluation of the system built in this project was done, leading to an analysis of both constraints and possible opportunities of current system.

Contents

CHAPTER 1

Introduction

In this chapter, brief outline of this project will be illustrated and brief background of Augmented Reality technology will be introduced. It will lead to a better understand of the concept of Augmented Reality as well as some typical techniques in this field. In the end of this chapter, aims and objectives of this project will be explained.

1.1 - Project Introduction and Motivations

Digital technology has been utilised to enrich our life in every possible way. It has developed rapidly in past decades. Those novel techniques have brought us impressive experience in virtual digital world. There are always new styles of human-computer interactions. Those new techniques have been adopted in variety fields, such as education, entertainment, and medical care. This project would be one of these techniques, which may deliver a relative novel approach to user-computer interactive system. In general, the whole project is about developing a system for playing a board game against a computer by using projected Augmented Reality. The key point of this project is that it utilises Augmented Reality techniques as well as traditional physical pieces of the game. It tries to combine the digital experience together with traditional game interactions. Ideally, it would construct a newly developed interactive system that avoids becoming a purely digital system. The game here for this research mainly refers to chess game, but in future it maybe many other similar board games. The system has three main components: a computer, a webcam and a projector (this would be explained in details later). My responsibility within this project is mainly focus on the work of projector. The projector is supposed to show visual feedbacks onto the chessboard, so that users can get information such as game tips, scores during the game. More technically, the projector should be able to project graphic annotations onto the proper locations of the chessboard in the end of this project. Therefore, briefly, the aim of this project became to develop a automatic projector calibration system. Once this system was completed, it should be able to combined with the whole system, and then realise the idea of 'Augmented Reality Chess Game'.

1.2 - Augmented Reality Technology

Augmented Reality as a newly developing concept has gained an impressive progress. A large amount of techniques in this field have been invented and developed in past decades. People start to be aware of this new technology, and various of applications have emerged.

1.2.1 - Definition and history of Augmented Reality

In the precomputer age, things could only be presented with real-world properties, such as tangible objects, sense of space. Therefore, in that time, human may only have two types of interactions within the world: human-to-‘real world’ interaction and human-to-human interaction. Afterwards, because of the emergence of the computer, which showed human the capabilities of the virtual world, people attempted to blend the real and virtual elements together. Accordingly, on one hand, the original physical interaction experience with the real-world could be enhanced by supplementing some virtual elements, such as computer graphic annotations and special sound effects; on the other hand, the computer technology itself had brought a new style of interaction, which is human to the virtual-world interaction. This then was split into two main perspectives: augmented virtuality (which refers to the situations that the real elements are added to the virtual environment) and augmented reality (which in contrast, refers to the situations that the virtual elements are added to the real environment) [1]. Basically, the system of ‘Augmented Reality Chess Game’ is a newly conceived interactive system that brings virtual elements to the traditional board game. Therefore, when talk about project, we mainly refer to the second perspective: the augmented reality.

What is augmented reality? There are amount of deliberate definitions. However, in general, an augmented reality system blends the virtual and real objects with each other in the real world in the real time. That is, making the virtual objects integrated in the real world, or, depending on perspective, augmenting the real world with computing technologies. According to Sutherland’s work, the origin of the augmented reality (AR) can be traced back to 1960s, which was about using a see-through head-mounted display (HMD) device to show 3D graphics [21]. Nevertheless, AR has been defined as a research field only in recent decade, first by a survey written by Azuma in 1997 [4]. During the past decade, AR has developed rapidly, and has been implemented in variety of technologies. On of the most typical technology is called head-mounted display (HMD), and it is a special presenting equipment

Figure1.1

(a) (b)

Figure 1.1 a). It is the VR Pro ST with see-through and high resolution 1920×1080 (source from online: http://www.vrealities.com/vrprost.html); b).It is the 5DT HMD 800 series of Head Mounted Display (source from online: http://www.vrealities.com/5dt.html)

used in the same way as a pair of glass to display 3D graphics (showed in Figure 1.1a-b).

Followed by this technique, there appear many other novel implementations of AR, such as some handheld displays, projection displays. Most of these techniques are used to display virtual scenes to coexist in the corresponding space in the real world. These types of display applications have been used in various occasions, normally in some museums, libraries, or even some technical conferences. After years development, AR tends to integrate in humans daily lives, such as games, educations. In so doing, interactions between human and the real world can be enriched by embedding these virtual graphic presentations into the real world surroundings [1]. Because of the remarkable development and progress of the AR technologies, they have been utilised in divers areas: medical, entertainment, education, manufacturing and even the military applications [4].

1.2.2 - Classic Augmented Reality techniques and utilisations

Nowadays, due to the development of the AR, there is an innovation within the interactive system technology. Amount of purely virtual computer interactions seem to be brought back to the real world. From another perspective, many traditional, real-world interactions are integrated with augmented virtual elements to become more attractive. These approaches and technologies are designed to create more exciting interaction experience. For the instance of games, although the purely virtual games have already provided fantastic experience with nice computer graphics and various sound effects, people are expecting something different, which can enrich their interactions within the game in any possible way[2, 3, 7]. However, computer games have their own drawbacks as they decrease the players' physical activities and social interactions. Conventionally, computer games make users focus on the computer screen with other typical manipulating devices: a keyboard and a mouse [2].

To address this problem, physical movement and social interaction have been brought back into games, while still keeping the benefits of digital techniques. Usually, augmented view with 3D graphic annotations will be registered so that they seem to exist in the corresponding space in the real world. Generally, there are three main approaches of utilising augmented reality for games according to Carsten [2]:

- ***Using head-mounted displays:***

  There are two kinds of displays within this category. One is the video see-through augmented reality, and the other is the optical see-through augmented reality.

- ***Using images projected on real-world surfaces***
- ***Using hand-held devices***

Among all ranges of augmented reality games, tabletop games seem to play a significant role within the pervasive gaming paradigm. It is probably because of its planar surface, on which essential components can be mounted [5]. Unlike many other approaches, which attempt to transform the real world properties into a game board, augmented tabletop games build directly on the traditional board games. It is aim at combining the traditional social interaction metaphors with novel computer techniques, thus achieving higher attractiveness.

Researches have been done on different augmented tabletop programs to investigate both opportunities and drawbacks. There is a more complicated experimental platform called STARS [2, 22]. It integrates dedicated components such as a public vertical displays and personal digital assistants (PDAs) with a smart interactive table. Several applications based on this platform have been developed and studied to show the opportunities for such kind of augmented tabletop games. The primary element of each STARS game is the smart interactive table, on which users can play AR games (Figure 1.2). Physical game pieces are provided on the tangible interface with graphic annotations on it. Most of these games utilise a touch-sensitive display set. So that, users can manipulate the system by touching the table surface just like the other traditional board games.

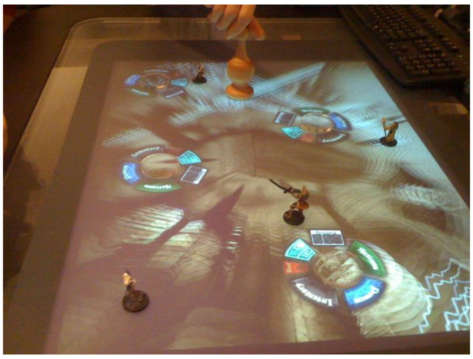


Figure 1.2. A STARS tabletop platform

1.2.3 - Physical interaction combined with Augmented Reality techniques

Despite the advent of the digital entertainment technologies, traditional tabletop games such as Chess or Go still show their constant attractiveness to people. Their continuing popularity does not seem to fade out, and can undoubtedly reveal the fact that the direct interaction and communication between the players as well as physical game pieces still motivate people's engagement during such games. In the final system of this project, tangible game pieces of chess are supposed to be used to add richness to the AR Chess. Therefore, in this part, researches about impacts of physical interactions with tangible game apparatus on players will be demonstrated.

In recent years, there is a growing number of studies trying to find out how and what can those tangible game components influence the players. Many of them argue that physical interactions amplify user experience with playfulness and emotional engagement [9, 10, 11, 12, 13]. In researches, people use 'Tangible' to refer to the physical artefacts, which are augmented and enhanced digitally to launch diverse digital events. In this project, visual feedbacks will be projected on the chessboard, in order to offer essential information for players during the game. According to previous studies, physical artefacts combined with an associated digital world to become a new form of tangible entertainment environment, thus providing people a new way of interacting in the real world with some known and familiarity. It assumed that this could provoke higher levels of freedom to explore, manipulate and react in games with physical pieces as well as the digital effects on them [9].

A research about a learning appliance called the Learning Cube has been done. It claimed that a physical appliance with digitally augmented can highly increase children's engagement and exploration when playing with it [13]. As according to Smets' study, difficulties exist when children interacting with traditional computer control device such as the mouse, the monitor or the keyboard, due to their spatially separated perception [27]. A Learning Cube is generally a cube box with six faces (Figure 1.3). 3D objects created by digital drawing is designed to be in the box, and projected orthogonally on each face of the box. In addition to the projection technique, there is also a speaker inside it. The experiment was taken among a group of children. Like all other original physical device, it can be touched, picked up, rotated, threw and shaken. Through all this interactions with this augmented cube box, results showed that this kind of physical supported learning could actually lead to a great initial engagement. Most children in this experiment tended consider the cube as a toy, rather than a learning tool [13].

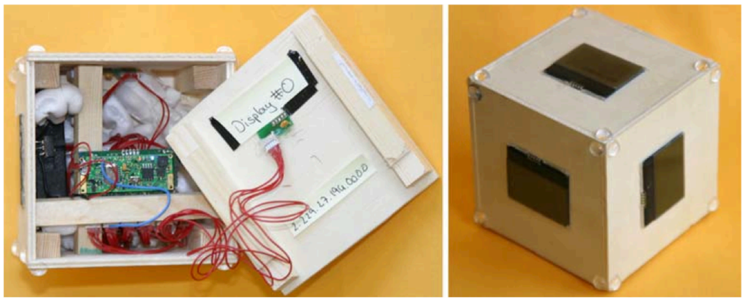


Figure 1.3. The Learning Cube [R13]

Another important study was done to compare the children's pleasure and engagement using physical, graphical and tangible user interfaces. Physical user interface (PUI) here refers to the traditional interface without any computing functionalities. Graphical user interface (GUI) is the digital representations on digital devices. Tangible user interface (TUI) can be a good example of extension of conventional physical system by utilising brand-new digital techniques. School-aged children were chosen to take part in this experiment by playing with a jigsaw puzzle, which had been implemented using these three interface styles: physical, graphical and tangible. Their engagement, enjoyment and collaboration were examined during series of tests. Pairs of participants were encouraged to play the puzzle together. Results showed that it took longer and met more difficulties when completing puzzles in the graphic style interface. Based on their observations and interviews with these children, this mainly because of the indirect interaction with the virtual 2D screen. Additionally, higher degree of engagement in PUI and TUI conditions had been shown through the repeat play test [10].

These findings imply a significant fact in an interactive system design. Digitally augmented reality system combined with physical pieces, which is visible and touchable, can bring better performance than those conventional purely virtual interactive systems. With the direct interactions, it will maintain higher intrinsic motivation for users. Therefore, in this project, the Chess will remain its traditional playing style while being augmented by some digital effects and functionalities.

1.3 Introduction to Chess game

As the system of this project is based on the game of Chess, knowledge and rules of Chess seems to be essential before designing the system. In this section, knowledge of Chess will be briefly introduced.

The origins of chess are unsure, however, it became popular since the 15th century in Europe. Nowadays, it is becoming a popular board game around the world. Chess is a traditional board game played by two people, where the objective is to checkmate the other king. Checkmate refers to the situation when the king is to be captured and cannot escape. The game board has 64 squares in total of alternating colours. At the beginning of the game, each player has 16 pieces: 1 king, 1 queen, 2 rooks, 2 bishops, 2 knights and 8 pawns. The chess pieces are supposed to be arranged the same way every time the game begins. In the first row, which is the edge row faced the player, the rocks go in the corners, the knights are next to them, and then the bishops, finally it is the queen, and the king is put in the remaining space. The pieces usually come in two colours, one player plays with black pieces, and in contrast, the other player plays with white pieces. The one with white pieces always moves first. Then followed by black, then white again, then black and until the end of the game. During the game, each of the six different kinds of pieces moves in different manner. There is usually a chess game embedded within a PC. In the final system of the whole project, a basic chess program is expected to be assembled in the AR Chess system, and more attractive features will be implemented later to enrich the game system.

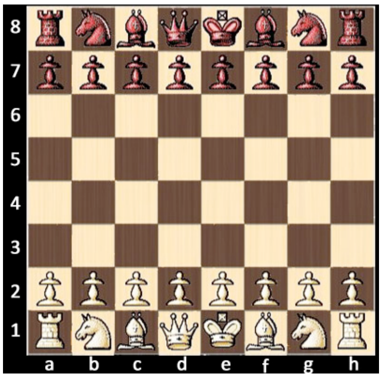


Figure 1.4. The Chess Game

1.4 Aims and Objectives

In general, as mentioned, the whole project is about developing an AR interactive system as opposed to a pure digital game system for playing a board game. The game here particularly refers to the chess. During the game, a real chessboard and pieces will be used and the computer's feedbacks will be projected on the board visually. Hence, a portable projector and a webcam that can recognise pieces and monitor their movements are required to be mounted above the board. My responsibility within this project was focus on the projection part, mainly referring to auto-calibration for the projector. Therefore, the aim is to develop an auto-calibration system for the projector to project graphic annotations onto the proper locations on the chessboard.

In order to fulfil the aim of the this project, the following objectives were set to be achieved:

- ***Literature Review***: Do some background researches on projector calibration, especially those about vision-based auto-calibration for projector. Study some algorithms that can be used to figure out the geometric relationships between webcam, projector and the chessboard, focusing on the concept and realisation of ‘Homography’.

- ***Set up Equipment***: Try to mount both webcam and projector above the chessboard, connecting with computer. Make sure that the chessboard is within the view of camera and the projection area of the projector.
- ***Programming and Tests***: C++ was chosen as the programming language in this project, and OpenCV library was selected as it allowed many computer vision and image processing functions to be used. Furthermore, C++ works better in terms of computational efficiency for projects that involved in computer vision and image processing programs.
- ***Experiment on the Real Equipment***: Implement the program on the equipment, and experiment the system in different circumstances. Figure out the working efficiency and accuracy of the program. Summarise the constraints of program implementation, environment and equipment.
- ***Explore possible Future Improvement***: Discuss and try to find out the possibilities to refine the system. Think about potential ways to modify and optimise the system in order to alleviate the effects caused by different constraints. Attempt to discuss ways in which can probably yield better results and provide users with better experience.

1.5 Structure

This report consists of five main sections. Firstly time will be spend demonstrating some background and related work of automatic calibration system for projector. It will focus on the vision-based calibration system, which utilise multiple or single camera to fulfil this task. In the second section, definitions and some details of methodologies used in this project will be explained. This may offer people a basic understanding of the uses of those methods in reconstructing geometric relationships between components. In the implementation section, both the programming language used and approaches tried during the project will be illustrated and discussed. There were mainly two trials during the project. Problems met during the implementation process will be explained as well. The next section is about total evaluation of the project implementation, equipment constraints and environment constraints. Finally, experiments for testing the performance of this system in some special occasions will be talked about, and some suggestions on possible improvements that are worthy further investigating will be enumerated.

CHAPTER 2

Background and Related Work

In this chapter, some background of projector utilisation in the Augmented Reality techniques will be talked about. Then it will concentrate on details of some classic projector calibration systems. This may provide this project with some general idea of automatic vision-based calibration system for projector.

2.1 Projectors in Augmented Reality interactive systems

As a whole system, a typical augmented reality interactive system contains not only a computer, but also devices such as projector, camera, or any other sensors. Interactive tabletop systems, same as other AR interactive system, access its capabilities by using techniques such as capacitive sensing, RFID, active infrared emitting devices, or computer vision-based recognition and detection [5]. There are many current games that are implemented in the similar way as the Augmented Reality Chess designed to be. Following are some examples of current available AR tabletop interactive systems.

2.1.1 False Prophets

This is a hybrid board-video game system, which keeps advantages of both physical and digital media. In so doing, it can enhance players' interaction experience [23]. By moving tangible game pieces on a touch-sensitive smart board with a digitally projected game board, players gain higher interactions with both other players and the game itself. Digital annotations draw players' attentions and make them pay more attentions to changing clues. Buttons for simple game operations are supplied on the game pieces. A computer would be provided to deal with more complicated operations and information.

2.1.2 PlayAnywhere [6]

It is a front-projected computer vision-based interactive table system. The PlayAnywhere system is motivated by the advent of brand-new sensing and display technology. It is based on the combination of projection techniques for display as well as computer vision based techniques for sensing. As its name implied, this system is supposed to be portable, which means users can use this system on any surface. Figure 2.1 shows the prototype of PlayAnywhere, which consists of a NEC WT600 DLP projector to project a 40" diagonal

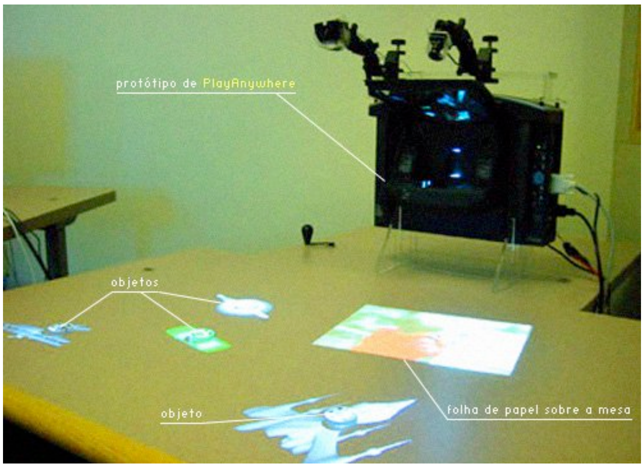


Figure 2.1. PlayAnywhere prototype placed on a normal table surface. And four game pieces and a real piece of paper are detected.

image onto a tabletop surface, a Sony ExView analog grayscale CCD NTSC camera that is highly sensitive to the near infrared domain, an Opto Technology OTLH-0070-IR high power LED package, and an assembled infrared illuminant. In terms of sensing, user interaction of PlayAnywhere system is based on a shadow-based touch detection algorithm. It allows the touch detection by examining the shadows of moving objects. Here it refers to the fingers. Basically, it requires two fingers, one as reference finger that is above the surface and the other as index finger that is touching the surface. By examining the differences of shadows of two fingers, it can finally detect the fingers' manner while using the system.

2.1.3 Augmented Reality Go [8]

This system seems to be the most similar system with the system in this project. Nonetheless, it is still slightly different from the Augmented Reality Chess in some way, which will be explained later. Generally, the AR Go system is an interactive board system, which contains a projector to display visual feedbacks on the real board of go, a camera to sense the movements of game pieces and a tough pane with IR sensor attached next to the interactive area to sense whether the finger placed on line. Meanwhile, virtual buttons will be displayed on the touch pane for users to manipulate the game (Figure 2.2 a-b). As encouraged by the benefits of traditional games, the AR Go allows users to use real stones and game board to play with, which may enrich the interaction process between the users and the game. Additionally, users can choose either the competition mode played by two players or the self-learning mode.

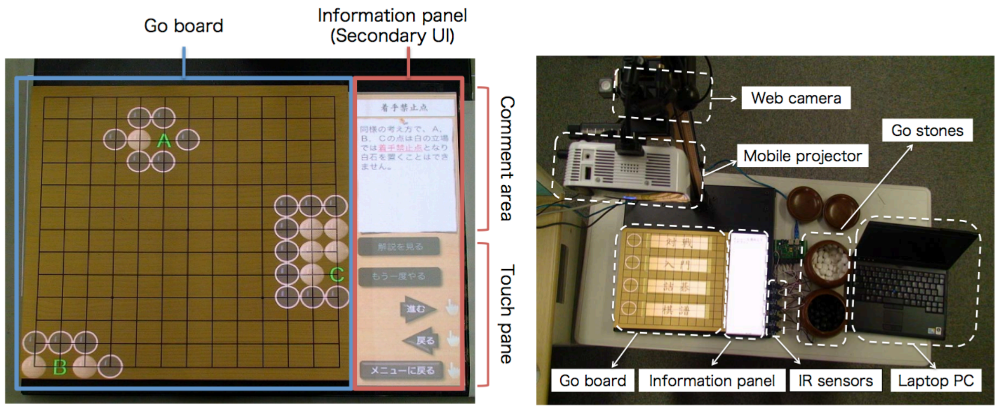


Figure 2.2 a,b, (a-left): overview of AR Go system user interface; (b-right): overview of AR Go hardware configuration.

As illustrated above, there has been a remarkable development in augmented reality interactive system. Here we limit discussion to those projection-vision based systems, since these systems share the most similarities with the design of this project. To sum up from current projection-vision based interactive research prototype systems, there are generally three kinds of hardware setup [6]. One common approach is to set a camera and projector in front of the projection surface (or above the projection surface), which is in the similar situation as the Augmented Reality Go (Figure 2.3 a). A second way is to use a cabinet, in which to mount the camera and projector to produce rear projection (Figure 2.3 b). Finally is the situation similar as the PlayAnywhere system, in which the camera and projector are placed the side of the active surface (Figure 2.3 c).

Figure 2.3:

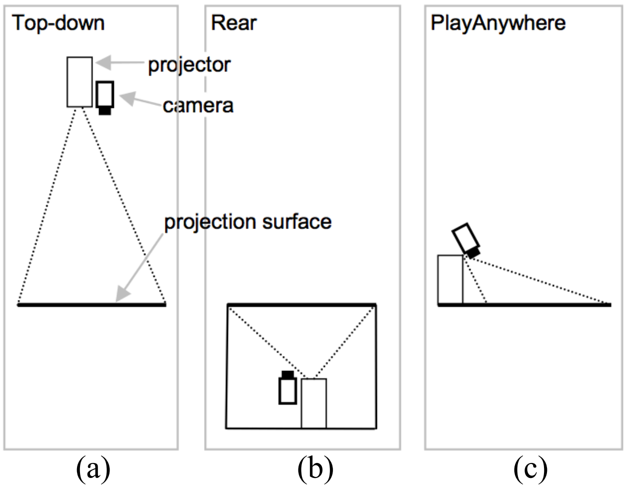


Figure 2.3. a). projector and the camera are mounted above the projection surface; b). projector and camera are mounted in a cabinet; c). projector and camera are placed at the side of the projection surface.

2.2 Related work in automatic projector calibration

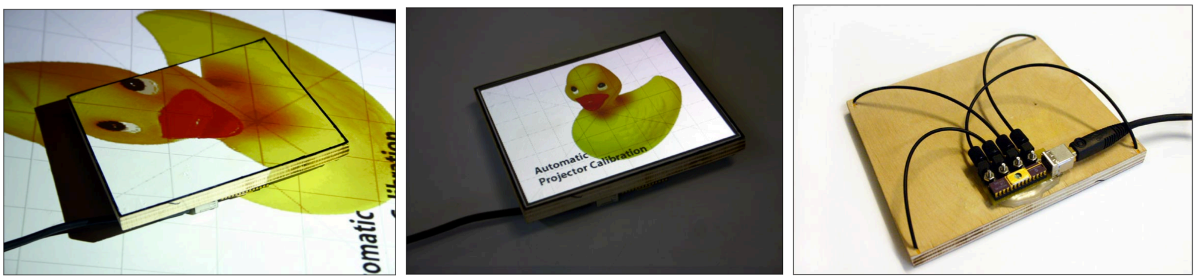
As proposed above, in this project, a projector and a camera will be used. The projector will be responsible for projecting visual feedbacks onto the chessboard. Consequently, first of all, a calibration system is required to adjust the projection image to make it automatically fit the shape of the board. Then, the project will be able to project graphic annotations onto the proper locations on the chessboard. However, as a fact of projection, alignment is required almost every time before a presentation when using the portable projectors. The process is usually done manually and thus becoming tedious and impractical to align perfectly if the projector is placed in an awkward position. Hence, the technique called auto-calibration for projectors has been induced.

2.2.1 Related work in projector calibration

Because of the great development of digital projection technology, it now has been implemented in a variety of areas as an important multimedia technique. For example, it can be used to assist presentation in work place or school, public display, and entertainment display. It has been an attractive alternative to other traditional display technologies, such as using the black/white board, optical projection. Nevertheless, several constraints still remain on the geometric relationship between the projector and the projection surface. These constraints can induce the problem of distorted, incorrectly sized image and perspective transformation. Therefore, people start to look for ways to solve this problem. The adjusting procedure here is named calibration. To make this process simpler for users, higher level of automatic is demanded. As a result of this, a great number of researches have been done to obtain a simple, robust and low-cost method for automatic projector calibration. Conventionally, for a standard projection system, a projection surface and a projector are needed. To achieve the auto-calibration, extra devices may be required. For different considerations, people may utilise different components, like multi-projectors, light sensor, tilt sensor, or camera [14, 15, 16, 17, 18, 19].

According to Lee and his team's work, an automatic high-quality calibration method with embedded light-sensors in the target planar has been created (Figure 2.4) [14]. To identify the geometric correspondence between the projector and the target projection surface, pre-determined structured light patterns are projected on the projection surface. The light sensors, which have been placed within the surface, therefore, can discover the locations of the light patterns. All the location data then will be used to pre-warp the source image from the computer connected to the projector. So that, the image can be matched with the feature locations on the projection surface. However, the drawback of this approach is that it requires special equipment conditions (needs to place several light sensors within the projection surface). This goes against the principle of this project: to make this projection can be done on arbitrary traditional chessboard (with different size etc.).

Figure 2.4:



(a) (b) (c)

Figure 2.4. a-b-c, (a-left). image before calibration; (b-middle). image after calibration; (c-right). Embedded light sensor connect optical fibers at each corner.

The similar approach has been exploited in many other applications. For instance, the game of go, which has been mentioned in previous section, it utilises a touch panel with some IR sensors to monitor the finger movements, thus projecting visual feedbacks onto the proper locations. However, in order to require as few equipment as possible, people begin to study how to use limited common equipment to calibrate the projector. Usually, equipment such as a normal webcam or another mobile projector will be adopted. Because these equipment are usually easy accessed. Consequently, this leads to a kind of calibration that is vision-based.

2.2.2 Vision-Based Camera-Projector calibration system

There are some other methods that are based on the geometric construction of a projector-screen-camera system. According to a research from Tohoku University, they have studied an auto-calibration system (Figure 2.5) [16]. This system consists of a planar screen, multiple projectors and a camera. The poses of those projectors are unknown initially. Structured light patterns are supposed to be projected on the projection screen, and meanwhile the stationary camera will take the images of patterns periodically. Then poses of such multi-beam projectors can be calculated from the camera-captured images of beam spots projected on the screen. In the next step, the geometrical relationship between the screen and the camera can be estimated and this is sometimes served as screen-to-camera homography. People sometimes may consider this process as camera calibration. Similarly, during this process, the projector-to-camera and projector-to-screen homography can be computed as well. Finally, it obtains the geometric relationships between all these three components and ultimately achieving the calibration. However, the camera-based computer vision technologies share some common constraints, which affect the resulting images. For instances, the camera resolution, reflective properties of the projection screen (planar surface), lighting conditions and the background separation. In this circumstance, it can become even more complex if calibrating onto non-planar surfaces. But this situation will not be discussed further here, because the projection surface in this project is the chessboard, which can be counted as a planar surface.

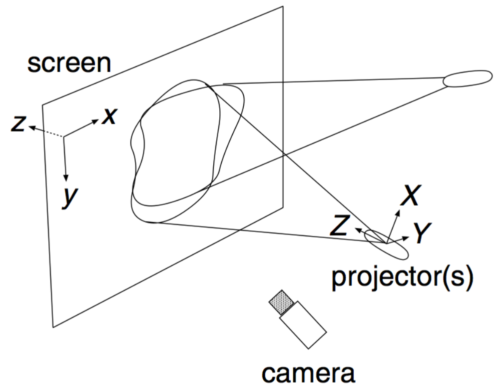


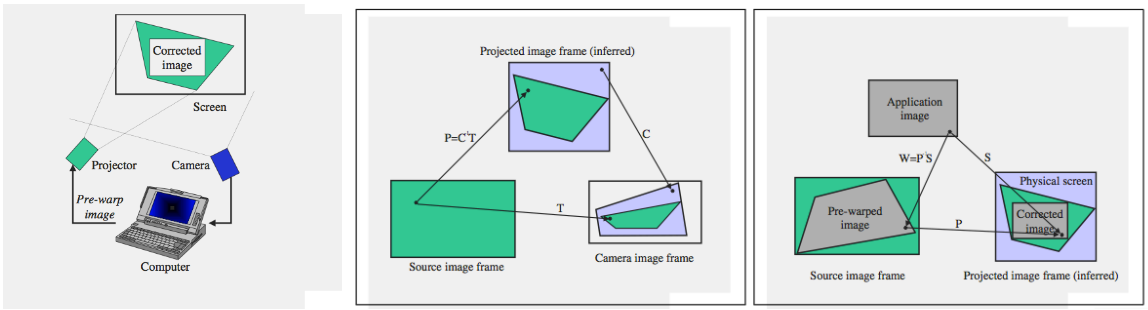
Figure 2.5. A projector-screen-camera system

This project is supposed to be user friendly. That is, users can implement the system easily without requiring for any other specific devices. A simple system of projector-board-camera will be adopted. Researches on this type of calibration system have been done in recent years. This is because high level of flexibility is often demanded. People normally prefer system, in which the equipment can be set up easily, and this will always save valuable time in special occasions, such as meetings, conferences.

As illustrated above, in the system of AR Chess, only a standard computer, a normal digital camera, and a standard projector will be included, so that a vision-based auto-calibration system is required for the projector. So far, there have been numbers of studies investigating and creating such type of system. Actually, when placing the projector, it is impractical to set it perfectly aligned to the projection surface. Therefore, it may cause distortion or perspective transformation to the resulting image. To solve this problem, researchers came up with a solution called keystone (perspective) correction to eliminate the keystoning effects (distortions or perspective transformations caused by projecting image onto a surface at an angle). According to the work from a team of Sukthankar and his colleagues [17, 19], the procedure usually contains five main steps:

1) Determine the corresponding points of the source image (computer display) in the camera image.
2) Specify the mapping between boundaries of projection image and the camera image.
3) Deduce a possible mapping from the source image to the projection image.
4) Identify a desired arrangement for the image on the projection surface.
5) Pre-warp all application images for keystoning.

Figure 2.6



(a) (b) (c)

Figure 2.6 a,b,c, (a): the apparatus of this camera-assisted presentation system; (b): procedure of mapping the source image to the projection image; (c): the process of getting pre-warping transformation W.

During this process, it is assumed that the poses and optical parameters of both projector and camera are unknown; perspective transforms can be used to model both projector and camera's optics; the projections surface is planar. The whole system is basically similar as shown in figure 2.6 a [19], but in the project of AR Chess, the location arrangement of each component maybe a slightly different from this one. In the following parts, the methodologies adopted in this project and implementation process will be explained in details.

CHAPTER 3

Theoretical Background and Methodologies

In this chapter, theoretical background and basic concepts of methodologies adopted in this project will be introduced. As we have already known, there are three pairs of unknown geometric relationships need to be revealed: chessboard and calibration; camera and projector; projector and the camera. To fulfill the calibration of projector, camera is demanded as a significant component. Because it is the only component in this project that is capable of capturing the scene of real world (chessboard). Therefore, the camera model and calibration will be explained firstly. In the following part, the concept of 'Homography' will be introduced in details. All of these concepts and algorithms were involved in the implementation of the project.

3.1 Camera model and calibration

In the case of this project, the webcam was perceived as a typical pinhole camera model. In this section, it will start with an explanation of a pinhole camera model in order to help express the basic geometry of webcam.

3.1.1 Pinhole model

Before we go to pinhole model, the basic concept of light needs to be expanded, because light contribute to all the visible scenes in the world. In the world of vision, the light can be seen as rays emanating from light source, and being reflected or refracted when hitting objects. A ray can be considered as a straight line with direction and can be expressed in the form of vector. Among all the camera models, the pinhole model is the simplest one. The lenses of the pinhole camera can cause deviations when present the scenes on image view. Usually, these deviations result in perspective transformation, and distortion to some extent. Camera calibration is adopted in order to correct those deviations. Another main purpose of using camera calibration is that it can measure the geometric relationship between itself and the real three-dimensional world. Therefore, camera calibration is often used to reconstruct the three-dimensional scene. Furthermore, the relation between camera's conventional units (pixels) and the physical units (e.g. meter) in the real world is critical. So definitions of unit should be given in advance. There are two main mathematic definitions involved in the process of camera calibration. One is called 'intrinsic matrix' and the other is called 'extrinsic matrix'. These will be explained in details later in this section.

• ***Camera model: Intrinsic Model***

In the model of pinhole camera, light is going through a tiny hole from the real world scene to the image plane within camera. The image plane is where the three-dimensional scene has been captured and presented in two-dimensional view. The size of image relative to the distance of real three-dimensional scene is determined by the 'focal length'. The 'focal length' refers to the distance from the pinhole aperture to the image plane. The following figure shows the structure of simple pinhole camera model.

Figure 3.1:

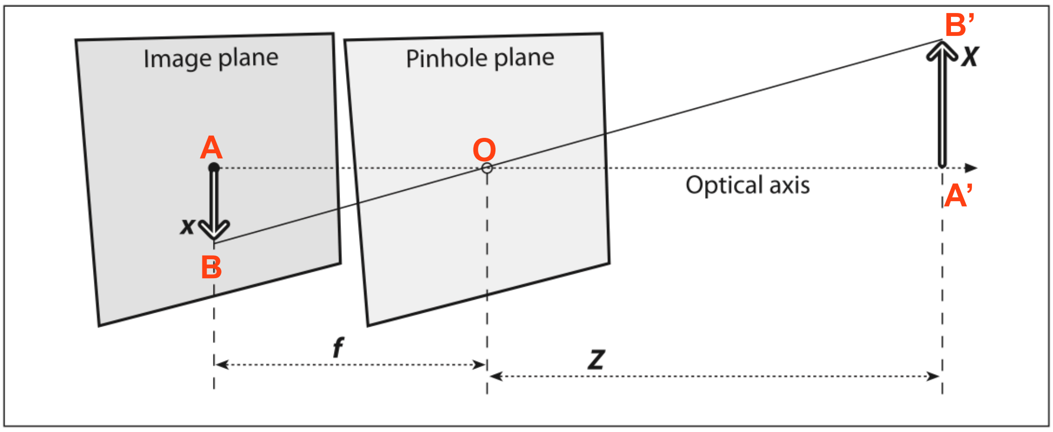


Figure 3.1. the pinhole camera model: rays (lights) go through the tiny pinhole and project on the image plane.

In the figure 3.1, 'X' can be seen as the physical height of the object in the real world. 'x' is the relative height of the projected object view on the image plane. 'f' represents the focal length of camera. 'Z' here refers to the distance from 3D object to the pinhole aperture, which is also referred as the depth of the scene. As can be seen in this figure, there is a pair of similar triangles: the triangle $\triangle ABO$ and the triangle $\triangle A'B'O$. So, as relationship between similar triangles, there is a equation that:

$$\frac{x}{f} = \frac{X}{Z}$$ Equation 3.1. Intrinsic model of camera

Accordingly, if three of these four parameters can be known, we can easily access to the last one left. Usually, in practical situation, 'x' can be got from the 2D image and it is usually in the unit of 'pixel'.

Compared with other camera models, there is a main difference between a pinhole camera model and other models. It is that the object in the 2D image view is shown as upside down. Additionally, the pinhole point 'O' is usually described as the center of projection, and the center of image plane is usually interpreted as principal point.

The next figure shows the relationship between a 3D point in the real world and its corresponding 2D point on the image plane.

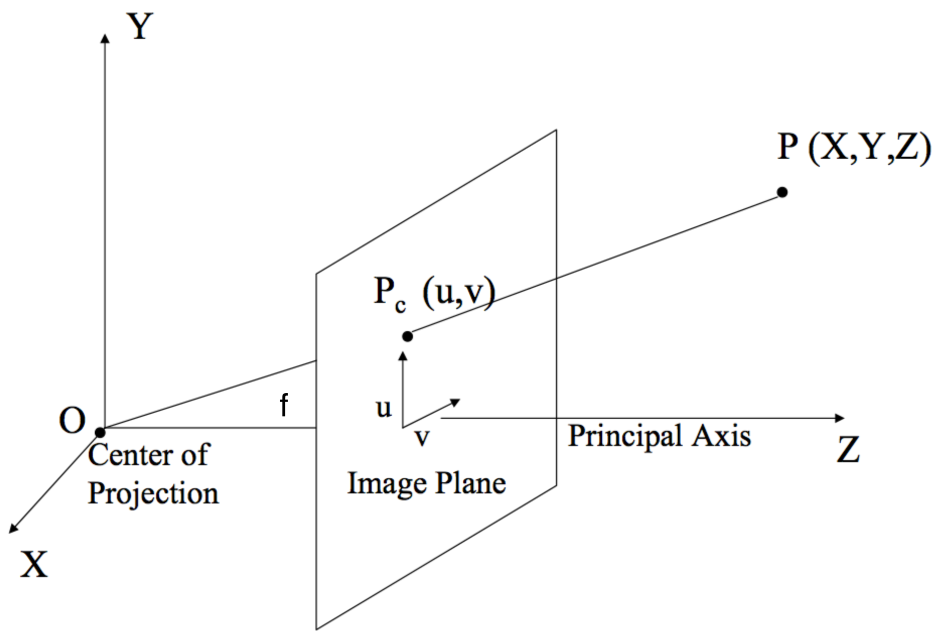


Figure 3.2. The geometric principle for a 3D scene displayed on the image plane. A point P=(X,Y,Z) in the real world and its 2D corresponding point P_c=(u,v) on the image plane.

As can be seen in the figure 3.2, the principal axis goes through both the center of projection and the principal point (center of the image view plane). There is a coordinate system in this figure, and its origin is just the center of projection 'O'. This coordinate system is known as the camera coordinate system. 'f' here is the focal length of the camera, and as mention before, it is the distance from the center of projection to the principal point. In this figure, there is a 3D object point represented as P = (X, Y, Z) and its 2D correspondence P_c = (u, v) shown on the image plane. u, v here are the pixel coordinates in the 2D coordinates system of image plane. The ray from the center of projection goes through the point P_c on the image view to the object point P in the real world. By adopting the principle of similar triangles again, there is a relation as:

$$\frac{f}{Z} = \frac{u}{X} = \frac{v}{Y}$$

Therefore, there is a transform as:

$$u = \frac{Xf_x}{Z}, v = \frac{Yf_y}{Z}$$

When using the homogeneous coordinates to express P_c, and we can get a relation between P_c and P that:

$$P_c = MP$$

$$M = \begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \qquad \text{Equation 3.2}$$

Equation 3.2. Relation between 2D point coordinates and the 3D corresponding point coordinates

By multiplying this out, we can find out a fact in this equation that w equals Z. However, in most practical cases, the center of the image plane is normally not in the origin of the image coordinates system. Thus, two parameters named C_X and C_Y have been induced to model the possible displacement of the center of the image plane. Therefore, the basic geometric relationship between 3D point P and the 2D image point P_c becomes like:

$$\begin{pmatrix} u \\ v \\ w \end{pmatrix} = \begin{pmatrix} f_x & 0 & \boxed{C_x} \\ 0 & f_y & \boxed{C_y} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} \qquad \text{Equation 3.3}$$

Where $$M = \begin{pmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{pmatrix}$$

In terms of 3D world geometric study, this relationship can be interpreted as the perspective transformation. Besides, from the computer vision perspective, the matrix M is defined as the intrinsic matrix of camera.

However, in the real world, no lens can be perfect. In practice, lens normally introduce distortions to some extent. This is why sometimes when people take photos may find out that a cubic object in the real world may become ‘spherical’ object int he 2D image view. This is conventionally known as the ‘fisheye’ effect. Generally, two main kinds of distortion are involved when calibrating camera. They are: radial distortions, which are affected by the shape of lens; tangential distortions that are usually resulted from the camera assemble process. As a consequence, during the real time calibration, parameters of distortions maybe introduced to correct the image view, reducing the deviations [24].

• ***Camera to world model: Extrinsic Matrix***

Extrinsic matrix is another significant element that is involved in the process of camera calibration. It can be divided into two main parts: rotation matrix and the translation vector.

In practice, it usually becomes easier when objects are identified in their own coordinate system, and relate each of them by calculating out the geometric relationships between their coordinate systems. Consequently, in the case of camera calibration, both of the camera coordinate system and the object coordinate system will be defined. This is shown in the following figure:

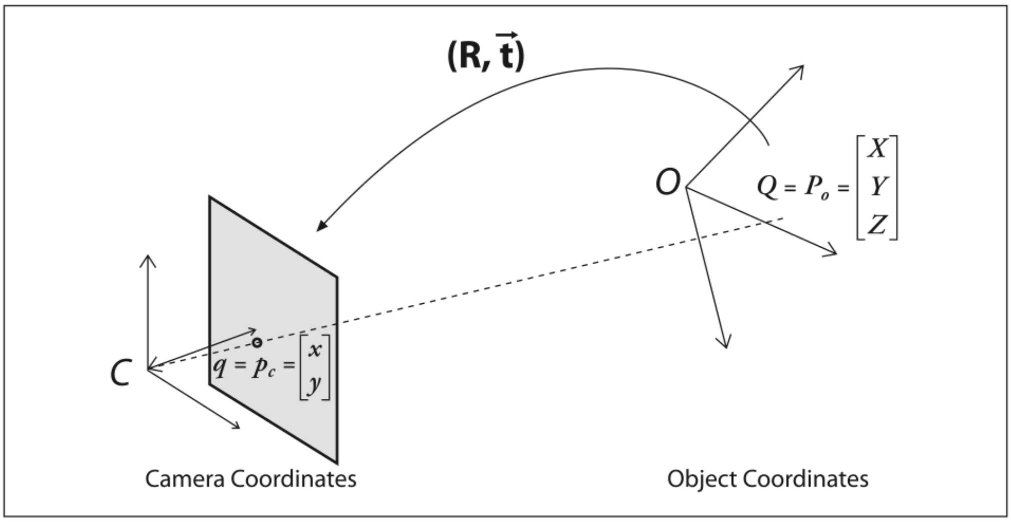


Figure 3.3. the geometric relationship between camera coordinates and the object coordinates

According to the figure 3.3, in order to convert the object coordinates system to the camera coordinates system, these two basic matrix R (rotation matrix) and $\vec{t}$ (translation vector) are needed. In this case we can define the angle around which the object coordinate system needs to rotate as θ.

Usually, the rotation will come first, so that these two coordinate systems will become parallel to each other, only left with some displacements of the origins of both coordinate systems. To align these two systems perfectly together, the translation vector $\vec{t}$ will be introduced afterwards. As a result of this, the appropriate translation vector can be simply expressed as:

$$\vec{t} = origin_{(Object)} - origin_{(Camera)}$$

After converting these two coordinates systems, the point q in the camera coordinate system can be related to the point Q in the world coordinate system as:

$$q = R(Q - \vec{t}) \qquad \text{Equation 3.4}$$

Briefly, this equation shows that the world point Q firstly translate $\vec{t}$, so that the origins of both coordinates systems can be attached together. Then the world coordinates system rotate by the matrix R, thus matching Q and q together.

3.2 Homography [16, 24]

Homography is a geometric mathematical concept and it is an invertible transformation. From the perspective of computer vision, homography is most commonly used to define the relationship between two planar surface. In other words, it is conceived as best estimation for perspective mappings between corresponding features in two planar views. It is usually presented in the form of a deformed square matrix with eight degrees of freedom, although it has nine elements actually. Therefore, it can be written as:

$$H = \begin{pmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 \end{pmatrix}$$

Equation 3.5. Homography Matrix

Equation 3.5. Homography Matrix, this is constrained by |H|=1.

Commonly, the homography matrix H can be modified and corrected by bringing in a scalar factor. Since P_9 in the H is usually initialised as 1, so the scalar factor can be multiplied with P_9. However, in real time computation, the scalar factor is usually applied to the whole homography matrix.

If identify a point in the first planar surface as (x, y), and another point in the second 2D surface as (X, Y), then the relationship between them can be indicated in homogeneous coordinates as:

$$\begin{pmatrix} wx \\ wy \\ w \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 \end{pmatrix} \begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} \qquad \text{Equation 3.6}$$

w here refers to the scalar factor that is used to refine the mapping.

It can be inferred from this equation that at least four 2D points correspondences are required to determine the value of H. The fact is that best estimation of H is developed when these four correspondences are in a least-squares sense. Sometimes the system can give more correspondences to refine H iteratively. This concept will be explained in further details in the following sections in accordance with the various situations met during the project.

3.3 Camera calibration by using chessboard [24]

In most cases of camera calibration, people tend to use a reference image, particularly some image with special pattern. This is because regular features in the pattern can be detected and recognised more easily than those irregular patterns. As a result of this, chessboard pattern is one of the most common patterns due to its connected black and white squares. It maybe easier to extract the connection points of the squares. Fortunately, in this project, an actual chessboard was used rather than only a projected chessboard pattern.

Sometimes, camera calibration relies on reconstructing 3D scene. However, a flat chessboard is much easier to deal with. Because a flat chessboard can be seen as a planar surface without depth. Assume there is a point on the chessboard is (X, Y, Z), as a result of neglecting the depth, the coordinates of this point should become (X, Y, 0) (where Z=0). We can define this point as Q. Once the camera captured an image of this view, and this point is contained in the detected image, it would has a corresponding point q in the 2D image view. q can be expressed in pixels coordinates of image plane, which is (x, y). Accordingly, homography between these two points can be interpreted as:

$$q = sHQ \qquad \text{Equation 3.7}$$

s here is a scalar factor.

The homography H within the procedure of camera calibration is mainly consists of two parts. One refers to the location and orientation transformations of object coordinates frame, the other is the camera intrinsic matrix, which results in projection of the view.

In the following figure, it shows the homography between the object plane and the image plane.

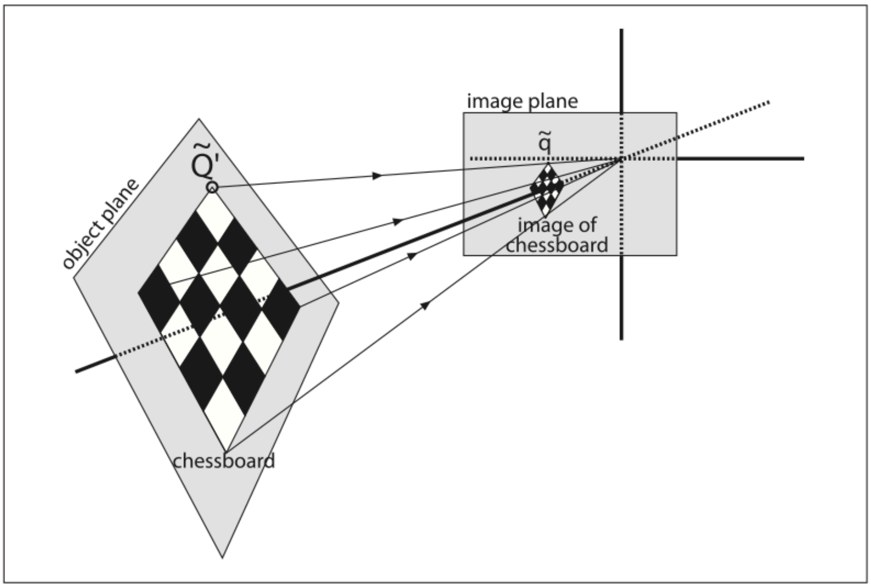


Figure 3.4. The geometric relationship between the image plane and object plane

As mentioned in previous, the physical location and orientation transformations can be counted as the combined effects of both rotation matrix R and the translation vector $\vec{t}$. Furthermore, R and $\vec{t}$ together represent the extrinsic matrix W of camera.

$$W = \begin{bmatrix} R & \vec{t} \end{bmatrix}$$

$$W = \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \quad \text{Equation 3.8}$$

In addition to this extrinsic matrix W, the intrinsic matrix M is also essential for camera calibration. M can be expressed in homogeneous coordinates as:

$$M = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix}$$

Ultimately, the homography H in the equation 3.7 can be replaced by W and M. Thus, the relationship between q and Q becomes:

$$q = sMWQ \quad \text{Equation 3.9}$$

If indicate the q and Q of equation 3.9 in homogeneous coordinates, the equation then turns to:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = sM \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \qquad \text{Equation 3.10}$$

As talked about above, when calibrating camera with chessboard, the depth of chessboard can be neglected. Therefore, the chessboard becomes a simple planar surface, which is easier for calculation. As a consequence, Z in the equation 3.10 can be considered as zero. And the equation can be written as:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = sM \begin{bmatrix} r_1 & r_2 & r_3 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix}$$

Since Z has been replaced by zero, we can eliminate r_3. Then the equation expression is as:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = sM \begin{bmatrix} r_1 & r_2 & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

It is also need to be stressed that as in the physical world, the chessboard can be measured in some physical unit, such as meters, inches. However, for the convenience of calculation for arbitrary size of chessboard, a individual square on the chessboard is defined as a unit in this project. In so doing, we can multiply the number of units by measured length of each square to get the actual size of the chessboard if needed.

In the next chapter, the implementation process of this project will be demonstrated in details, and all these methodologies adopted will be explained depending on the certain occasions.

CHAPTER 4

Implementation and Results

This chapter will mainly demonstrate the process of implementation during the whole project. The choice of programming language and library used to implement this project will be illustrated. It will also explain different attempts that have been experimented throughout the project. Issues and difficulties within the project will be described as well. Furthermore, during the calibrations of both camera and projector, the inner corners of chessboard were used as the corresponding features. The reason for doing so will be illustrated in following paragraph.

4.1 Programming Language and Library

As mentioned in the introduction, the program is implemented in C++, using OpenCV computer vision and image processing library [24]. In general, for this project, it is aimed at developing a program that can project visual feedback onto the target position on chessboard. As there are a large amount image data need to be processed, the system is required to work efficiently. C++ matches this requirement well. In addition, OpenCV has provided users with a robust library for dealing with computer vision and image processing projects. In this project, the library was utilised for camera calibration and the detection of chessboard inner corners since it contains incredible number of functions to solve these problems. For instance, one of the most commonly used function is cvCalibrateCamera2(). It requires many arguments such as object points in the three-dimensional world and image points in the two-dimensional image view. This will be expanded later in this chapter.

In fact, there are some other libraries and toolboxes can be used to calibrate the camera by adopting chessboard pattern. For example, one of the most frequently used toolbox besides OpenCV library is the matlab camera calibration toolbox. It has structured user interface for people to easily calibrate the camera as well as showing the graphs of results analysis. However, the reason for applying C++ with OpenCV instead of matlab for camera calibration in the final project is that C++ plays better at the aspect of efficiency. Normally, matlab is usually used for system simulation before programming and data analysis. Even though the matlab toolbox maybe easier to use and does not need much knowledge of programming, there is drawback emerging during the runtime. The memory usage of matlab is extremely high, which means it probably takes hundreds time of that C++ can do. Hence, in the pre-stage tests, matlab maybe the appropriate one, whereas for the final system, C++ with OpenCV library maybe a better choice.

4.2 Equipment setup

This project is not only about programming, it also includes the stage of setting up equipment and figuring out the relationships between those components. Three main components were involved in the system in addition to a computer: a webcam, which can capture images of chessboard and locate the chessboard relative to its own location; a projector, which is determined to project visual feedback onto target positions on the chessboard; a chessboard placed within the view of camera and projection area of projector. Theoretically, the webcam and projector can be mounted in flexible locations relative to the chessboard as long as their maximum views can cover the area where the chessboard is placed. However, in practice, the webcam and projector are normally mounted above the chessboard to get better view of chessboard, so as in this project.

Figure 4.1

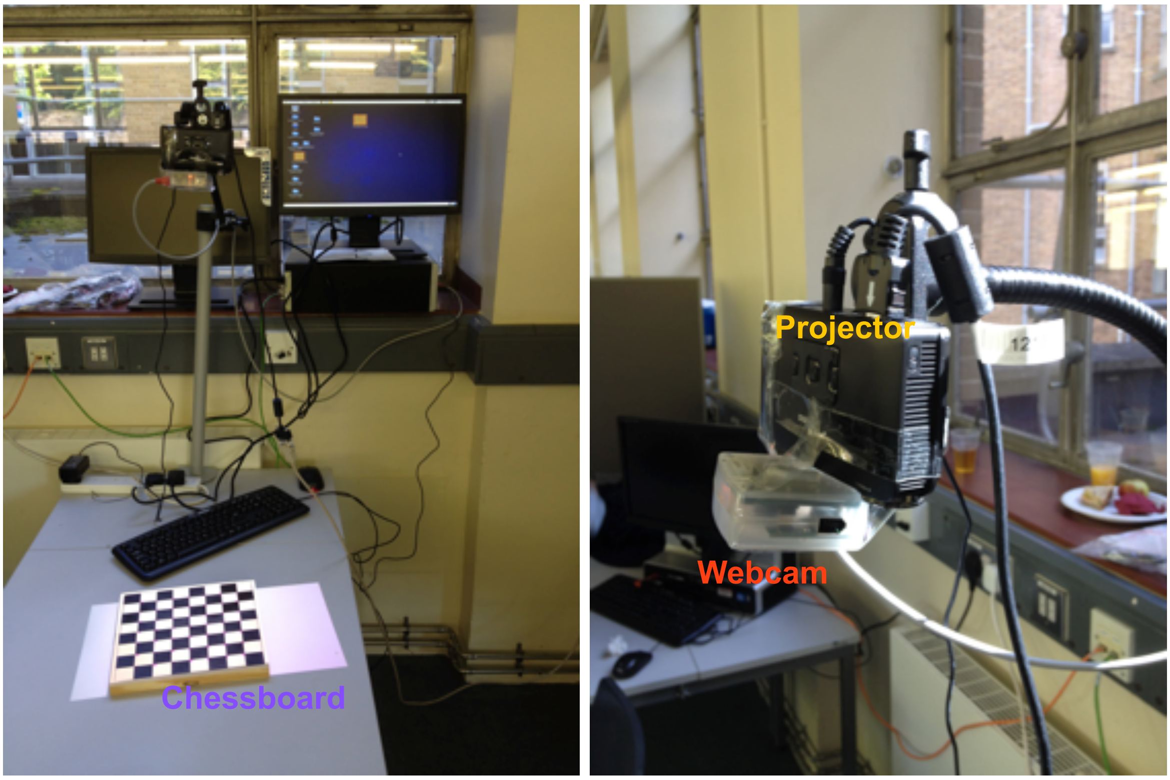


Figure 4.1. The equipment setting up in the project. There is a portable projector and a narrow lens webcam connected with computer. A chessboard is placed under these two components.

As it is shown in this figure, there is a normal narrow lens webcam bound with a portable projector above the chessboard. Both the webcam and projector have been connected to the computer on which users can extract the image data and control the projection image. There is no any extra special equipment is required, so that this system can be easily accessed and conveniently implemented in future. Nonetheless, there were some constraints and they brought difficulties when developing the system. More details discussed in more details later.

4.3 Implementation process

Within this project, there are three pairs of geometric relationships: chessboard and webcam; webcam and projector; chessboard and projector. Actually, during the project, only the images of webcam and project were involved rather than the physical equipment. Images were analysed to extract the geometric information of those equipment.

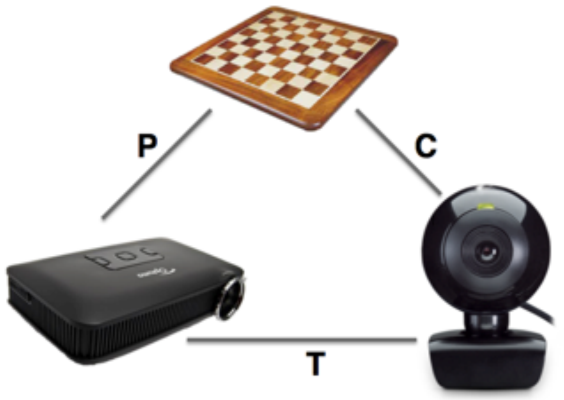


Figure 4.2. Geometric relationships between camera, projector and the chessboard in the project.

In the figure 4.2, the transformation between chessboard in the real world and images projected by projector is defined as ‘P’; the transformation between chessboard and camera captured image is defined with ‘C’; the homography between projected image and camera captured image is represented as ‘T’. Relationships between them will then be described by using these three letters.

In convention, the procedure sequence of this system should be like:

(8) The chessboard is placed on the table top firstly.

(9) Then the webcam captures the image of chessboard.

(10) Use calibration algorithms to locate the chessboard in the coordinates system of webcam.

(11) Calibrate projector by referencing to the location information of chessboard.

(12) Project visual feedback onto the proper positions of chessboard.

It needs to be clarified that the ultimate aim of this project is to simplify project graphic annotations onto the target area of the chessboard. So the final relationship that is required to be calculated out is ‘P’ in figure 4.2.

However, the question was whether it would be better if worked out ‘P’ directly, or it might be better if calculated out ‘C’ and ‘T’ firstly and then worked out ‘P’ by multiplying transformations ‘C’ and ‘T’ indirectly. Both of these two ideas have been attempted during the project, and each had its own limitations. The direct way of calculating ‘P’ was tried firstly, but not successfully in the end. Afterwards, the indirect way was used to figure out the relative geometric relationship between the chessboard and projector. In the following sections, these two trails will be demonstrated in details as well as the difficulties met during the project.

4.4 1st Trail - Reconstruct ‘P’ (referencing to figure 4.2) directly

Generally, this method is based on a research has been done by Falcao, Hurtos and Massich [25]. The key point of this projector calibration method is using the idea of ‘ray-plane intersection’ to reconstruct the three-dimensional coordinates of features on chessboard. Ideally, the whole process is following the steps:

1) Calibrate the camera to get its intrinsic matrix as well as the extrinsic matrix.
2) Use those two matrixes got from the camera calibration to calculate the equation of the projection surface (here, it refers to the flat chessboard surface).
3) Project a chessboard pattern onto the projection surface (it means the tabletop in this step).
4) Detect inner corners of the chessboard pattern from the image captured by the webcam.
5) Compute the rays (as vectors) that go through the corners of the camera image to that of the projected patter on the projection surface.
6) Apply ray-plane intersection to calculate out the three-dimensional positions of each corner on the projection surface (within the camera coordinates frame).
7) Calibrate the projector by corresponding the 2D points (the inner corners) of the projected pattern image with those 3D points calculated in the previous step.

4.4.1 Camera calibration - get intrinsic matrix and extrinsic matrix

This step is aimed at getting the intrinsic matrix and extrinsic matrix of the camera. During the programming, this step seems to be the most stable one as there are some existing functions in OpenCV library working on it.

In OpenCV library, there is a function called ‘cvFindChessboardCorners()’. It is used to determine whether the image contains a view of chessboard and extract the pixel coordinates of each inner corners of the chessboard from this image. When calling, there are several arguments need to be imported.

```
int cvFindChessboardCorners(
  const void*   image,
  CvSize        pattern_size,
  CvPoint2D32f* corners,
  int*          corner_count = NULL,
  int           flags        = CV_CALIB_CB_ADAPTIVE_THRESH
);
```

`image` refers to the 2D image of chessboard or chessboard pattern, and it must be an 8-bit grayscale image. `pattern_size` recodes the number of corners in each row and column. The third parameter points to an array in which pixel coordinates of corners are stored. `corner_count` records the total number of interior corners of the chessboard. The final parameter `flag` denotes to the behavior of this function for locating the corners. Another function is often used to refine the results of this function to subpixel accuracy. The function is `cvFindCornerSubPix()`. This function runs iteratively to reach its best estimation. Users can define the number of iteration by setting one of its arguments.

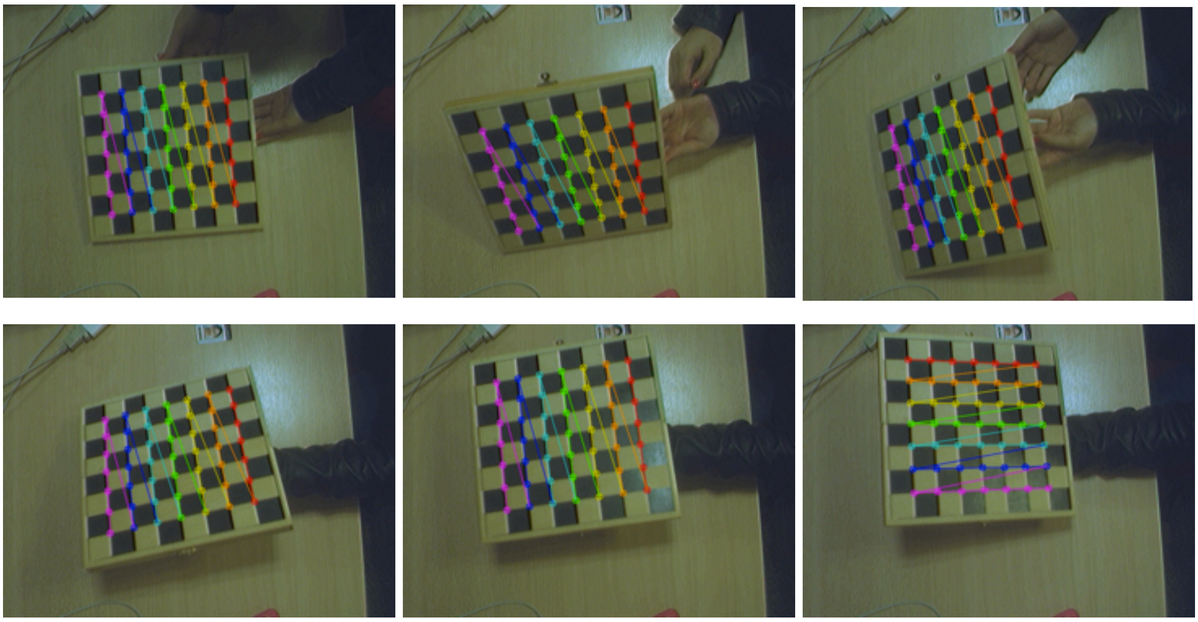


Figure 4.3. A group of camera-captured images, in which inner corners of chessboard are located and highlighted out by coloured circles. The function `cvDrawChessboardCorners()` is used to show the corners found successfully. If the detection of corners failed, then this function will return an image with only those detected corners highlighted in red circles.

After extracting all the pixel information of corners, camera calibration can be done by calling the function `cvCalibrateCamera2()`.

```
void cvCalibrateCamera2(
  CvMat*    object_points,
  CvMat*    image_points,
  int*      point_counts,
  CvSize    image_size,
  CvMat*    intrinsic_matrix,
  CvMat*    distortion_coeffs,
  CvMat*    rotation_vectors    = NULL,
  CvMat*    translation_vectors = NULL,
  int       flags               = 0
);
```

This function can help to get the intrinsic matrix and extrinsic matrix (rotation matrix and translation vector) of camera together with the distortion coefficient of its lens. It is important to keep that in mind for this function, several views of chessboard in different poses are required. It utilises those views to compute the relative locations of chessboard in different views to the camera. The first argument is the object points (interior corners on the real chessboard), however the physical unit of the chessboard here is defined by grid. Additionally, as mentioned before, during the process of camera calibration, we assumed that the depth of the chessboard is zero. Therefore, the coordinate of the object points looks like the format as (x, y, z): (0,0,0), (1,0,0), (2,0,0), (3,0,0), (4,0,0)… The argument `point_counts` works just like `corner_counts` in the function '`cvFindChessboardCorners()`'. `image_size` defines the size of image in pixel level. All views need to be in the same image size. The intrinsic matrix and distortion coefficient will be used in the following calculation.

After using this function, we apply another function to work out the relative pose of projection surface to the camera. The function is `cvFindExtrinsicCameraParams2()` with inputs: image points, object points, intrinsic matrix and distortion coefficient. Finally, it outputs the rotation matrix and translation vector separately. Since the intrinsic matrix and distortion coefficient had already been attained by `cvCalibrateCamera2()`, and the object points were simply as listed before. It was easy to work out the extrinsic matrix. Furthermore, the rotation matrix here obtained directly by this function is in the form of $[r_x, r_y, r_z]^T$. Therefore, it needs to be expanded into 3×3 matrix by using the function `cvRodrigues2()`. Here, we defined the final rotation matrix and translation vector as R and T:

$$R=\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, T=\begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Therefore, we can get the extrinsic matrix by combining these two components together. The extrinsic matrix can be expressed in the homogeneous coordinates as:

$$K_{ext}=\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

4.4.2 Compute the equation of the projection surface

Once the parameters such as the camera matrix, rotation matrix R and translation vector T had been known, it would become easier to compute the plane equation of the projection surface (the surface that the chessboard lies on). In terms of ray tracing, a plane is commonly defined by a point ‘a’ lies on it as well as a normal ‘n’ to it [26]. In this case, as talked before, the lights go through the corners on the image to that on the real chessboard were treated as rays. Therefore, the extrinsic matrix K_{ext} actually gives us the coordinates of a point ‘a’ on the plane corresponding to the origin point of the image plane. Thus, the coordinates of ‘a’ was the column of translation vector. It was also easy to infer that the third column of K_{ext} represents a normal ‘n’ to this plane.

$$K_{ext} = \begin{array}{c} \begin{array}{cccc} & & & \text{point ‘a’} \end{array} \\ \begin{bmatrix} r_{11} & r_{12} & \boxed{r_{13}} & \boxed{t_x} \\ r_{21} & r_{22} & \boxed{r_{23}} & \boxed{t_y} \\ r_{31} & r_{32} & \boxed{r_{33}} & \boxed{t_z} \\ 0 & 0 & \boxed{0} & \boxed{1} \end{bmatrix} \\ \begin{array}{cccc} & & \text{normal ‘n’} & \end{array} \end{array}$$

If a ray-plane intersection point is defined as ‘p’, then the vector from a to p is p-a, and it is also on the plane of projection surface. In a three-dimensional space, the dot product of two vectors that are perpendicular to each other equals to zero. As a result of this principle, we can get the equation that:

$$(p - a) \bullet n = 0 \qquad \text{Equation 4.1}$$

where $n = a\hat{x} + b\hat{y} + c\hat{z}$, $p = x\hat{x} + y\hat{y} + z\hat{z}$, $d = -a \bullet n$

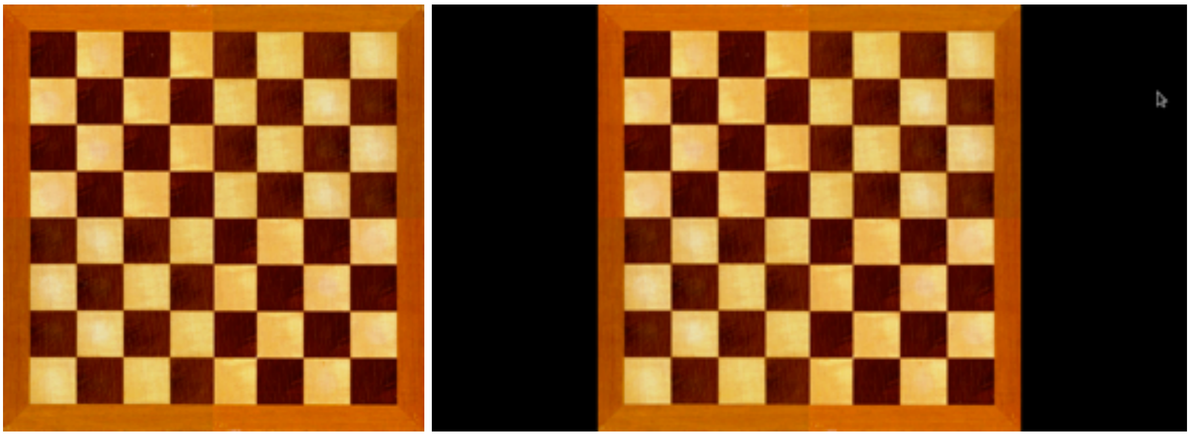
As a result, the plane equation of projection surface can be determined by:

$$ax + by + cz + d = 0 \qquad \text{Equation 4.1. Equation of projection surface plane}$$

where a, b, c and d are four known real numbers. a, b, c are not all zero. (x, y, z) is the coordinates of the ray-plane intersection point. This means all points on this plane should satisfy this equation.

4.4.3 Project a chessboard pattern onto the projection surface

In this step, a chessboard pattern (Figure 4.4 a) needs to be projected onto the projection surface in order to calibrate the projector. Once the pattern has been projected, a screenshot of the projection image should be taken (Figure 4.4 b). It is noticeable that the screenshot needs to be in full screen size, because when connecting to the monitor, the projector will project the full screen image onto the projection surface. The screenshot of the chessboard pattern will be used in later step to detect the pixel coordinates of interior corners. Meanwhile, the camera needs to take photos of the projection pattern. And corners in the camera-captured pattern images will be detected and recorded for further computations.



(a) (b)

Figure 4.4 a,b: (a). The chessboard pattern projected onto the projection surface; (b). The full screen shot on the computer monitor.

4.4.4 Detect inner corners of the chessboard pattern from the camera shots

Once camera has taken the image of the projection pattern, pixel coordinates of those inner corners need to be extracted. Here, we define the coordinates as (C_x, C_y).

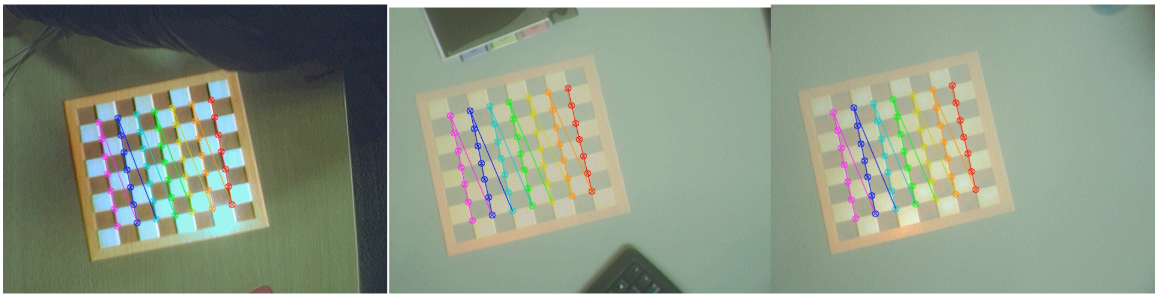


Figure 4.5: Camera-captured images of projected pattens, and successfully detected all the interior corners in the images.

4.4.5 Compute the rays (as vectors) that go through the corners of the camera image to that of the projected patter on the projection surface

In this step, since we have already got the intrinsic matrix of camera as well as the extrinsic matrix to show the relative location of the projection surface to the webcam, we can easily express the 3D rays that mentioned in previous step. The pixel coordinates of those detected corners in the camera images combined with the projective transformation can produce the equation of 3D rays. If present the points and rays in homogeneous coordinates, the equation can be written as:

$$\begin{bmatrix} sR_x \\ sR_y \\ sR_z \\ s \end{bmatrix} = \underbrace{\boxed{\begin{bmatrix} K_{\text{int}} & K_{ext} \end{bmatrix}^{-1}}}_{\text{Projective transformation}} \begin{bmatrix} C_x \\ C_y \\ 1 \end{bmatrix}$$

Equation 4.2: 3D ray vectors

's' is a scalar factor. (R_x, R_y, R_z) is the coordinates of the 3D ray vector. (C_x, C_y) represents the coordinates of corner points in the camera image. From this equation, the coordinates of the 3D rays can be calculated. Although the 's' here gives the ray a particular length, we still need to do the ray-plane intersection computation to get a scalar precisely.

4.4.6 Apply ray-plane intersection – compute the 3D positions of those interior corners on the chessboard

From previous steps, the equation of the projection surface has already been calculated as well as the coordinates of ray vectors.

Plane Equation: $ax + by + cz + d = 0$,

Ray Vector: (R_x, R_y, R_z)

The coordinates of the intersection points can be expressed by multiplying the ray vectors with a scalar, which is used to determine distance from the optical center of camera to the projection surface that a ray needs to travel. Here, we defined the scalar as 's'. Furthermore, as the intersection points are supposed on the projection surface, so the expressions of those points need to satisfy the equation of the projection surface. Therefore, we can substitute point coordinates expression in the plane equation as:

$$a(sR_x) + b(sR_y) + c(sR_z) + d = 0$$

Thus, s can be calculated and finally the 3D position of corners being projected can be obtained.

4.4.7 Calibrate the projector by corresponding the 2D points (the inner corners) of the projected pattern image with those 3D points.

In this step, the first thing needs to be finished is detecting the pixel coordinates of corners in the projected pattern. From previous work, a screenshot of the projected pattern has been taken. So, the detection of those 2D points is simply done by using OpenCV functions to extract the corners from that screenshot (referencing to the figure 4.4 b). Furthermore, the 3D position of corners on the projection surface have been obtained, thus the correspondence between those 2D points and the 3D points can be easily reconstructed. However, although matches can be found between those points, the transformation matrix was hard to calculate. Because of the inconsistent of the dimensional expressions, homography is no longer suitable for this resolving this problem. Therefore, this 'direct' method for computing 'P' (referencing to the figure 4.2) ended up in failure. But if this transformation can be resolved in the future, then this method should work.

4.5 2nd Trail – Reconstruct 'P' (referencing to figure 4.2) Indirectly

Generally, this approach is based on a research about developing a 'smart presentations system' [16]. The key point of this projector calibration method relies on the assumption that all the entities here are considered as a planar surface. That is, the chessboard adopted here is no longer an object in the 3D world coordinates system; it will only be regarded as a planar surface similar to an image plane. The coordinates of its corners then become in the physical unit of grid: (0,0), (1,0), (2,0), (3,0)… Ideally, the whole process can be divided into following stages:

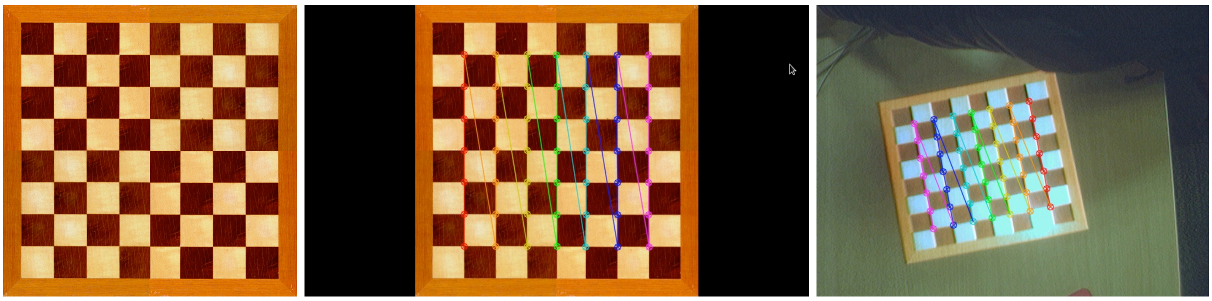
1) Camera calibration – get the intrinsic matrix of camera
2) Project chessboard pattern onto the projection surface, detecting corners on both camera image and projector image
3) Compute homography T between the projector image and camera detect image
4) Put the chessboard onto the surface and calibrate camera to get the extrinsic matrix - solve the geometric relationship of C (referencing to figure 4.2).
5) Use T and C to solve the geometric relationship (P) between projector and the chessboard surface.

4.5.1 Get the intrinsic matrix of camera

This stage is just similar to the first step in the previous method. Different views of chessboard need to be taken for analysing. `cvFindChessboardCorners()` and `cvFindCornerSubPix()` are utilised to extract the information of corners' coordinates from the camera images. Then, `cvCalibrateCamera2()` is invoked to get the intrinsic matrix of camera, as well as extrinsic matrix in each view. However, only the intrinsic matrix of camera would be recorded since it was consistent for a camera with a fixed focal length. We can define the intrinsic matrix of camera here as K_{int}. Basically, this step is similar to the first step in previous approach demonstrated above.

4.5.2 Project chessboard pattern – detecting corners

After calibrating the camera, we need to calibrate the projector. In order to build up the maps between camera and projector, a chessboard pattern is projected onto the projection surface (the same table top that the chessboard placed on). The pixel coordinates of corners in the projected image (on the monitor connecting to the projector) can be worked out effortlessly, so as that in the image captured by the webcam. Similar to the first trial method, the procedure of detecting the interior corners in the projected image should be taken on the full screen shot, shown as following:



(a) **(b)** **(c)**

Figure 4.6 abc: (a).Chessboard pattern being projected; (b). Full screen shot of the projected pattern on the monitor, and inner corners within this pattern has been detected; (c). Camera-captured image of the projection image ont he table surface.

Arrays of pixel coordinates of corners in both projection image and camera-captured image are stored for further usage.

4.5.3 Compute homography T between the projector image and camera detect image

As mentioned before, homography is commonly used to express the geometric relationship between two planar surfaces in computer vision. In this case, two digital image views are considered as two planar surfaces. One is the camera-captured image, the other is the projected image. The corresponding features of these two images are those interior corners of chessboard pattern, because they share the same pattern image. Since we have already obtained the pixel coordinates of corners in both camera detect image and projected image, T can be computed directly. The OpenCV function `cvFindHomography()` is used to take those correspondences and output the best estimation of homography T between those corresponding features iteratively:

```
void cvFindHomography(
    const CvMat* src_points,
    const CvMat* dst_points,
    CvMat*       homography
);
```

It needs to be noticed that the homography matrix is a 3×3 matrix and invertible. The relation between these three arguments can be presented as:

$$\text{dst\_points} = \text{homography} \cdot \text{src\_points}$$

Therefore, if using the obtained homography T to compute the `src_points`, the homography needs to be inverted before multiplying with the known `dst_points`. So far, we only need to record the homography matrix T.

4.5.4 Put the chessboard onto the surface and calibrate camera to get the extrinsic matrix - solve the geometric relationship C

After all those preparations, the chessboard used for the game is finally put on the table. Thereafter, the relative location of this chessboard to the camera needs to be calculated now. Then, combined with the intrinsic matrix, the transformation matrix between the chessboard surface and the camera image view can be done by a simple multiplication.

In the first step, the intrinsic matrix K_{int} of camera has already been worked out as well as the distortion coefficient. These can be used as inputs in the OpenCV camera calibration function `cvFindExtrinsicCameraParams2()`, and the function will give back both rotation matrix and translation vector. Consequently, the extrinsic matrix K_{ext} is easily obtained (similar to the calculation of K_{ext} demonstrated in section 4.4.1).

The calculation of geometric relationship C between the camera image view and the chessboard surface is simply interpreted as:

$$C = K_{\text{int}} K_{ext}$$

Alternatively, C can also be calculated by computing the homography between chessboard planar surface and the camera image directly if calling the function `cvFindHomography()`. Only the homography here has two parts: the physical relocation of the chessboard relative to the location of camera; the projection transformation caused by camera intrinsic properties.

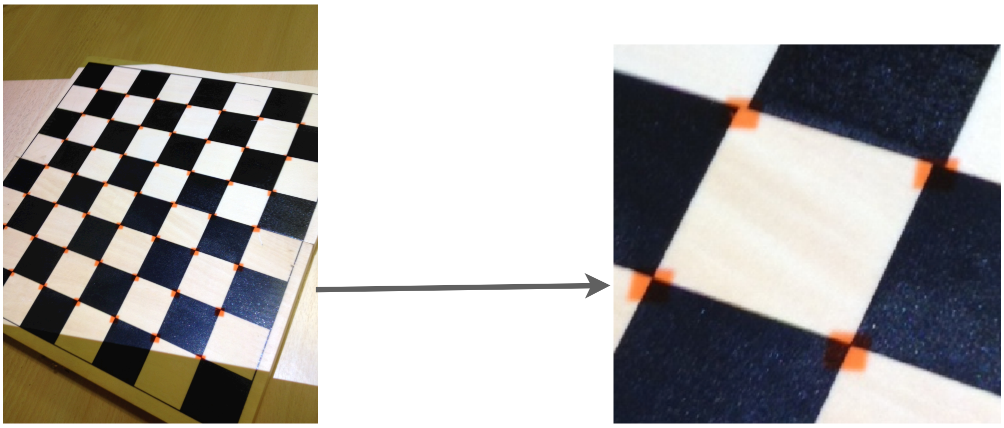
4.5.5 Using T and C to solve the geometric relationship (P) between projector and the chessboard surface

Since T and C have been worked out, it becomes possible to reconstruct the geometric relationship P between projector and chessboard indirectly. Theoretically, it can be done by multiplying the coordinates of chessboard corners with C and T, sometimes needs to invert the matrix. If the pixel coordinates of features (those graphic annotations match the interior corners of chessboard) in projection image used to give the visual feedbacks is defined as '`p_points`', and 2D coordinates of corners (as (0,0), (1,0), (2,0), (3,0)…described before) on the chessboard are represented as '`c_points`', then the relationship between these two entities can be interpreted as:

$$\text{p\_points} = T \bullet C \bullet \text{c\_points}$$

In doing so, those corresponding feature pixels on the projected image can be computed. To see the accuracy of the results, small coloured squares with edge length of 11 pixels were shown centered on the feature points in this project (Figure 4.7).

Figure 4.7.



4.6 Refine 2nd Trail – Make it simpler

So far, two trails have been demonstrated. The first approach failed at last, while the second worked somehow. However, the second approach was still found can be improved. Theoretically, it can be simplified to some extent. Look back to the final equation of the second method:

$$p\_points = T \bullet C \bullet c\_points$$

Where p_points refers to the corresponding features in the projected image (on the monitor); c_points is an array of coordinates of those interior corners of the real chessboard (in the form of (0,0), (1,0), (2,0), (3,0)…); T is the homography matrix between camera image and projected image; C is the projection transformation between the chessboard and webcam. (T and C can be referred to the figure 4.2)

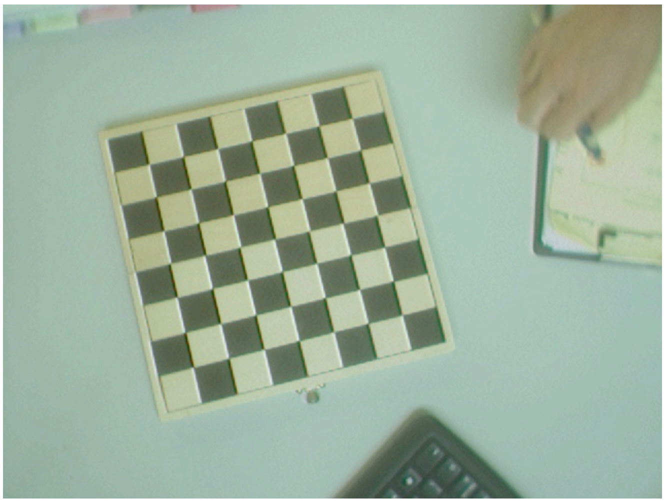


Figure 4.8. Camera captured image of chessboard used during the game.

The sequence of this calculation starts with C•c_points, and the results of this are supposed to be the corresponding features in the camera image. In this project, the results should be the coordinates of corresponding locations of interior corners in the camera image view. Nonetheless, because the calculation of C is just an estimation of the transformation, the accuracy is still constrained. Hence, if the inference is correct, it maybe better to use the camera-captured image directly. Then, we only need to extract the coordinates information of corners

The figure 4.8 shows the camera-captured image of the chessboard that has been put for game.

Finally, the coordinates information of camera image can be used to get their correspondences in the projected image by multiplying with T. Small coloured squares with edge length of 11 pixels are projected onto the chessboard to show the accuracy of calculation.

4.7 Result and Assessment

The projection image results for different chessboard poses of the final system are shown in figure 4.9. The final projection results are presented in figure 4.10. The displacement from the center of the small square to its corresponding interior corner on the chessboard represents the calculation error of this projection system.

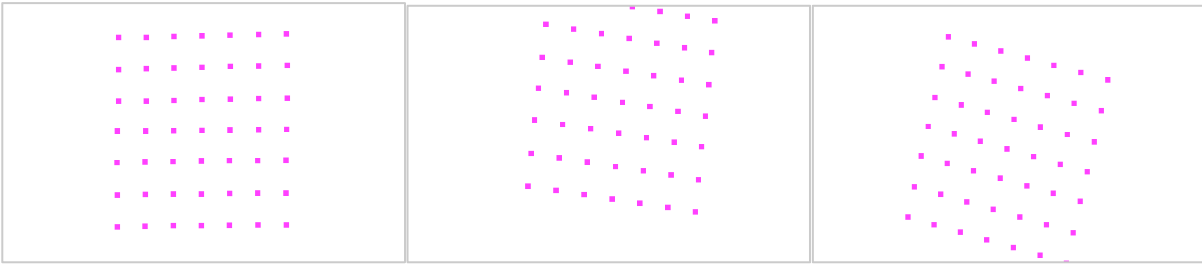


Figure 4.9. The projection image results for different chessboard poses

The system worked successfully most of the time, depending on the accuracy of the estimation of homography T between the camera image and projected image. However, if there was high level of distortion (especially the radial distortion) of the camera lens or projector lens, the result might be not as good as current result. This is due to the neglect of distortion coefficient, since homography between two planar surfaces is better for estimation of perspective transformation.

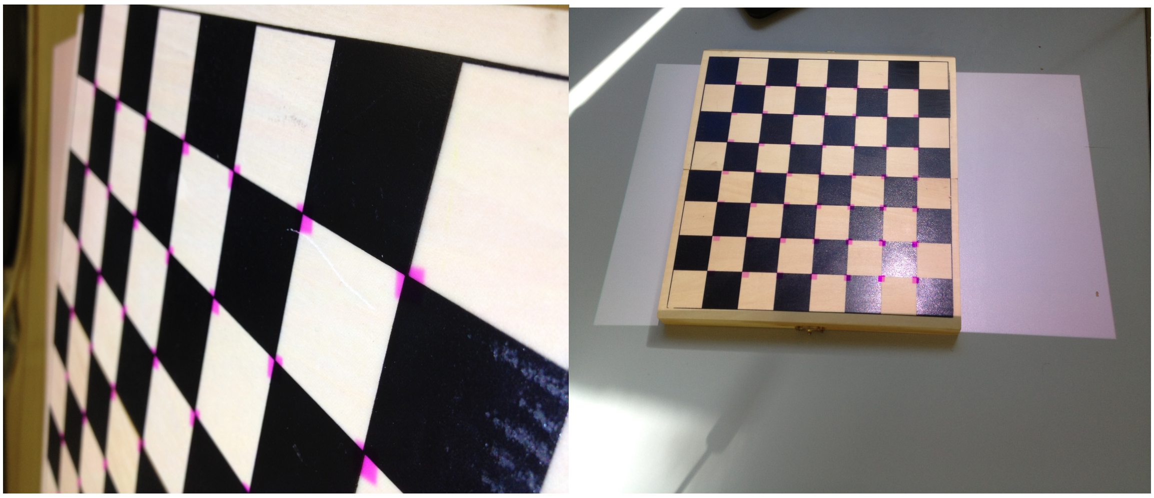


Figure 4.10. The final showoff of the projection results

Furthermore, different colors were tried to determine which color maybe better seen on the chessboard. Since the chessboard contains both black squares and white squares, the color should be suitable for both of these colors. The white squares were much easier for various colors to show off, whereas the black squares were much more difficult for those colors to show. Among all the experimented colors, the warm colors such as red (RGB:(255,0,0)) and fuchsia (RGB:(255,0,255)) are best shown colors. In contrast, colors such as blue and green are much more difficult to be seen, especially against the darker background color.

More problems and constraints have been realised during the project and they will be discussed in the next chapter in details.

CHAPTER 5

Evaluation and Constraints

In this chapter, the evaluation of the whole project from both the aspect of user friendly and the aspect of system efficiency will be illustrated. Then, constraints found during programming as well as that of the equipment will also be enumerated with examples. This information maybe helpful for possible improvements that can be done later.

5.1 Whole project evaluation

The whole project pursues a relative high level of automation. That is, the system should be implemented in the way that is user friendly. It aims at making a system that for most of the time users only need to focus on the game itself rather than manipulating the equipment. Hinske et al. [3] once proposed a guideline to design this kind of interactive system. According to the guideline, “the game should still be playable (in the “traditional” way) even if technology is switched off or not working” and “the focus should remain on the game and the interaction itself, not on the technology”. Therefore, all the other system setting up tasks will be better done by the system automatically. In the real time, for the purpose of being user friendly, the system required limited extra equipment except the traditional chessboard game package. Therefore, only one portable projector and a common webcam were adopted in this system, while normally more projectors or webcam would be used in other computer vision calibration tasks. Meanwhile, once the game starts, the poses of webcam and projector would normally keep unchanged, unless for some special occasion. Due to these reasons, there were constraints in the choice of the approaches for reconstructing the three-dimensional scene. Thus, bringing difficulties in building up the geometric relationships between the chessboard, webcam and the projector during the project. It took more time to find the better way to reconstruct the transformations between these three main components: chessboard, webcam and the projector (figure 4.2). Different approaches were tried to find out which idea was better for mapping features of real chessboard to pixel coordinates in the projected image (refer to chapter 4). However, fortunately, the whole project worked well, and the results showed that the final approach is potent and efficient. But there are still some possible improvements can be done in future, and these will be discussed in next chapter.

5.2 Problems during Programming

As talked about, in this project, because of the huge amount of image data processing, C++ were chosen as the programming language due to its high efficiency in memory usage. OpenCV computer vision and image processing library was utilised to deal with camera calibration and the detection of chessboard inner corners since it has numbers of functions to solve these problems. Those existing functions had benefited this project to a large extent. They can be implemented easily and provide high performance.

Nevertheless, although the functions have these advantages, some problems may still be arisen during the implementation process. For example, the corner detection function `cvFindChessboardCorners()`:

```
int cvFindChessboardCorners(
  const void*    image,
  CvSize         pattern_size,
  CvPoint2D32f* corners,
  int*           corner_count = NULL,
  int            flags        = CV_CALIB_CB_ADAPTIVE_THRESH
);
```

It worked successfully most of time, but still had its own limitations in locating the corners. Same problems happened when calling the conner detection function `cvFindCornerSubPix()`. According to the official description of this function, it requires some white squares or background around the chessboard to make this function more robust. This is because, basically, the detection of internal corners on chessboard is based on the point detection, where those black squares connect with each other. Therefore, if the background was too dark, it might affect the segmentation of those outer black squares, and can lead to a failure in the following detection of the interior corners.

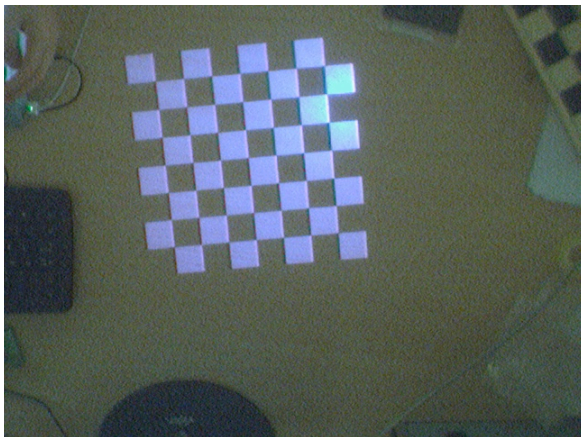


Figure 5.1. The background is too dark to segment from the outer lack squares of the chessboard pattern.

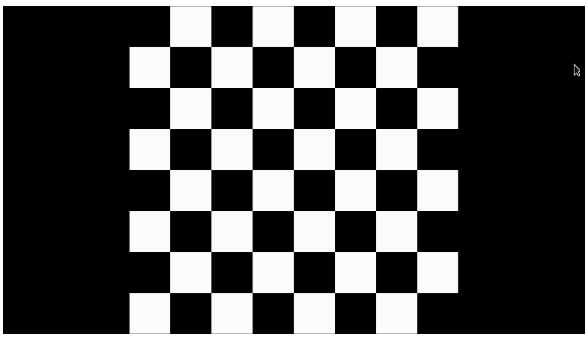
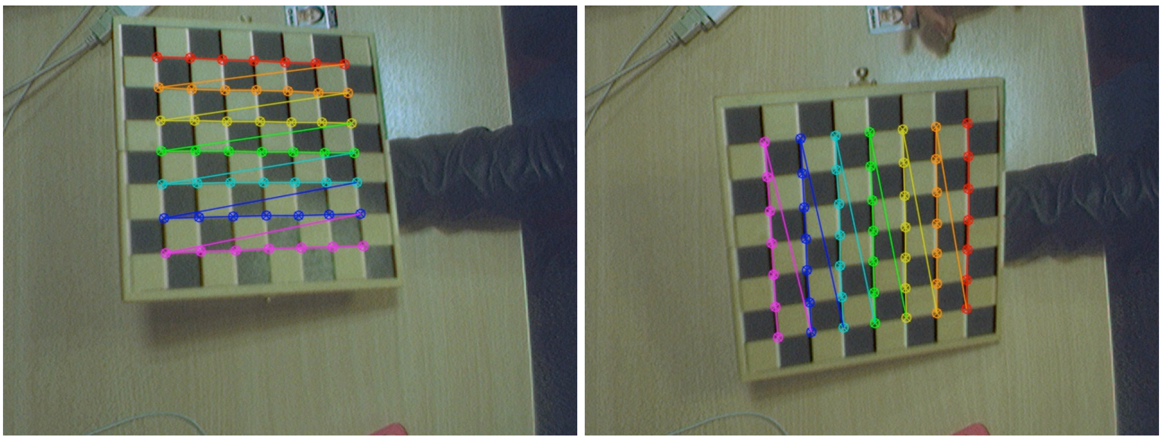


Figure 5.2. The surrounding color is as same as the outer black squares, detection failed.

These images show the situations when the background was too dark to segment it from the outer black squares on the chessboard. This usually happened when projected a black and white chessboard pattern onto the projection surface (figure 5.1). When projecting, the pattern image on the monitor would be shown in full screen view, if the image was not as large as the full screen size of the monitor, the rest of the screen view would be presented in black (figure 5.2). As a consequent of this, the chessboard pattern showed on the projection surface would be surrounded by dark background. Thus, it may cause failures in detecting corners.

There was another problem occurred during the corner detection. The difference was that this problem happened sometimes after the internal corners had been detected. Generally, after the corners had been detected successfully, the coordinates information of those corners would be recorded within some arrays in certain orders (Figure 5.3 a and b). The order in which those corners had been stored was vital in this project, because it related to the mapping of correspondences in two images. If the corresponding corners in two images were stored in different orders, then the geometric relationships found between them would be seriously wrong.



(a) (b)

Figure 5.3 a,b: (a). The corners are detected in a horizontal order; (b). Corners are detected in a vertical order.

This problem might induce another series of difficulties when mapping features in two images. In fact, if the function worked in a fixed way in which the detection order could be predicted with certain conditions, there might be some solutions to this problem. However, the results showed that the detection order varied randomly. As a result of this, it is necessary to find out some solutions, or the mapping of features in following steps could all go wrong. Fortunately, there is a function in the OpenCV library used to show the successfully detected corners in certain orders called `cvDrawChessboardCorners()`. The highlighted corners in figure 5.3 a and b were done by using `cvDrawChessboardCorners()`.

This function attempts to draw corners detected as circles with different colors for different rows or columns if all internal corners on the chessboard have been detected. By using this function, the corner detection orders can be checked, and the problem illustrated above can be realised in time and solved even though it may take some time.

5.3 Constraints

Although from the final results, we can tell that the system worked well and all the current equipment were suitable for such a projector calibration system, there were still some parts could be improved. Actually, during the whole project, besides the perspective of software development, there were also some constraints arisen from the equipment and surrounding environment conditions. Due to some reasons, such as for better corner detection or for better presentation, there were some requirements to the equipment and the environment conditions. This subsection will talk about these constraints in two categories. Firstly, the constraints of equipment will be illustrated. Then some typical environment constraints will be demonstrated.

5.3.1 Equipment Constraints

In this project, as presented before, there were three main components: a chessboard, a webcam and a projector. For these three components, the chessboard has almost no constraints as far as it is in the common style of black and white squares. In contrast, for the other two components, they each has own various constraints. Those of most important constraints and can affect the results of this project in a large extent will be described here.

- ***Image Quality***

  As a fact of this project, it contained a large amount of computer vision and image processing dealing methods. Thereafter, the quality of images then becomes particularly important. This attribute is one of the most significant factors that could affect the results of the project seriously, especially during detecting the chessboard corners – poor quality of image may result in worse detection. When people talk about image quality, they usually think about the image resolution of a computer image shown on the screen. In addition to image resolution, the quality of image is also affected by the focal length setting of the equipment. In this project, there are three different components need to be taken into consideration: the webcam, the projector and the computer.

  Firstly, in terms of the computer, the resolution of image matters especially when the chessboard pattern needs to be projected in this system. If the resolution of the monitor were low, the quality of image presented would be limited. For instance, the first monitor used to connect with the projector during the project had its highest resolution of 848*680. Therefore, even though the pattern image that had been chosen was in higher resolution than this, it would be limited by the resolution of the monitor. This had been proved after monitor with higher resolution was used and other conditions kept the same, then the detection process seemed to work better.

Besides the computer, webcam also plays an important role in this system. Within this project, webcam was involved in numbers of steps. It firstly used when images of chessboard in different poses needed to be captured for camera calibration. In this step, if the resolution or the focal length setting of the webcam were poor, the quality of images taken would be restricted. Thus, it can influence the performance of functions used to detect the internal corner points of chessboard image.

After this step, the webcam was used for projector calibration later for capturing images of projected chessboard pattern shown on the projection surface. This was similar to the previous step, in which poor resolution or focal length setting would lead to poor detection of the corners. The situation could be worse if the quality of image projected had already been in poor conditions. Then the detection errors could be doubled.

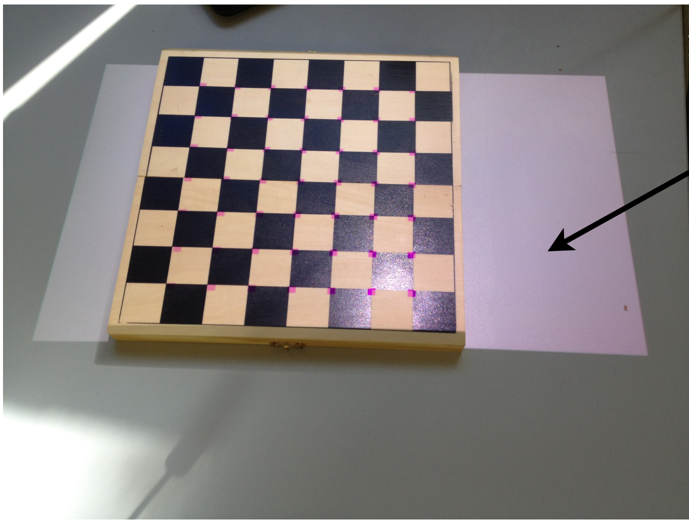
Finally, the quality of projector is as vital as those two components discussed above. Here, we mainly refer to the resolution of the projector. Once the resolution of the projector has been constrained, the quality of projection image might be restricted. This is just similar to monitor. However, most projectors are designed to present large image view, rather than present graphic annotations within a small area. Hence, in fact, most of projectors people used now are with appropriate resolutions.

- ***Projector Brightness***

  The brightness of projector is a significant attribute to be reviewed when choosing a projector. Because the brightness of projector can affect the quality of image projected directly. If the image was not shown bright enough, it might be hard to segment the image from the background. This could be a serious problem to this project, since the projector needs to project a chessboard pattern to the projection surface (table) and then inner corners on this pattern needs to be detected. To segment the chessboard pattern projected from the background was therefore compulsory and particularly vital. The brighter projected pattern would have higher level of contrast against other objects within the scene, and it could contribute to a higher possibility of successful detection.

- ***Maximum Viewed and Projection Area***

  There are another two issues need to be noticed. This is the maximum area in which projector can project image onto it and the maximum area where the camera view can cover. Usually, the higher these two components have been mounted, the larger their projection and view area are. Consequently, when mounting them above the chessboard, they were mounted as higher as possible on the stander. It seemed had less problem with the camera view area, the webcam commonly was able to cover the area that was needed. The chessboard therefore could always be in view. In contrast, as figure? shows, the projector mounted at a certain height would have limited projection area. As can be seen in this figure, the bright area (projection area) can just fit the size of the chessboard. If changing the chessboard to a bigger size, then the projection area will no longer be able to cover it. As a result of this, a larger projection area should be required.



The maximum projection area is limited. If the chessboard was bigger, then the projection area might not be able to cover the whole board.

Figure 5.4: Limited projection area

- ***Others***

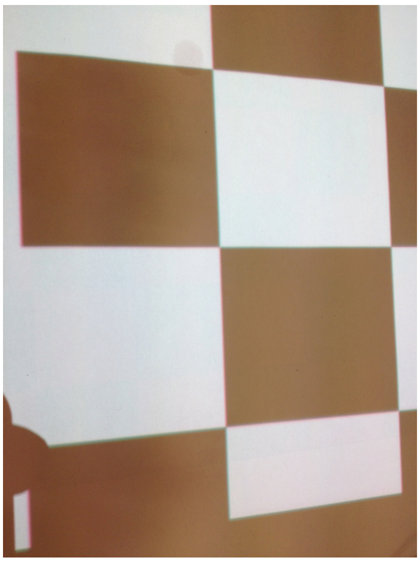


Figure 5.5: Colored borders around the squares. Maybe not so obvious.

In addition to all constraints talked about above, there maybe some other constraints that do not matter so much to this project. For instance, when projecting chessboard pattern onto the projection surface, because of some unknown optical problem or digital color presentation problem, the squares within the pattern seem to have some color borders just as shown in figure 5.5 . This may influence the corner detection due to the different colors in the connection point of squares. Furthermore, when the projection area was limited, if there was an equipment that can make the projector head rotate to a free degree, then the problem could also be solved.

5.3.2 Environment Constraints

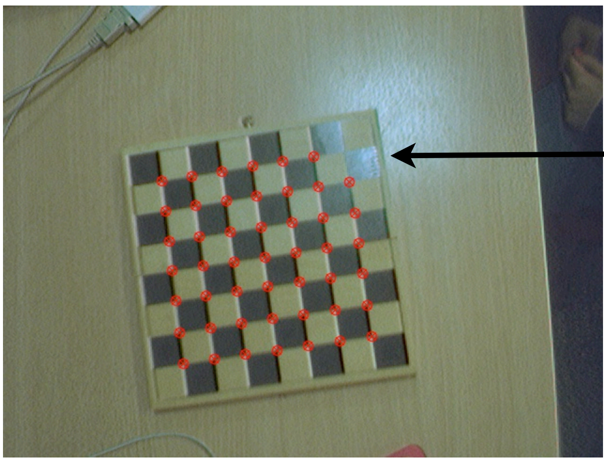
Except for those equipment constraints, some environment factors may also affect the performance of this auto-calibration system. Normally, these environment factors affect the system by influencing the performance of the equipment. In the following part, some examples will be illustrated.

- ***Light Condition***

  Since during this project, the projector was used frequently and it was one of the key components. However, the quality of projection image presented could be influenced by the room light condition in a large extent. If the room light were too bright, then the contrast between the projection image and the projection surface background would be weakened. This may result in errors when segmenting the projection pattern from the other background colors. Thus influencing corner detection during the project. Furthermore, it would influence the graphic annotations that projected at last too. Besides the projector, bright light could also affect the images taken by the webcam. Those images would be presented in brighter condition, and then influence the corner detection effect in the project.

- ***Others***

  There maybe many other external factors may have an effect on the project. Similar to the room light condition, the light reflection of the projection surface can also has an impact on the projection quality. Accordingly, the material and the smoothness of the projection surface may influence the projection results.



The bright light spot cause strong reflection on the chessboard surface, which affect eh corner detection results.

Figure 5.6: Failed to detect the corners due to some bright reflection on the board.

CHAPTER 6

Experiments and Future Directions

For the purpose of this project, the system should be able to work under flexible external conditions and it should be able to adjust itself to any incident. Possible experiments were designed to test the robustness of the current system and investigate some valuable further improvements. In this chapter, it will firstly demonstrate some experiments designed to test the flexibility and robustness of this system. Then it will give some more suggestions about this system that worth further investigating.

6.1 Experiments

Review the working results got so far, the system has shown its solid ability to calibrate the projector by only given a webcam and a projector connected with computer. In this section, more detail experiments will be implemented to test if the system is capable to adjust itself to different incidents, such as moving part of chessboard outside the projection area, changing room light condition and changing projection brightness.

6.1.1 Chessboard movement

In real situation, when people play the chess game, chessboard maybe moved during the game accidentally. Sometimes, it maybe moved in the position, which may not be aligned with the orientations of projector and webcam. The system then should be able to deal with this; it should be able to detect the chessboard as long as the board is within the camera view and projection area. An experiment was conducted to test whether the system was still able to work properly when the part of the chessboard had been moved outside the projection area. The result is positive. In this experiment, small colored squares were shown on the corresponding inner corners of chessboard, unless those were outside the projection area. The result proved that the system could operate properly even when some features (corners) of the view had lost.

6.1.2 Changing room light condition

As mentioned in previous chapter, room light condition can restrict the performance of the system sometimes, especially when projecting some graphic annotations onto the projection surface. The bright light will make the colors in projection image less contrast, thus reducing the quality of projection annotations. In this experiment, the room light condition was tested as a changing parameter. However, this was not tested for the projection effect directly, it was mainly tested for the effect that light condition can have on the corner detection process. During the experiment, all other parameters such as projection brightness and camera resolution were kept the same, and three light conditions were used: dark, normal and bright. It showed that the system was able to work under almost all these three room light conditions.

6.1.3 Projection brightness

Similar to the light condition, projection brightness has also been counted as a condition that can limit the performance of the system. During the project, experiment was also executed on this condition to test whether different projection brightness would influence the accuracy of corner detection. For the projector used in this project, it can be adjusted to three different level of brightness: dark, normal, and bright. Basically, the experiments were still some image processing tests. Different conditions contributed to different image qualities. The experiment results show that the system normally worked better when the projection brightness level was set to 'bright level', which resulted in big contrast in colors. The edges between colors in the image then became clearer, thus making the segmentation process of the image effortlessly. Nonetheless, there was one exception, if the room light was dark, then the reflection of the bright projection light would become strong. Consequently, the strong reflection light would influence the image captured by the webcam since there would be an extreme bright area in the image. This would obviously affect the detection of inner corners in the image. But in normal conditions, the system should be able to work efficiently and project image onto the proper corresponding positions of the chessboard.

6.2 Future Directions

To sum up, the auto projector calibration system developed so far can work stably, and shows relative high accuracy for matching corresponding features of projector image view as well as the real chessboard. Although, the system has some constraints and shortages, it satisfies almost all the targets and can work under various conditions. However, to achieving a better augmented-reality system for a real time game play, it still has a long way to go. There are still many details can be refined and further functions can be added to the system if given more time.

Generally, the approach to calibrate the projector automatically demonstrated in this article is very amenable to further refinement or improvements. The biggest limitation of the whole auto-calibration system is that it cannot recalibrate automatically when some external conditions change. For example, as demonstrated above, the experiments showed that this system could work if the chessboard was moved to a direction that was not so perfectly aligned with the orientations of webcam and projector. But this can only be done manually, because the process of exporting the captured images from the webcam software is separately from this auto projector calibration system. Conventionally, this can hardly work in real time; users cannot recalibrate the system manually every time when the chessboard has been relocated. As a result of this, the system should be refined to be able to recalibrate automatically itself.

Furthermore, for the algorithm implemented to realise this system, it may neglected the effect caused by some factors. For instance, the radial distortion caused by the optical lens of webcam or the projector. If this was taken into consideration, the system would be more complicated since different lens may result in different distortion coefficient. Nevertheless, in this project, it had not been taken into consideration when reconstructing the homography between webcam and projector. Because the radial distortions caused by the webcam and projector in this project were not obvious. During future development, the system can be possibly more robust if take those radial distortion coefficient of lens into account.

In addition, once the automatic projector calibration system has been refined, more projection graphic annotations can be designed. Those different colored annotations are supposed to enrich the user experience during the game. Users are expected to get as much game information as possible from those projected annotations. Therefore, what kind of annotations would increase the attractiveness of the game should be worthy exploring.

Finally, in the future, when the whole ‘Augmented-Reality Chess Game’ system has been completed, it is supposed to be able to present different graphic annotations to give the players fantastic visual feedbacks during the game. As imagined, in the final system, it needs to show users a projected visual menu on the chessboard. Then, users should be able to choose game mode, such as self-learning, human-computer competition, and human-human competition. In the self-learning mode, the system can provide users with visual feedbacks about game strategies, and colored arrows can be shown to present the movement directions of game pieces. With such a system, players can gain more interests and knowledge about chess during the game. Ideally, this system can be developed to be more robust and flexible. The game it implemented to will not be limited by the chess game, but also some other similar board games, such as draughts. Thus, it brings users more fun during the traditional games with these advanced augmented reality techniques.

CHAPTER 7

Conclusion

In conclusion, the calibration approach used in this project is potent and efficient. The assumption of chessboard in the real world to be simplified as a 2D planar surface is quite efficient in solving this kind of problem. Because, in doing so, the physical size of the chessboard can be neglected and the calculation can be accomplished in some relative units. Generally, the final results provide the essential evidence for the succeed of this automatic vision-based projector calibration system. The system can yield relative stable mappings between three main components (chessboard, webcam and the projector) with relative high accuracy. However, it still need to be kept in mind that there are some constraints for this system, such as image quality, projector brightness and light condition. All of these problems need to be resolved in future development. Some possible suggestions have been given on the areas which may be worthy further investigating. Hopefully, some day in future, the whole completed Augmented Reality Chess game system can provide people with a novel fantastic experience in playing the improved traditional chess game.

Bibliography:

1. Ronald A., Yohan B., Reinhold B., Steve F., Simon J. and Blair M., Recent Advances in Augmented Reality. In *IEEE Computer Graphics and Applications*. vol. 21, issue 6, Nov/Dec. 2001, pp. 34 - 47.

2. C. Magerkurth, A. Cheok, R. Mandryk, and T. Nilsen. Pervasive games: bringing computer entertainment back to the real world. *Computers in Entertainment*, 3(3), 2005

3. S. Hinske and M. Langheinrich. W41K: digitally augmenting traditional game environments. In *TEI'09: Proceedings of the SIGCHI conference on Human factors in computing systems*, 1997, pp. 234 - 241.

4. R. Azuma, A Survey of Augmented Reality, *Presence: Tele-operators and Virtual Environments*. vol. 6, no. 4, Aug. 1997, pp. 355 - 385.

5. Andrew D. Wilson, Depth-Sensing Video Cameras for 3D Tangible Tabletop Interaction, In *IEEE Horizontal Interactive Human-Computer Systems,* TABLETOP'07*, Second Annual International Workshop*, Oct. 2007, pp. 201 - 204.

6. Andrew D. Wilson, PlayAnywhere: A Compact Interactive Tabletop Projection-Vision System, In Proceedings of the 18th annual ACM symposium on User Interface software and technology, NY, 2005, pp. 83 - 92.

7. Steve Benford, Carsten Magerkurth, Peter Ljungstrand, Bringing the Physical and Digital in Pervasive Gaming, In *Communications of the ACM*, vol. 48, no. 3, March. 2005, pp. 54 - 57

8. Takahiro Iwata, Tetsuo Yamabe, Tatsuo Nakajima, Augmented Reality Go: Extending Traditional Game Play with Interactive Self-Learning Support, In *Embedded and Real-Time Computing Systems and Applications (RTCSA) IEEE 17th International Conference*, Aug. 2011, pp. 105 - 114.

9. S. Price, Y. Rogers, M. Scaife, D. Stanton, and H. Neale., Interacting with Computers: Using 'tangibles' to promote novel forms of playful learning. In *Interacting with Computers*, 2003, pp. 169 - 183.

10. L. Xie, A. N. Antle, and N. Motamedi., Are tangibles more fun?: comparing children's enjoyment and engagement using physical, graphical and tangible user interfaces. In *TEI '08: Proceedings of the 2nd international conference on Tangible and embedded interaction*, 2008, pp. 191 - 198.

11. D. Johnson and J. Wiles. Effective Affective User Interface Design in Games. In *Ergonomics*, 2003, pp. 1332 - 1345.

12. H. Kaufmann and B. Meyer. Simulating educational physical experiments in augmented reality. In *ACM SIGGRAPH ASIA 2008 educators programme*, Dec. 2008.

13. L. Terrenghi, M. Kranz, P. Holleis, and A. Schmidt., A cube to learn: a tangible user interface for the design of a learning appliance. In *Personal and Ubiquitous Computing*, 2006, pp. 153 - 158.

14. Johnny C. Lee, Paul H. Dietz, Dan Maynes-Aminzade, Ramesh Raskar, Scott E. Hudson, Automatic Projector Calibration with Embedded Light Sensors, In *Proceedings of the 17th annual ACM symposium on User Interface Software and technology*, vol. 6, issue. 2, Oct. 2004, pp. 123 - 126.

15. M. Kimura, M. Mochimaru, T. Kanade, Projector Calibration using Arbitrary Planes and Calibrated Camera, In *Computer Vision and Pattern Recognition CVPR '07 IEEE Conference*, June. 2007.

16. T. Okatani and K. Deguchi, Autocalibration of a Projector-Screen-Camera System: Theory and Algorithm for Screen-to-Camera Homography Estimation, In *Proceedings of the 9th IEEE International Conference on Computer Vision*, vol. 2, Oct. 2003, pp. 774 - 781.

17. R. Sukthankar, R. G. Stockton, M. D. Mullin, Smarter Presentations: Exploiting Homography in Camera-Projector Systems, In *Proceedings of Computer Vision ICCV 8th IEEE International Conference*, vol. 1, 2001, pp. 247 - 253.

18. Raskar, R., Beardsley, P. A., A Self-Correcting Projector, In IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Dec. 2001, pp. 504 - 508.

19. R. Sukthankar, R. G. Stockton, M. D. Mullin, Automatic Keystone Correction for Camera-Assisted Presentation Interfaces, In Advences in Multimodal interfaces (ICMI), vol. 1948/2000, 2000, pp. 607 - 614.

20. E. Dubrofsky and R. J. Woodham, Combining Line and Point Correspondences for Homography Estimation, In Advances in Visual Computing, vol. 5359/2008, 2008, pp. 202 - 213.

21. I. Sutherland, A Head-Mounted Three-Dimensional Display, in *Fall Joint Computer Conf., Am. Federation of Information Processing Soc.* (AFIPS) *Conf. Proc. 33*, Thompson Books, Washington, D.C., 1968, pp. 757-764.

22. Magerkurth, C., Engelke, T., and Memisoglu, M., Augmenting the virtual domain with physical and social elements. In *Proceedings of the international Conference on Advancements in Computer Entertainment Technology*, June. 2004, pp. 163 - 172.

23. Mandrk, R. L., Maranan, D. S., And Inkpen, K. M., False prophets: Exploring hybrid board/video games. In *Extended Abstracts of the Conference on Human Factors in Computing Systems*, 2002, pp. 640 - 641.

24. Adrian Kaehler, Gary Bradski, *Learning OpenCV Computer Vision with the OpenCV Library*, O'Reilly Media, 2008, printed in United States of America.

25. Kevin Suffern, *Ray Tracing from the Ground Up*. A K Peters, 2007.

26. Cabriel Falcao, Natalla Hurtos, Joan Massich, Plane-based calibration of a projector-camera system, 2008.

27. Smets GJF, Stappers PJ, Overbeeke KJ, Van der Mast CAPG, Designing in virtual reality: implementing perceptual-action coupling and affordances, In *Proceedings of the virtual reality software and technology conference*, World Scientific Publishing, 1994, pp. 97 - 110.