SPHERICAL IMAGE PROCESSING FOR IMMERSIVE VISUALISATION AND VIEW GENERATION

by

Xiaoyin Guan

A thesis submitted to the University of Central Lancashire
in partial fulfilment for the requirements of the degree of

Doctor of Philosophy

November 2011

The work presented in this thesis was carried out in the Applied Digital Signal and Image Processing (ADSIP) Research Centre, School of Computing, Engineering and Physical Science at the University of Central Lancashire, Preston, England

Declaration

I declare that while registered with the University of Central Lancashire for the degree of Doctor of Philosophy I have not been a registered candidate or enrolled student for another award of the University of Central Lancashire or any other academic or professional institution during the research programme. No portion of the work referred to in this thesis has been submitted in support of any application for another degree or qualification of any other University or Institution of learning.

Signed _____

Xiaoyin Guan

Acknowledgements

There is a saying from the great Chinese philosophy ‘A tutor for a day, a father for a lifetime’. I hereby give great thanks to my supervisor Prof. Lik-Kwan Shark, for his hard work and kind heart, valuable guidance and consistent support, for each of our meetings that focused my mind, and renewed my strength.

Special thanks to Dr. Wei Deng and Dr. Geoff Hall, for their logical minds and kind support.

I am also grateful for more technical helps from Dr. Wei Quan, Dr. Bogdan Matuszewski, Dr. Yan Zhang and Dr. Andriy Kurekin. Their innovative opinions helped me through many difficulties.

I would like to thank my parents for their unconditional love, my friends Miss Yu Yi and Miss Yingzhu Li for their companionship.

Finally,

To the summer of 2010, my little friend Miss Xuan in Heaven, and God Himself

ABSTRACT

This thesis presents the work carried out on studying and processing of panoramic spherical images for immersive visualisation of real environments and generation of in-between views based on two views acquired. It studies the properties of spherical images covering 360 degrees horizontal and nearly 180 degrees vertical field of view, against planar images taken by conventional digital cameras. The spherical image is taken by a rotating line spherical camera and a camera calibration method based on perpendicular panels covered by checkerboard patterns is proposed for estimating the projection parameters and recovering the camera pose.

For visualisation based on one spherical image, the surrounding environment is modelled by a unit sphere mapped with the spherical image and the user is allowed to view the modelled scene from within. The mapping function is essentially the transformation between pixels in their image coordinates and their projection on the spherical sensor, and viewing is based on a small viewpoint window interpolated with pixel values from the spherical image. Once the user moves away from the centre of the unit sphere, which is the point where the spherical image is taken from in the real world, image distortion occurs to the viewpoint due to the nonlinear characteristics of the camera sensor. A distortion correction method is proposed and implemented to remove this distortion thereby allowing the user to navigate within a certain range inside the modelled sphere and see correct visual results. This is seen to improve the user's visualisation experience by enabling more self-immersion into the modelled scene than the viewpoint being fixed at a point.

For visualisation based on two spherical images, a view generation algorithm is proposed for modelling an indoor manmade environment. If two spherical images are taken at different positions within a room, geometry constraints exist between the two spherical views of the same scene. The camera poses are recovered from the calibration and furthermore, the scene is modelled by a simple geometry in order to generate control points for image warping. The view generation algorithm deals with background image and foreground objects separately. For each occluded area caused by a foreground object in one spherical image, its texture is found from the other spherical image. The new view is rendered from a combined occlusion free background image by using thin-plate spline surfaces as mapping functions to accommodate the nonlinear characteristics of the spherical camera sensor, and the rendered texture of any

foreground object is then added to the background. This allows the scene to be modelled using multiple spherical images and the user to move smoothly from one sphere mapped image to another one by going through in-between sphere mapped images generated. By combining the proposed method with a user position and orientation tracking system, the work is seen to form a solid base for implementation of an immersive and interactive visualisation environment.

Table of Contents

Chapter 1 INTRODUCTION

1.1 Visualisation based on Image Processing

During the past two decades, visualisation technologies have advanced rapidly and achieved a whole new level due to the ever increasing performance as well as ever decreasing costs of computer hardware. Traditionally, visualisation of environments falls in the field of computer graphics (Blundell, 2008) (Watt, 2000), whereby 3D geometric models are generated to describe the surrounding environment and new views are rendered directly from the 3D geometric models. The issues of this geometric-based modelling and rendering are the requirement of high performance hardware or time and the complexity in generating representations of a scene from primitive polygons, polynomial curves or space subdivision (Guttmann, 2009). Furthermore, it is difficult to create natural realism as in the real world. This is because the intricate geometry of real world scenes as well as the unavailability of an accurate lighting and reflection model. Therefore, there comes an alternative approach named image-based modelling and rendering (Shum et al., 2007) to address the problem. By processing images captured directly from the real environment instead of construction of its geometric model, image-based modelling and rendering combines processing algorithms under the fields of computer vision (Shapiro and Stockman, 2001) and computer graphics to generate photorealistic views from viewpoints different from the original positions where the input images are captured from (Burschka et al., 2003) (Shum and Kang, 2000).

1.1.1 Visualisation and View Generation Principles

For visualisation based on image processing, the world is essentially perceived as an interaction of light rays with the surface of the objects. Since an image of the scene is formed as a result of a set of light rays passing through the eye and being focused on the retina, every visible object could be described by a dense array of light rays in the space. As a digital representation, the *plenoptic* function was first introduced by (Adelson and Bergen, 1991) to describe the structure of the light. It is essentially a 7D function describing the amount of light passing through the camera lens at every location, and every possible viewing angle of the scene for every possible wavelength at every time instant. Arbitrary views of the scene can be generated if the full *plenoptic* function is given.

However, the *plenoptic* function is in practice impossible to compute due to an unmanageable amount of data to acquire and to process. It can be reduced in its dimensionality by imposing certain scene constraints under certain assumptions (Shade, 2002). For example, if only one snapshot at a time is concerned and the light is considered to be monochromatic, the function is reduced to 5 spatial dimensions with 3 parameters describing the location and 2 describing the viewing direction. Even for a 5D *plenoptic* function (McMillan, 1997), it is still very difficult to construct a new view from a different viewpoint, as it often requires solving the feature correspondences problem. A further reduction of the dimensionality can be achieved by resampling the function to a 4D light field (Levoy and Hanrahan, 1996) or lumigraph (Gortler et al., 1996) by restricting viewpoints to move within a bounding box. In addition, a 3D function can be achieved by restricting viewpoints to move along a 2D circle (Shum et al., 2005); and a 2D function by placing the viewpoint at fixed points (Chen, 1995).

Visualisation and new view generation by image-based modelling and rendering contains subtle techniques which are the interest of various research communities such as computer graphics, computer vision, image processing (Jähne, 1997), and virtual reality (Shumaker, 2011). Very often, rendering results depend on the scene complexity contained in the input images or constraints existed in the scene. (Chen and Williams, 1993) proposed a view interpolation method to generate novel views by interpolating the optical flow between corresponding points. Dense correspondences between the pixels in reference images are established by geometry-based rendering and the reference images are mapped to the desired viewpoint by image morphing (Watkins et al., 1993) with linear interpolation between corresponding points. Although this method generates accurate results, there is a problem when the geometric model is absent and the method requires the change in viewing positions to be small.

(Laveau and Faugeras, 1994) presented a method for representing a 3D scene without an explicit geometry reconstruction. The scene is represented as a collection of images related by algebraic relations and perspective depth information of the desired views is calculated by homogenous transformations between the projections of the reference and the desired views. However, this method might require several images to assure an unambiguous visibility solution.

(Lhuillier and Long, 2003) introduced the joint view triangulation (JVT), which is an image representation created by the parallaxes between image pairs for solving the

visibility and occlusion problems. Together with an incremental edge-constraint and a pseudo-painter's rendering algorithm, novel views could be painted according to a triangulated disparity map. Difficulties arise when the scene contains a mixture of natural and manmade objects.

Other methods including combining some amount of geometric modelling based on user inputs, and image based processing together to improve the rendered result. (Oh et al., 2001) presents a scene as layered collection of depth images assigned by the user and the image based objects in the scene can be modified and viewed from different viewpoints. (Jabi, 2000) visualises and investigates an architectural space modelled by images and allows the user to sketch over the modelled scene and that the sketched artifacts rotate correctly when his view shifts.

Tour into the picture (TIP) is also suggested for generating a sequence of walk-through images from a single reference picture (Kang et al., 2001) (Horry et al., 1997). 3D scene model is constructed from the picture assuming the picture has one vanishing point. Convincing 3D effects can be generated through this simple perspective geometry method.

1.1.2 Panoramic images

Before modern panoramic imaging devices being made commercially available (the major products on the market are to be introduced in Chapter 2), special techniques had already been used by researchers to achieve a wider field of the view of the scene projected onto images. This type of images increases the approximation of the *plenoptic* function as it describes bigger amount of light rays interacting with the scene at greater viewing angles. It effectively makes more scene information available for modelling and rendering. Techniques for creating panoramic images are divided into two categories, one is in the area of image processing, and the other is in the area of image acquisition by making use of special devices or structures.

On image processing, the most common technique is image mosaicing (Capel, 2004) dated almost as early as photography by using manual methods initially. With computers available, image mosaicing is achieved through different algorithms such as image stitching (Szeliski, 1996) from a sequence of images acquired using a controlled camera motion (panning). (Szeliski and Shum, 1997) presented a method without

constraints on how the images are taken by using a set of motion transforms. It creates seamless panoramic mosaics from a number of overlapping images. An omnidirectional multi-baseline stereo reconstruction method was proposed by (Kang and Szeliski, 1997), but dense correspondences need to be recovered to reconstruct the panoramic scene with depth information. At each camera location in the scene, sequences of images are captured while rotating the camera about the vertical axis passing through the camera optical centre. Panoramas at each camera location are produced by compositing each set of images.

On image acquisition, special devices such as mirrors, fisheye lenses, rotating tripods are used to set up the capturing system for panoramic image acquisition. (Shum and He, 1999) constructed concentric mosaics by mounting a number of cameras on a rotating horizontal beam supported by a tripod, to obtain a collection of mosaic images of the scene. Novel views can be rendered by combining the appropriate captured light rays from the mosaics. (Bakstein and Pajdla, 2002) proposed a camera model with fisheye lenses with a field of view larger than 180 degrees. Mapping functions of the projection are estimated from a special camera calibration method, and 360 by 360 degrees mosaics can be composed from the projection model. (Bakstein and Pajdla, 2003) adopted an omnidirectional camera equipped with a lens with a field of view of 183 degrees to acquire a sequence of images, which are used for composing omnidirectional concentric mosaics to enable rendering stereo pairs of views and allowing arbitrary viewing directions and vergence angle of the two eyes of a viewer. The viewer movement is at a fixed point but allowed to look sideways to see behind the occluding objects.

The above image acquisition set-up and image processing methods often involve manual work as they are not error free, and sometimes precise craftsmanship is required for generating high quality images.

1.1.3 Immersive Visualisation Hardware

Panoramic imaging results can be applied to immersive visualisation hardware, which range from high resolution monitors with stereoscopic display (3D monitors) to viewer-centred perspective interfaces, and fully immersive projection systems in virtual reality.

Examples of small scale visualisation systems include Head Mounted Display (HMD) and Binocular Omni-oriented Monitor (BOOM). For HMD, it consists of a pair of stereo displays to provide the user with the visual signal received, and effectively isolates the user from the real world by immersing the user in the displayed imagery (VirtualWorldlets-HMD) (Fiala, 2005). Using a head-tracking device in conjunction with HMD to provide the location and orientation of the viewer, the displayed image is adjusted according to the position of the head to simulate the correct view, thereby creating the illusion of physical presence within the virtual environment. Like the HMD, BOOM (see Figure 1-1) mounts small displays on an articulated arm which measures its position and orientation in order to provide a correct view to the user (VirtualWorldlets-BOOM).

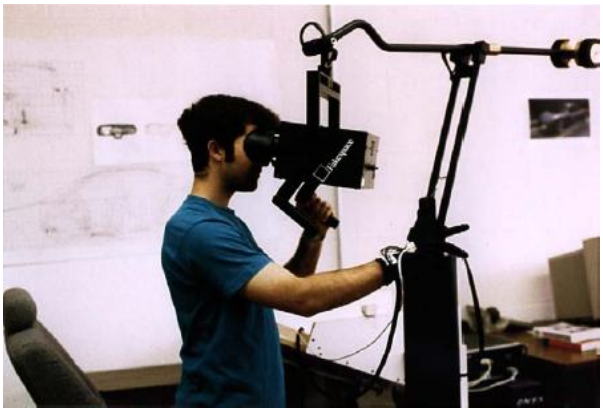


Figure 1-1 BOOM system from (Márquez, 2002)

Large scale visualisation facilities are projection based multiscreen display systems such as CAVE Automatic Virtual Environment System (Cruz-Neira et al., 1993), (Cruz-Neira et al., 1992) with hemispherical displays and Cybershpere (Fernandes et al., 2003). For the former, the viewer observes the scene back projected onto the hemispherical wall, and is free to move within confines of the room. By tracking the movement of the viewer using wired or wireless devices, the correct perspective and stereo projection of the environment appear on the display screen. Cybersphere is a fully immersive spherical projection system that completely encloses the user in the virtual world. It comprises of a large hollow translucent sphere, and walking movements of the viewer cause the sphere to rotate, thereby allowing the user to navigate and explore the virtual world in a natural manner in all directions.

1.2 Aims and Motivation

Panoramic images have already become the modern interest of not only researchers but common consumers nowadays, due to the fast development of digital imaging technologies and availability of camera devices offering high resolution and high quality images at affordable costs.

Moreover, panoramas with the cylindrical prototype have the simplicity for image processing as they are essentially planar images stitched together to yield a wide horizontal field of view and warped to a cylinder. Each slit of the cylinder when un-warped is considered as image columns comprising a planar image. Although cylindrical panoramas involve multi-perspective projection, there is no projection distortion. As the projection does not cover the sky and ground of the actual scene, its lack of fully immersive sensation to the user when being applied to advanced virtual reality equipment is a disadvantage.

Spherical images with the sensor prototype being a sphere have the advantage of full view coverage of the scene, but it becomes problematic for visualisation and view generation in image rendering due to the nonlinear characteristics inherent with the sensors. Previous methods have been developed, such as that used in the Google Map's Street View (Google, 2011), to interpolate part of the spherical image to the user's viewing window, but the viewing is restricted at a fix point. In order to simulate a virtual walkthrough (or navigation through the scene), the user has to 'jump' from one spherical camera position (scene sampling point) to another.

The aim of this project is to develop image processing methods for immersive visualisation of real-world environments modelled by spherical images. While different solutions are available to generate a spherical image, a spherical camera is used for image acquisition to ensure the projection model estimated for the images to be accurate.

The scene is modelled and simulated either by one or two spherical images with each spherical image mapped on a unit sphere. With only one spherical image, the fact that no depth information being available limits the viewpoint being at the point of image acquisition (or at the centre of the unit sphere). Hence, the specific objective for image processing of one spherical image is to develop algorithms to allow the user to observe the correct deformation of the scene content when moving away from the original viewpoint, thereby enhancing the immersiveness in visualisation of one sphere mapped

image by providing a certain range of free movements within the sphere, instead of just simple zooming in and out in the current visualisation systems.

A more advanced user navigation could be achieved by linking spherical images captured at different positions. This is especially difficult for a closed indoor manmade environment with objects of rigid shapes. Unlike an open outdoor environment with surroundings often distant from the image acquisition point, the space is often narrower indoors when structures like walls are closer to the camera position. It is straightforward to understand that when the scene (e.g., landscape) is distant from the camera, depth information is not important which can be approximated to a unity value, and this suits the modelling based on only one spherical image where the scene is mapped onto a unit sphere centred at the camera position. Hence, the specific objective for image processing of two spherical images is to develop algorithms to enable new views to be generated based on two nearby spherical images acquired, thereby enhancing the immersiveness in visualisation of multiple spherical images by allowing the user to move smoothly from one sphere mapped image to another in an indoor environment, instead of simple jumping from one image acquisition position to another with visual discontinuity.

1.3 Thesis Structure

The thesis is structured into six chapters. Chapter 2 introduces the spherical camera, SpheroCam®, used for image acquisition and studies the properties of the spherical images provided by this camera. Chapter 3 describes a calibration method for this camera based on a simple calibration object, and includes the experiments carried out to assess the calibration accuracy. In Chapter 4, immersive visualisation based on one spherical image is introduced and simulated. The system is shown to allow the user to navigate away from the original image acquisition point and generates correct visual results for the user through an image distortion correction algorithm. Chapter 5 studies two-view geometry for spherical images and extends the immersive visualisation by generating new views based on two spherical images. The work focuses on visualisation of indoor environments by giving the user more freedom to walk within the modelled room and providing the accurate panoramic views rendered from a mixed texture from both of the spherical images. Chapter 6 concludes the work and indicates future research interests.

Chapter 2 PANORAMIC IMAGING AND SPHERICAL CAMERA

The word panorama means ‘all’ (*pan*) and ‘sight’ (*horama*) in Greek terms (Merriam-Webster, 2007). In a more technical term, a panorama is defined as a picture with an unlimited view in all directions, which synonymously means ‘omnidirectional’ (PARIAN, 2007). In this project, a special spherical camera, SpheroCam®, is used for acquiring panoramic images, and this chapter mainly introduces the different approaches to achieve panoramic imaging by modern panoramic imaging devices, followed by the properties of the adopted spherical camera and its output images.

2.1 Panoramic Imaging

Panoramic imaging techniques can generally be divided into two categories, catadioptrics and dioptrics (Parian, 2006). Dioptrics is the science of refracting elements (lenses), and catoptrics is the science of reflecting surfaces (mirrors). The combination of refracting and reflecting is called catadioptrics (Gross, 2005).

In catadioptrics, panoramic imaging can be achieved by using a single special mirror or a combination of multiple planar mirrors, to capture images with a wider field of view. Special types of mirrors include conical, spherical, ellipsoidal, and hyperbolic mirrors, each with a unique mathematical projection model. An example of a catadioptric panoramic camera is FullView® (FullView, 1999), which consists of four cameras clustered to capture images from planar mirrors configured in a pyramid shape.

In dioptrics, panoramic imaging is achieved by using only cameras and lenses, and methods are grouped into four categories: camera cluster, fisheye lens, direct scanning and stitching.

The first category is to combine multiple cameras or one camera with multiple sensors looking out directly to different directions to capture views from different directions. Dodeca® 2360 Camera System (Media, 2011) developed by Immersive Media is an example, whose technique has also been adopted by Google Map’s Street View (Google, 2011).

The second category for panoramic imaging is to use a single camera looking out through a fisheye lens, which gives the sensor a 180-degree angular field of view. Panoramas can then be produced by processing these fisheye images taken by the camera using appropriate software. A surveillance camera with a fisheye lens is an example, such as IPIX (Minds-Eye-View, 2009) and ImmerVision (ImmerVision, 2010).

The third category is through direct scanning such that a single camera is looking in different directions at different instants. This type of cameras is normally known as a rotating camera, and so is the SpheroCam® used in this project. Other examples are PanoCam®, Panoscan® (Panoscan), and Eyescan (Scheibe et al., 2001). The image data from different directions are collected during camera rotation without the need of stitching. The projection prototype can be modelled as a sphere or a cylinder, depending on whether a fisheye lens is used. When the camera is attached to a fisheye lens, a wider vertical field of view (nearly 180 degrees) is obtained. Although the scanned image is with great distortion when being viewed through a 2D window screen, the distortion could be removed when the image is 'mapped' onto a sphere. A spherical image viewer software provides the right way to appreciate these images.

The fourth category produces panoramic images by mosaicing or stitching a number of normal images together. This has been studied by various researchers as introduced in Chapter 1. The available panoramic stitchers software are the *Hugin* project (Hugin, 2004), for photo mosaic construction, and *Montage* (Montage) for astronomers to create images of regions of the sky that are too large to be captured by astronomical cameras.

2.2 The Spherical Camera

This section gives an introduction of the SpheroCam®, which has been used for acquiring the image data for this project. Its functionalities, properties and use for acquisition are discussed in Sub-sections 2.2.1 and 2.2.2, respectively.

The SpheroCam® is developed by SPHERON-VR AG, Germany (SPHERON-VR), who launched their first panoramic camera (PanoCam®) in 1998. Their product family includes PanoCam®, SpheroCam®, and SpheroCam HDR®, producing three levels of panoramic images. PanoCam® uses a normal optical lens, other than a fisheye one adopted by the latter two, and takes images without covering views of the sky and the ground. This type of images is defined as a cylindrical panorama as it can be mapped onto a cylinder. SpheroCam HDR® is an advanced version of the SpheroCam® used in this project. It takes spherical panoramas with a high dynamic range (Reinhard, 2006), up to 26 f-stops to enable higher contrast for both bright and dark areas in the same image.

2.2.1 System Components

The SpheroCam® scanning system is illustrated in Figure 2-1 (SpheroCam) below. The system consists of the following main elements: a camera head and lens attached to a turntable, a tripod, and a portable computer attachable to the tripod.

Figure 2-1 SpheroCam® scanning system

The camera head and its turntable (see Figure 2-2) are specially designed to allow a line CCD sensor at the camera centre to rotate around a vertical axis (perpendicular to the ground), and to return to its original panning position after rotation. The turntable is driven by an integrated DC gear motor and is powered by a rechargeable battery pack, which could be attached to one leg of the tripod. The line CCD sensor scans vertically pixel-by-pixel each time when it is moved to a new position. When the round turntable is positioned horizontally level using the tripod, and the sensor rotates and scans a complete cycle of 360 degrees, the image data will constitute a seamless panorama covering everything visible in the surroundings, except a bottom circular area where the tripod stands.

Figure 2-2 Camera head and its turntable

The fisheye lens (see Figure 2-3) is the key to the spherical imaging in this context. Specifically, a 16mm f/2.8D AF Nikon Fisheye lens (Nikon, 2011) is adopted by the SpheroCam®. Most fisheye lenses fall into two categories, namely, circular fisheye lenses and full frame fisheye lenses (Kumler, 2000). Circular fisheye lenses cover the field of view of 180 degrees in all directions, and only expose a disc in the centre of the image frame. Full frame fisheye lenses, on the other hand, expose the whole sensor area, but only cover 180 degrees in the diagonal direction (See Figure 2-4). As this Nikon fisheye lens has a full frame design, the actual vertical field of view of its imaging device will be significantly less than 180 degrees. The actual vertical field of view is regarded as an important parameter of this camera and will be discussed in details in Chapter 3.

Figure 2-3 Nikon fisheye lens from (Nikon, 2011)

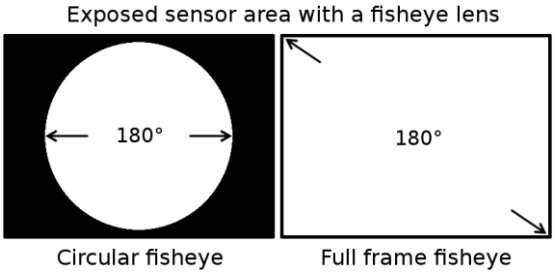


Figure 2-4 Fisheye lens sensor exposure area from (Photography)

The tripod (see Figure 2-5) is a Manfrotto® 055MF3 (Manfrotto, 2010), designed with a special connection head to accommodate the camera, and with two spirit bubble levels to indicate whether the scanning system is positioned level. The tripod's minimum height is 11cm, and could be fully stretched to the maximum height of 169cm, which is suitable for taking images indoor and outdoor. The battery pack is attached to one leg of the tripod, and the portable computer could be attached to another.

Figure 2-5 Manfrotto® tripod

The portable computer is installed with SPHERON-VR's managing software to operate the camera and store image data. It controls the camera head and turntable to rotate and scan. The software can also be used for processing the raw output image data after scanning. A spherical image viewer, *SpeheroViewer*, is also installed in the computer for viewing the scanned image in a correct perspective. The camera is connected to the computer through a simple USB port.

2.2.2 Software Functionalities

The managing software, *SpheronCam*, operates the camera scanning system to capture images. It is also used for post-processing of raw image data. In this sub-section, the functionalities of the software are discussed with respect to its use for image acquisition and processing. These functionalities are divided into the following three areas:

1) Camera Control

The software controls the camera head's rotating direction, i.e. clockwise or anticlockwise, and adjusts the camera viewing direction at the start point before scanning. The latter avoids the person operating the camera being caught by the scan. Also for images obtained with different rotating directions, the camera coordinates whose origin is at the camera centre should be defined as either a left-handed or right-handed 3D coordinate system.

The horizontal angular scanning range of the camera can be controlled according to the pre-settings before the scanning starts, or by operating the 'start' and 'stop' buttons provided by the software. The software also enables the captured image to be monitored in real-time while scanning.

The software controls a number of camera settings for optimal scanning outcome. Basically it includes white balance, ISO sensitivity, focus distance, aperture setting, resolution, exposure time, and rotation angle in degrees. The significance of these photography technical terms can be found from (Drew, 2005).

When assembling together the different components for the camera scanning system, i.e. the camera head and the tripod, etc., the software reads the camera's built in electronic level and shows it on the computer screen so that the camera head's levelling position can be adjusted accordingly by hand with respect to the horizon to reduce the misalignment in the output images.

2) Lens Focusing and Calibration

The software helps the lens to focus on a desired area in the output image. Its image scanning preview screen indicates the manual adjustment outcome of the focusing ring on the lens, and the sharpness of the image could be improved in this way.

Although the lens provided with the SpheroCam® is carefully calibrated by the manufacturer, and the calibration data are attached to the lens through the software, the user is still able to follow the calibration process suggested in the camera manual (SPHERON-VR, 2004). The calibration is achieved by scanning a black and white stripe target, then to generate the correction data for the lens through the 'lens analyzer' function. The quality of the correction data can also be accessed by the software.

3) Image Post-processing

Once a scan is finished, the image data are stored in the computer in the form of a .sph file. The software can convert the raw data to other image file formats, such as TIFF, JPEG, PNG, GIF, etc. for its final output. The following post-processing operations are also provided by the software:

- Correct raw scan data for fisheye lens distortion and chromatic aberrations
- Crop overlapped spherical scans (with a default setting of 400 degrees horizontally) to 360 degrees, blending the seam
- Convert to different image geometries, e.g. generate six cube faces from full spherical scans, create plane projection cut-outs of full spherical scans
- Resize the image, pixel re-sampling
- Perform colour space transformations
- Perform unsharp maskings

For more details of the above functions, refer to (SPHERON-VR, 2004).

2.3 Spherical Images

Spherical images are produced and stored in a normal image file after scanning and post-processing through the software. Figure 2-6 and Figure 2-7 below show two typical spherical images taken by the spherical camera in an indoor and outdoor environment, respectively. The horizontal field of view is set to 360 degrees (a full scan) and the images are both converted to JPEG format.

Figure 2-6 A spherical image of an indoor office environment

Figure 2-7 A spherical image of an outdoor environment

2.3.1 Image Properties

From Figure 2-6 and Figure 2-7, it is clearly seen that the spherical images have a much broader field of view than normal images taken by conventional cameras by covering views of the scene from every direction. The image contents distribute in such a way that the ceiling (sky) and the floor (ground) appear to be much more distorted from our perspective than the middle part horizontally across the whole image. This is due to the way the fish eye lens collects lights and the rotating line CCD sensor, which result in the prototype of the camera being modelled as a spherical focal surface. The mathematical expression of the camera's projection model is discussed in Chapter 3.

There are four default sizes for the output spherical image. With the horizontal field of view being set to 360 degrees (a full scan), Table 2-1 shows the data size and the scanning time of the output images.

Size	Resolution (pixels)	Data Size (megabytes)	Scanning time with shutter speed at 1/60s	Scanning time with shutter speed at 1/30s
Minimum	640 x 1571	6	0:36	1:07
Medium	1280 x 3141	24	1:16	2:21
High	2624 x 6283	100	2:25	4:28
Maximum	5312 x 12566	403	7:38	8:56

Table 2-1 Spherical image properties

It is seen from Table 2-1 that the camera produces high resolution panoramas up to 66 megapixels. The scanning time is limited by exposure time or data transfer. For small images, the aperture and shutter speed settings tend to determine the scanning time, as

seen from the table that the scanning time nearly doubled when the shutter speed is decreased to its half. For large images, the date transfer speed limit is crucial (the USB connection to the camera is capable of transferring one megabyte of data per second) hence it takes at least 400 seconds to finish the scan when the data size increases to 400 megabytes, regardless of other impacts like the resolution and the shutter speed.

The camera uses 16 bit colour CCD technology with each image pixel represented by 3 x 16 bits in RGB. The output with 16 bit depth per colour, rather than the normal 8 bit colour, provides a higher colour fidelity for its image.

2.3.2 The Spherical Image Viewer

The images captured can be restored to a normal perspective on a 2D screen, when being viewed by the spherical image viewer software, *SpeheroViewer*. The software provides the user a viewing window to observe the image from inside of a sphere which the image has been mapped onto. The user can pan, tilt, zoom in and zoom out the image by operating the mouse, but cannot move away from the centre of the sphere which is the camera's viewpoint of the image when it is taken.

The principle under the spherical viewer is to project a certain part of the spherical image onto a planar plane. This is achieved by image interpolation of the appropriate part of the spherical image onto the user's viewing window, according to the viewing orientation selected by the user (Benosman and Kang, 2001).

2.4 Concluding Remarks

In this chapter the general properties of the spherical camera as well as its output images are presented. In comparison with other panoramic imaging devices on the market, the rotating line SpheroCam® with a fisheye lens provides seamless scan of spherical images in an easy operation manner. The spherical image with a panoramic view of the scene can be observed through a spherical image viewer. However, this viewing is restricted to a fix point for the user, and leads to the work described in the following chapters to allow user more freedom of movements in visualisation of sphere mapped images.

Chapter 3 SPHERICAL CAMERA CALIBRATION

3.1 Introduction to Calibration Approaches

Camera calibration has been a subject area with significant importance to the majority of researches in digital imaging. Essentially, calibration is to estimate the intrinsic and extrinsic parameters of the camera's projection model (Trucco and Verri, 1998). The intrinsic parameters describe the camera's sensor projection, i.e. how 3D points with coordinates referenced by the camera centre are being projected onto the image sensor (image plane). The extrinsic parameters describe the transform between 3D points under the camera coordinate system and their world coordinate system, in other words, they describe the camera's position and orientation with respect to the real world coordinates. The intrinsic and extrinsic parameters together form the camera's projection matrix (Fusiello et al., 2000), which describes directly the projection from any 3D points to their image.

The most straight forward method to calibrate a camera is to produce equations linking the projection of a group of 3D points and their image, based on picture(s) taken of a 2D calibration chart or a 3D object (Hartley and Zisserman, 2003). Other methods employ no calibration object are so called self-calibration (or auto-calibration), where the camera parameters are recovered from a set of 'uncalibrated' images using constrains on the parameters themselves or in the imaged scene (Mendonca and Cipolla, 1999). As errors often occur due to the camera and/or lens imperfection, some of following additional parameters causing projection errors are often taken into account to increase calibration accuracy: radial distortion, decentering distortion, principal distance, principal point offset, skew factor, etc (Wei and Ma, 1994).

For calibration of conventional cameras, (Tsai, 1987) uses multiple images and requires more than eight feature points per image to solve the linear equations based on the radial alignment constraint with a second order radial distortion model. (Zhang, 2000) requires a planar checkerboard grid pattern where corner features are extracted to compute the projective transformation between image points of multiple images, up to a scale factor. (Ayache and Lustman, 1991) describes a stereo system which requires calibration of the projection matrix, but not explicit knowledge of the camera parameters.

For special cameras for panoramic imaging as explained in Sub-section 1.1.2 and Section 2.1, different calibration methods have been proposed. (Parian and Gruen, 2004)

and (Schneider and Maas, 2003) use a large calibration room/field attached with target points such that calibration markers are distributed in the image sparsely and equally. Although this method has the advantage of accuracy with more than 20 imaging parameters generated for the camera model, it has the disadvantages of a large space requirement and time consuming preparation. To enable direct use of real world environments, a calibration method has been proposed based on straight line segments extracted from man-made environment (Hirota et al., 2006) and it relies on a sufficient number of straight lines available in the scene. Further calibration methods, such as (Fujiki et al., 2007) and (Svoboda and Pajdla, 2002), require multiple views, whereby the epipolar geometry (Cyganek and Siebert, 2009) between two views is utilized to enable image correspondences being searched for camera calibration.

Different projection models have also been used in calibration of these special cameras. For cameras with fish-eye lenses, the proposed projection models include the simple equidistance function (Bakstein and Pajdla, 2002), a general higher order polynomial form with the result modified by circular control points (Juho, 2006), and a rational function with image points 'lifted' to higher power and the size of the camera projection matrix increased to 3 by 6 (Claus and Fitzgibbon, 2005).

This chapter focuses on calibration of the SpheroCam®, which is essentially a rotating line spherical camera with a Nikon fisheye lens. The method is developed with simplicity in mind. It uses a calibration object based on a small five-sided open box with a checkerboard pattern on each plane, a simple equidistance projection model with an additional camera parameter to characterise the vertical field of view, and a simple iterative computation process to estimate the required camera projection matrix. Also presented are the experimental results to show the achievable accuracy of the calibration method as well as the influence of the number of corner points and the number of planes on calibration accuracy.

This chapter is organised in six sections. Section 3.2 introduces the spherical camera's projection model, whilst Section 3.3 presents the basic settings and image pre-processing for the calibration. A special calibration object is introduced as well as the methodology to pre-process the spherical image. This is followed by the simple calibration methodology based on the feature corner points on multiple planes in Section 3.4, and experimental results in Section 3.5 to show the accuracy as a function

of the number of corner points and planes used for calibration. The last section concludes the work.

3.2 Spherical Camera Projection Model

In this section, the prototype of the spherical camera projection model is introduced. For each 3D point in its world coordinates capture by the spherical camera, it is projected onto the camera sensor and then transformed to an image pixel in the output spherical image. The process involves transformations between different coordinate systems, which are described in the following subsections. Basically, the prototype of the projection is modelled as a sphere (which represents the camera sensor) centred at the camera origin. As the camera is unable to cover the bottom circular area where the tripod stands, the sphere is incomplete with a small part of the bottom section removed. This is illustrated in Figure 3-1(a) (SpheronVR) where the camera field of view is less than 180 degrees for each vertical scan-line, and in Figure 3-1(b), where the full scan forms a sphere with an empty flat bottom as the projection model of the camera sensor.

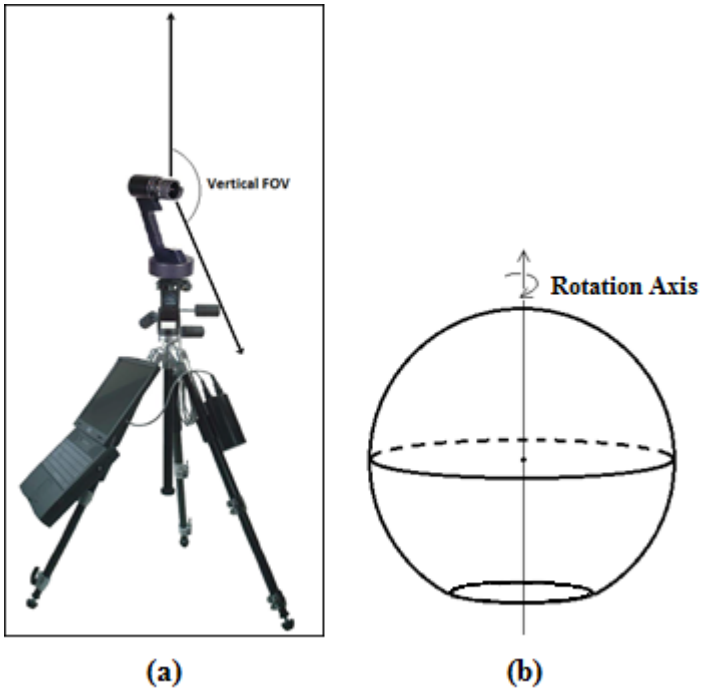


Figure 3-1 (a) Spherical camera vertical field of view and (b) Spherical camera projection model

3.2.1 2D Coordinate Systems

The 2D coordinate systems involved in the spherical projection model are the 2D sensor coordinate system referencing the vertical fisheye lens projection, and the image coordinate system defined for the output image.

1) 2D fisheye lens projection on the image sensor

There are a few projection models adopted by different types of fisheye lenses. The most common ones are listed below:

Orthogonal projection	-	$v = f\sin(\varphi)$	**Equation 3-1**
Equidistance projection	-	$v = f\varphi$	**Equation 3-2**
Stereographic projection	-	$v = f\tan\left(\frac{\varphi}{2}\right)$	**Equation 3-3**
Equisolid angle projection	-	$v = f\sin\left(\frac{\varphi}{2}\right)$	**Equation 3-4**

where φ is the angle between the optical (principal) axis and the incoming ray, i.e. the elevation angle, f is the focal length of the lens (the radius of the sensor sphere, or the distance between the sensor and the optical centre), and v is the distance between the image point and the principal point (image centre in the vertical direction) defined in the 2D sensor coordinate system.

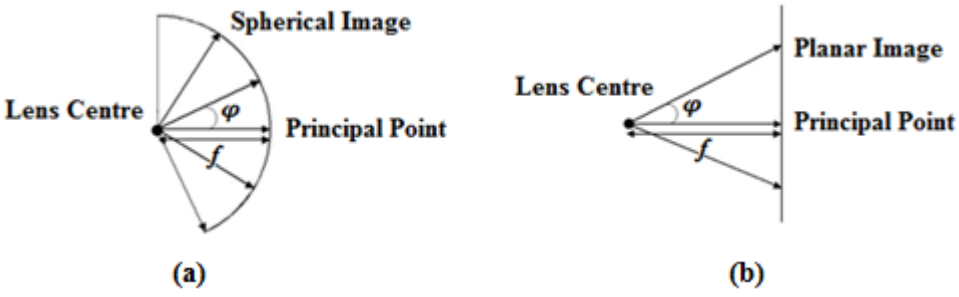


Figure 3-2 (a) Equidistance fisheye lens projection and (b) Perspective projection

The Nikon fisheye lens used in this project (with a vertical field of view less than 180 degrees) employs the equidistance projection as illustrated in Figure 3-2 (a) (Hirota et al., 2006). As a comparison to show the difference, Figure 3-2 (b) shows the perspective projection for a conventional pin-hole camera producing planar images with the projection equation given by

Perspective projection - $v = f\tan(\varphi)$ **Equation 3-5**

The 2D sensor coordinates are defined as (u, v), where u is the physical distance between the image point and the initial scan-line on the sensor, and v is as explained in the fisheye lens projection. These coordinates are used as auxiliary coordinates for transformation between the different coordinate systems later on.

2) Definition of the image coordinate system (x_i, y_i)

The image coordinate system is defined in pixels for referencing each image point in the output spherical image, where x_i and y_i are the column and row numbers respectively, with the column-axis pointing right and row-axis pointing down (as shown in Figure 3-3). The image can be 'mapped' onto a sphere with each row corresponding to a latitude circle and each column corresponding to a longitude of the sensor prototype sphere.

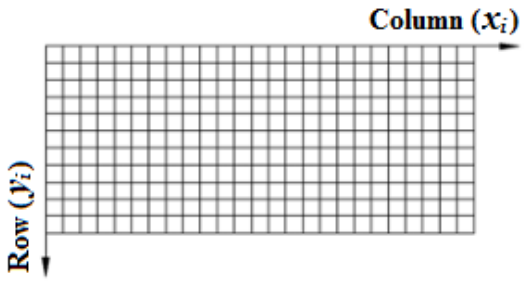


Figure 3-3 Image coordinates of a spherical image

If each pixel on the sensor model has an actual size of s_u by s_v along the horizontal and vertical directions respectively, the transformation between the sensor coordinates and the image coordinates is:

$$u = s_u x_i \quad \textbf{Equation 3-6}$$

$$v_0 - v = s_v y_i \quad \textbf{Equation 3-7}$$

where v_0 is the principal point on the image sensor.

With the sensor model being a sphere, introduction of the azimuth and elevation angles results in u and v being expressed by:

$$u = f\theta \quad \text{Equation 3-8}$$

$$v_0 - v = f\frac{\pi}{2} - f\varphi = f\left(\frac{\pi}{2} - \varphi\right) \quad \text{Equation 3-9}$$

where θ is the azimuth and φ is the elevation of the sensor sphere.

From the above equations x_i and y_i could be derived as:

$$x_i = \frac{f}{s_u}\theta \quad \text{Equation 3-10}$$

$$y_i = \frac{f}{s_v}\left(\frac{\pi}{2} - \varphi\right) \quad \text{Equation 3-11}$$

With the lens properties provided by the camera's manufacturer company, the values of s_u and s_v are known with $s_u = s_v = 8\mu m$ (when the output spherical image is set to its maximum resolution size), and the focal length of the camera $f = 16mm$. Note that the values of s_u and s_v change accordingly when the output image is set to different sizes.

If the output spherical image has the resolution of m-by-n pixels, the image coordinates can also be calculated proportionally by:

$$x_i = \frac{\theta}{2\pi}m \quad \text{Equation 3-12}$$

$$y_i = \frac{\left(\frac{\pi}{2} - \varphi\right)}{\varphi_v}n \quad \text{Equation 3-13}$$

where φ_v is the actual vertical field of view of the camera, $\theta \in [0, 2\pi]$ and $\varphi \in \left[-\left(\varphi_v - \frac{\pi}{2}\right), \frac{\pi}{2}\right]$.

3.2.2 3D Camera Coordinate Systems

The camera coordinate systems are defined with respect to the camera centre. They come in two forms, namely, the Cartesian and spherical coordinates. The definition of and transformation between them are explained here.

1) Definition of the 3D camera coordinate systems

The 3D Cartesian camera coordinates $(\boldsymbol{x^c}, \boldsymbol{y^c}, \boldsymbol{z^c})$ are defined with respect to the origin at the camera centre. With the z^c-axis pointing upwards, x^c-axis and y^c-axis are defined in either a right-handed or a left-handed coordinate system, to reference each

point on the spherical sensor model. For equation consistency, the choice of using a right-handed or left-handed coordinate system depends on the camera's rotation direction and this is explained later on.

The 3D spherical camera coordinates $(\boldsymbol{\theta}, \boldsymbol{\varphi}, \boldsymbol{r})$ are defined for each point on the sensor model using two incident angles and the radius of the spherical sensor. The angles, θ and φ, are the azimuth and elevation angles, respectively, as introduced in the previous section, and r is the radius of the spherical sensor model (focal length). Angle θ is defined so that the initial position of the rotation is at the first scan-line of the sphere passing through (1,0,0) and φ is defined against the x^c-y^c plane (see Figure 3-4 below).

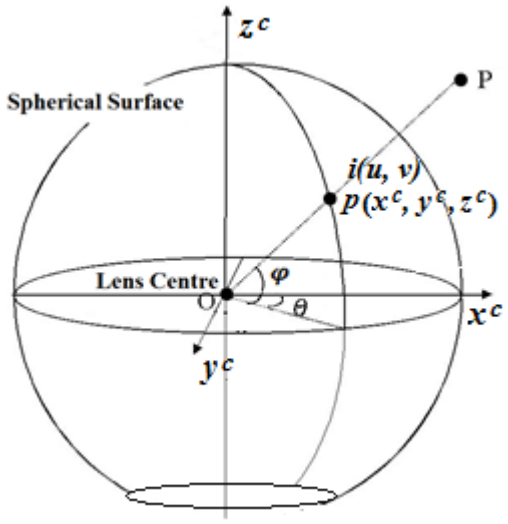


Figure 3-4 3D camera coordinates

2) The 3D left-handed and right-handed coordinate system

The camera head, controlled by the operating software, can rotate in either a clockwise or an anti-clockwise direction when one looks down the camera head from above.

If the camera rotates in a clockwise direction, the Cartesian camera coordinates are defined as a 3D left-handed coordinate system, where the optical centre of the lens is aligned with the origin of the coordinates denoted by o. The angle θ in the spherical coordinates is defined in a clockwise direction against the positive x^c-axis as shown in Figure 3-5.

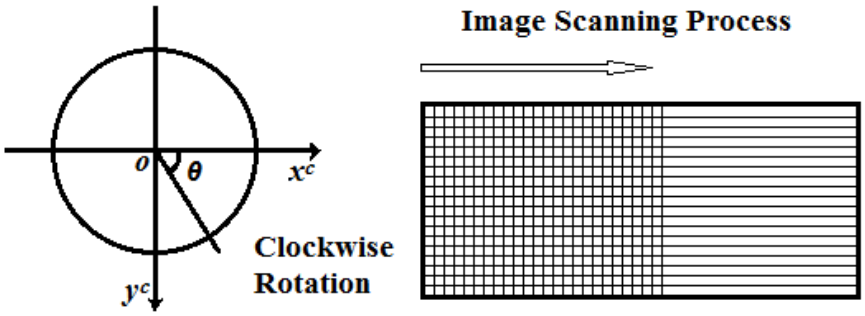


Figure 3-5 Coordinates of clockwise camera rotation (bird's eyes view)

If the camera rotates in an anti-clockwise direction, the Cartesian camera coordinates are defined as a 3D right-handed coordinate system. The angle θ in the spherical coordinates is defined in an anti-clockwise direction against the positive x^c-axis as shown in Figure 3-6.

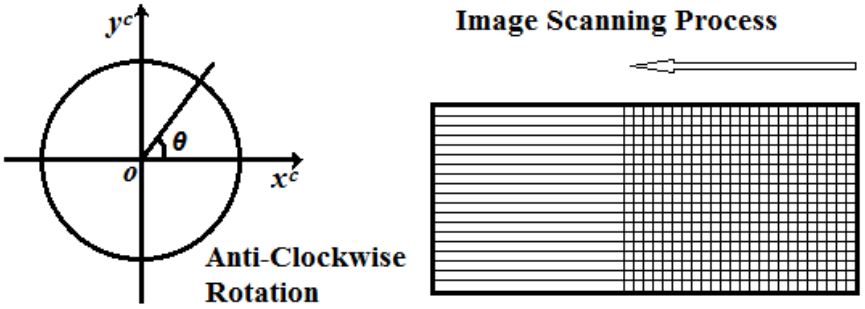


Figure 3-6 Coordinates of anti-clockwise camera rotation (bird's eyes view)

The purpose of the above definitions is to yield consistent transformation equations between the Cartesian coordinates and the spherical coordinates for either a left-handed or a right-handed coordinate system.

3) Transformation between the Cartesian and the spherical camera coordinates

For each of the points on the spherical sensor model, the relationship between its Cartesian coordinates and spherical camera coordinates can be derived as:

$$x^c = r \cdot cos(\varphi)\cos(\theta) \qquad \textbf{Equation 3-14}$$

$$y^c = r \cdot cos(\varphi)\sin(\theta) \qquad \textbf{Equation 3-15}$$

$$z^c = r \cdot sin(\varphi) \qquad \textbf{Equation 3-16}$$

And reversely,

$$\theta = \tan^{-1}\frac{y^c}{x^c} \qquad \textbf{Equation 3-17}$$

$$\varphi = \tan^{-1}\frac{z^c}{\sqrt{x^{c2} + y^{c2}}} \qquad \textbf{Equation 3-18}$$

$$r = \sqrt{x^{c2} + y^{c2} + z^{c2}} \qquad \text{Equation } \textbf{3-19}$$

Note that $r = 1$ if a unit spherical sensor model is assumed.

3.2.3 3D World Coordinate System

The 3D world coordinate system (X^w, Y^w, Z^w) is defined for points in the real world. The transformation between points in the Cartesian camera coordinates and the world coordinates can be described linearly as $[R, \boldsymbol{t}]$ which consists of a 3-by-3 rotation matrix denoted by *R* and a 3-by-1 translating vector denoted by ***t***, and

$$\begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = [\, R \quad \boldsymbol{t}] \cdot \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} \qquad \textbf{Equation 3-20}$$

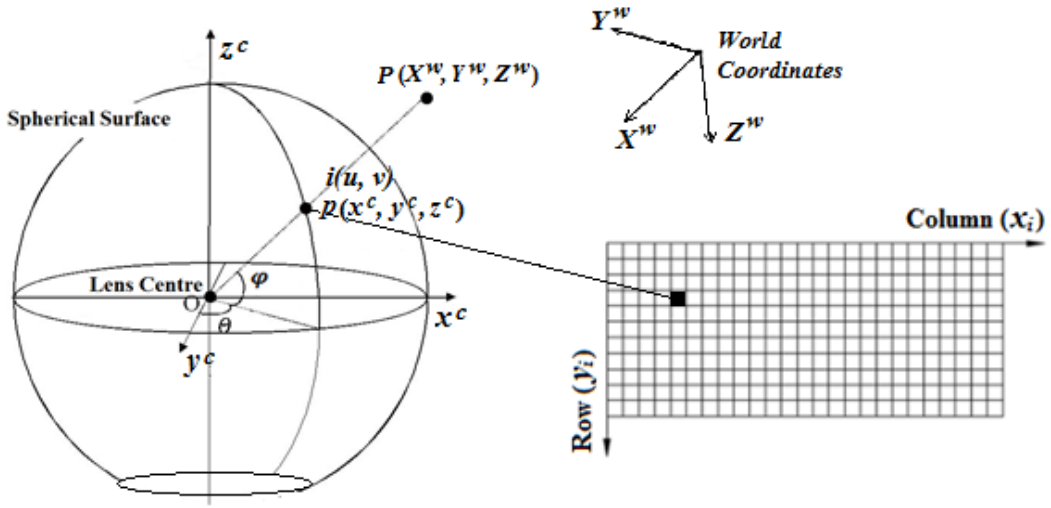


Figure 3-7 Transformation among all coordinate systems

Figure 3-7 above illustrates the transformations among all the coordinate systems described in this section. The transformation between the image coordinates and the world coordinates can be derived by linking the equations given. These equations are involved in the computation for the proposed calibration methodology which will be explained in Section 3.4.

3.3 Calibration Set-up and Image Pre-processing

The proposed calibration method is performed by taking only one spherical image of the checkerboard grid patterns on multiple planes. The image is then pre-processed for the purpose of feature point extraction.

3.3.1 The Calibration Object

A special calibration object based on a five-sided box was constructed. As shown in Figure 3-8, the calibration box has an inner physical size of 57 x 57 x 27cm (height x width x depth) and its insider faces are covered by checkerboard patterns with each square having a size of 3 x 3cm.

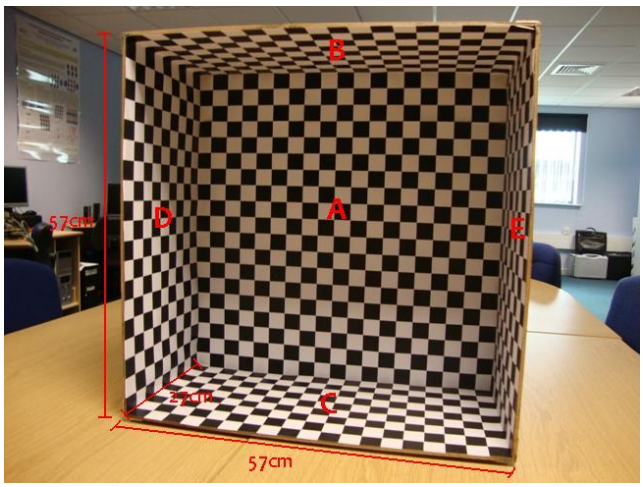


Figure 3-8 Five-sided calibration object

In the image acquisition set-up, the camera is placed around the middle of the calibration box facing directly towards the front panel denoted by **A** in Figure 3-9 (a), with the panels above and below the camera denoted by **B** and **C**, and the panels on the left and right sides of the camera denoted by **D** and **E**. Furthermore, the camera head is dipped into the calibration box (as near to panel **A** as possible) to ensure a hemisphere coverage by the checkerboard patterns. During image acquisition, the horizontal field of view for the SpheroCam® is set to 180 degrees (half of a full scanning cycle) to capture only half of its surrounding.

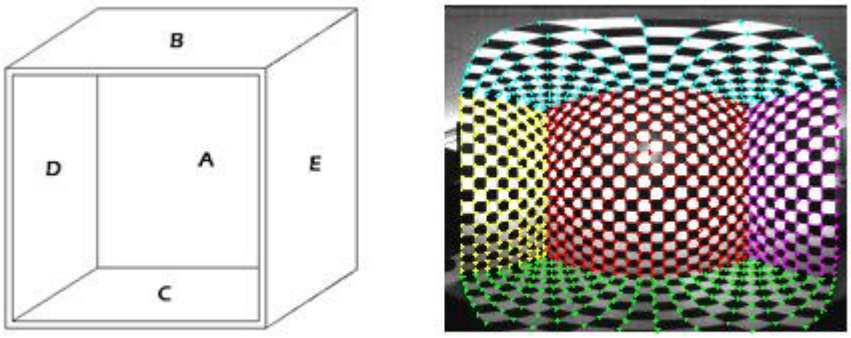


Figure 3-9 (a) Calibration box panel notations and (b) Hemispherical calibration image

Figure 3-9 (b) is an example of the hemisphere calibration image acquired (640 x 785 pixels), where corner points of the checkerboard squares on five different sides of the calibration box are highlighted using different colours. It should be apparent that the proposed calibration method is based on a reasonable assumption of symmetrical hemisphere. Since the bottom panel (panel **C**) cannot be captured fully by the camera due to the limitation of its vertical field of view, the actual vertical field of view, φ_v, is considered as a very important camera parameter, which is however commonly approximated to be 180 degrees in many other researches (Parian and Gruen, 2003). This has led to the developed calibration method to determine the value of φ_v.

3.3.2 Image Pre-processing

To perform calibration, the feature corner points in the output hemispherical image need to be marked and recorded by their coordinates. Together with their physical dimensions in the world coordinates, the computation of the camera parameters can then be carried out.

The feature corner points $\boldsymbol{p_i} = [x_i, y_i]^T$ on the calibration image can be found automatically by common corner detection methods such as Harris corner detection (Harris and Stephens, 1988). To determine whether a pixel is a corner point, its value is compared with the surrounding pixels to find the changes of intensities. If the change is significant along certain directions (depending on the definition of corners in degrees), the pixel is considered as a corner.

Harris corner detector shifts a small window all over the image to determine the change in intensities of the windowed pixels. If the shifting yields no change in all directions,

the windowed area is considered as a 'flat' area with no feature of interests; if the shifting yields no change along one direction but not the others, an 'edge' is detected along that direction; and if the shifting yields significant change in all directions, a 'corner' is detected.

Change of intensity for the shift $[u, v]$ can be described as follows:

$$E(u,v) = \sum_{x,y} w(x,y)[I(x+u, y+v) - I(x,y)]^2 \qquad \textbf{Equation 3-21}$$

where $w(x,y)$ is a window function, $I(x+u, y+v)$ is the shifted intensity and $I(x,y)$ is the original intensity.

If a patch of image pixels is constant with not much intensity change, the value of E would be near 0. If the patch has features, the value of E would be large.

The function $I(x+u, y+v)$ could be approximated by the first order Taylor series to give

$$I(x+u, y+v) \approx I(x,y) + uI_x(x,y) + vI_y(x,y) \qquad \textbf{Equation 3-22}$$

where I_x and I_y are the first partial derivative of x and y respectively.

Therefore Equation 3-21 can be written as:

$$\begin{aligned} E(u,v) &\cong \sum_{x,y} w(x,y)\big[I(x,y) + uI_x(x,y) + vI_y(x,y) - I(x,y)\big]^2 \\ &= \sum_{x,y} w(x,y)\big[u^2 I_x(x,y)^2 + 2uvI_x(x,y)I_y(x,y) + v^2 I_y(x,y)^2\big] \\ &= [u,v] \sum_{x,y} w(x,y) \begin{bmatrix} I_x(x,y)^2 & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y(x,y)^2 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \\ &= [u,v]\, Q \begin{bmatrix} u \\ v \end{bmatrix} \qquad \textbf{Equation 3-23} \end{aligned}$$

where Q is the square difference matrix given by

$$Q = \sum_{x,y} w(x,y) \begin{bmatrix} I_x(x,y)^2 & I_x(x,y)I_y(x,y) \\ I_x(x,y)I_y(x,y) & I_y(x,y)^2 \end{bmatrix}$$

and it only comprises of the window function (rectangular or Gaussian) and the products of components of the gradients I_x and I_y. These gradient vectors can be treated as a set of derivatives of x and y with a centre of mass defined at $(0,0)$.

The corner measure response can then be determined using:

$$W = \det(Q) - k(trace(Q))^2 \qquad \textbf{Equation 3-24}$$

where k is a small constant called the sensitivity factor. If the value of W is large, a corner is detected. The smaller the value of k is, the more likely the sharp corners can be detected.

The above method has been applied to the hemispherical image for corner detection, and a difficulty has been observed in selection of an optimum value for the sensitivity factor due to a mixture of angles to be dealt with including not only right angle corners close to 90 degrees on panels **A**, **C** and **D**, but also various rhombic corners much more or less than 90 degrees on panels **B** and **C**. To ensure the quality of the calibration and all the feature points on the patterns being included for accuracy assessment purpose, the corner points are manually adjusted after the detection.

3.4 Methodology for Computation of Camera Parameters

The calibration is to recover the camera parameters based on the equations linking the physical points on the calibration object and their image. As shown in Figure 3-10, a pair of points $\boldsymbol{P}$ and $\boldsymbol{p_i}$ could be linked through a matrix which is the projection matrix. The aim here is to recover the projection matrix as well as the vertical field of view as an additional parameter.

The fisheye lens delivered with the camera is precisely calibrated by the manufacturer company to compensate the lens imperfection, and the correction details (see Appendix A) are provided to the camera by its operating software. Therefore the following lens errors affecting the projection model mentioned in some other researches are excluded: radial lens distortion, shift of principal point, tilt and inclination of the linear array with respect to the rotation axis, and eccentricity of the projection centre from the origin of the sensor coordinate system (Schneider and Maas, 2004).

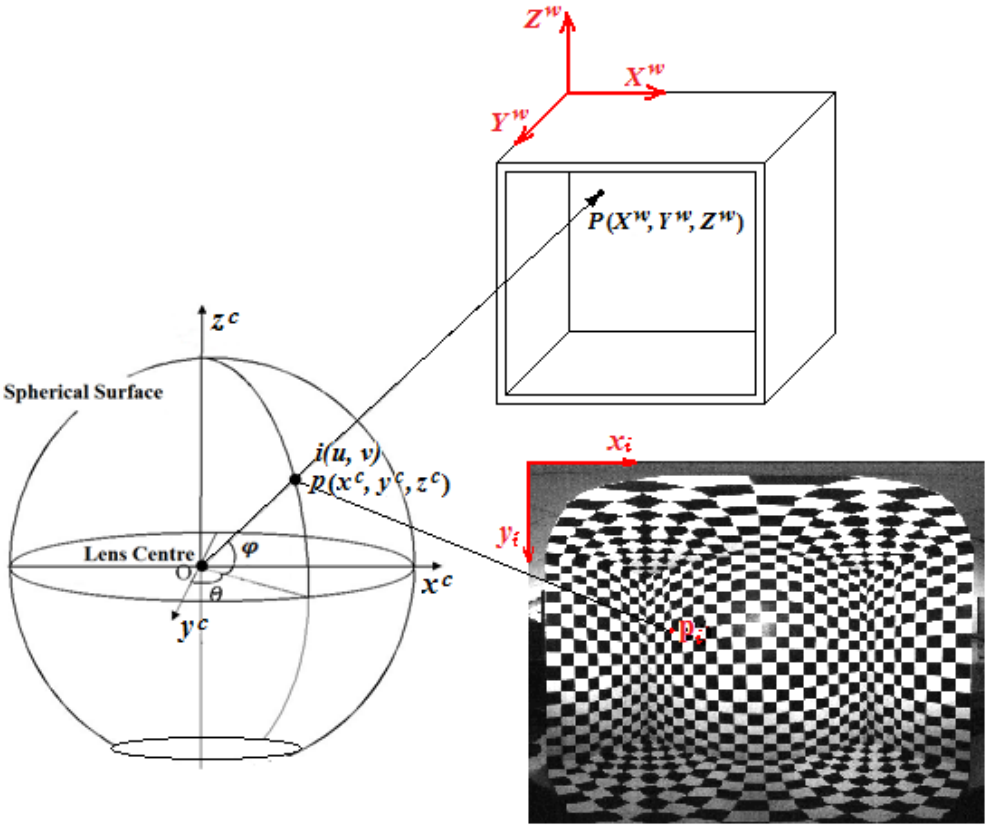


Figure 3-10 Methodology for calibration

3.4.1 Iterative Computation of Camera Parameters

By rearranging Equation 3-12 and Equation 3-13, the two incident angles of each point on the spherical sensor can be estimated from its image coordinates as follows:

$$\theta = \frac{x_i}{m} 2\pi \qquad \textbf{Equation 3-25}$$

$$\varphi = \frac{\pi}{2} - \frac{y_i}{n} \varphi_v \qquad \textbf{Equation 3-26}$$

As the scanned image is half a spherical image, the image size along the horizontal direction, m_0, is half the value of a full scan and the actual horizontal field of view $\theta_0 \in [0, \pi]$, Equation 3-25 is re-written as:

$$\theta_0 = \frac{x_i}{m_0} \pi \qquad \textbf{Equation 3-27}$$

Since the image coordinates of each corner point, (x_i, y_i), are known and the image size, m_0 and n, are known, if the vertical field of view, φ_v, is also known (or set to a value for the iterative calculation process), the angles, θ_0 and φ, can be computed from Equation 3-27 and Equation 3-26 for each feature corner point. Using Equation 3-14,

Equation 3-15 and Equation 3-16, the 3D points in their Cartesian camera coordinates (x^c, y^c, z^c) could then be calculated from θ_0 and φ obtained. If the radius of the sensor sphere, r, is set to one for modelling a unit spherical sensor, then the projection of physical feature points (X^w, Y^w, Z^w) to their camera coordinates (x^c, y^c, z^c), in their homogeneous form can be written as:

$$\begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = M \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} \qquad \textbf{Equation 3-28}$$

where matrix M is the projection matrix to be recovered up to a scale factor and it has a size of 3 by 4. In order to solve M, a skew-symmetric matrix of $[x^c, y^c, z^c]^T$ is defined as $\begin{bmatrix} 0 & -z^c & y^c \\ z^c & 0 & -x^c \\ -y^c & x^c & 0 \end{bmatrix}$ and since

$$\begin{bmatrix} 0 & -z^c & y^c \\ z^c & 0 & -x^c \\ -y^c & x^c & 0 \end{bmatrix} \begin{bmatrix} x^c \\ y^c \\ z^c \end{bmatrix} = 0 \qquad \textbf{Equation 3-29}$$

where $[x^c, y^c, z^c]^T$ can be substituted by Equation 3-28 so that Equation 3-29 becomes

$$\begin{bmatrix} 0 & -z^c & y^c \\ z^c & 0 & -x^c \\ -y^c & x^c & 0 \end{bmatrix} M \begin{bmatrix} X^w \\ Y^w \\ Z^w \\ 1 \end{bmatrix} = 0 \qquad \textbf{Equation 3-30}$$

Given a set of the image coordinates of i checkerboard corner points extracted from the hemisphere calibration image (transformed into their 3D camera coordinates by the method stated above) and their corresponding physical points on the calibration object, a set of equations can be written for Equation 3-30 to form a linear equation system to solve the elements vector $\boldsymbol{u} = [M_{11}, M_{12}, \dots, M_{33}, M_{34}]^T$ of M. The equation system can be written as:

$$A\boldsymbol{u} = 0 \qquad \textbf{Equation 3-31}$$

where

$$A = \begin{bmatrix} 0 & 0 & 0 & 0 & -z_1^c X_1^w & -z_1^c Y_1^w & -z_1^c Z_1^w & -z_1^c & y_1^c X_1^w & y_1^c Y_1^w & y_1^c Z_1^w & y_1^c \\ z_1^c X_1^w & z_1^c Y_1^w & z_1^c Z_1^w & z_1^c & 0 & 0 & 0 & 0 & -x_1^c X_1^w & -x_1^c Y_1^w & -x_1^c Z_1^w & -x_1^c \\ -y_1^c X_1^w & -y_1^c Y_1^w & -y_1^c Z_1^w & -y_1^c & x_1^c X_1^w & x_1^c Y_1^w & x_1^c Z_1^w & x_1^c & 0 & 0 & 0 & 0 \\ & & & & & . & & & & & & \\ & & & & & . & & & & & & \\ & & & & & . & & & & & & \\ 0 & 0 & 0 & 0 & -z_i^c X_i^w & -z_i^c Y_i^w & -z_i^c Z_i^w & -z_i^c & y_i^c X_i^w & y_i^c Y_i^w & y_i^c Z_i^w & y_i^c \\ z_i^c X_i^w & z_i^c Y_i^w & z_i^c Z_i^w & z_i^c & 0 & 0 & 0 & 0 & -x_i^c X_i^w & -x_i^c Y_i^w & -x_i^c Z_i^w & -x_i^c \\ -y_i^c X_i^w & -y_i^c Y_i^w & -y_i^c Z_i^w & -y_i^c & x_i^c X_i^w & x_i^c Y_i^w & x_i^c Z_i^w & x_i^c & 0 & 0 & 0 & 0 \end{bmatrix}$$

Hence the elements of M can be computed by at least $i = 4$ pairs of correspondences using singular value decomposition (SVD) to solve the simultaneous system. If $A = UDV^T$, the last column of V corresponds to the zero singular value of A which is the solution of vector $\boldsymbol{u}$. When more corresponding points are employed, the elements of M can be solved by more equations satisfying Equation 3-31 in a least square error manner, where the result is considered to be more accurate.

For estimation of the vertical field of view φ_v, its value is increased from minimum to maximum expected angles in Equation 3-26 to yield different φ values for each given y_i (the increase can be made in a small angular interval in order to give a sufficient angular resolution). Since different φ values will result in different sets of (x^c, y^c, z^c) being generated from Equation 3-14, Equation 3-15 and Equation 3-16, a number of different camera projection matrices will be generated for M. By applying each camera projection matrix estimated and the corresponding φ_v value used in estimation, the physical positions of the checkerboard feature corners are re-projected onto the image plane. If the re-projected image points are denoted by $p_i' = [x_i', y_i']^T$, then the root-mean-square (rms) error between the re-projected image points and the corner points appeared in the hemisphere calibration image p_i is given by

$$RMS\ Error = \frac{1}{k}\sum_{i=1}^{k}\sqrt{(x_i - x_i')^2 + (y_i - y_i')^2} \qquad \textbf{Equation 3-32}$$

where k denotes the total number of the checkerboard feature corner points used. With the rms error varying for different φ_v values, it forms the cost function of the iterative search for the minimum rms error with the corresponding φ_v value and the camera projection matrix identified to represent the camera parameters.

3.5 Experimental Results and Accuracy Assessment

Presented in this section are some experimental results to demonstrate the achievable accuracy as well as the effect on the accuracy due to the numbers of control points and planes used.

3.5.1 RMS Error and Vertical FOV

Figure 3-11 shows the characteristics of the rms error between the re-projected image points and the corner points appeared in the hemisphere calibration image as a function of φ_v for determination of the vertical field of view. It is generated based on all the corner points available in all 5 planes as shown in Figure 3-9(b) with φ_v increasing from 155 to 175 degrees in a step of 0.1 degree. The minimum rms error of 1.2155 pixels is seen to occur at 163.5 degrees (or $109/120\,\pi$), resulting in it being taken as the vertical field of view of the spherical camera.

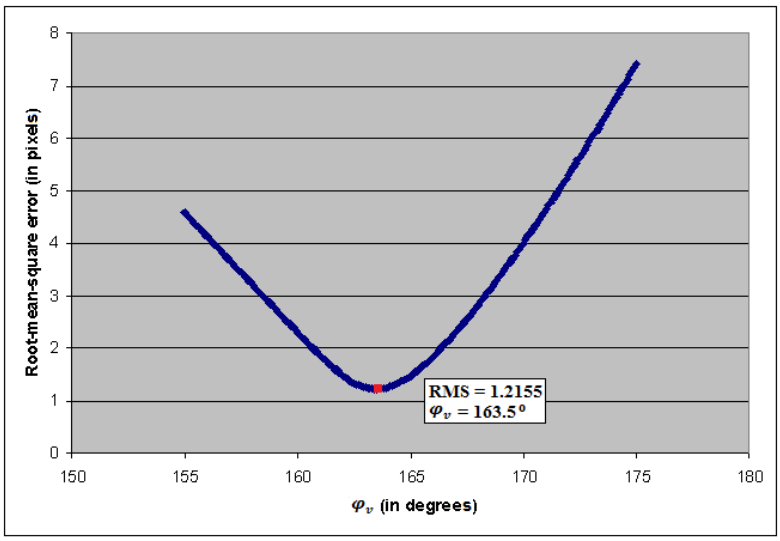


Figure 3-11 Estimation of φ_v against RMS error

3.5.2 Different Plane Combinations

Referring to the hemisphere calibration image shown in Figure 3-9(b), the first investigation on calibration accuracy was based on all checkerboard corner points available in different plane combinations. The use of corner points in any single plane for calibration was found to yield high rms error (between 200 and 400 pixels). Table 3-1 lists the results obtained for non-coplanar corners points from the different number of planes from 2 to 5 with all possible combinations, and the lowest rms error of 1.2155 pixels with $\varphi_v = 163.5$ degrees is seen to be generated by all the available corner points

in all the planes. In terms of the number of planes used for calibration, an increase in the number of planes, resulting in a higher number of corner points involved in the calibration process, is seen to reduce the variation of rms errors in each group. In terms of the number of control points used for calibration within each group, an increase in the number of control points by using the large front plane does not lead to a decrease in the rms error. For the group using all available corner points in 2 planes, the lowest and the highest rms errors (1.2837 pixels and 2.0404 pixels) are seen to be generated by using similar numbers of corner points, with the combination of two side planes (**D** and **E** planes) involving the lowest number of corner points to produce the lowest rms error. The same outcome is also seen from the group using all available corner points in 3 planes. The lowest and the highest rms errors (1.2159 and 1.4034 pixels) are again generated by using similar number of corner points, with the combination of two side plane plus the bottom plane (**D**, **E** and **C** planes) involving the lowest number of corner points to produce the lowest rms error. Although the same could not be said for the group using all available corner points in 4 planes, the difference between the rms errors in this group is much smaller (less than 0.0907 pixels).

No. of planes	Plane combination	No. of corner points	Error in pixels	φ_v in degrees
2	AB	580	1.5847	164.7
	AC	570	1.3102	163
	AD	562	1.3163	163.3
	AE	562	1.412	163.6
	BC	350	1.5055	163.9
	BD	342	1.7176	163.5
	BE	342	2.0404	164.2
	CD	332	1.8567	163.5
	CE	332	1.8958	163.8
	DE	324	1.2837	163.6
3	ABC	750	1.2272	163.6
	ABD	742	1.3113	163.8
	ABE	742	1.359	163.7
	ACD	732	1.2833	163.2
	ACE	732	1.286	163.4
	ADE	724	1.2666	163.5
	BCD	512	1.4034	163.9
	BCE	512	1.3852	163.8
	BDE	504	1.3771	163.9
	CDE	494	1.2159	163.4
4	ABCD	912	1.222	163.5
	ABCE	912	1.2192	163.5
	ABDE	904	1.2854	163.6

	ACDE	894	1.2222	163.4
	BCDE	674	1.3099	163.6
5	ABCDE	1074	1.2155	163.5

Table 3-1 Assessment 1 on plane combinations

3.5.3 Different Plane Combinations with Less Feature Points

The second investigation on calibration accuracy was based on half of the corner points available in different plane combinations, and Table 3-2 lists the results obtained based on 10 sets of corner points randomly selected from each plane. Comparing Table 3-2 with Table 3-1, there are various similarities between them which include the combination of planes to yield the lowest and highest average errors for the groups using corner points in 2 and 3 planes. The lowest average errors in these two groups are again seen to be produced with the least number of corner points without using the front plane (**A** plane). The trend in error variation is similar between the two investigations. The maximum error is seen to be just over 2 pixels based on corner points selected from 2 planes and is reduced to well below 2 pixels by using corner points in 3 or more planes.

No. of planes	Plane combination	No. of corner points	Averaged error in pixels	φ_v in Degrees	Max error in pixels	Standard deviation
2	AB	290	1.57343	164.58	1.7072	0.055
	AC	285	1.34333	163.03	1.3891	0.030
	AD	281	1.37834	163.32	1.4663	0.050
	AE	281	1.42485	163.65	1.5939	0.095
	BC	175	1.53478	163.87	1.5789	0.032
	BD	171	1.7024	163.52	1.8651	0.091
	BE	171	1.97444	164.23	2.1044	0.100
	CD	166	1.89414	163.52	2.1264	0.156
	CE	166	1.96923	163.53	2.1062	0.106
	DE	162	1.33643	163.5	1.389	0.043
3	ABC	375	1.25144	163.51	1.2875	0.022
	ABD	371	1.33164	163.76	1.3865	0.029
	ABE	371	1.37083	163.69	1.417	0.026
	ACD	366	1.29771	163.16	1.3346	0.017
	ACE	366	1.31731	163.43	1.3516	0.022
	ADE	362	1.26916	163.49	1.308	0.017
	BCD	256	1.39906	163.81	1.4559	0.032
	BCE	256	1.39379	163.82	1.4424	0.024
	BDE	252	1.39087	163.86	1.435	0.034
	CDE	247	1.22742	163.43	1.2588	0.012
4	ABCD	456	1.22658	163.47	1.2556	0.013

	ABCE	456	1.23557	163.42	1.2539	0.012
	ABDE	452	1.29772	163.54	1.3234	0.017
	ACDE	447	1.24246	163.37	1.2666	0.015
	BCDE	337	1.31627	163.57	1.3412	0.018
5	ABCDE	537	1.22322	163.46	1.2499	0.011

Table 3-2 Assessment 2 on plane combinations

3.5.4 Number of Feature Points Used

With the error reduced by using corner points in increasing number of planes, the third investigation on calibration accuracy was based on the number of corner points from all five planes, and Figure 3-12 shows the results obtained based on 10 sets of randomly selected corner points. Although the average, maximum and minimum errors as well as the standard deviation are seen to increase as the number of the corner points reduces, the proposed method is seen to be able to provide a calibration error well within 2 pixels by using 18 corner points or more.

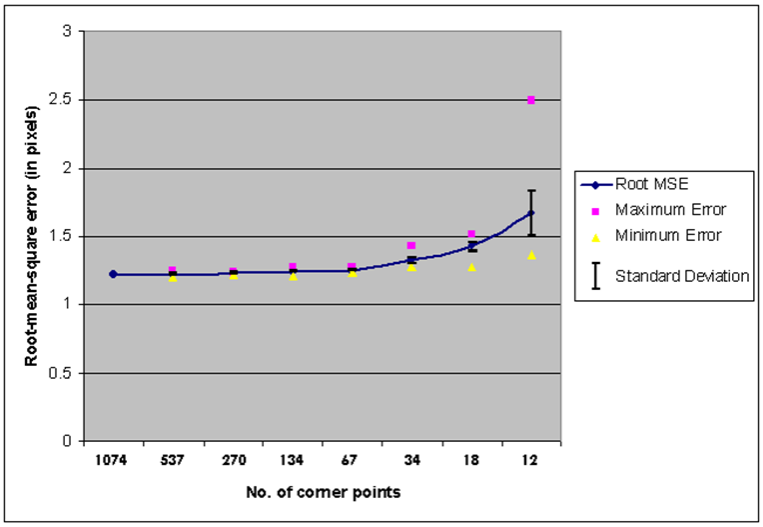


Figure 3-12 Assessment on no. of feature points used

3.6 Conclusion

In this chapter, a simple method for calibration of a rotating line spherical camera has been presented. It is based on a single hemispherical image acquired from a five-sided open box with the insider faces covered by checkerboard patterns. By including the vertical field of view as an additional camera parameter, the proposed method is seen to provide good calibration accuracy and robustness without requiring a large number of

corner points as shown by various experiments. In particular, the influence of the corner points selected from different combination of planes on calibration accuracy has been shown, the achievable accuracy is seen to be near to 1 pixel, and stable results are seen to be obtained consistently with an error bound of well within 2 pixels by using corner points selected from at least 3 different planes.

Chapter 4 VISUALISATION BASED ON ONE SPHERICAL IMAGE

4.1 Introduction

Computer simulated visualisation has been developed fast due to the ever increasing performance and the decreasing costs of the hardware equipment. Visualisation is traditionally achieved by creating 3D computer graphic objects and new views are then rendered directly from the geometric models. The difficulty of this geometry-based modelling is the complexity in generating polygons to model real world objects and reach realism, also the unavailability of an accurate lighting and reflection model.

The other way to achieve visualisation is to use images taken from the real world and new views are rendered by processing the acquired images. The disadvantage is the lack of accuracy and freedom of movement, but the advantage is having photorealistic views at positions different from where the images were taken.

This chapter presents generation of new views for environment visualisation from one spherical image. The SpheroCam® is used to take one spherical image at a certain image acquisition point. The surrounding is scanned and projected onto the camera sensor to produce a full spherical image, except a bottom circular area where the camera stands. This results in the surrounding scene to be modelled by a unit sphere mapped with the scanned image. A Graphical User Interface (GUI) is created for the user to visualise the sphere-mapped image. Unlike the *SpeheroViewer* described in Sub-section 2.3.2 and Apple's Quick Time VR (Kitchens, 1998), which do not allow the user to move away from the original camera viewpoint when visualising the sphere-mapped image, the developed GUI allows the user to change the viewing position, viewing direction and field of view, and renders correct views based on one sphere-mapped image for immersive and interactive environment visualisation.

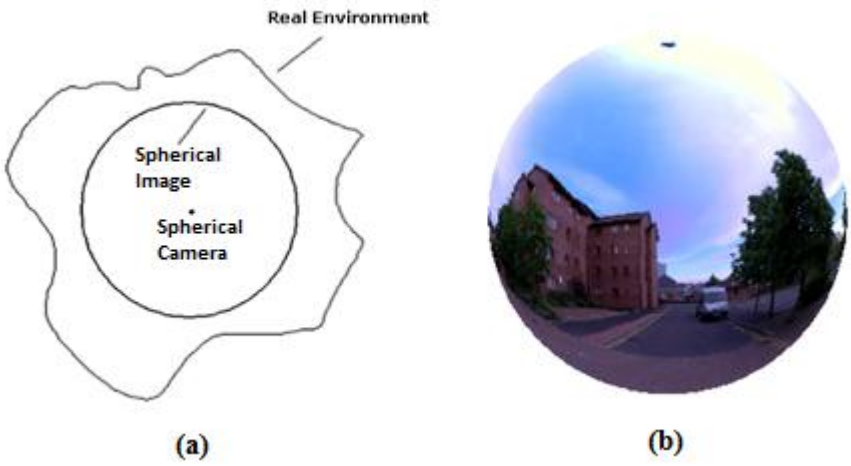


Figure 4-1 (a) Environment modelling and (b) Sphere-mapped image

Figure 4-1(a) shows the plan view of the environment modelling principle. The SpheroCam® is placed roughly at the centre of a certain scene and it takes a spherical image of the environment. A visualisation model of the environment is created by mapping this image onto a unit sphere as shown in Figure 4-1(a) and (b). A virtual camera is then created and placed within the sphere, to capture and provide the view which should be seen by the user. New views are generated each time as the user changes the viewing position and direction. Figure 4-2 illustrates the system flow chart. The main steps of the visualisation are to simulate the virtual camera and to interpolate the appropriate images for the user's viewing screen. The interpolation step also involves image distortion correction, and this is explained in Section 4.3.

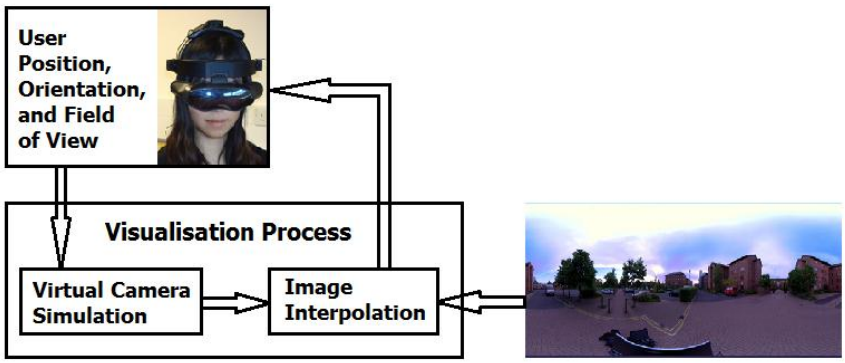


Figure 4-2 Visualisation system flowchart

The rest of the chapter is organised as follows. The algorithms developed for visualisation are explained in Section 4.2, with viewing results shown. Section 4.3 discusses the image distortion inherent with the sphere visualisation model and the

methodology for distortion correction. The results and analysis are presented in Section 4.4.

4.2 Visualisation by Virtual Camera and Image Interpolation

The virtual camera is modelled as a simple pin-hole camera following the law of refraction for thin lenses (Ray, 2002). For the part of the sphere wall captured by the virtual camera according to its position and viewing direction as well as field of view, the corresponding pixels on the spherical image are located and interpolated to generate an appropriate image for display on the user's viewing screen.

4.2.1 Virtual Camera Simulation

A camera with perspective projection is very similar to the human visual system and it is used to simulate the capture of the view which should be seen by the user when visualising a sphere-mapped image. The principal elements of a digital camera are the lens and the sensor, acting similarly as the lens of human eyes to refract lights and the retina to project the image. The lens collects and refracts lights passing through the camera and the sensor converts lights into electrical charges to give each pixel a colour value. Figure 4-3 below shows the geometric model of a simple pin-hole camera. For an object point ***P*** seen by the camera, one stream of light collected by the optical centre (camera origin) passes through its sensor, leaving ***p*** (the projection of ***P***) onto the image sensor plane. This applies to all the points captured by the camera within its field of view and the contents in the image plane form the output image.

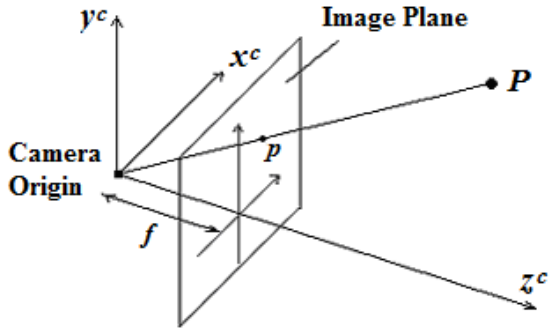


Figure 4-3 Virtual camera model

The projection from point ***P*** to ***p*** is accomplished through the camera's projection matrix. As shown in Figure 4-4, ***P*** and ***p*** are a pair of corresponding points under

different coordinate systems: point $\boldsymbol{P}$ with coordinates (X^w, Y^w, Z^w) in the world coordinate system and point $\boldsymbol{p}$ with coordinates (x_i, y_i) in the image coordinate system. Similar to that explained in Chapter 3, where the projection matrix for spherical cameras is shown to transform the world coordinates of real points to their corresponding 3D camera coordinates on a sphere, in here it transforms 3D points to their corresponding image coordinates in a 2D image. Again, the transformation is achieved via the camera coordinate system defined within the camera.

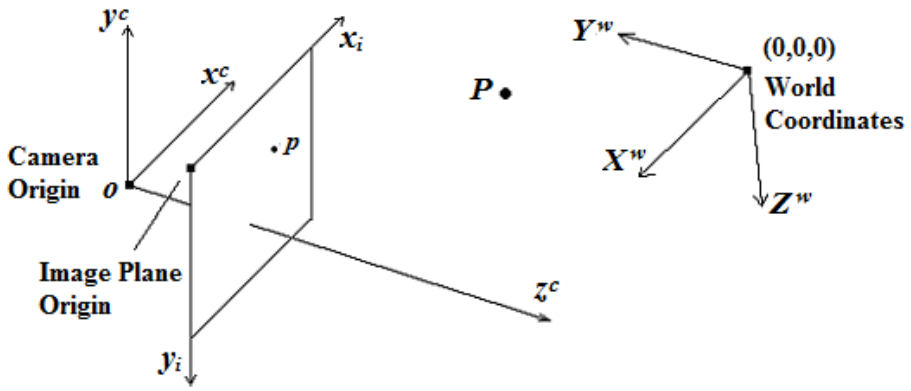


Figure 4-4 Virtual camera projection geometry

As shown in Figure 4-4, the camera coordinate system o-x^c-y^c-z^c is defined by the camera origin and the principal axis o-z^c pointing at and passing through the centre of the image plane. The o-x^c and o-y^c axes define a plane that is perpendicular to the principal axis. The image coordinate system is defined based on the output image, with its origin situated at the top left corner of the plane and with the column and row numbers of any image point defined as x_i and y_i, respectively. The world coordinate system is defined with reference to any known point in the world. In this case, as the environment is modelled by a unit sphere where point $\boldsymbol{P}$ is always situated on, the origin of the world coordinates is defined at $(0, 0, 0)$, namely, the centre of the unit sphere.

The projection matrix consists of two matrices, namely, the extrinsic matrix and the intrinsic matrix. The extrinsic matrix transforms the world coordinate system to the camera coordinate system; and the intrinsic matrix transforms the camera coordinate system to the image coordinate system.

The extrinsic matrix contains parameters related to the position and orientation of the camera, i.e. where the camera is and which direction the camera is looking at, in the world coordinate system. The matrix form can be written as:

$$M_{est} = \begin{bmatrix} R & \boldsymbol{t} \\ \mathbf{0} & 1 \end{bmatrix}$$ **Equation 4-1**

where R is a 3-by-3 rotation matrix containing orientation parameters and $\boldsymbol{t}$ is a 3-by-1 translation vector containing the position parameters.

In environment visualisation, matrix R is set by the user to indicate the viewing direction and is given by

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\varphi & -\sin\varphi \\ 0 & \sin\varphi & \cos\varphi \end{bmatrix} \cdot \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ **Equation 4-2**

where θ and φ are the angles defined as panning and tilting (same as the azimuth and elevation angles defined in Chapter 3). Although a general 3D rotation matrix often has another degree of freedom which is rolling, this parameter is ignored because it is not a normal human observation movement.

Vector $\boldsymbol{t}$ translates the camera centre away from the origin of the world coordinates at $(0, 0, 0)$ and the position of the camera centre is set by the user to indicate the viewpoint.

The intrinsic matrix is an inner parametric matrix of the camera. It contains five parameters which are:

f	–	focal length
s_x , s_y	–	pixel size of the sensor
o_x , o_y	–	centre of the image plane

These parameters are set to some fixed values to model the virtual camera placed inside the unit sphere. Specifically, with respect to the sphere with radius of 1, the focal length is set to 0.1, pixel size of the sensor is set to 0.0005 by 0.0005, and the size of the image plane is set to 120 rows by 160 columns with the centre of the image plane at (61, 81). The camera is modelled as an ideal one so no other intrinsic parameters are involved. This results in the intrinsic matrix given by:

$$M_{int} = \begin{bmatrix} {}^{f}/_{s_x} & 0 & o_x \\ 0 & -{}^{f}/_{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix}$$ **Equation 4-3**

In order to control the virtual camera's field of view, the user is allowed to change one parameter from the intrinsic matrix which is the focal length, with the default value set to 0.1. A broader field of view is achieved by reducing the focal length, and vice versa. Another way to change the user's field of view is to change the number of pixels in the virtual camera's image plane (with the default values set to 120 by 160 pixels). However the control of focal length is chosen because changing this achieves a constant zooming in and out effect, which is more straightforward.

4.2.2 Image Interpolation

At the start of environment visualisation, the virtual camera is first positioned at the centre of the sphere $(0, 0, 0)$ as the default viewpoint. The user is allowed to change the viewing position and direction to visualise the sphere-mapped image by moving the 3D coordinates of the virtual camera centre, $\boldsymbol{C}$, inside the unit sphere and controlling the azimuth (panning) and elevation (tilting) angles, θ and φ.

In order to achieve correct image interpolation, the first step is for the virtual camera representing the user view to capture the correct part of the mapped scene from the inner surface of the unit sphere. This involves identification of the area on the inner sphere surface that should be covered by the rays projected from each point in the image plane of the virtual camera and intersecting with the sphere wall. The second step is to transform the inner sphere surface points identified in the 3D camera coordinate system of the spherical sensor model to their corresponding image coordinates by using the knowledge from Chapter 3 and these coordinates are then used for interpolation of the original spherical image to give a correct image for display in the user's viewing window.

In the first step of locating the points on the sphere being captured by the virtual camera, each captured point $\boldsymbol{P} = (X^w, Y^w, Z^w)$ on the unit sphere is defined by a light ray that is originated from the camera centre, goes through the corresponding image point $\boldsymbol{p}$ on the image plane, and intersects the sphere wall. Using the geometry illustrated in Figure

4-5, with the centre of the sphere denoted by $\boldsymbol{O}$ and the camera position denoted by $\boldsymbol{C}$, point $\boldsymbol{P}$ can be determined by applying the triangle law of vector addition, i.e. vector $\overrightarrow{\boldsymbol{OP}}$ is given by:

$$\overrightarrow{\boldsymbol{OP}} = \overrightarrow{\boldsymbol{OC}} + \overrightarrow{\boldsymbol{CP}} \qquad \textbf{Equation 4-4}$$

While vector $\overrightarrow{\boldsymbol{OC}}$ is known from the coordinates of the virtual camera centre $\boldsymbol{C}$ according to the user viewpoint, vector $\overrightarrow{\boldsymbol{CP}}$ needs to be derived and it can be expressed in terms of its modulus (length) denoted by d and unit direction vector denoted by $\boldsymbol{v}$, respectively. While length d is the distance between the virtual camera centre and point $\boldsymbol{P}$ on the sphere, $\boldsymbol{v}$ is a unit length vector originated from the world coordinate origin and indicating the direction of $\overrightarrow{\boldsymbol{CP}}$.

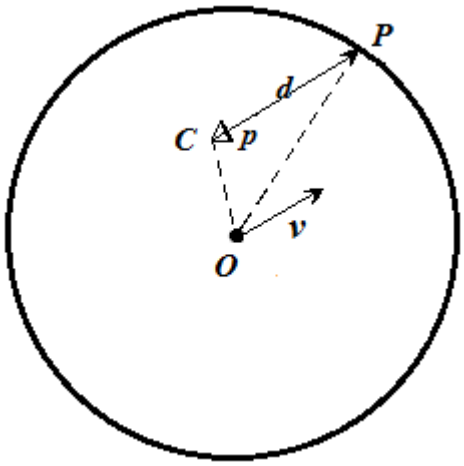


Figure 4-5 Locating *P* on the sphere wall captured by virtual camera (plan view)

Vector $\boldsymbol{v}$ needs to be determined for each image point originated from the virtual camera centre $\boldsymbol{C}$. Given the size of the image plane, the coordinates of every image point $\boldsymbol{p}$ are known, and they can be transformed back to their camera coordinates by applying the inverse of the intrinsic matrix M_{int}. The unit direction vector of these camera coordinates in the world coordinate system is then given by a further inverse transformation based on rotation matrix R and normalisation to a unit vector. These operations to obtain the unit vector in the world coordinate system for each image point $\boldsymbol{p} = (x_i, y_i)$ can be expressed as:

$$\boldsymbol{v} = \left\| R^{-1} {M_{int}}^{-1} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \right\| \qquad \textbf{Equation 4-5}$$

With the unit vector determined and the known coordinates of the virtual camera centre $\boldsymbol{C}$ with respect to the sphere centre (the origin of the world coordinate system), Equation 4-4 can be rewritten to determine point $\boldsymbol{P}$ on the unit sphere:

$$\boldsymbol{P} = \boldsymbol{C} + d\boldsymbol{v} \qquad \textbf{Equation 4-6}$$

The coordinates of $\boldsymbol{P} = (X^w, Y^w, Z^w)$ satisfy the unit sphere equation given by:

$$X^{w2} + Y^{w2} + Z^{w2} = 1 \qquad \textbf{Equation 4-7}$$

Substituting Equation 4-6 in Equation 4-7 and solving for d gives

$$d = \frac{-m + \sqrt{m^2 - 4ln}}{2l} \qquad \textbf{Equation 4-8}$$

where $l = \boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{v}$, $m = 2\boldsymbol{v}^{\boldsymbol{T}}\boldsymbol{C}$ and $n = \boldsymbol{C}^{\boldsymbol{T}}\boldsymbol{C} - 1$.

With the values of $\boldsymbol{v}$ and d obtained, all the points on the unit sphere being captured by the virtual camera can be located by using Equation 4-6.

In the second step after locating the points on the inner spherical surface captured by the virtual camera, the coordinates of these points denoted by $\boldsymbol{P}$ under the 3D Cartesian system (X^w, Y^w, Z^w) are transformed to their spherical coordinates based on the geometrical relationship established between them in Chapter 3, by applying Equation 3-17 and Equation 3-18. Then these 3D points on the unit sphere are transformed to their corresponding image coordinates by applying Equation 3-12 and Equation 3-13. The derived image coordinates are used for interpolation to generate appropriate views to the user's viewing window.

Basically, the interpolation with a target-to-source mapping (Nixon and Aguado, 2007) is used to assign colour pixel values from the sphere-mapped image to each image point of the virtual camera image plane. The process is illustrated in Figure 4-6, where the black squares in the smaller rectangle represent the pixel positions in the camera image plane (the actual size of the camera image plane is 120 by 160 pixels) and the black squares in the bigger rectangle represent the pixel positions on the sphere-mapped image.

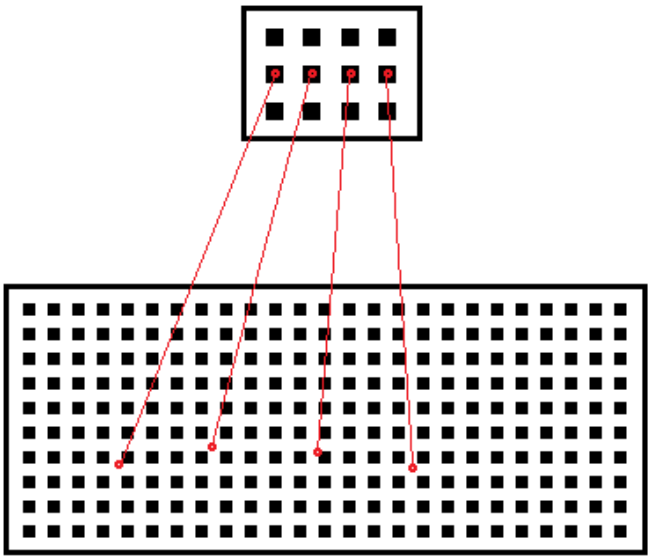


Figure 4-6 Image interpolation

As shown in Figure 4-6, the image point at each integer pixel position in the virtual camera image plane is projected to the sphere-mapped image as illustrated by the red lines. Since such projection does not necessarily fall on the integer pixel position of the sphere-mapped image, the colour values at the neighbouring pixel positions are interpolated to provide an appropriate colour value for the corresponding image point in the camera image plane. Common methods for image interpolation include ‘nearest’, ‘bilinear’, ‘bicubic’ and ‘spline’ .

For nearest interpolation, the colour value for the camera image plane is assigned with the colour value of the nearest pixel position. This method is simple and the fastest one as it requires the least calculation, but the interpolated image often suffers from a rough appearance.

For bilinear interpolation, each colour value to fill the camera image plane is calculated by using the locations and colours of the four neighbouring pixels, with the colour contribution made by each neighbouring pixel weighted by the distance to the projected pixel position. This method is also fast because only four pixels are needed to compute each output pixel value, and it gives a smoother appearance.

For bicubic interpolation and other higher order interpolation methods such as spline, a larger neighbourhood is needed for calculating each colour value to be assigned onto the new image, though it provides a smoother output image. For the trade-off between time and quality, the implementation in this project employs the bilinear interpolation method.

4.2.3 Initial Visualisation Results

Using the spherical image shown in Figure 2-7, Figure 4-7 and Figure 4-8 show the results of two images generated for the user's viewing window, based on their virtual camera positions and orientations given on the left, where the field of view is remained at its default setting. Also shown on the left of each figure is the corresponding 3D visualisation model, where the virtual camera inside the unit sphere is shown at the position denoted by $\boldsymbol{C}$, the camera image plane (or use's viewing window) is shown as a small green rectangle in front of the camera position, the principal axis is shown by the green arrow going through the camera and the middle of the camera image plane, four corners of the projection from the camera image plane to the unit sphere are shown by four pink arrows, the area on the inner sphere captured by the projection is shown by blue boundary, and the world coordinate system whose origin is at the centre of the sphere is shown as a reference in red.

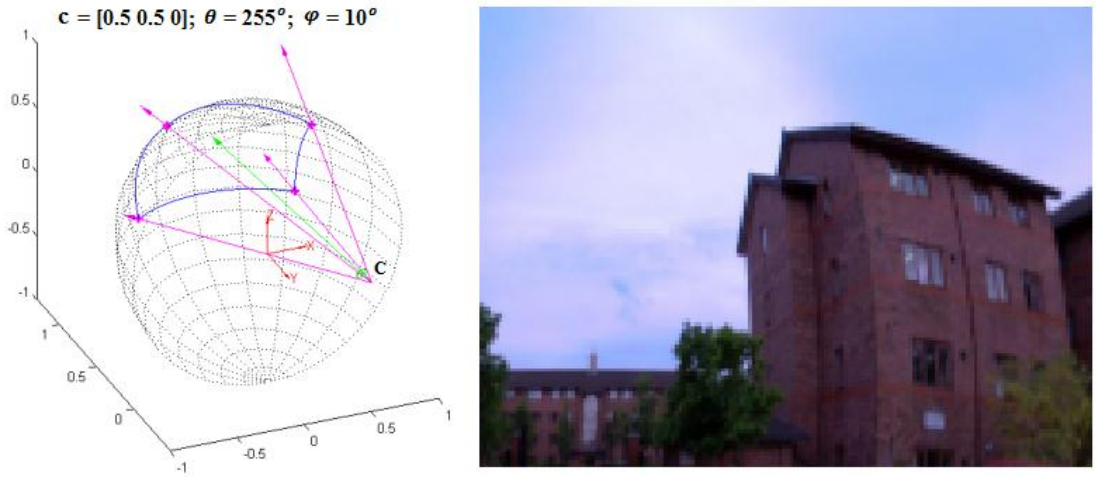


Figure 4-7 Real image viewpoint result 1

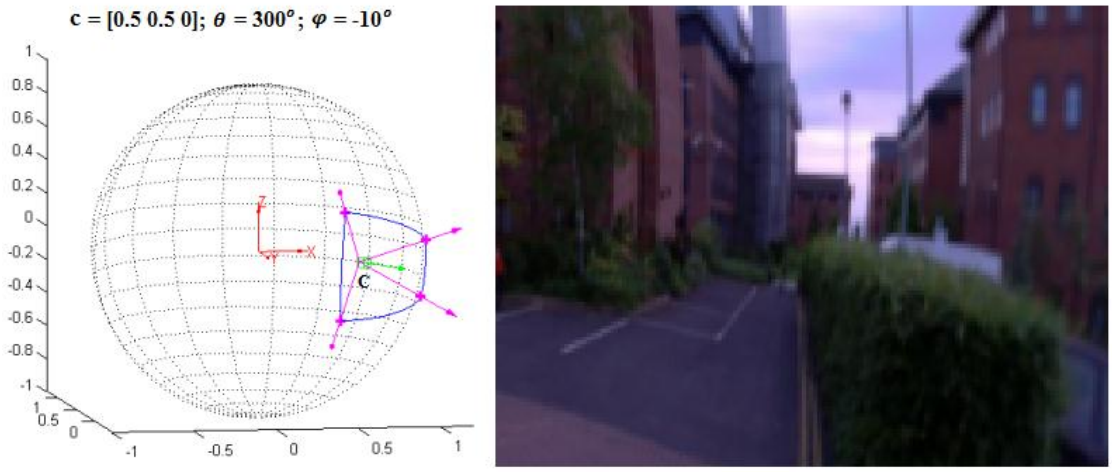


Figure 4-8 Real image viewpoint result 2

4.3 Image Distortion Correction

By placing the virtual camera in various positions and orientations within the sphere, it was observed that an undistorted view can be generated when the virtual camera is placed at or near the centre of the sphere. As the virtual camera moves away from the sphere centre, distortion begins to appear, and becomes severe when the virtual camera is positioned near the inner sphere wall. The distortion is clearly visible in Figure 4-7 and Figure 4-8 shown in the last section, where the vertical buildings are seen to be curved outwards.

4.3.1 Distortion Illustration

In order to show the distortion effect in a clear manner, a regular texture map based on multiple strips (shown in Figure 4-9) is used for image generation. The corresponding views generated based on the virtual camera being placed at the same viewing points as those in Figure 4-7 and Figure 4-8, are shown in Figure 4-10 and Figure 4-11, respectively.

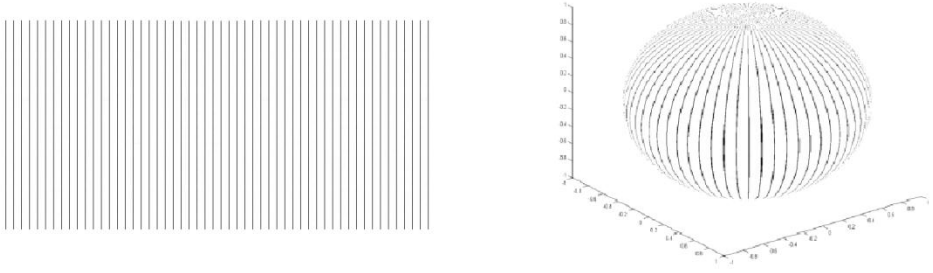


Figure 4-9 Texture map based on strips

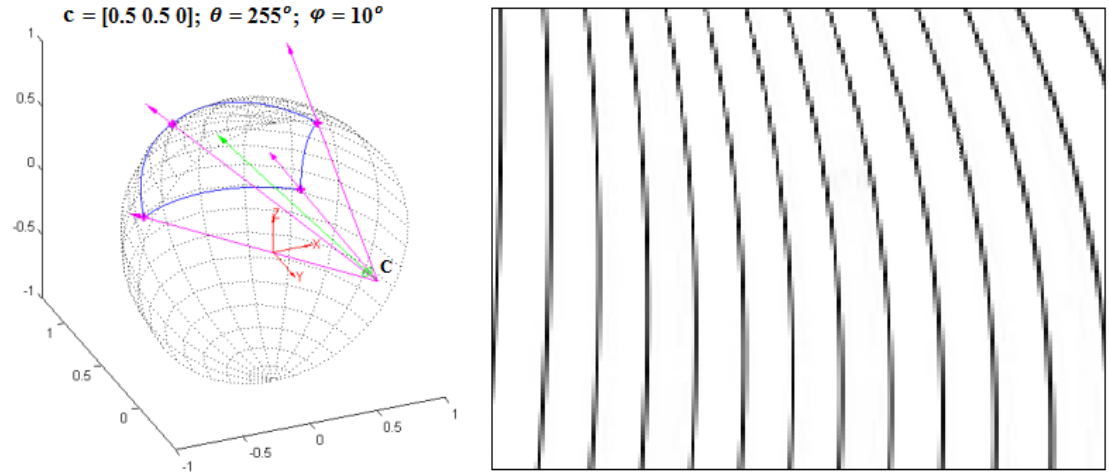


Figure 4-10 Texture map viewpoint result 1

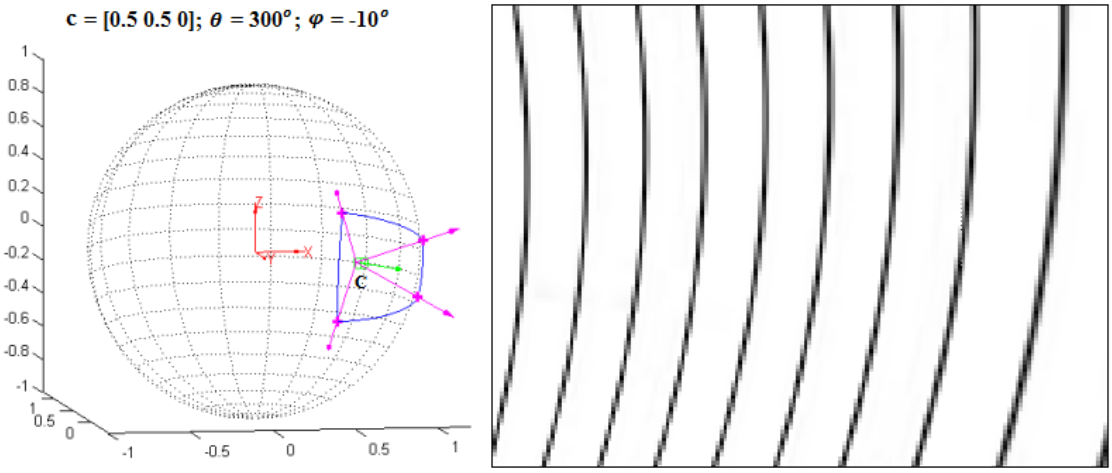


Figure 4-11 Texture map viewpoint result 2

Based on the image results shown above, the cause of the distortion was analysed by comparing the virtual camera positions with and without distortion. As the original spherical image is acquired at the centre of the unit sphere, this is the distortion-free point because the scene captured by the virtual camera at this point corresponds to the scene captured by the real camera. If the virtual camera is moved forward or backward with its viewing direction maintained at normal with respect to the sphere wall, an undistorted view is also generated as shown in Figure 4-12 with the virtual camera placed at $(0, 0.5, 0)$, $\theta = 270^o$, and $\varphi = 0^o$.

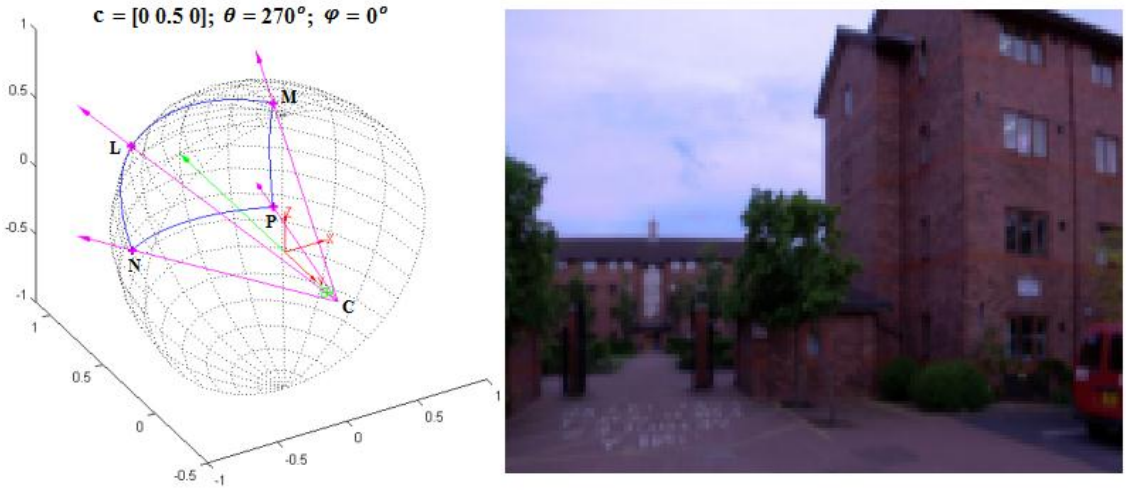


Figure 4-12 Distortion free viewpoint result

As shown by the 3D visualisation model on the left of Figure 4-12, the centre of the virtual camera and the four corners of the captured area form a pyramidal polyhedron $\boldsymbol{C}$–$\boldsymbol{LMNP}$. The base of the pyramid $\boldsymbol{LMNP}$ is a regular spherical surface, the principal axis (the green arrow) is perpendicular to the base, and the four edges are equal in length, i.e. $\boldsymbol{CL} = \boldsymbol{CM} = \boldsymbol{CN} = \boldsymbol{CP}$.

When the virtual camera is moved away from the sphere centre, and the principal axis is no longer perpendicular to the base (as shown on the left part of Figure 4-7 and Figure 4-8 in the last section), the polyhedron becomes an oblique pyramid. The irregularity of the pyramid can be measured by comparing the different lengths of the four edges $\boldsymbol{CL}$, $\boldsymbol{CM}$, $\boldsymbol{CN}$, and $\boldsymbol{CP}$. The greater the difference is, the worse the distortion occurs.

4.3.2 Methodology and Implementation

This sub-section presents the methodology for correcting the image distortion, and the method is based on generation of an intermediate projection plane acting as the base of the polyhedron to provide a better balance between the lengths of the four edges. The introduction of the intermediate projection plane results in two projecting steps for image generation, namely, projection from the spherical surface to the intermediate projection plane, followed by projection from the intermediate plane to the image plane of the virtual camera.

There are a few possible positions where an intermediate projection plane could be placed. Using the coordinates of the original points on the sphere being captured by the virtual camera as reference positions (see Figure 4-13(a)), Figure 4-13(b), Figure

4-13(c), and Figure 4-13(d) show three options for the placement of the intermediate projection plane.

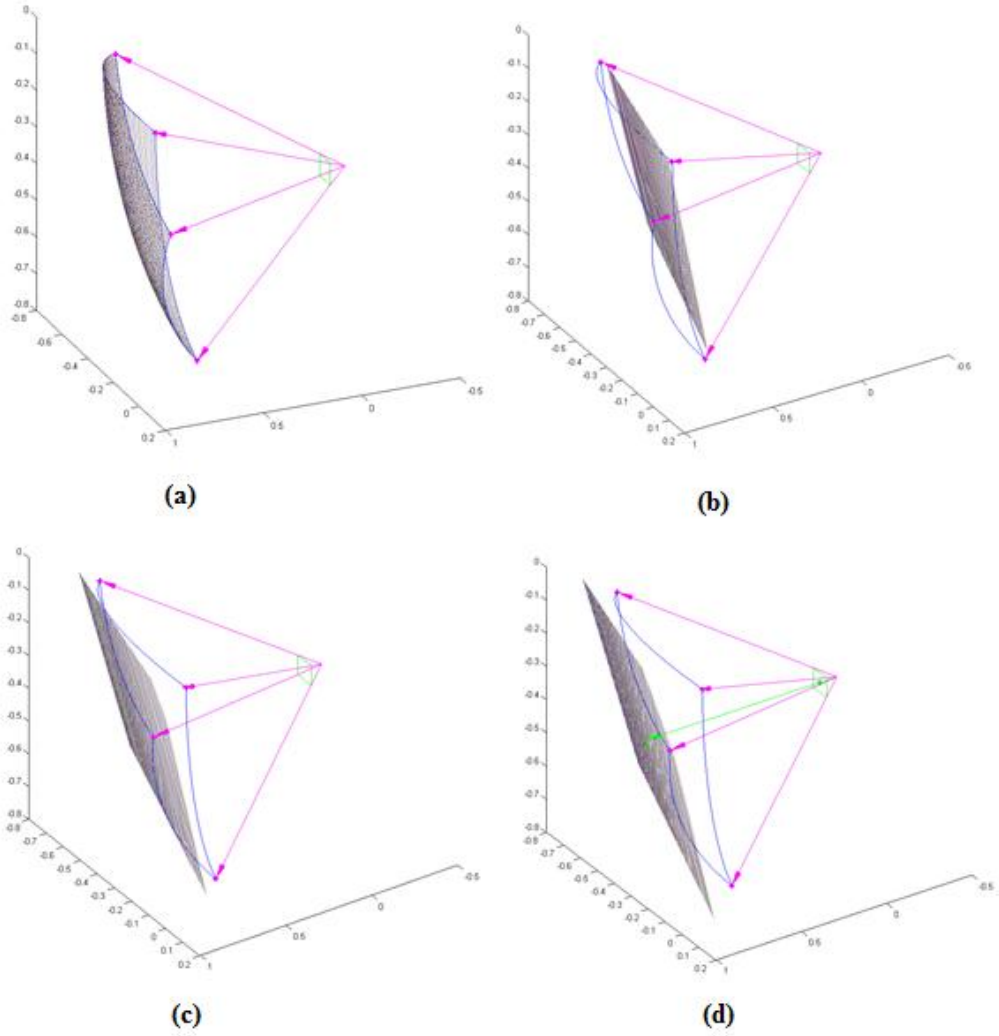


Figure 4-13 (a) Original points on the sphere captured by virtual camera (b) Option 1 for intermediate projection plane (c) Option 2 for intermediate projection plane (d) Option 3 for intermediate projection plane

Whilst Figure 4-13(b) shows the intermediate projection plane being anchored at the four corner points of the captured area, Figure 4-13(c) shows the intermediate projection plane being placed based on the result of least squares plane fitting (Li and Zhang, 1998) of all the captured points on the sphere, and Figure 4-13(d) shows the intermediate projection plane being placed at tangent to the captured spherical surface with respect to the principal axis of the virtual camera (shown as the green arrow).

To choose from the three options, some tests have been carried out to compare their results. Although all of the options were seen to reduce the curvature distortion,

different positions of the intermediate projection plane were seen to cause the image content in the generated view to be shifted by different extents, due to the varying distance between the intermediate projection plane and the original spherical surface position. With the image contents found to be shifted the least from the original view based on Option 2, it is considered as the optimum intermediate projection plane position.

With the optimum position for the intermediate projection plane decided, the algorithm for visualisation with the virtual camera being placed at an arbitrary position within the sphere can be divided into four steps. The first step is to estimate the immediate projection plane. To do this, the fundamental part of the visualisation process presented in the last section is repeated firstly, i.e. the virtual camera is placed within the unit sphere to capture points on the sphere. The captured points are then used for fitting a plane among themselves in a least squares sense. If points on the fitted plane (or the immediate projection plane) are denoted by $\boldsymbol{P'}$, and $\boldsymbol{P_0}$ is any point on this plane, the application of least squares fitting gives:

$$(\boldsymbol{P'} - \boldsymbol{P_0})\boldsymbol{n} = 0 \qquad \textbf{Equation 4-9}$$

where $\boldsymbol{n}$ is the direction cosines of the normal to the fitting plane.

The second step is to locate the points captured by the virtual camera on the intermediate projection plane, and this is similar to locating the captured point on the sphere as explained in Sub-section 4.2.2. For a point denoted by $\boldsymbol{P'}$ on the intermediate projection plane with the distance to the camera centre denoted by d, if $\boldsymbol{C}$ denotes the coordinates of the camera centre, and $\boldsymbol{v}$ denotes the direction vector of $\boldsymbol{P'}$, then following the approach described in Sub-section 4.2.2 gives:

$$\boldsymbol{P'} = \boldsymbol{C} + d\boldsymbol{v} \qquad \textbf{Equation 4-10}$$

Substituting Equation 4-10 in Equation 4-9 gives:

$$(\boldsymbol{C} + d\boldsymbol{v} - \boldsymbol{P_0})\boldsymbol{n} = 0 \qquad \textbf{Equation 4-11}$$

Solving Equation 4-11 for d gives:

$$d = \frac{(\boldsymbol{P_0} - \boldsymbol{C})^T\boldsymbol{n}}{\boldsymbol{v^T n}} \qquad \textbf{Equation 4-12}$$

Substituting Equation 4-12 in Equation 4-10, all the points on the intermediate projection plane being captured by the virtual camera can be located.

The third step is to find out the corresponding points of $\boldsymbol{P'}$ on the sphere. As the sphere is modelled as a unit one, this is done simply by normalising the modulus of $\boldsymbol{P'}$ to one.

The fourth and final step is to do image interpolation based on the coordinates of the points on the sphere (same as that explained in Sub-section 4.2.2) and produce the distortion corrected results. Target to source mapping with bilinear interpolation is again used.

4.4 Improved Visualisation Results and Analysis

4.4.1 Improved Images

With the virtual camera being placed at the same position and orientation as those shown in the previous section, the improved images for Figure 4-7 and Figure 4-8 as well as Figure 4-10 and Figure 4-11 are shown in Figure 4-14 and Figure 4-15 as well as Figure 4-16 and Figure 4-17 with distortion correction.

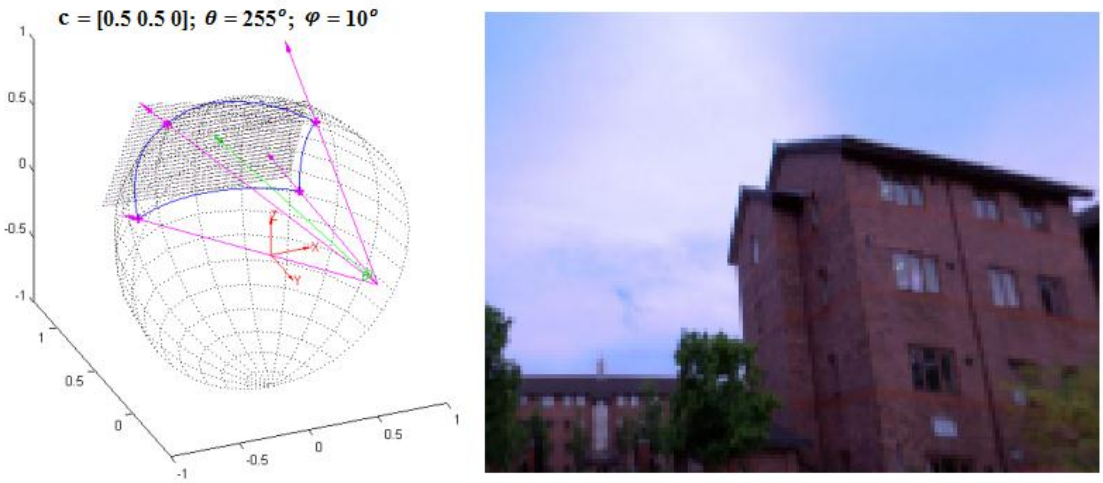


Figure 4-14 Improved real image viewpoint result 1

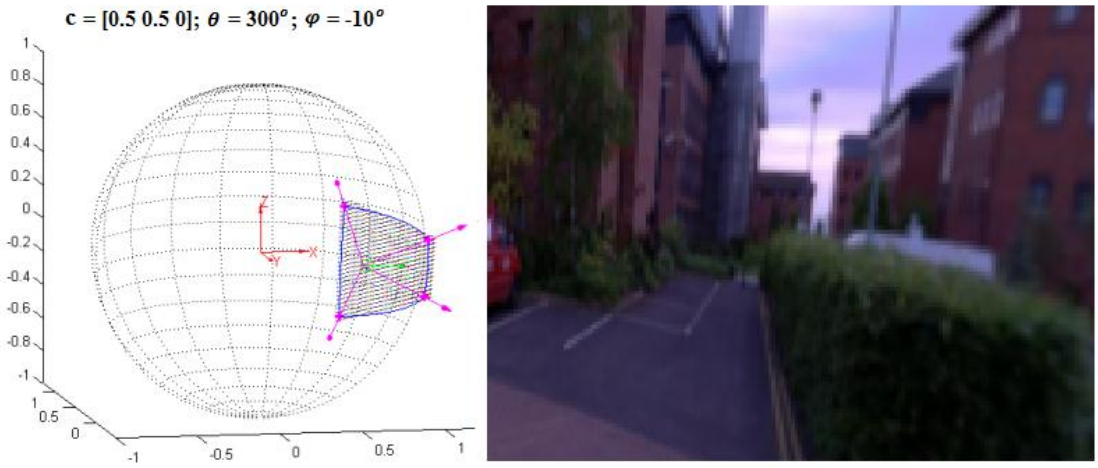


Figure 4-15 Improved real image viewpoint result 2

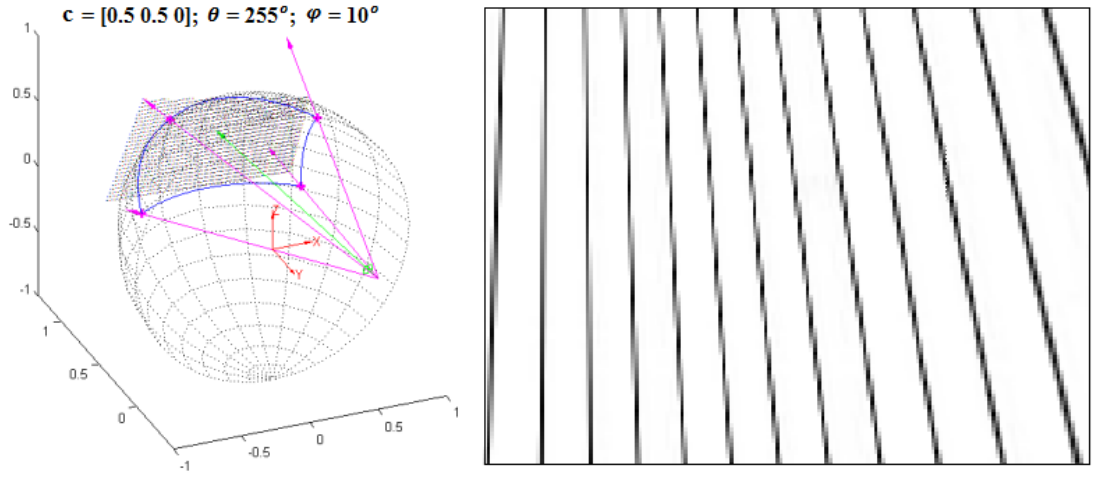


Figure 4-16 Improved texture map viewpoint result 1

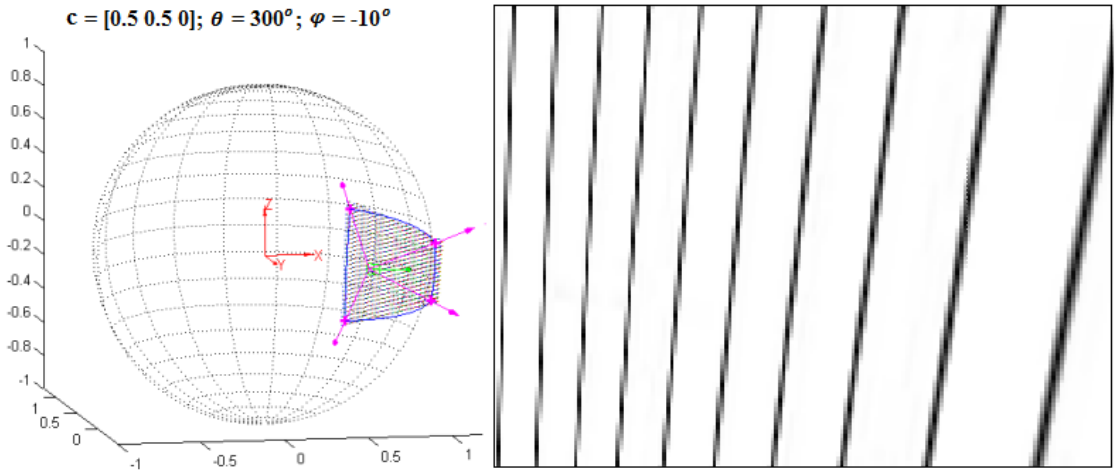


Figure 4-17 Improved texture map viewpoint result 2

Compared these four figures with the corresponding ones in Sub-sections 4.2.3 and 4.3.1, it should be apparent that the proposed distortion correction method has removed the curvature effect with straight buildings walls and strips.

With the virtual camera placed at $(0.5, 0, 0)$, a further six face cubic map is generated in Figure 4-18 and Figure 4-19 to demonstrate the effectiveness of the proposed distortion correction method. The six views are shown with (θ, φ) set to $(0^o, 0^o)$ for the right view; $(90^o, 0^o)$ for the front; $(180^o, 0^o)$ for the left; $(270^o, 0^o)$ for the back; $(0^o, 90^o)$ for the top and $(0^o, -90^o)$ for the bottom.

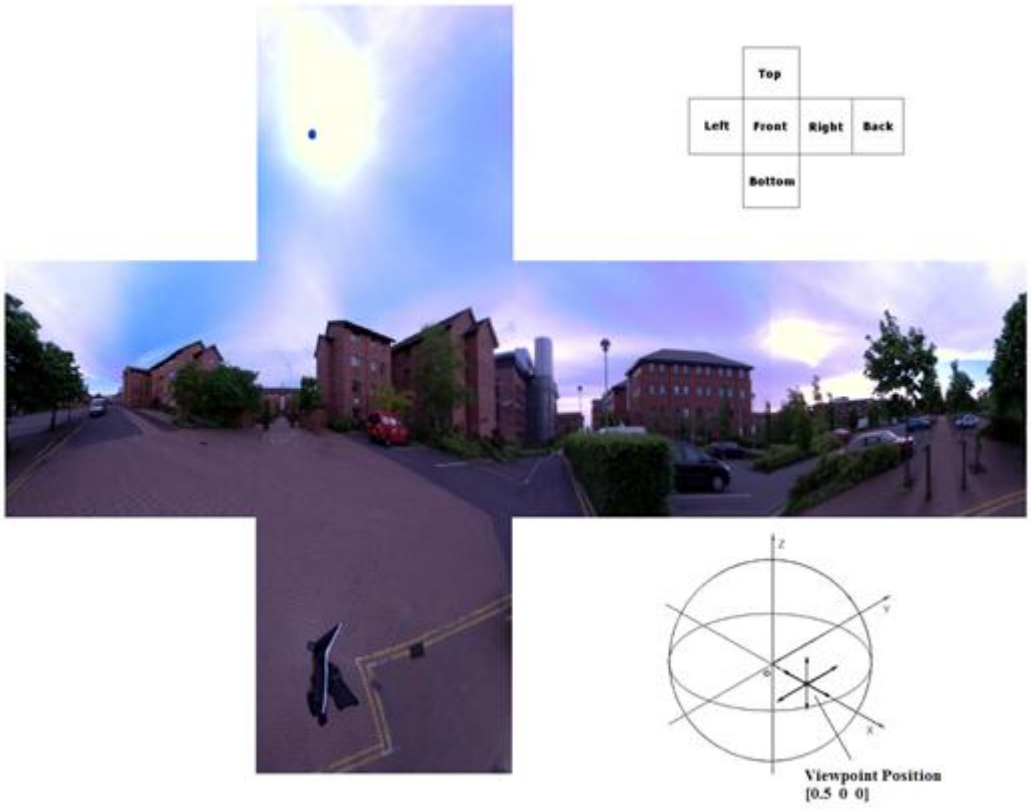


Figure 4-18 Panoramic real image result

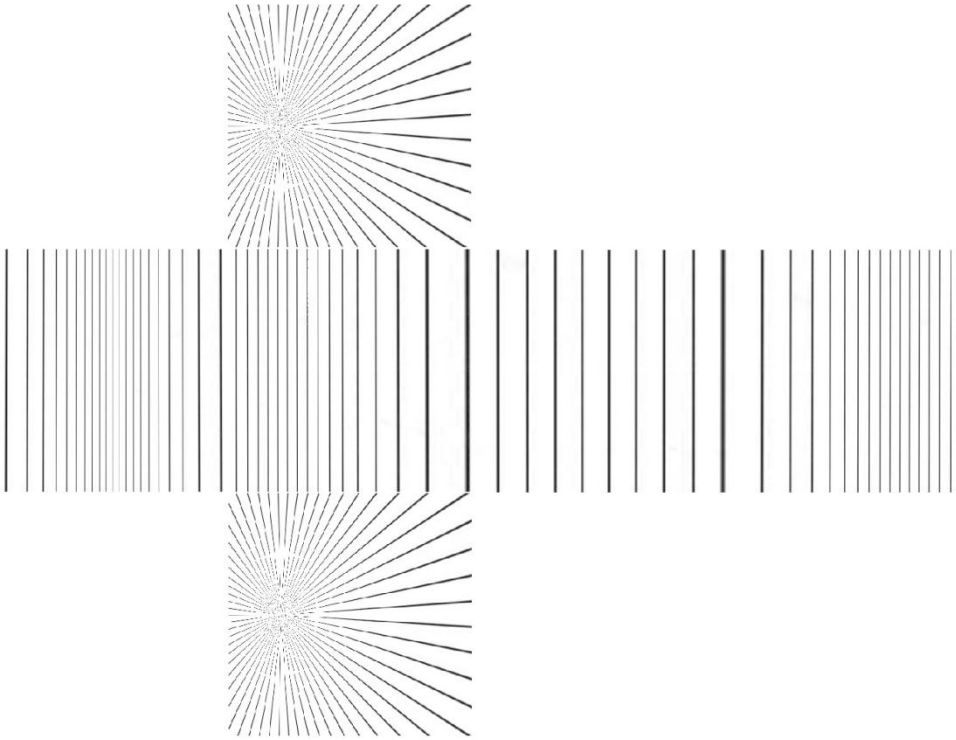


Figure 4-19 Panoramic texture map result

For comparison, Figure 4-20 shows the six views generated based on the strip pattern without distortion correction.

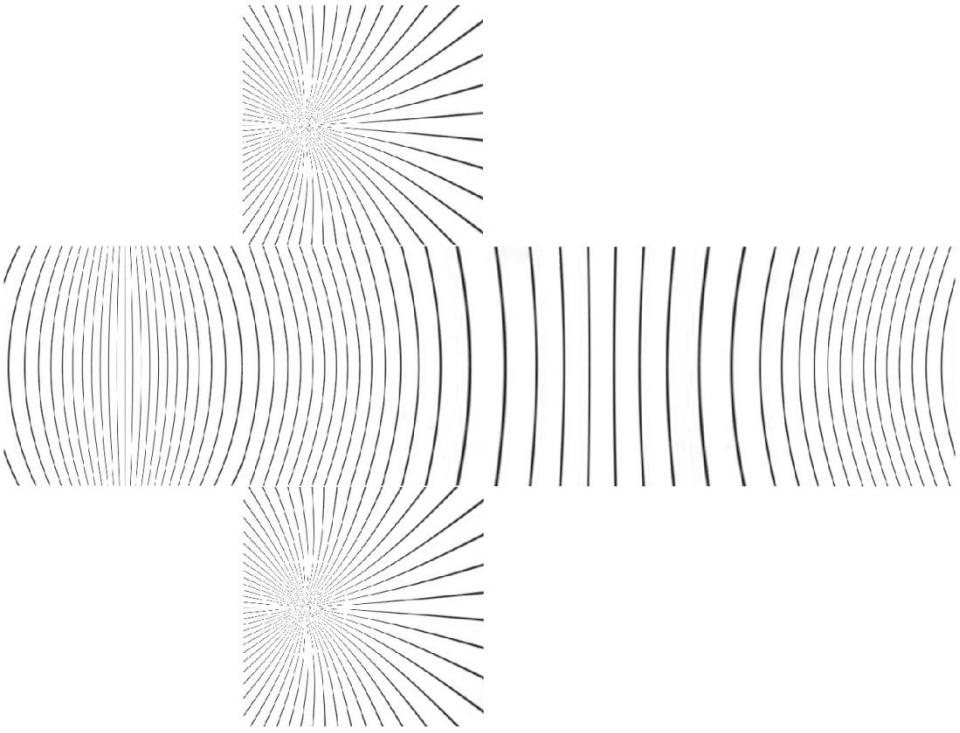


Figure 4-20 Panoramic texture map result without distortion correction

4.4.2 GUI and Navigation

The visualisation algorithm being implemented in MATLAB has been developed into a Graphical User Interface (GUI) (Gonzalez et al., 2004) to allow the user to move freely within the environment constructed by mapping the spherical image acquired on a sphere. The interface is shown in Figure 4-21 with four parts of display and seven sliding bars for user controlled position, orientation and field of view. Extensive visual evaluation experiments were conducted to compare position and direction dependent views generated from spherical images with and without distortion correction. Still using the spherical image shown in Figure 2-7, Figure 4-22 illustrates five viewing positions and directions along a particular virtual camera's movement route within the unit sphere in a plan view.

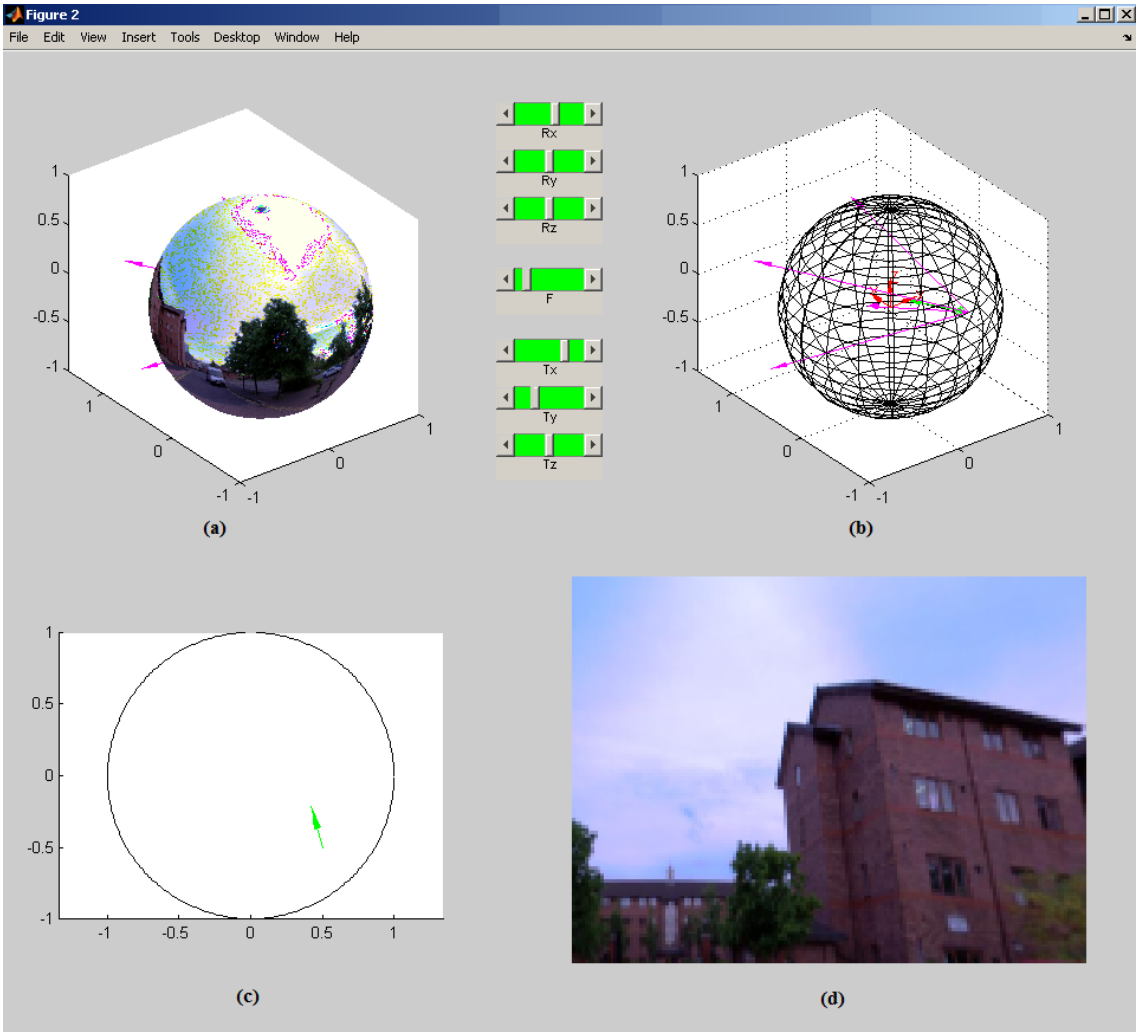
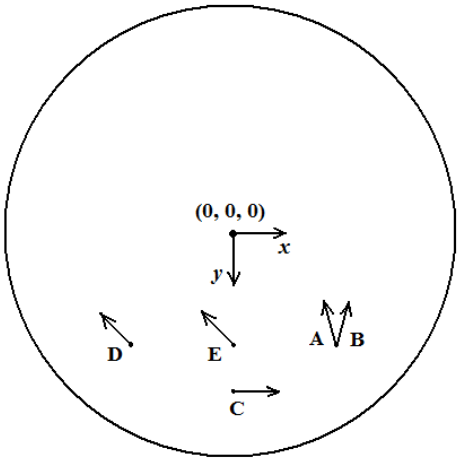


Figure 4-21 GUI with (a) Sphere-mapped image, (b) 3D visualisation model, (c) plan view of the virtual camera position and orientation and (d) viewpoint image result



$A: (0.5, 0.5, 0)$ $\theta = 255^o;$ $\varphi = 5^o$

$B: (0.5, 0.5, 0)$ $\theta = 285^o;$ $\varphi = 5^o$

$C: (0, 0.7, 0)$ $\theta = 0^o;$ $\varphi = 0^o$

$D: (-0.5, 0.5, 0)$ $\theta = 225^o;$ $\varphi = 0^o$

$E: (0, 0.5, 0)$ $\theta = 225^o;$ $\varphi = 0^o$

Figure 4-22 Navigation demonstration

Figure 4-23 and Figure 4-24 show the two corresponding sequences of the views generated for the five virtual camera positions and orientations along the route without and with distortion correction. It should be apparent that the object distortion (building curving inwards and outwards) in Figure 4-23 has been corrected in Figure 4-24 with good visual quality. From a number of tests carried out based on the outdoor spherical images, the proposed distortion correction method was found to provide visually good results if the virtual camera is placed within a radius of 0.5 of the unit sphere.

Figure 4-23 Image sequence result without distortion correction

Figure 4-24 Image sequence result with distortion correction

Furthermore, a movie has been generated according to a certain user navigation route with frames of the movie shown in Appendix B.

4.4.3 Conclusion

Visualisation based on one spherical image is to take a spherical image at a certain scene and model the surrounding by mapping the image onto a unit sphere. This chapter presents a complete method for visualisation within the modelled spherical environment, from construction of the sphere model to position and direction dependent view generation from the sphere-mapped image using a virtual camera. In particular, by using an intermediate project plane to reduce the effect of spherical curvature, a new image rendering method has been developed to remove image distortion in the new views generated. Using real outdoor spherical images, the user is allowed to change the viewing position, orientation and field of view through a GUI program to visualise the surroundings within the sphere. The presented method is shown to generate good visual image results according to the user's movement without apparent distortion, thereby providing a more immersive navigation experience than visualisation fixed at the sphere centre (like in Google Map Street View, Apple QuickTime VR).

The visualisation results in a 'moving range' for free navigation, i.e. within half of the model sphere's radius. A further research is presented in the next chapter for visualisation based on two overlapping spherical images taken within the same environment.

Chapter 5 VIEW GENERATION BASED ON TWO SPHERICAL IMAGES

5.1 Introduction

By enabling novel (intermediate or in-between) views to be generated from the two views acquired, view generation based on two spherical images is a further investigation of the previous chapter. This chapter focuses on image processing methods applied to the two view geometry between a pair of spherical images and the research interest is to bring about immersive user navigation within a virtual environment modelled by multiple spherical image data.

Image based rendering techniques have been well established for new view generation from planar images. If two images of the same objects or a certain scene are taken from different angles by two cameras or a camera moving along a path from one position to another, a number of methods could be applied to generate a new image for a viewpoint different from the original two viewpoints. Point correspondences which are images of the same physical points appeared in two different views can be searched through detection of image features and using the constraints applicable to the two view geometry. Depth information can be recovered if dense point correspondences are available which effectively enables reconstruction of the 3D scene for a new view to be generated. Image warping (Glasbey and Mardia, 1998) and morphing are often applied when there are less control points, and sometimes the images being processed in small patches or triangle based meshes are created for rendering. If more images are taken, multiple views of the scene are available to make the geometry more complete, and more control points could be used to improve the novel view results.

In the case of geometry and view generation between two panoramic views, only limited researches have been exercised. One example is from (Fu et al., 1999) which suggested a method to subdivide cylindrical images into variable-sized triangles and to apply image warping to these triangles by drawing the warped triangles in a specific order (with the drawing order determined from epipolar geometry). No depth information of the scene is needed. However, triangles are not efficient for merging two warped images seamlessly especially for complex scenes.

(Chan et al., 1999) proposed a panoramic-based walkthrough system for cylindrical images. The point correspondence search between cylindrical images is achieved by

allowing users to identify correspondences interactively. By defining the corresponding patches on the panoramic image, a triangular mesh for the panorama based on the given patches is generated, and free navigation of the scene can be provided between cylindrical images acquired by means of a triangle based image warping technique.

In the method proposed by (Shum and He, 1999), it constrains the user viewing motion to planar concentric circles and creates concentric mosaics using a manifold mosaic for each circle. Concentric mosaics index all input image rays naturally in 3 parameters: radius, rotation angle and vertical elevation. Novel views are rendered by combining the appropriate captured rays in an efficient manner at rendering time.

(Bradley et al., 2005) constructs a virtual environment by processing cubical panoramas and let the user walk from one static viewpoint to another. Navigation is achieved by densely sampled panoramic images at various viewpoints. Furthermore, (Fiala and Roth, 2005) proposes a method to align a sequence of cubic panoramas by image feature detection to form a camera moving route. New views can then be generated in-between the two aligned panoramas if the viewpoint is on the camera route.

(Takahashi et al., 2000) takes panoramic images of the scene along to a camera running line and these panoramic images are divided into vertical slits for new view reassembling. The new navigating route can be parallel or perpendicular to the camera run line and have new views rendered and occlusions shown correctly. This method deals with cylindrical panoramas.

In this chapter, spherical images acquired at different sampling positions within the space are used to generate in-between views of the scene for indoor environments. With a camera calibration box placed within the scene to obtain the camera projection information, a new viewpoint position can be chosen by the user for a virtual spherical camera to capture a new panoramic view of the scene, and the new view can be applied to a spherical image viewer for visualisation.

Section 5.2 gives a general introduction to the two view geometry and explains the imaging geometry for spherical cameras and a pair of panoramas captured. Section 5.3 proposes the new view generation algorithm based on camera calibration and geometry constraints. In Sections 5.4 and 5.5, the proposed view generation algorithm is illustrated by using computer simulated scene and real spherical images of an indoor environment, respectively. This is followed by Section 5.6 with further results and their

detail analysis to demonstrate the effectiveness of the proposed view generation algorithm. Finally in Section 5.7, the performance of the proposed view generation algorithm is summarised and the issues in practical applications are discussed.

5.2 Two View Geometry for Spherical Cameras

The term of 'two view geometry' is borrowed from researches studying the geometry of two perspective views taken from conventional cameras (Trucco and Verri, 1998). These two views are either acquired simultaneously by using a stereo camera rig or sequentially by moving the camera along the scene. Both of these two cases are geometrically equivalent. In this section, the two view geometry for cameras taking panoramic images is studied. The terms 'cameras' and 'Cameras *A* and *B*' used later on generally refer to two camera positions, i.e. the camera is placed at one position to produce a spherical image, and then moved to another position to produce another one. This is not to be confused with the case of using a pair of twin cameras (usually carefully calibrated) for image capture at the same time, as high precision or stereoscopic reconstruction is not the issue in this scenario. While the two view geometry for general cases is introduced in Sub-section 5.2.1, the constraint applicable to spherical images is presented in Sub-section 5.2.2.

5.2.1 Two View Imaging Geometry

In order to generate a new view from a two view geometry, the two views taken by conventional cameras are normally required to have a large overlap in their image contents, i.e. the camera positions being not far apart and in similar viewing directions. The overlapping area ensures sufficient image correspondences between the two views to be found to enable the geometries between them to be determined. The description of imaging geometries can be divided into the correspondence geometry, camera geometry and scene geometry (Hartley and Zisserman, 2003).

Correspondence geometry is the constraint for two corresponding image points of the same physical point to appear in the two views. In other words, given an image point in the first view, the geometry constrains the position of its corresponding point in the second view. The constraint is also known as the epipolar geometry.

Camera geometry is the geometry of camera motion, i.e. the projection (camera) matrices describing the cameras' projection of the scene and their physical position and

orientation. The matrices for the cameras can be estimated in relation to each other or up to a projective ambiguity, and the ambiguity can also be solved by supplying additional information of the cameras or the scene.

Scene geometry is understood as the reconstruction of the geometry structure. From the image correspondences obtained and the camera geometry, the physical points in the scene captured by the cameras can be recovered. This is often achieved by *triangulation* which is the intersection of each pair of the camera rays back-projected from the correspondences.

For two view geometry for spherical cameras with each camera capturing a panoramic image of the scene, the acquired two views are largely overlapped due to the camera's super large field of view. This is an advantage enabling numerous correspondences, though the appearance of the corresponding image points suffers from distortion due to the characteristics of the camera sensor. Cross correspondence search for these images can be simplified by applying the epipolar geometry existing between the two spherical views as well as planar views.

5.2.2 Epipolar Geometry for Spherical Images

The epipolar constraint between two spherical images is essentially a geometry describing the intersection of the two spherical sensors with a group of planes passing through their camera baseline (the line joining the two camera centres).

Figure 5-1 shows the epipolar geometry between two spherical cameras capturing a single physical point. For point $\boldsymbol{P}$ in the space, it is acquired by the spherical camera on the left and leaves an image pixel $\boldsymbol{p}$ as its projection on the spherical sensor. The second spherical camera (on the right) also captures point $\boldsymbol{P}$ but the projection on its image sensor $\boldsymbol{p'}$ is unknown. Imagining a plane defined by the light ray from the left camera to point $\boldsymbol{P}$ (passing through $\boldsymbol{p}$ on the sensor) and the baseline of the cameras, this plane intersects the two spherical sensors and the intersection on the right camera sensor (illustrated in grey) is a circular plane passing through the camera centre. This intersection leaves a full circle on the spherical surface which is called the epipolar circle or epipolar curve. The image of point $\boldsymbol{P}$ on the right camera sensor is constrained to be on the epipolar curve which is the epipolar constraint. For different physical points captured by the two cameras, the epipolar geometry exists in each pair of their image

correspondences. The baseline joining the two camera centres together intersects the two spherical sensors at two points, which are called epipoles (denoted as $\boldsymbol{e}$ and $\boldsymbol{e}'$). All the epipolar curves intersect at their epipole, which is the projection of one camera centre on the sensor of the other camera.

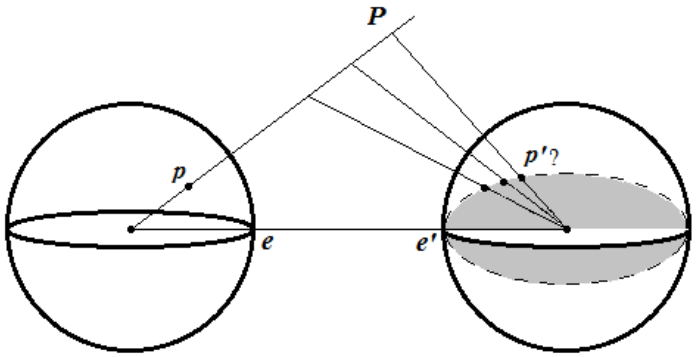


Figure 5-1 Epipolar geometry

For both conventional and spherical cameras, the epipolar geometry is independent of scene structure and it only depends on the cameras' projection parameters and relative poses. The fundamental matrix regarded as the algebraic representation of epipolar geometry is normally studied in the case of conventional cameras. Basically it is the mapping between an image point and its epipolar line (instead of a curve for spherical cameras), or a link between two corresponding image points in different views. The fundamental matrix is of size of 3-by-3 but only of rank 2 as it is geometrically a mapping from a point in 2D to another point or line in 2D.

The essential matrix of rank 3 is more suitable for describing the mapping between a pair of 3D points on the two spherical sensors. It contains parameters of rotation and translation, constraining the mapping from an image point on one spherical sensor to an epipolar curve on the other spherical sensor. It contains 5 degrees of freedom with scale ambiguity, and like the fundamental matrix, the essential matrix is a homogeneous quantity. For a pair of corresponding points on the two spherical sensors, $\boldsymbol{p}$ and $\boldsymbol{p}'$, the essential matrix linking them can be written as:

$$\boldsymbol{p}'^{\boldsymbol{T}} E \boldsymbol{p} = 0 \qquad \textbf{Equation 5-1}$$

As E is defined up to an arbitrary scale factor, it therefore has only 8 independent entries. It can be determined through a homogeneous linear system formed by writing Equation 5-1 for a set of at least 8 point matches. To expand the above equation, let the

coordinates of the *i*th pair of the corresponding points be denoted as $(p'_{xi}, p'_{yi}, p'_{zi})$ and (p_{xi}, p_{yi}, p_{zi}) . With the entries of E being organised to give $\boldsymbol{m} = [E_{11}, E_{12}, \ldots, E_{32}, E_{33}]^T$, the linear system for i pairs of correspondences can be written as:

$$Q\boldsymbol{m} = 0 \qquad \textbf{Equation 5-2}$$

where

$$Q = \begin{bmatrix} p'_{x1}p_{x1} & p'_{x1}p_{y1} & p'_{x1}p_{z1} & p'_{y1}p_{x1} & p'_{y1}p_{y1} & p'_{y1}p_{z1} & p'_{z1}p_{x1} & p'_{z1}p_{y1} & p'_{z1}p_{z1} \\ & & & & \vdots & & & & \\ p'_{xi}p_{xi} & p'_{xi}p_{yi} & p'_{xi}p_{zi} & p'_{yi}p_{xi} & p'_{yi}p_{yi} & p'_{yi}p_{zi} & p'_{zi}p_{xi} & p'_{zi}p_{yi} & p'_{zi}p_{zi} \end{bmatrix}$$

For the case of using more than 8 pairs of correspondences with $i \geq 8$, more equations are obtained for the system, which allows $\boldsymbol{m}$ to be estimated through least squares techniques with $Q\boldsymbol{m} \cong 0$ and provides more accurate results. Vector $\boldsymbol{m}$ can be recovered by computing the SVD of Q, where $Q = UDV^T$, and the last column of V corresponds to the null (in practice the smallest) singular value of Q.

Once the essential matrix is determined, for any known point $\boldsymbol{p}$ on one spherical sensor, its corresponding epipolar curve on the other spherical sensor can be expressed by:

$$\boldsymbol{u}' = E\boldsymbol{p} \qquad \textbf{Equation 5-3}$$

where $\boldsymbol{u}'$ is the direction vector perpendicular to the circular plane (the intersection shown in grey in Figure 5-1, with the dashed circle as the epipolar curve).

Epipolar geometry enables fast correspondence search between two view spherical images by limiting the search to a great circle on the spherical sensor (or along a curve across the spherical image). To simplify the search further, the spherical image can be 'rectified' by rotating the sphere mapped image until the epipole reaches the north pole of the sensor sphere (Fujiki et al., 2007). By doing this, the epipolar curves (all intersecting at the epipole) become longitudes of the spherical sensor, which in turn become vertical lines in the spherical image. With the correspondence search along vertical columns of the spherical image, it becomes a 1D search in implementation. This idea is similar to the epipolar geometry for normal cameras with the search along horizontal lines of the image after image rectification.

As an example, Figure 5-2 and Figure 5-3 show a pair of spherical images taken by the SpheroCam® in an office environment with the two camera positions placed roughly 1.5m apart. There are 12 pairs of point correspondences which are manually selected and marked as blue crosses on both of the images for estimating the essential matrix. By firstly transforming these points from their image coordinates back to the unit sphere sensor in 3D Cartesian camera coordinates (see Section 3.2), then using the point correspondences in 3D to form 12 equations of the linear system as Equation 5-2, the essential matrix can be determined using SVD.

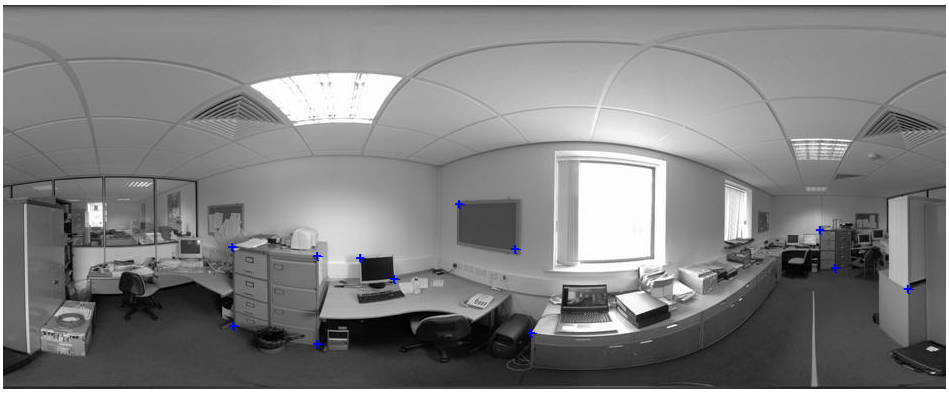


Figure 5-2 Spherical image - office view 1

Figure 5-3 Spherical image - office view 2

With the essential matrix obtained, the epipoles can be estimated directly from it since each epipole lies on all the epipolar curves on its spherical sensor. Hence, Equation 5-1 can be written as

$$\boldsymbol{p'^{T}} E \boldsymbol{e} = 0 \qquad \textbf{Equation 5-4}$$

for every $\boldsymbol{p'}$. Since E is not a zero matrix, this is only possible if

$$E\boldsymbol{e} = 0 \qquad \textbf{Equation 5-5}$$

Hence epipole $\boldsymbol{e}$ can be derived as the null space of E and similarly for $\boldsymbol{e'}$, which is the null space of E^T. The epipole is useful for rectifying the spherical images. If epipole $\boldsymbol{e} = [e_1, e_2, e_3]^T$ and the north pole of the unit spherical sensor is $\boldsymbol{N}^* = [0, 0, 1]^T$, rotation of the unit sphere to make $\boldsymbol{e}$ reach $\boldsymbol{N}^*$ can be expressed as:

$$\boldsymbol{R} = [\boldsymbol{v}]_\times + \frac{1 - \boldsymbol{e}^T\boldsymbol{N}^*}{\|\boldsymbol{v}\|^2}[\boldsymbol{v}]_\times^2 \qquad \textbf{Equation 5-6}$$

where $\boldsymbol{v} = \boldsymbol{e}^T \times \boldsymbol{N}^*$ and $[\boldsymbol{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix}$ is the anti-symmetric matrix of vector $\boldsymbol{v}$.

For the spherical images shown in Figure 5-2 and Figure 5-3, its epipolar geometry is shown in Figure 5-4, Figure 5-5 and Figure 5-6, where some points of interest are selected from the first view (see Figure 5-4) and their corresponding epipolar curves are plotted on the second view (see Figure 5-5) and the vertical epipolar lines (longitudes on the spherical sensor) are plotted on the rectified view (see Figure 5-6).

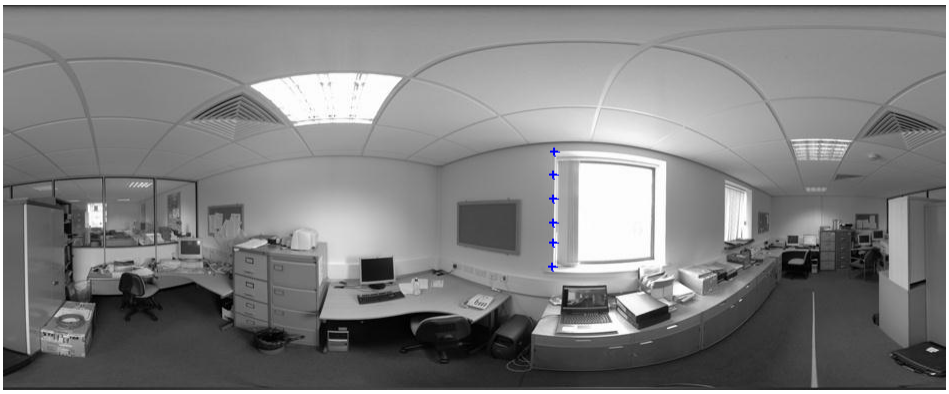


Figure 5-4 Points of interest on view 1

As marked by the blue crosses on Figure 5-4, six points are selected in the first view along the left side of the window. Applying Equation 5-3 to estimate their corresponding epipolar curves by using the essential matrix derived from 12 point correspondences, the derived curves are plotted as blue dotted lines on the second view (as shown Figure 5-5). It shows clearly that the epipolar curves cross the window edge on the left side, and pass through the correct positions of the six point correspondences accordingly. All the curves intersect at the epipole (marked by red), which is effectively the projection of the other camera's centre.

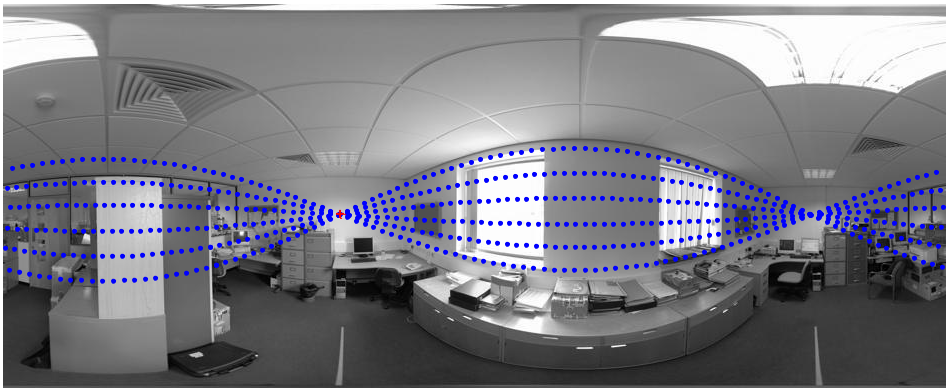


Figure 5-5 Epipolar curves on view 2

To reduce the correspondences search from a 2D curve to a 1D straight line, the view shown in Figure 5-5 is rectified by using the rotation method described and the six corresponding epipolar lines are plotted on the image (see Figure 5-6). It is seen that each epipolar line is cut into two halves as an epipolar circle yields two longitudes on the spherical sensor. It is also observed that the epipolar lines pass through the correct positions of the point correspondences on the rectified view.

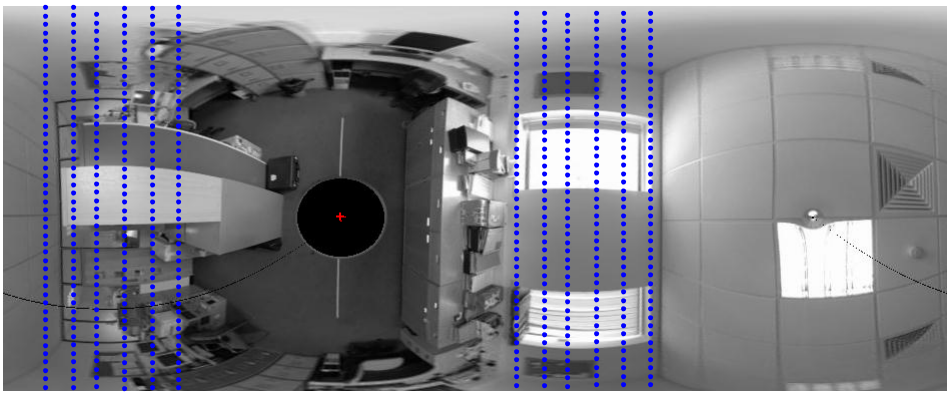


Figure 5-6 Epipolar lines on rectified view 2

5.3 Calibration Set-up and Proposed View Generation Approach

This section gives an overview of the view generation algorithm based on a pair of spherical images. The scenes concerned are indoor environments which can be approximated using multiple rectangular cubes. A calibration object is placed in the scene so that its image appears in both of the views produced by the two cameras for their calibration. The scene contains at least one foreground object (the calibration

object), which causes occlusion in the views captured. In generation of a new view, the occlusion is dealt with by applying textures from different views.

5.3.1 Camera Calibration for Two View Geometry

The calibration method for a single spherical camera was explained in Chapter 3. The conclusions were made that the feature points used should not be co-planar (i.e. they must be on at least two different 3D planes), and that a number of some 50 feature points are needed for getting a reasonably good calibration result.

For calibration of spherical cameras in a two view geometry, the calibration object is made slightly different to the one used in Chapter 3. Patterns of the checkerboard grids are now placed on the outside of the calibration box (on a pair of perpendicular planes, see Figure 5-7) to make sure they are not co-planar and easily seen by both of the cameras. The dimensions of the calibration pattern on the perpendicular planes are 48 x 36 x 27cm (height x width x depth) with each square the size of 3 x 3cm. The two views can be calibrated respectively by using the same method described in Chapter 3 and it results in one projection matrix for each camera. As the calibration object has a recognisable checkerboard pattern and the feature points as corners of black and white squares, a corner detector can be used for feature extraction and the epipolar geometry explained in Sub-section 5.2.2 can be applied to establish correspondences. To ensure the calibration quality, the corner points are again corrected by hand in the two spherical images in the evaluation of the proposed method.

Figure 5-7 Calibration object

With the projection matrices of the two cameras estimated respectively from the calibration, the steps of the view generation algorithm are explained in the next sub-section.

5.3.2 Proposed View Generation Algorithm

The proposed view generation algorithm consists of eight steps described in the following with the general notations used:

1) Recover both of the cameras' poses using the camera calibration method described in the previous sub-section.
2) Identify the corner points of the room boundary in both images and use these room corner points to recover the room geometry by intersecting the room corner points from the two camera centres.
3) Decide the position for a new view to be generated (based on user input), and determine which of the two spherical images is to be used as the main image for new view generation, based on which camera is closer to the new viewpoint (With two cameras denoted as Cameras A and B, and the new view to be provide by a virtual spherical camera denoted as Camera C, Camera A is assumed to be closer to Camera C, with Image A captured by Camera A to be used as the main image for generating the new view, Image C).
4) On Image A, define the boundary of the foreground object(s) and separate the foreground object(s) from the background (The extracted background is denoted as Image A^-, and the texture image of the foreground object(s) is $Object\ A$).
5) Find the area in Image B which corresponds to the occluded area in Image A, (this is denoted as $Occlusion\ B$), and this is done by applying the projection matrices obtained from Step 1) and room geometry obtained from Step 2). Define $Object\ B$ (in Image B) in the same manner as $Object\ A$. For each point of $Occlusion\ B \notin Object\ B$, transfer its pixel value back to Image A^- to fill the black holes in order to form a new background, Image A^+.
6) Render Image A^+ for the new view by generating regularly spaced grid points on the scene geometry, projecting them to Cameras *A* and *C* to produce a set of corresponding control points for warping, and applying a non-rigid transformation.
7) Render the texture of $Object\ A$ for the new viewpoint C by using control points, which are generated by projecting 3D intersection points of $Object\ A$ and B to Cameras A and C.
8) Combine the rendered background and foreground object texture to form Image C, any black holes still left are filled with the neighbouring pixels around them.

The overview of the whole system is illustrated in Figure 5-8. The two inputs, **Images *A*** and ***B*,** are used not only for camera poses and room geometry recovery, but also new view generation. For the latter, **Image *A*** is separated into the texture image, $Object\ A$, and the background, Image A^-, with some black holes left. $Object\ A$ is then rendered for the texture of the object(s) in the new view which is denoted as $Object\ C$. **Image *B*** is used for acquiring the occluded area in Image A, which is denoted as $Occlusion\ B$. However if any pixels of $Occlusion\ B$ are within the $Object\ B$ area, corresponding to object(s) being in front of the background in Image B, these pixels are excluded. $(Occlusion\ B - Object\ B)$ is then added to Image A^- and rendered for the new background. Finally, $Object\ C$ and the new rendered background Image C' are combined together to form the new view, Image C.

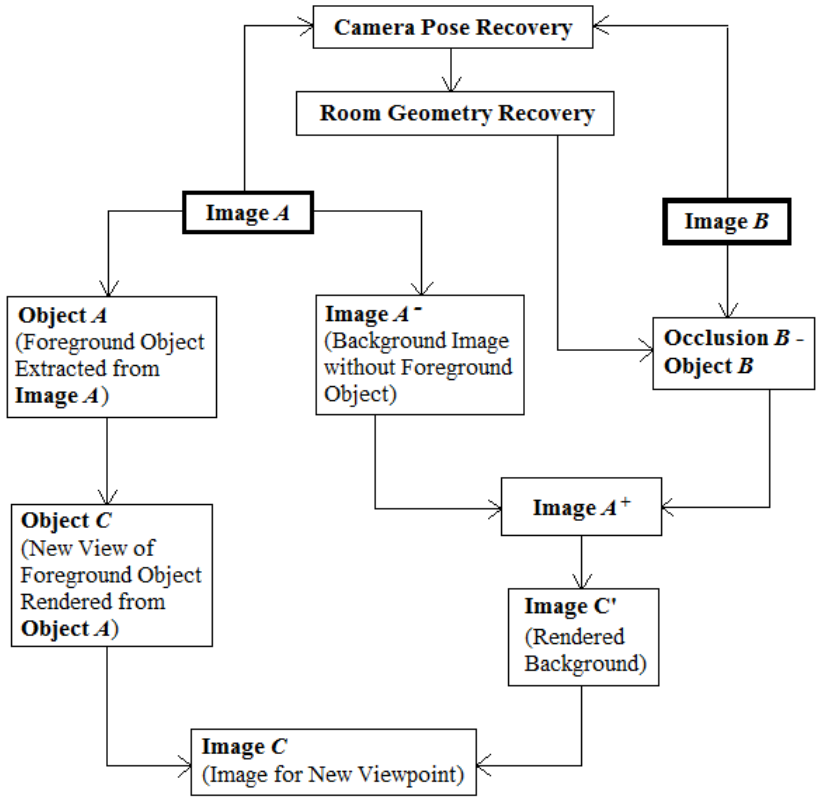


Figure 5-8 Flowchart of the system

5.4 View Generation Based on Computer Simulated Scenes

In this section, the proposed view generation approach is illustrated using synthetic spherical images generated by the computer based on a rectangular room geometry with one foreground object placed within it. The two view geometry is modelled by

specifying the projection matrices of two spherical cameras in turns of their viewing position and direction with respect to the modelled room geometry. After scene simulation presented in Sub-section 5.4.1, steps 4) to 8) of the view generation algorithm described in the previous sub-section are presented one by one in the following sub-sections.

5.4.1 Scene Simulation

The scene is generated as a 3D room with a shape of rectangular cube and with the dimensions of 8-by-4-by-4. It can be defined by using function $F(x, y, z)$ given by

$$F(x, y, z) = \begin{cases} x, & 0 \le x \le 8 \\ y, & 0 \le y \le 4 \\ z, & 0 \le z \le 4 \end{cases} \qquad \textbf{Equation 5-7}$$

A foreground object is created for the scene which is of the shape of two perpendicular panels and this modelled foreground object is shown in Figure 5-9. Its position in the room is defined by function $f(x, y, z)$ with

$$f(x, y, z) = \begin{cases} x, & 2.9 \le x \le 3.2 \\ y, & 2.5 \le y \le 2.7 \\ z, & 0.5 \le z \le 1.0 \end{cases} \qquad \textbf{Equation 5-8}$$

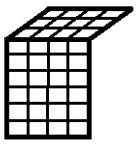


Figure 5-9 Modelled foreground object

Two cameras, denoted as A and B, are placed in the room with coordinates of $\boldsymbol{c_A} = [1.15, 1.4, 1.3]^T$ and $\boldsymbol{c_B} = [2.65, 1.4, 1.3]^T$, respectively.

The projection matrices for the two cameras, containing the rotation and translation parameters with respect to the world coordinates under which the room is defined, are constructed with

$$R_A = R_B = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad \textbf{Equation 5-9}$$

for rotation and

$$\boldsymbol{t_A} = -R_A \boldsymbol{c_A} \qquad \textbf{Equation 5-10}$$

$$\boldsymbol{t_B} = -R_B \boldsymbol{c_B} \qquad \textbf{Equation 5-11}$$

for translation. The rotation matrix has set the azimuth and elevation angels at $\theta = 90^o$ and $\varphi = 0^o$ which means both of the cameras are looking level to the horizon and sideways to the foreground object. The plan view of the room geometry containing the foreground object (shown in grey) and two camera positions (shown as circles) are shown in Figure 5-10.

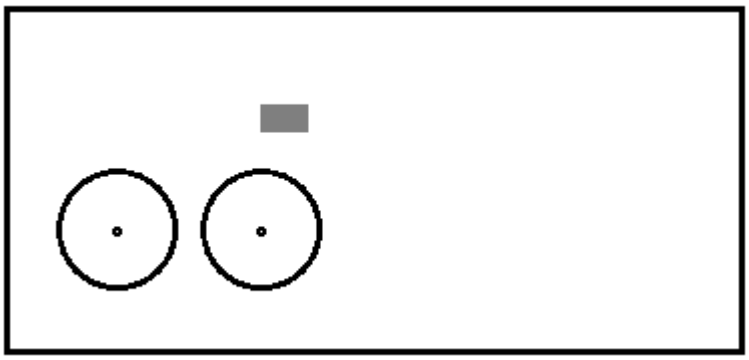


Figure 5-10 Plan view of the modelled room

The spherical sensors of the cameras are defined as unit spheres with focal length set to the value of 1. Hence, the projection matrices for the cameras can be directly written as $M_A = [R_A \, \boldsymbol{t_A}]$ for Camera A and $M_B = [R_B \, \boldsymbol{t_B}]$ for Camera B with no intrinsic parameters involved. For each synthetic scene point on each wall of the room or on each plane of the foreground object being captured by the cameras, it is first transformed by the projection matrix, and then multiplied by a scale factor to normalise its Euclidean length to 1 as its projection is on a unit sphere.

The images captured by Cameras A and B, denoted as I_A and I_B respectively, are produced by using dot grid sampling to cover the six walls of the modelled room and more dense dot grid sampling to cover the two planes of the foreground object. They are shown in Figure 5-11 and Figure 5-12. Taking Camera A as an example, for each sampling dot, $\boldsymbol{P} = [X^w, Y^w, Z^w]^T$, created in 3D, it is first transformed to the camera coordinates using

$$\boldsymbol{q} = M_A \boldsymbol{P} \qquad \textbf{Equation 5-12}$$

and its projection, $\boldsymbol{p} = [x^c, y^c, z^c]^T$, on the spherical sensor of Camera A is given by

$$\boldsymbol{p} = \frac{\boldsymbol{q}}{\|\boldsymbol{q}\|} \qquad \textbf{Equation 5-13}$$

where $\|\boldsymbol{q}\| = \sqrt{q_1^2 + q_2^2 + q_3^2}$ is the Euclidean length of $\boldsymbol{q}$. Finally the projection is transformed to its image coordinates in I_A as shown in Figure 5-11. Image points in I_B shown in Figure 5-12 are produced in the same way.

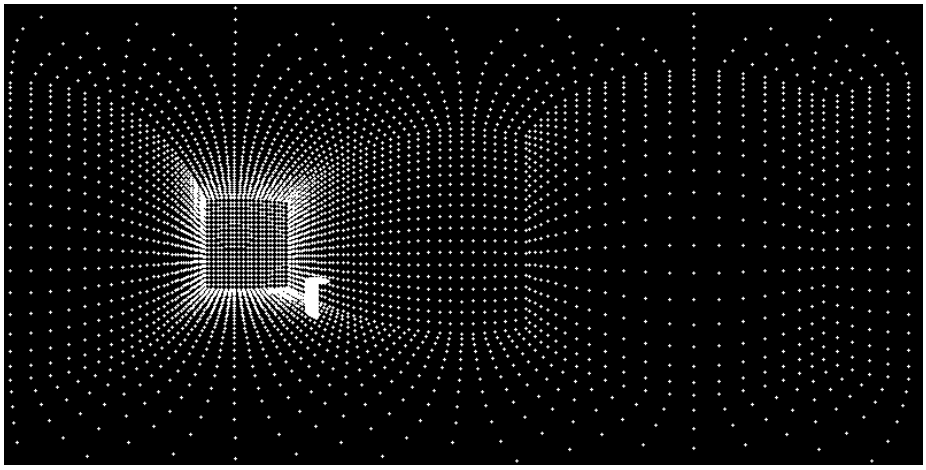


Figure 5-11 Synthetic spherical image - view 1

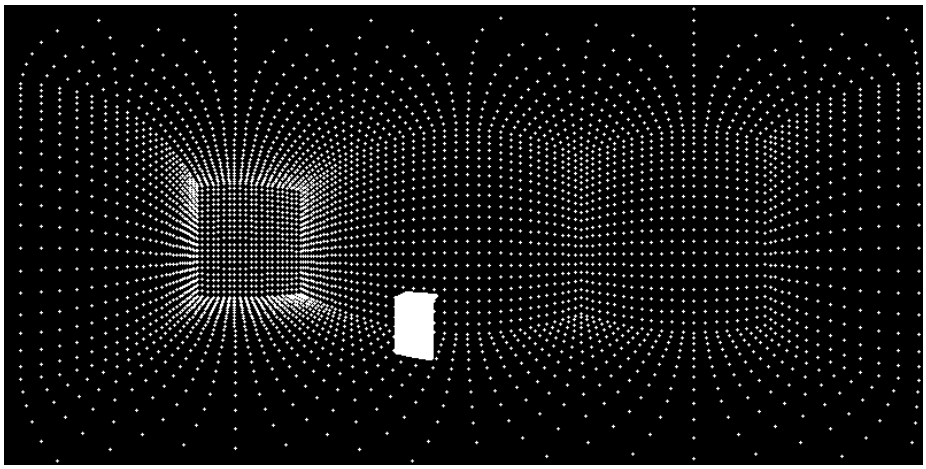


Figure 5-12 Synthetic spherical image - view 2

From the figures showing the panoramic views of the simulated room based on dot grid sampling, the relative camera positions used to capture the pictures are apparent. While the foreground object is seen to be at different positions and smaller in Figure 5-11 (as it is further away from Camera *A*) compared with that shown in Figure 5-12 (as it is closer to Camera *B*), the varying dot patterns are seen to indicate different distances from each spherical camera to each wall of the room.

5.4.2 Background Image for the New View

The new view will be generated from the two synthetic images simulated in the last section. The main image to be used for generation of the new view depends on where the new viewpoint (denoted as Camera C) is. With the coordinates of Camera C set to $\boldsymbol{c_C} = (1.8, 1.4, 1.3)$ which is slightly closer to Camera A than B, I_A is to be used as the main image for generating the new view, denoted as I_C. In the special case of the new viewpoint being exactly in the middle between the two camera centres, it does not matter which image is to be used.

5.4.2.1 Image Extraction

With I_A selected as the main image for new view generation, image extraction is carried out to separate the foreground object from the background image in I_A by drawing a bounding box surrounding the object. There are a number of image feature detection techniques which can be used to achieve automatic detection in this step, and selection of a particular method depends on the object feature and applicable scene constraints.

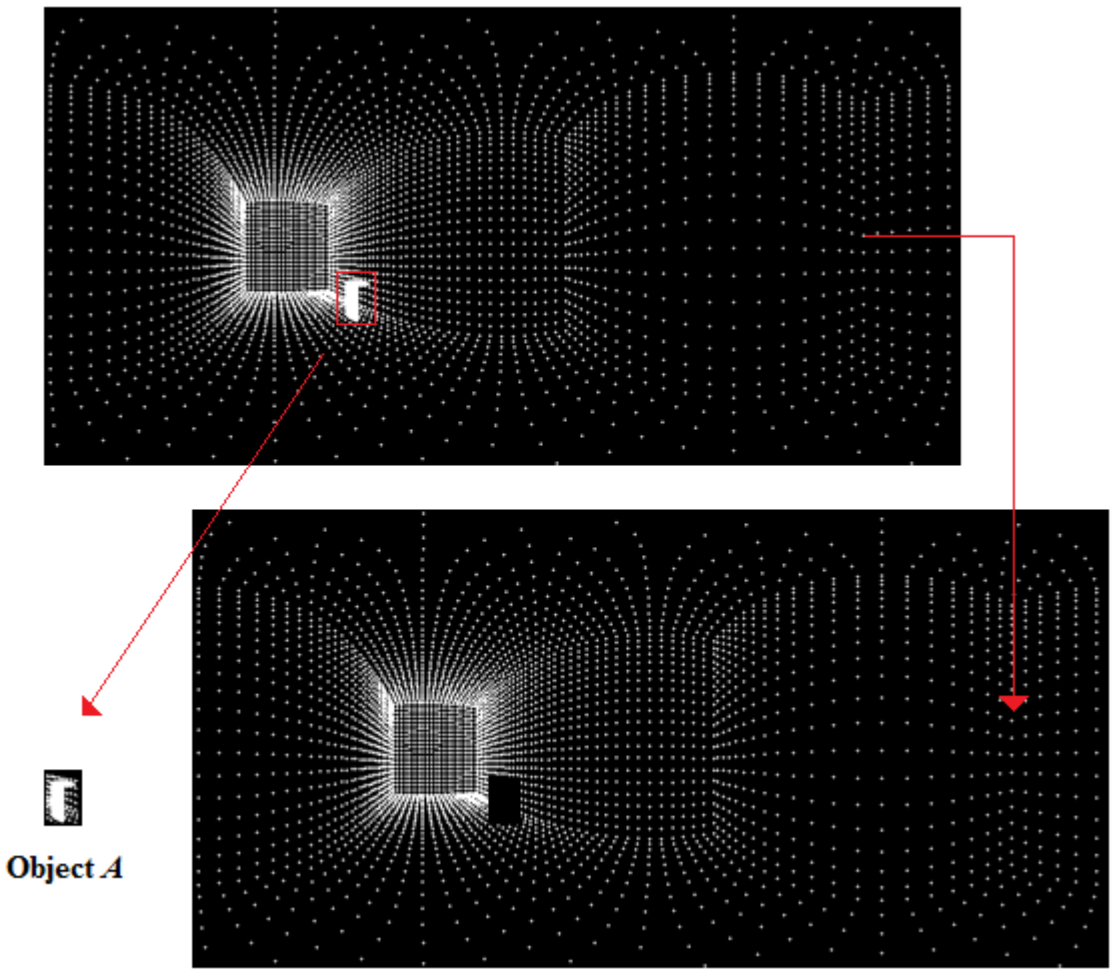


Figure 5-13 Image extraction

As shown in Figure 5-13, the extracted texture of the foreground object is referred as $Object\ A$ and the remaining background without the foreground object as I_{A^-}.

5.4.2.2 Background Image Make-Up

There are black holes left in I_{A^-} where the background is occluded by the foreground object after its extraction. Image pixels for this area need to be found from I_B. This process is illustrated in Figure 5-14. Firstly, every image point of $Object\ A$ is back projected onto the wall of the modelled room, which is the occluded background area. The 3D points within the occluded area are then projected onto Camera B to identify their corresponding image pixels in I_B, which is denoted as $Occlusion\ B$, and contains candidate image pixels for filling the black holes in I_{A^-}. On I_B, a bounding box is defined for the same foreground object and the points within this bounding box are defined as $Object\ B$.

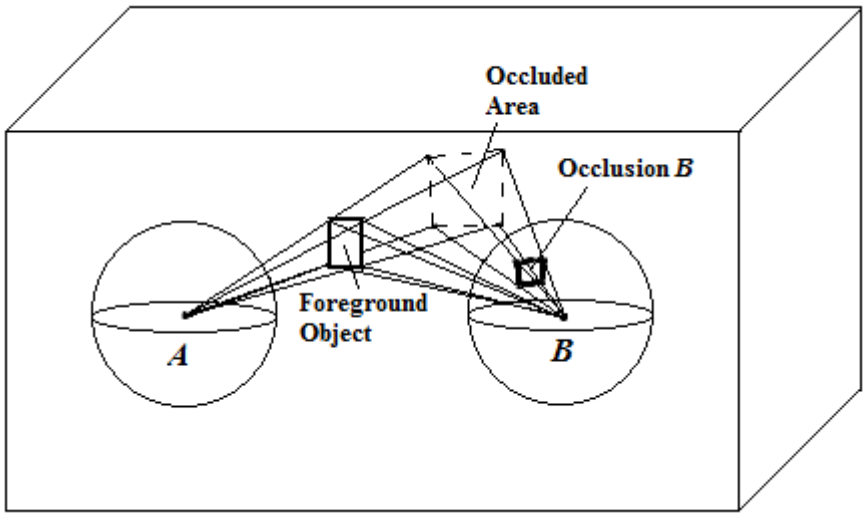


Figure 5-14 Locating occlusion

- If no point of $Occlusion\ B$ belongs to $Object\ B$, all candidate image pixels in $Occlusion\ B$ are used for filling up the hole in I_{A^-}.
- If a part of $Occlusion\ B$ belongs to $Object\ B$, i.e. the background area occluded from I_A is also partially occluded from I_B, only those candidate image pixels not belonging to $Object\ B$ are taken from $Occlusion\ B$ for filling up the hole in I_{A^-}.
- If every point of $Occlusion\ B$ belongs to $Object\ B$, the object is not considered as a foreground object but a part of the background.

For other cases with more than one foreground object, each foreground object is defined and dealt with individually to yield a number of different occluded areas and corresponding patches in I_B. With $Object\ B$ defined as the collection of all the foreground objects appeared in I_B, each patch in I_B corresponding to an occluded area in I_A is then compared with $Object\ B$ to identify the pixels for filling up the black holes in I_{A^-}.

To do back projection of the foreground object from I_A to the wall, vectors from the camera origin A to each point of $Object\ A$ on the spherical sensor are drawn to intersect with the wall(s) of the room, and the intersections are the 3D points on the wall being occluded from I_A by the foreground object. All the intersection 3D points on the wall are then projected to Camera B to locate their corresponding image patch in I_B, which is $Occlusion\ B$. Figure 5-15 shows the image of I_B superimposed with the image points of $(Occlusion\ B - Object\ B)$ coloured in blue.

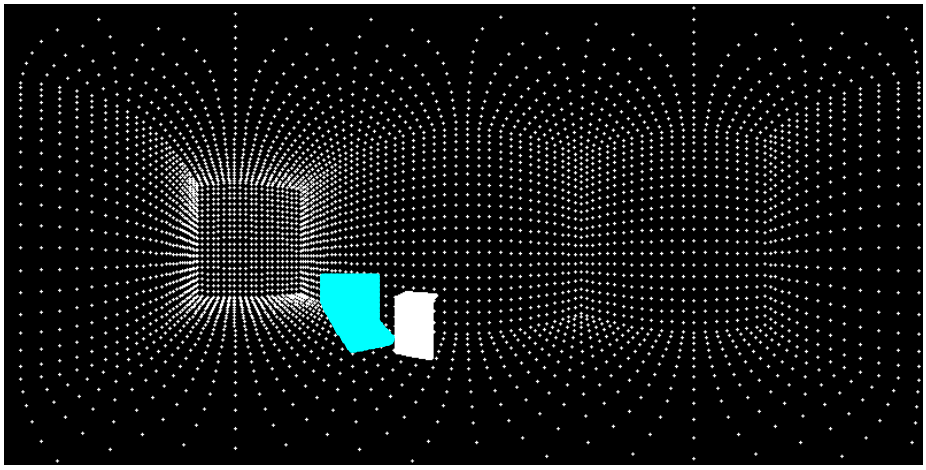


Figure 5-15 Patch found in I_B for making up the occluded area in I_{A^-}

Transforming the pixel values of $(Occlusion\ B - Object\ B)$ back to the black hole positions left in I_{A^-} and using linear interpolation (as explained in Sub-section 4.2.2) in their image coordinates, it yields the occluded background image shown in Figure 5-16:

Figure 5-16 Occluded background for I_A

By combining this small image patch with I_{A^-} shown in Figure 5-13, it results in an occlusion free image as shown in Figure 5-17.

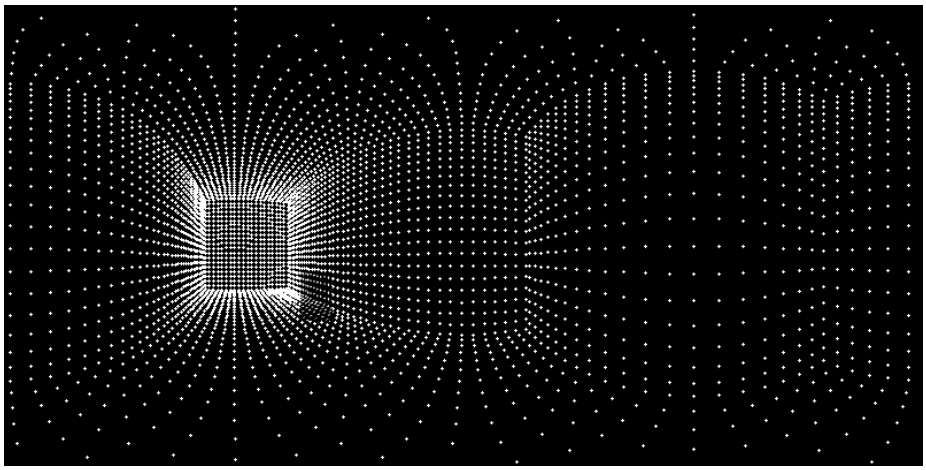


Figure 5-17 Occlusion free image *A*

The image shown in Figure 5-17 (denoted as I_{A^+}) corresponds to the image captured by Camera A without the foreground object being placed in the scene. It is to be used for rendering the background image of the new view. Comparing Figure 5-17 and Figure 5-13, it is observed that the original occlusion in I_{A^-} is mostly filled, but the new pixels appear to be blur (with less intensity levels). This is due to image interpolation and can be improved by using more sophisticated interpolation methods involving more time consuming calculations. Regardless of the interpolation deficiency, the dot grid pattern in this area is seen to be correct as it is consistent with the patterns in the surrounding area.

5.4.2.3 Background Image Rendering

In order to render the prepared background image I_{A^+} for the new viewpoint, a non-rigid transformation with control points is used to warp I_{A^+} to the new view I_C.

To provide the corresponding control points in I_{A^+} and I_C, regular grid points are generated on the walls of the modelled room and then projected onto Cameras A and C, respectively. In Figure 5-18, it illustrates the cameras placed in the room with the new viewpoint at Camera C closer to Camera A than Camera B. The points are generated sparsely and regularly on the walls and then projected onto Cameras A and C using the projection parameters specified before. The figure only shows the scenario with a small

number of points on the top of the room being projected. The projected points on the two spherical sensors are corresponding 3D points on unit spheres, which are used as control points for image warping.

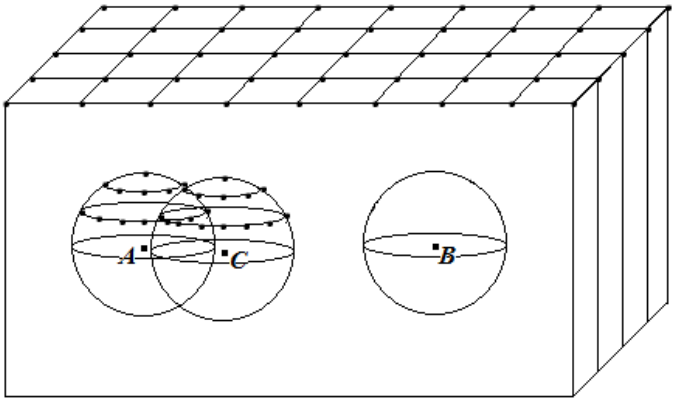


Figure 5-18 Correspondences geneartion

The transformation between the two groups of corresponding control points is nonlinear, because they were obtained by camera sensors with nonlinear characteristics. Therefore the image interpolation based on linear coordinate transformation is not applicable for the spherical camera case.

In this case, 3D correspondences on the sensors are first transformed into their 2D image coordinates. Thin-plate splines, based on an analogy of approximating the shape of a thin metal plate deflected by normal forces at discrete points, are then used to estimate the mapping function required for alignment of the two images using the corresponding 2D image coordinates as the control points.

The thin-plate spline method for image alignment was described by (Goshtasby, 1988). For n corresponding control points in the images $([(x_i, y_i), (X_i, Y_i)], i = 1,2, \dots, n)$, it maps one set of control points in one image to another set of control points in another image by using functions f and g with $(X_i = f(x_i, y_i), Y_i = g(x_i, y_i), i = 1,2, \dots, n)$ and maps other points in the images by interpolation. Imagining an infinite thin plate with point loads added to the plate at positions determined by the control points (x_i, y_i) in the reference image, the plate deflects at the control points and takes on values equal to the X-component of the corresponding control point (X_i, Y_i) in the target image. The surface obtained in this manner is the X-component mapping function between the two images and the Y-component mapping function can be obtained similarly. The surface

of an infinite plate obtained under the imposition of point loads is known as the surface spline and its equation is given in (Harder and Desmarais, 1972):

$$f(x,y) = a_0 + a_1 x + a_2 y + \sum_{i=1}^{n} F_i {r_i}^2 \ln {r_i}^2 \qquad \textbf{Equation 5-14}$$

where n is the number of point loads, r_i is the distance to the control point with ${r_i}^2 = (x - x_i)^2 + (y - y_i)^2$ and (x_i, y_i) denoting the position of the i^{th} control point in the reference image, and $f(x,y)$ is the elevation of the surface at point (x,y), i.e. the X-component of the point corresponding to (x,y) in the target image. The parameters of $a_0, a_1, a_2,$ and F_i with $i = 1, 2, \dots, n$ are determined by substituting the coordinates of corresponding control points into the above mapping function and solving the system of linear equations:

$$\begin{cases} \sum_{i=1}^{n} F_i = 0 \\ \sum_{i=1}^{n} x_i F_i = 0 \\ \sum_{i=1}^{n} y_i F_i = 0 \\ f(x_1, y_1) = a_0 + a_1 x_1 + a_2 y_1 + \sum_{i=1}^{n} F_i {r_{i1}}^2 \ln {r_{i1}}^2 \\ \vdots \\ f(x_n, y_n) = a_0 + a_1 x_n + a_2 y_n + \sum_{i=1}^{n} F_i {r_{in}}^2 \ln {r_{in}}^2 \end{cases}$$

Similarly for the thin-plate spline surface representing the Y-component mapping function between the two images, its equation is given by:

$$g(x,y) = b_0 + b_1 x + b_2 y + \sum_{i=1}^{n} G_i {r_i}^2 \ln {r_i}^2 \qquad \textbf{Equation 5-15}$$

For a set of 602 grid points generated on the walls of the modelled room, projected onto the two spherical sensors and transformed to their image coordinates as control points, applying the mapping functions derived from thin-plate splines to warp the background image I_{A^+} to the new viewpoint at Camera C yields the result shown in Figure 5-19.

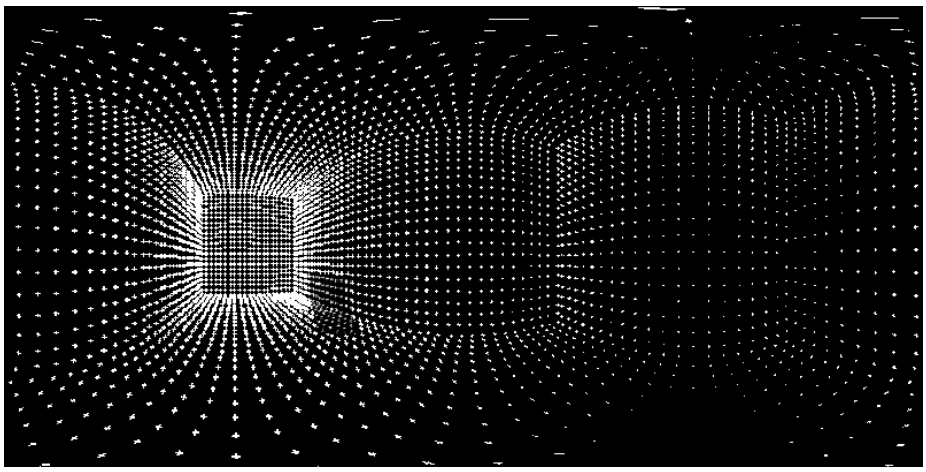


Figure 5-19 Rendered background for new viewpoint

The generated background image of the new viewpoint is compared with the original two spherical images shown in Figure 5-11 and Figure 5-12. The size and shape as well as the patterns on each wall shown in Figure 5-19 are seen to be correct as they are seen to be in-between of those shown in Figure 5-11 and Figure 5-12, and reflect the position of Camera C in relation to Cameras A and B.

From the tests conducted, the thin-plate spline surfaces as mapping functions have shown to be suitable for generation of new spherical views based on the two view geometry as they are sensitive to local geometric distortion between the images. This is due to the value of a surface at a point is determined by the nearby point loads and the influence of a point load on a point decreases as the distance of the point to the point load increases. This property ensures that local geometric distortion between the images will not be averaged equally all over the image but used locally in the warping process. The surfaces are therefore especially effective in mapping large images with less control points, images obtained from different viewing angles, and images obtained by a sensor with nonlinear characteristics (Goshtasby, 1988).

5.4.3 Foreground Object Texture for the New View

The image texture of the foreground object in I_A, denoted as $Object\ A$, is used for rendering the texture of the same foreground object in the new view. The process is similar to the one described in the previous section. The real coordinates of the 3D points on the foreground object can be found by the intersection method, which is to back project the light rays passing through the projections of the foreground object on

the spherical sensor from both Cameras A and B. The intersection results a 2D surface as point-to-point depth information cannot be recovered. Dense 3D points are then regularly generated on the object's surface and then projected onto Camera A and C as corresponding control points for the determination of the thin-plate spline surfaces.

The image of $Object\ A$ and its thin-plate spline based warping result (based on 105 pairs of control points) are shown in Figure 5-20. The view of the object from Camera C is slightly different and the object appears to have a bigger size as it is nearer to Camera C. This rendered texture is then added on top of the background image, I_{A^+}.

Figure 5-20 Rendered foreground object for new viewpoint

Figure 5-21 shows the final image generation result, I_C, for the new viewpoint. The image patch of the foreground object, whose coordinates are correctly preserved by the thin-plate spline mapping, appears to be at the correct position in I_C with its patterns matched with the background. Comparing I_C with I_A and I_B, it is clear to see that I_C is an intermediate view between I_A and I_B as Camera C is placed on the baseline of Camera A and B and slightly closer to Camera A. Some of the data points in I_C are again appearing with lower intensity values as the result of image interpolation.

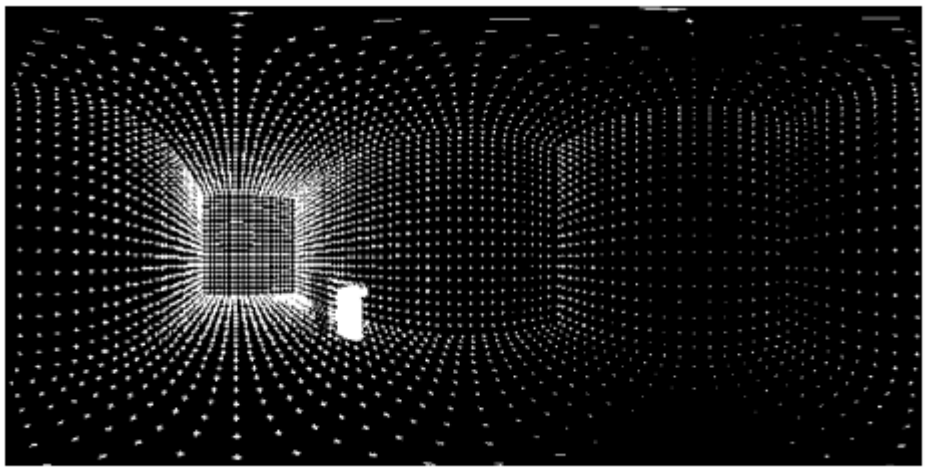


Figure 5-21 Synthetic spherical image - generated new view

5.5 View Generation Based on Real Scenes

In this section, the view generation algorithm is illustrated using real spherical images, which are taken from real scenes such as the lobby environment shown in Figure 5-22.

Figure 5-22 The lobby environment

The calibration object shown in Figure 5-7 is put into the scene for camera calibration, which will also be treated as a foreground object causing occlusion to the view captured. The floor plan of the lobby is shown in Figure 5-23, where (a) is the plan view of the floor with some chairs and a rubbish bin by the walls and considered as parts of the background, and (b) illustrates the imaging set-up with the two camera positions shown by two circles placed roughly in the middle of the room, and the calibration box with grid patterns visible to both of the cameras. The lobby is not in a regular rectangular shape, and it is connected with two short sections of corridors leading to double doors. By ignoring the two short corridors, the geometry of the floor is approximately a rectangle outlined by red in the figure.

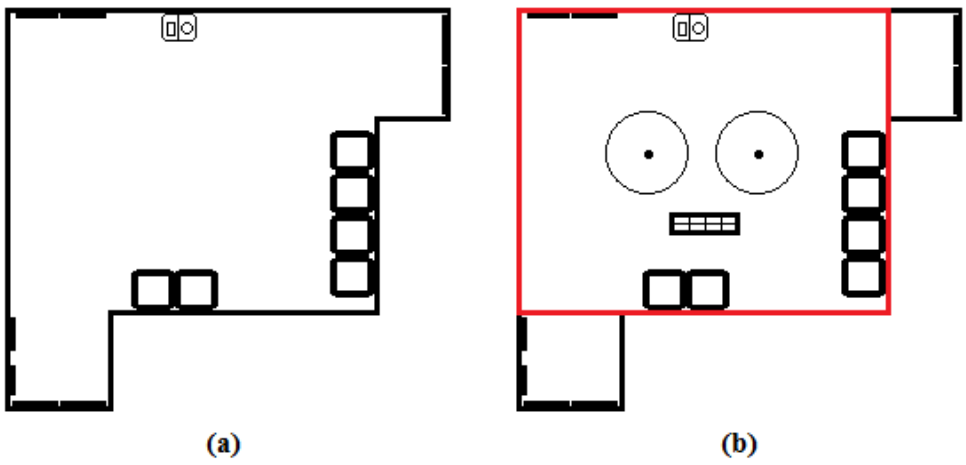


Figure 5-23 (a) Floor plan of the scene, and (b) Imaging set-up

5.5.1 Camera Pose and Room Geometry Recovery

The two spherical views of the lobby taken by the cameras are shown in Figure 5-24 and Figure 5-25.

Figure 5-24 Spherical image - lobby view 1

Figure 5-25 Spherical image - lobby view 2

From the images, it is seen that the surroundings are captured from two different viewpoints of A and B, and the calibration box is visible to both of the views but scanned from different angles. The red markers are not part of the scene but placed on the images to correspond to the corners of the rectangle cube outlined in Figure 5-23(b).

The calibration is performed as that described in Sub-section 5.3.1 and the two projection matrices are determined by using the projection of 84 corner points on the calibration box (using the method explained in Chapter 3). The projection matrices are defined up to a scale and denoted as M_A and M_B for Cameras A and B, respectively. With the project matrices containing the linear transformation from the world coordinate origin to the camera position and orientation, if M_A is written as $M_A =$

$[R_A\ \boldsymbol{t_A}]$ with R_A describing the rotation and $\boldsymbol{t_A}$ the translation (both scaled), and $M_B = [R_B\ \boldsymbol{t_B}]$ similarly, the coordinates of the camera origins A and B are given by:

$$\boldsymbol{c_A} = -{R_A}^T \boldsymbol{t_A} \qquad \textbf{Equation 5-16}$$

$$\boldsymbol{c_B} = -{R_B}^T \boldsymbol{t_B} \qquad \textbf{Equation 5-17}$$

By applying the above equations, the two camera positions are recovered with $\boldsymbol{c_A} = [-0.27, -0.32, -0.74]^T$ and $\boldsymbol{c_B} = [0.85, -0.31, -0.71]^T$, with respect to a chosen world coordinate origin located on the calibration pattern. The recovered poses of the cameras indicate the cameras were shifted along the x-axis. Furthermore, with the recovered rotation $R_A \cong sR_B$ where s is a small scale factor, both cameras were looking at roughly the same direction.

With the projection matrices and the poses for the two cameras estimated, the room geometry of the indoor environment can be recovered by a semi-automatic method, where room corner points are selected on the images (shown as the red markers on Figure 5-24 and Figure 5-25). 8 corner points of the room on each image are sufficient to recover the geometry by locating them on their spherical sensors and intersecting the light rays of each pair of the corresponding points. More points could be selected along wall edges to ensure the robustness of the final geometry estimation and this will be discussed in Chapter 6 as a future research interest. With points of interest selected in one spherical image, their locations on the second image can be estimated by using the epipolar constraint, as well as other constraints such as straight lines in the manmade environment. In this case, for simplicity on the implementation, 8 corner points are carefully marked by hand on each spherical image as shown before. They are marked on the edges of the main space in the lobby ignoring the two short corridors. As shown in Figure 5-26, with the camera projection geometry recovered, two light rays are drawn for each pair of the corner correspondences from the camera origins passing through the projection of the selected corner points on their spherical sensors until they intersect, and the intersection is the physical position of the room corner.

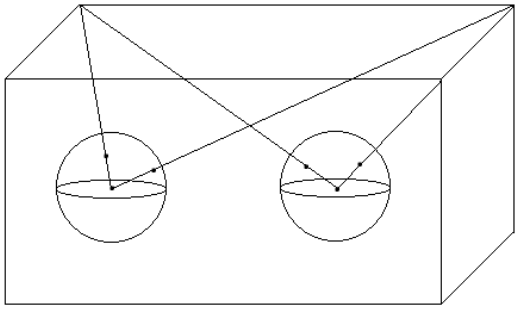


Figure 5-26 Room geometry recovery

Due to errors inherent with the camera pose recovery based on the calibration box and room corner points selected, the two lines from the camera origins do not often intersect. The intersection is hence defined as the midpoint of the shortest distance between the two un-intersected 3D lines (Bourke, 1998). It is also the case that the intersection points do not often form a set of corner points for a box, i.e., 4 of them do not lie on the same plane while they should. In order to correct this, six planes are fit between the points using an optimal fitting method with least squares error (least distance from the points to the planes) to yield the approximated geometry of the room. If more points on the room boundaries are selected for the estimation, the optimal fitting based on more point entries can produce a closer approximation of the physical room geometry.

With the room geometry recovered and the origins of the two cameras known, the new view corresponding to the selected viewpoint (Camera C) can be generated.

5.5.2 Background Image and Foreground Texture for the New View

Generation of background image and foreground texture of the new view is similar to that discussed in Sub-sections 5.4.2 and 5.4.3. As an example, let the new viewpoint be selected at the middle between the two original camera centres, and I_A (shown in Figure 5-24) be chosen as the main image for generation of the new view.

On I_A, the image patch containing the calibration box is separated from its background and this area is located in I_B by applying the camera poses and room geometry estimated. $Occlusion\ B$ (defined in the same manner as in the last section) is then found by image interpolation and is shown in Figure 5-27 .

Figure 5-27 Occluded background area for I_A

Apply this small image patch to fill the holes in the extracted background, image I_{A^+} for rendering the background of the new view is shown in Figure 5-28.

Figure 5-28 Occlusion free image *A*

Again the thin-plate spline surfaces are used for warping I_{A^+} to the new view based on 602 pairs of control points, and the rendered background is shown in Figure 5-29.

Figure 5-29 Rendered background for new view

Similarly the extracted foreground object image from I_A is rendered for the new view (based on 84 pairs of control points) and Figure 5-30 shows the foreground object texture before and after rendering.

Figure 5-30 Rendered foreground object for new view

Finally the generated new view for Camera *C* combining the foreground and background image is shown in Figure 5-31.

Figure 5-31 Spherical image - generated new view

From Figure 5-31, the surroundings in the generated image show correctly as an in-between view of the two original images shown in Figure 5-24 and Figure 5-25. The interpolation blur is not quite obvious as seen in the synthetic data before, even when observing with zooming. The new panoramic view generated at a novel viewpoint can be imported to a spherical image viewer as described in Section 2.3.2 to achieve the immersive navigation.

5.6 Results and Analysis

In this section, more experimental results are shown and analysed to demonstrate the effectiveness of the new view generation system. Results from both synthetic and real data are discussed.

5.6.1 Results from Synthetic Data

For the tests using synthetic data, the new view generated from two views in Section 5.4 is first assessed by comparing it with the absolute reference image generated directly from the simulated scene. This is then followed by more view generation results rendered from different camera separation distances.

5.6.1.1 Reference Image Based Assessment

In the first assessment, the new view generated for Camera C from two views of Cameras A and B in Section 5.4 is compared with the view generated directly from the scene. As the projection parameters for Camera C is known, the reference image I_r is generated in the same manner as using Cameras A and B to generate I_A and I_B. With

$$\boldsymbol{c_C} = [1.8, 1.4, 1.3]^T$$

and

$$\boldsymbol{R_C} = \boldsymbol{R_A} = \boldsymbol{R_B} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 5-32 shows the reference image I_r generated, and Figure 5-33 shows the superimposition of I_r and I_C (shown in Figure 5-21) for examination of their differences.

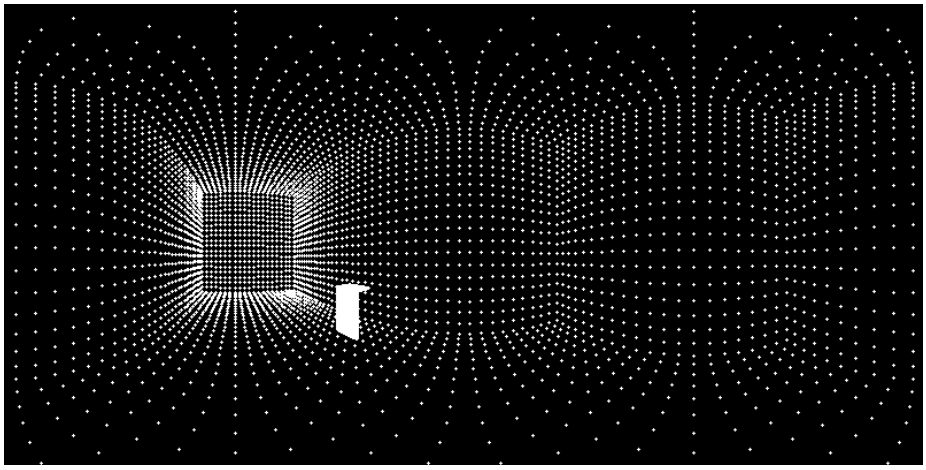


Figure 5-32 Synthetic spherical image - reference view

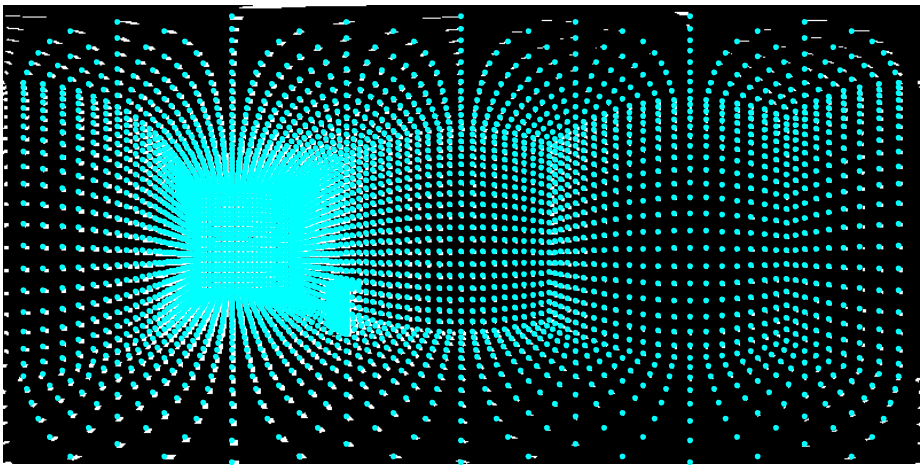


Figure 5-33 Point difference between reference and generated view

From Figure 5-33, it is observed that the reference data points of I_r (shown in blue) are well aligned with the corresponding data points of I_C (shown in white). The small differences between them are due to errors occurred at image warping by using thin-plate spline mapping functions. Furthermore, the differences are seen to increase towards the image borders.

5.6.1.2 Assessment of Wide Baseline View Generation

In this section, the baseline distance between two cameras is increased by moving Camera B away from Camera A along the x-axis of the world coordinates to evaluate the view generation results.

The first test is to fix the centre of Camera B at $\boldsymbol{c_B} = [3.65, 1.4, 1.3]^T$. A new I_B is generated for the new camera position as shown in Figure 5-34. Comparing this with I_A (Figure 5-11) and the original I_B (Figure 5-12), it is seen that the foreground object now appears to be captured from an opposite direction due to the camera position change.

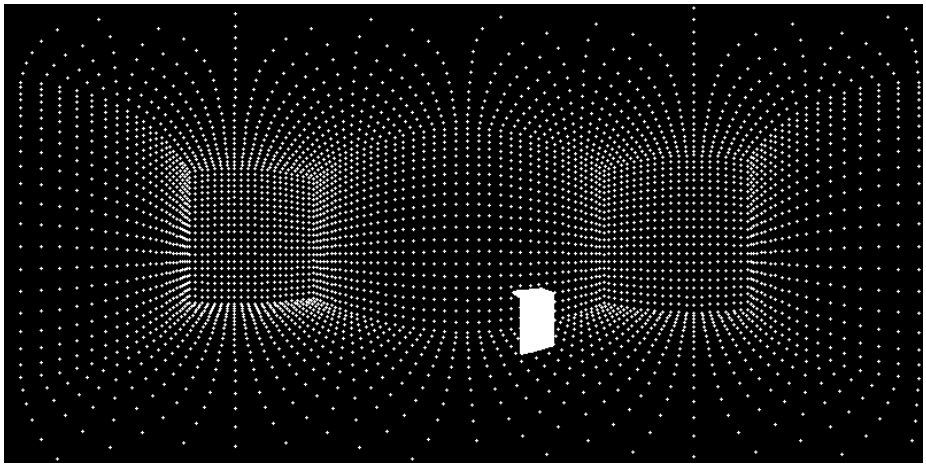


Figure 5-34 Test 1 on synthetic data - image I_B

A new view is generated with Camera C at the middle between Cameras A and B. With I_A chosen to be the main image for new view generation, the result is shown in Figure 5-35.

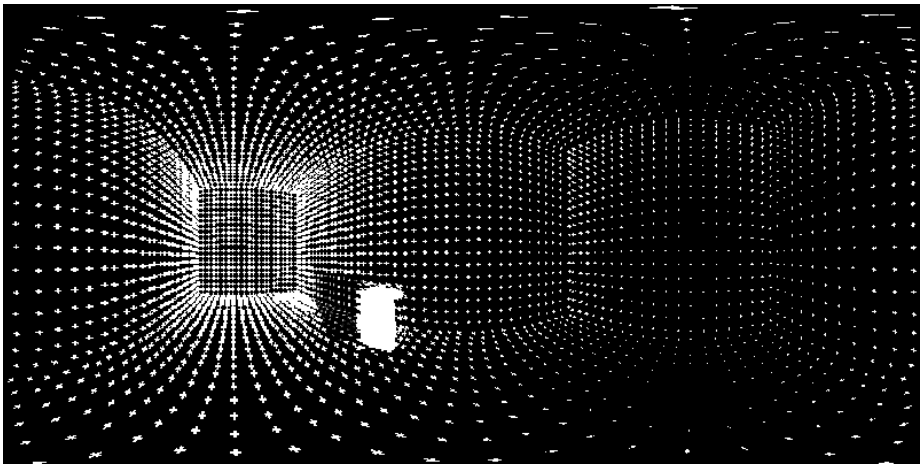


Figure 5-35 Test 1 on synthetic data - image I_C

Again a reference view (shown in Figure 5-36) is created to compare with the generated view and the differences of their point positions are shown in Figure 5-37.

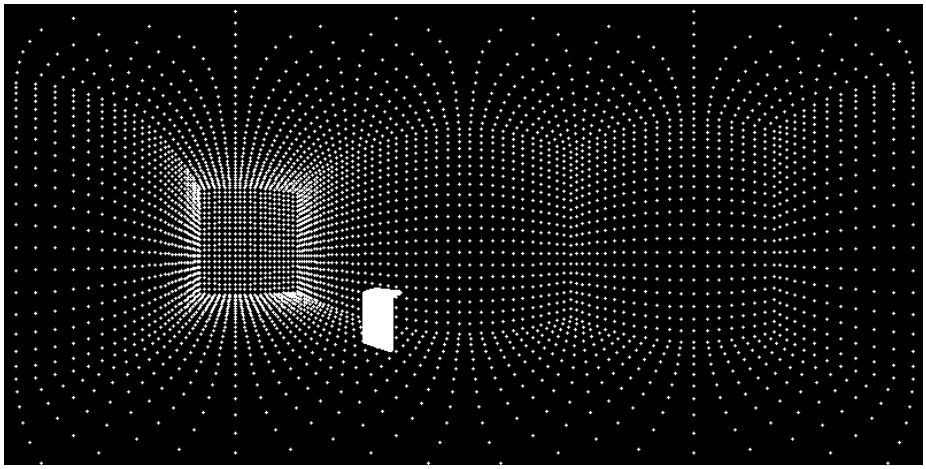


Figure 5-36 Test 1 on synthetic data - image I_r

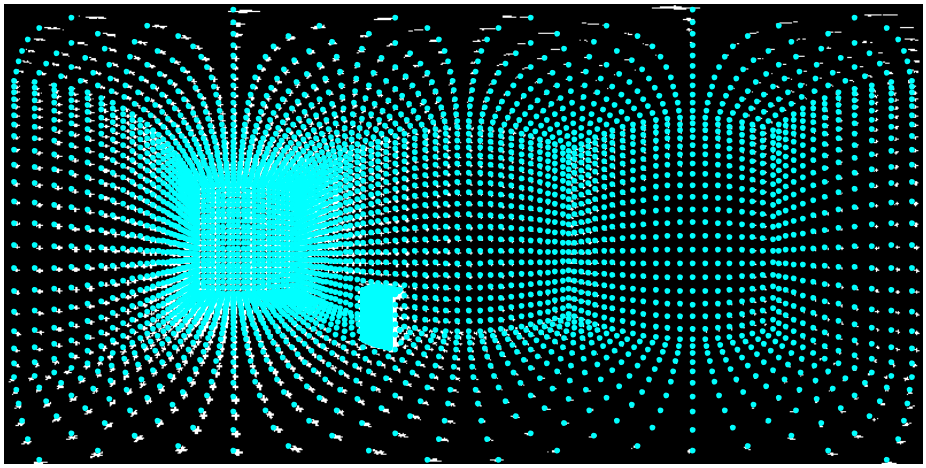


Figure 5-37 Test 1 on synthetic data - point difference

The second test is to fix the centre of Camera B at $\boldsymbol{c_B} = [6.65, 1.4, 1.3]^T$, which is much more farther away from Camera A. The new image generated for I_B is shown in Figure 5-38, and it is seen that Camera B is at a relatively long distance from the foreground object as indicated by the tiny size of the foreground object.

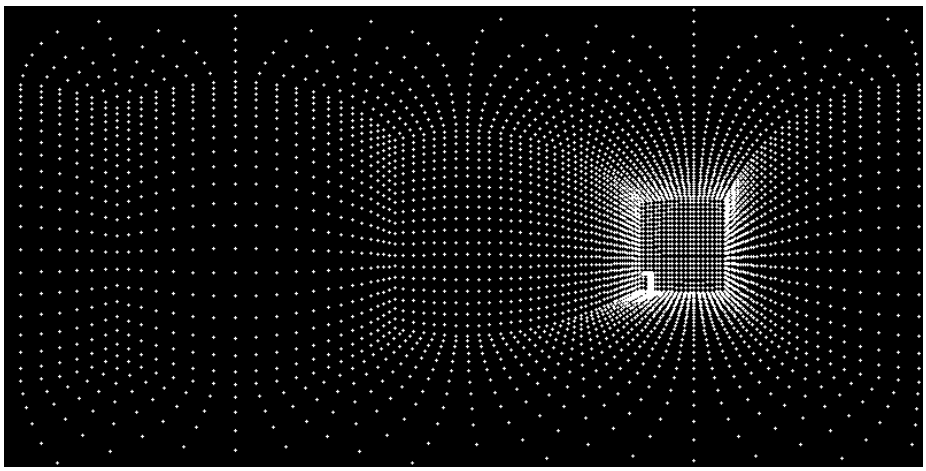


Figure 5-38 Test 2 on synthetic data - image I_B

The new view is generated again with Camera C at the middle point between Camera A and B. With I_A again used as the main image for rendering, the result of the generated new view is shown in Figure 5-39.

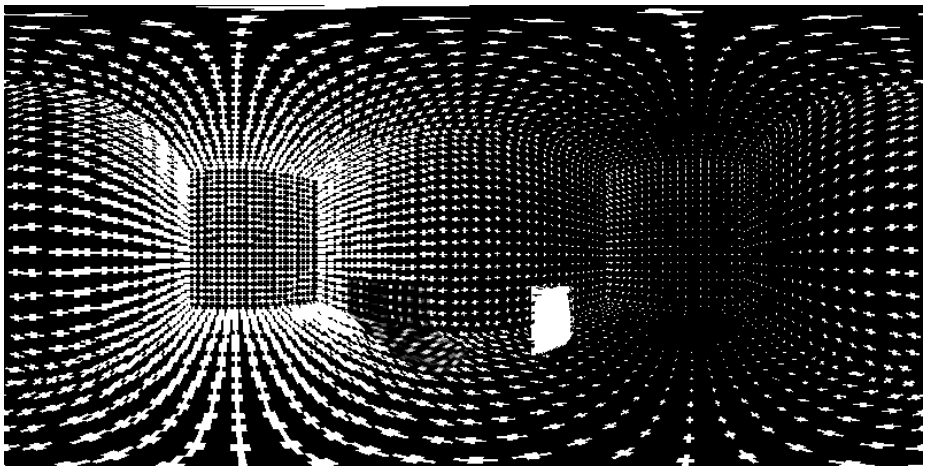


Figure 5-39 Test 2 on synthetic data - image I_C

Figure 5-40 shows the reference view created for comparing with the generated view, and Figure 5-41 shows the differences of their point positions.

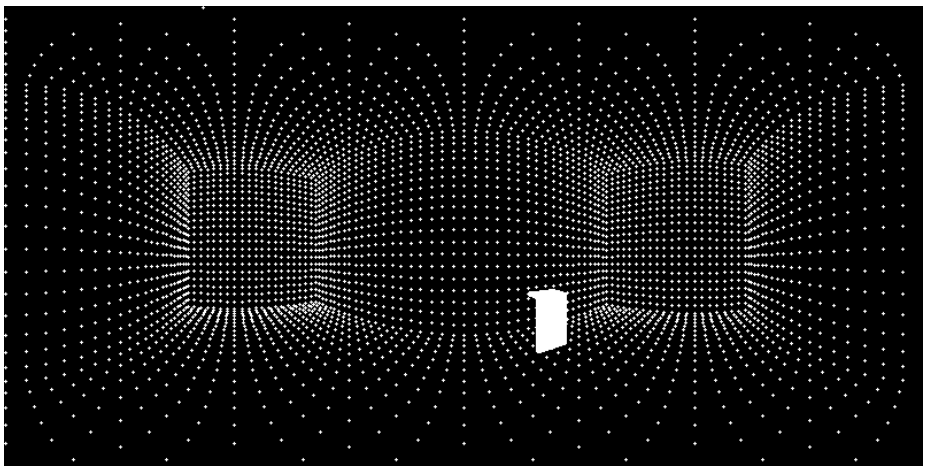


Figure 5-40 Test 2 on synthetic data - image I_r

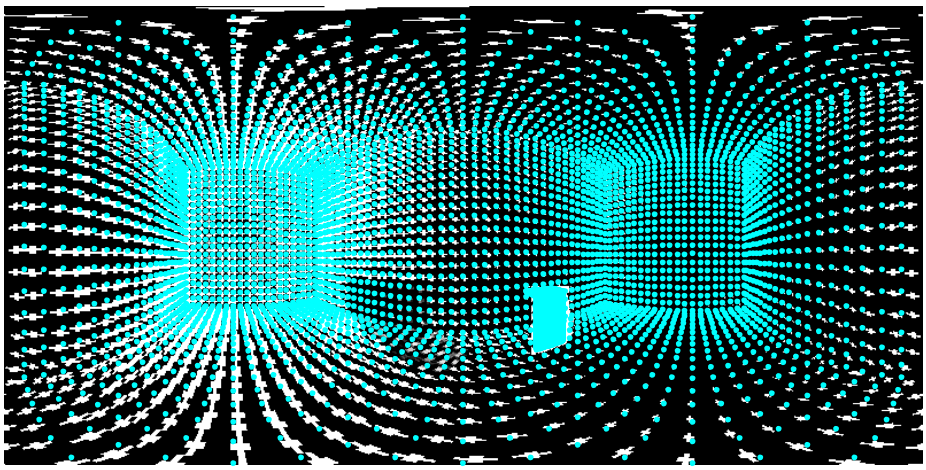


Figure 5-41 Test 2 on synthetic data - point difference

From the above two tests with the camera baseline increasing as the second camera is moved away from the first one, it is seen that position differences between the generated points and the reference become greater, and this trend is especially noticeable near the border area of the spherical image . The plan view of the imaging set-up in Test 1 and 2 is shown in Figure 5-42, where the two camera positions are illustrated for different tests in comparison with the original position of Camera *B* (shown in dashed circle). Furthermore, as the camera baseline increases, the effect of interpolation blur becomes more significant. Comparing the view generated from two views with wider baseline shown in Figure 5-39 with that generated from two views with narrower baseline shown

in Figure 5-35 and Figure 5-21, it is seen that the sizes of the interepolated data points in the former become more stretched.

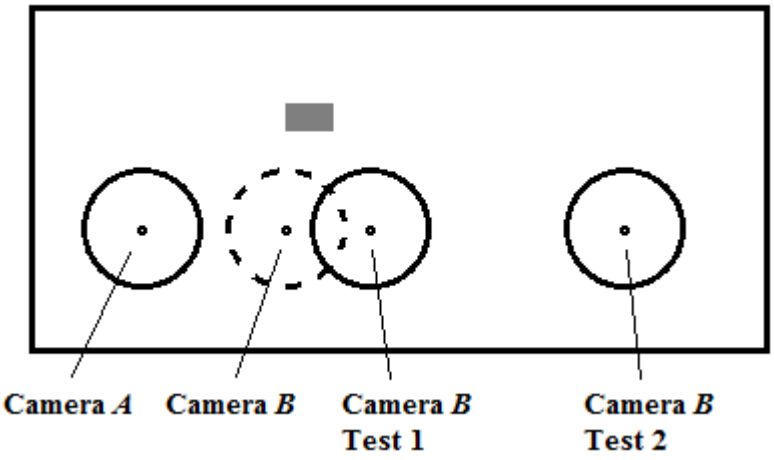


Figure 5-42 Plan view of tests on synthetic data

5.6.1.3 Assessment of Narrow Baseline View Generation

In the third test, the baseline distance between the two cameras is decreased by moving Camera B near to Camera A at $\boldsymbol{c_B} = [2.15, 1.4, 1.3]^T$. Since the two cameras are close to each other, the occluded background area in I_A is partially occluded in I_B as well. Figure 5-43 shows the image generated at Camera B and the occluded background area in I_A found in I_B (in blue), and it is seen that the view of this area is also partially blocked by the foreground object.

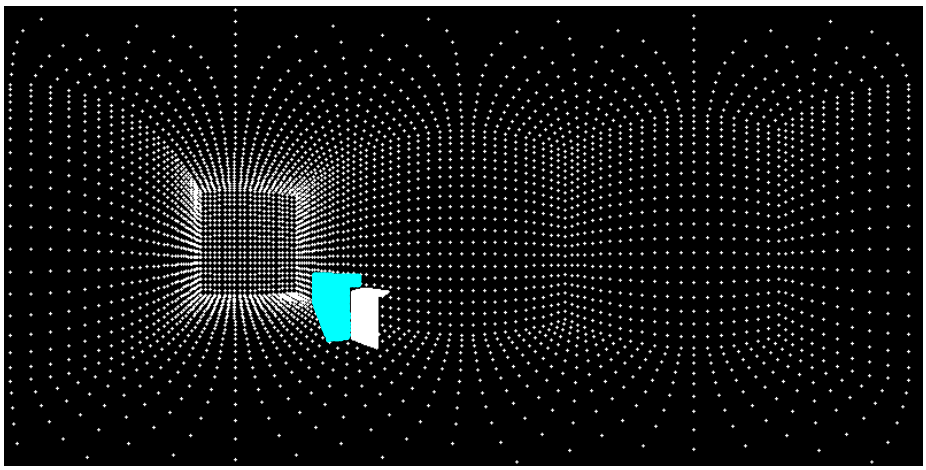


Figure 5-43 Test 3 on synthetic data - image I_B

The new view at Camera C is again generated at the middle point between Cameras A and B with I_A selected for rendering, the result of the new view generated is shown in Figure 5-44.

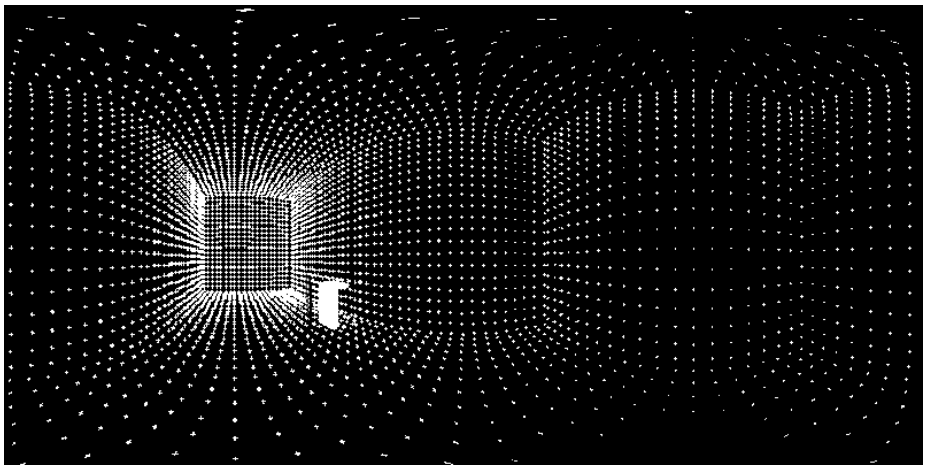


Figure 5-44 Test 3 on synthetic data - image I_C

It is seen that although the two images share mutual occlusion which results in black holes being left in the background image used to generated the new view, there is no black holes seen in the final new view generated as it is covered by the texture of the foreground object due to the camera positions being close to each other. The reference data point positions (in blue) plotted on top of the generated new view are shown in Figure 5-45.

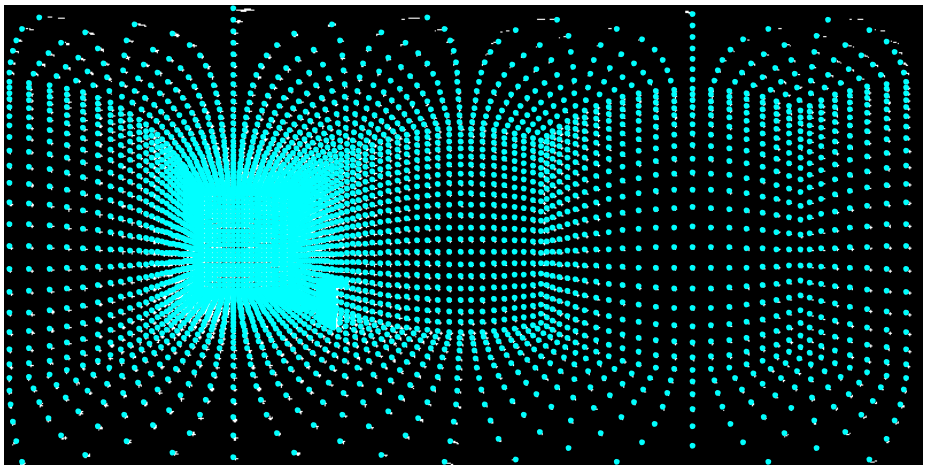


Figure 5-45 Test 3 on synthetic data - point difference

With little point differences between the new view generated and the reference image as shown in Figure 5-45, the best results are seen to be provided by the narrow baseline

with two camera positions close to each other. Furthermore, as shown in Figure 5-44, the blur effect in the new view caused by interpolation is also seen to be less noticeable.

5.6.2 Results from Real Data

5.6.2.1 Walkthrough Result

Using the same real image data shown in Section 5.5, two more spherical views are generated along the baseline of Cameras A and B. Together with I_A and I_B and the middle view generated before, these five views enable a route to be constructed as if walking from the position of Camera A to the position of Camera B. The five spherical views are shown in Figure 5-46 on the next page.

From Figure 5-46 showing a series of panoramic views along a route, it should be apparent that a panoramic navigation can be carried out by generating arbitrary in-between views from a pair of spherical images acquired at two different positions. Although some initial settings are made manually in terms of feature selection for camera calibration, room geometry identification and foreground object(s) extraction, the main algorithm proposed for view generation has shown to be able to generate new views automatically according to the user's navigating intention. The rendered images when applied to a spherical image viewer give the user all directional views as if immersed into the scene.

5.6.2.2 Assessment of Number of Control Points

Another interest to assess the result is to use different number of control points for warping the new view. The image result of Figure 5-31 was generated based on 602 pairs of control points. In this assessment, the number of control points is reduced gradually to observe the image quality of the generated view. It is observed that it makes no visual difference to the results until the number of control points drops to about 100. Figure 5-47 shows the rendered result of I_C generated at the middle point between Camera A and B when the number of control points is reduced to 98, with warping defects appeared in the image e.g. the pipes on the wall (near the calibration object) appear to be bent.

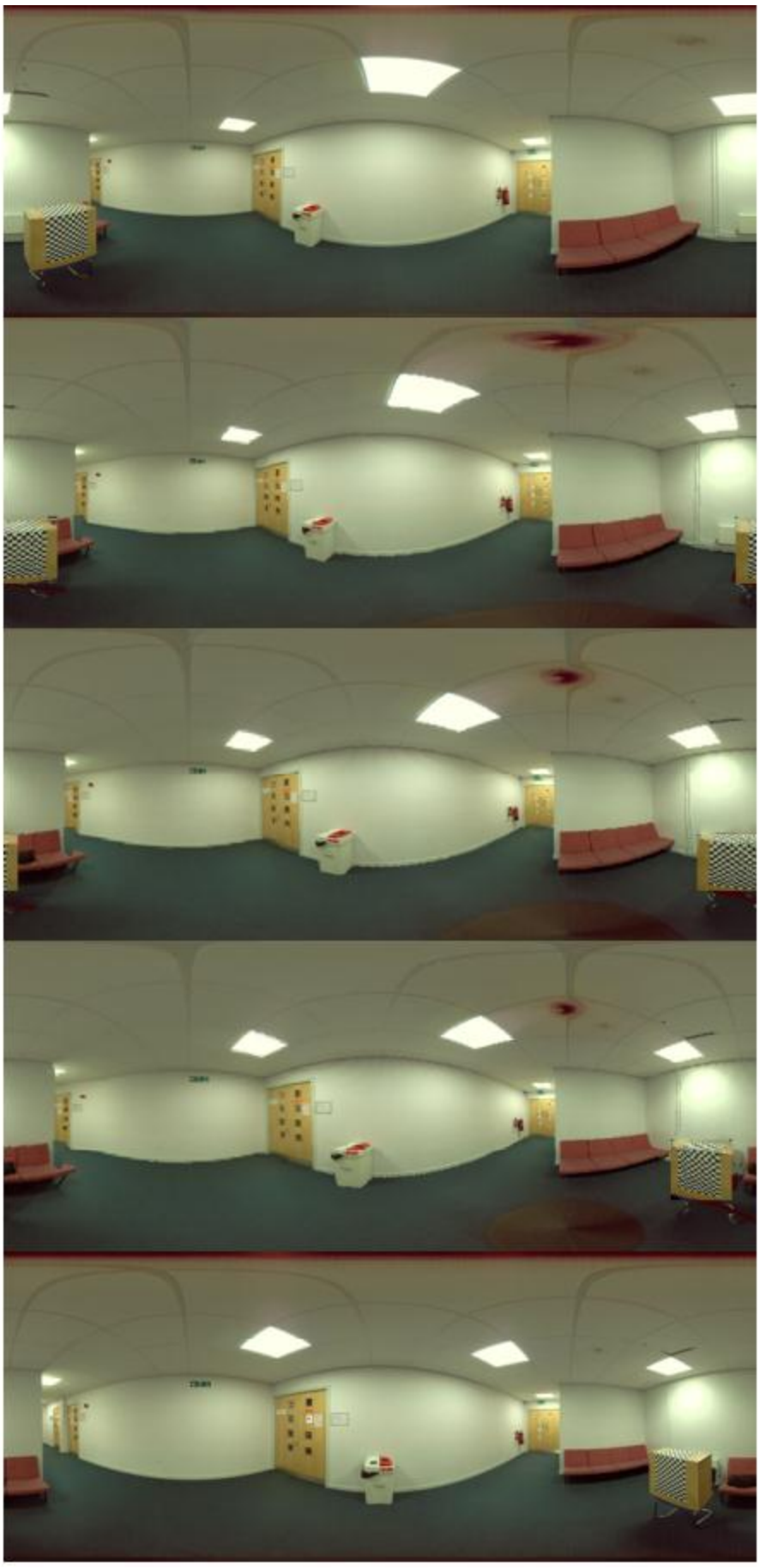


Figure 5-46 Test on real data - navigation view

Figure 5-47 Test on real data - 98 control points

Figure 5-48 and Figure 5-49 show the same generated new view but with the control points reduced to 56 and 26, respectively. It is seen from Figure 5-48 that the pipes bend more severely, the patterns on the ceiling become more distorted and the black object on the chair is missing. In Figure 5-49 the wall on the left is no longer straight.

Figure 5-48 Test on real data - 56 control points

Figure 5-49 Test on real data - 26 control points

It is shown that the denseness of the control points does affect the image warping result and more than 100 points control points across each acquired image are required for a smoothly generated new view. Note that all the real spherical images in this chapter are reduced to the size of 263 x 629 pixels to speed up the experiments. The control points for image warping need to be equally distributed onto the images with certain density as areas with less control points result in large deformation or missing of small parts of image content.

5.7 Conclusion

In this chapter, the algorithm of view generation from two spherical images is proposed and demonstrated using both simulated data and real scenes. The algorithm has shown to be suitable for indoor environments which can be approximated using multiple rectangular cubes, such as an office, a corridor or a lobby. Although foreground objects causing occlusion to the spherical cameras can be dealt with one by one, it is advisable to avoid them when taking the original spherical images of the scene due to the complexity it adds to the implementation.

The potential of the proposed algorithm was demonstrated by using computer simulated data without the need of camera calibration and room geometry recovery. The result shows a whole panoramic view being correctly generated at a novel camera position including even the correct background that has been occluded by the foreground object. A series of results has also been provided to show the goodness of the views generated from two cameras with close and wide separation distances. Traditionally camera distance and viewing angles affect significantly the result of new view generation as conventional cameras have limited field of view. In this case of spherical cameras being all directional, viewing direction becomes less an issue as the cameras captures all the scene visible to the viewpoint. As an obvious outcome of the experiments, the separation distance between the two camera positions has shown to have the key influence on the generated new view, i.e. the closer the two cameras are placed, the better rendered image result is in the generated view.

For view generation of real scenes based on the proposed method, spherical images are taken from different camera positions with a calibration object placed in the scene. With the camera poses and the room geometry recovered using the calibration object, new views can be generated by placing a virtual camera between two camera positions with

actual spherical images. This new image can be applied to the spherical image viewer developed in Chapter 4 for immersive visualisation. By generating more views in-between the original two camera positions, it enables immersive walkthrough of an environment by the user to appreciate all directional views of the surroundings along the route.

Furthermore, unlike Google Map's Street View function for outdoor roads, the proposed method allows new views to be generated freely between consecutive positions of the camera through user control, thereby enabling smooth immersive visualisation along an arbitrary route without jumping from one camera position to another with visual discontinuities.

Chapter 6 CONCLUSION

6.1 Concluding Remarks

This project contributes to image based immersive visualisation and new view generation by successfully developing a set of algorithms for rendering spherical images acquired from a rotating line spherical camera, and demonstrating their effectiveness through both computer simulated and real scenes. In particular, the work done and contributions are grouped as follows:

1) Properties of the spherical camera and its output images

- The properties of the spherical images have been studied and the nonlinear fisheye lens sensor projection is proved to be an equidistance projection. Particular significant finding is vertical field of view that is less than 180 degrees and is an unknown parameter.
- The transformation between any pixel point in the image coordinates and its physical coordinates in real world has been established. This transformation is via the projection on the camera's spherical sensor expressed in terms of the camera coordinates.
- A calibration method has been proposed based on perpendicular rigid panels covered by checkerboard patterns as the calibration object. The vertical field of view was recovered as well as the camera's projection matrix.

2) Visualisation based on one spherical image

- By modelling the scene using a unit sphere mapped with one spherical image, a method of image mapping has been developed to allow the user to navigate within the unit sphere by changing the viewing position and orientation as well as field of view. This allows the appropriate part of the scene to be projected onto the viewpoint window when the user moves. A GUI has been created for the user to interact with the system in real-time.
- An algorithm has been developed to correct image distortion for visualisation within one sphere mapped image when the user moves way from the sphere centre. This is based on a 3D intermediate projection plane and the distortion free navigation results are shown in frames of a walkthrough video, as well as on a six-faced cubical panorama generated for Virtual Reality visualisation facilities such as the CAVE.

3) View generation based on two spherical images

- Epipolar geometry between a pair of spherical images has been studied. With a pair of spherical images captured at two different positions within the same scene, the essential matrix can be estimated from a number of image correspondences, i.e. 10 to 12 scattered pairs throughout the images.
- The spherical image pair can be rectified for the purpose of correspondences search. It is shown that the search along an epipolar curve (located by the essential matrix) becomes 1D and on an image column of the spherical image after rectification.
- The scene can be modelled by using two spherical images taken from different camera positions. For an indoor environment modelled by synthetic data, a third camera is virtually put into the scene to capture a spherical view in relation to the existing two. Different virtual camera positions are tested for the algorithm.
- For real image data, by using rectangular cubes to approximate indoor environment and camera poses recovered through a calibration object, control points can be generated for image warping for the new view. An algorithm has been developed to process the background image and foreground object separately and to find the occluded area. The new view is generated through thin plate spline surfaces as mapping functions.
- Various experiments are carried out using both synthetic and real image data. A walkthrough between two camera positions is demonstrated with in-between spherical views generated. The system enables new arbitrary spherical viewpoint to be placed within the modelled scene hence to carry out free user navigation.

6.2 Future Work

The future research interests including algorithm development and applications can be summarised into four aspects described in the following sub-sections.

6.2.1 Correspondence Search via Epipolar Geometry

Correspondence search between a pair of images has always been a research interest in image processing and computer vision. Image correspondences enable the link to be established between the image pair and two view geometry within the scene. A disparity depth map and 3D reconstruction can also be achieved for new view generation if dense correspondences are available.

Different methods have been proposed for correspondences search between a pair of planar images, such as cross-correlation, RANSAC (Fischler and Bolles, 1981) and Scale invariant feature transformation (SIFT) (Lowe, 1999). These methods have been experimented on the spherical image pairs in this project and they do not perform well for the search or depth map generation, as the images are obtained from a nonlinear spherical sensor.

However, the epipolar geometry existing between the two spherical views is a useful constraint for correspondence search as the search can be narrowed to along the epipolar curve or along a 1D vertical image column when rectified. From the rectified image shown in Figure 5-6 in Chapter 5, if the other spherical image is also rectified, the two rectified images can be used together to find pairs of corresponding columns containing the same image content. In order to observe this clearly, both of the rectified images are rotated by 90 degrees and one of them is shifted (as if rotating its sensor 90 degrees horizontally) to let the corresponding image rows match. As shown in Figure 6-1, the same image content appears on each pair of the horizontal rows but stretched or compressed with different proportions. Hence, one possible solution for correspondence search is to match the image row profiles by local adjustment with different scales. Another suggestion is to detect the features like corners and lines to match the image features.

6.2.2 Room Geometry from Epipolar Geometry

It is observed from the spherical images (e.g., Figure 5-2 and Figure 5-3) that vertical lines are preserved through the spherical projection as the walls appear to be straight on the image. This is due to the camera's self-rotation which scans vertical lines linearly. On the other hand, it is seen from a rectified image such as Figure 5-6 that some horizontal lines in the scene are preserved and become vertical on the rectified image, e.g., the long table and the top and bottom edge of the windows.

It is known from Figure 5-6 that vertical image columns are epipolar curves being rectified to longitudes of the sensor sphere, therefore the conclusion can be made that the epipolar curves on the image preserve some of the horizontal lines in the real world. Note that this is true when the two cameras are placed on the same horizontal level, which is natural for image sampling of the scene.

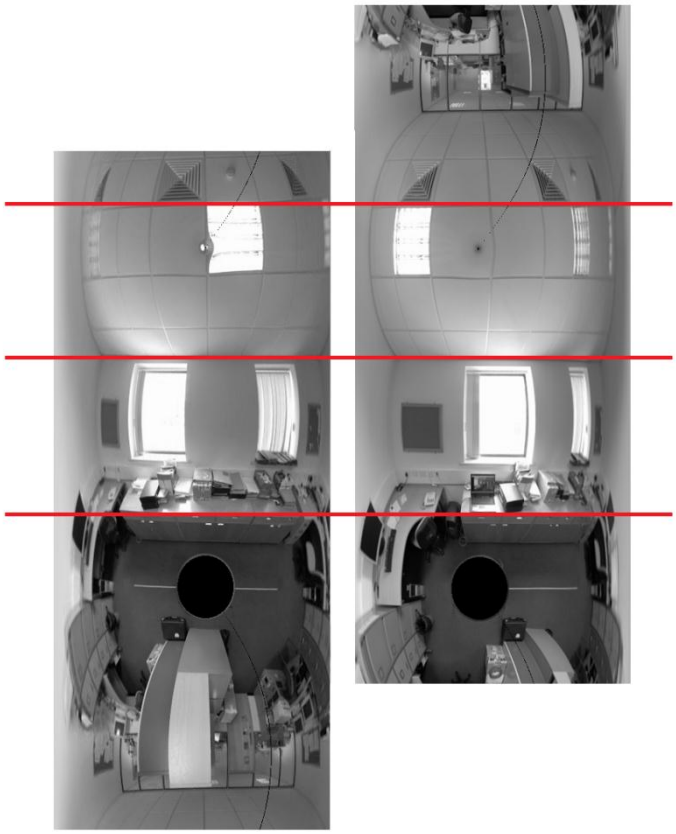


Figure 6-1 Row correspondences on rectified spherical images

This is useful for the room geometry recovery especially when not all the corner points are visible in the image. The detection of straight lines on the image is also made easier through the above constraint. For a horizontal straight line in the real world, as shown in Figure 6-2, if its position is parallel to the baseline of the two spherical cameras, it is preserved to become part of the epipolar curves on the cameras' sensor spheres.

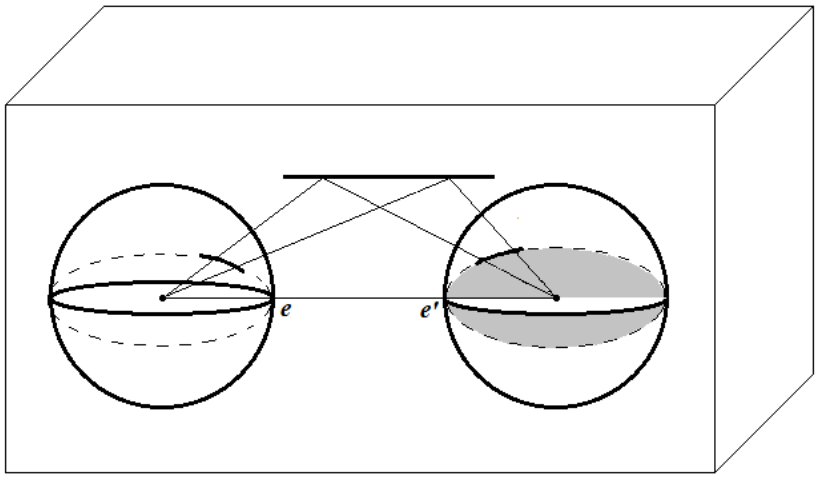


Figure 6-2 Epipolar geometry for a horizontal straight line

Hence for the boundaries of the room often parallel or perpendicular to the cameras' baseline, the vertical edges are preserved by the spherical image itself and some of the horizontal edges are part of the epipolar curves. These added constraints can be applied to room geometry recovery to improve the accuracy of the results and reduces the dependency on the image contents. More points on the room edges can also be found through these constraints to estimate the result and improves the robustness of the recovered room in terms of its geometry.

6.2.3 Virtual Reality Application

Another future work interest is the application of the VR system to the developed algorithms. The available equipment includes HMD and CAVE with InterSense tracking system (InterSense, 2004) in the Visualisation Lab at the University of Central Lancashire.

A Head Mounted Display normally consists of a head tracking device and a pair of special 3D display-glass mounts to the user's head and connected to a computer which generates the correct views for the display. A picture of the HMD is shown in Figure 4-2 in Chapter 4. The head tracker has a full six degrees of freedom i.e. x, y, z, pan, tilt, and roll, and the displayed view for the user has a natural horizontal field of view to human of 160 to 170 degrees.

The CAVE stereoscopic display system has three large screens and behind each screen there are two back projectors displaying images for the left and right eye to achieve the 3D stereoscopic effect. Immersive navigation can be achieved if the generated panoramic view of the scene is projected onto the screens and the user's movement is obtained by the InterSense tracking system attached to the user and the signal receiver installed on the ceiling of the display room.

In both of the above cases, if a stereo image pair is acquired instead of individual spherical image at the image acquisition step by using a special tripod, both of the stereo image results can be generated for the new view and hence to provide a 3D illusion.

6.2.4 A Complete Visualisation Model

The final possible work for the future is to combine the visualisation with one and two spherical images to achieve a more complete model of an indoor multi-room environment.

An indoor environment consisting of rooms and an L shaped corridor is shown in Figure 6-3. For the corridor area, it is straight forward to model the environment by multiple spherical images marked from 1 to 5. Each adjacent spherical image pair can be treated by the developed algorithm and the visualisation can be modelled based on every two images as long as they are taken with a large overlapped area. Recovered room geometry can be verified through different pairs of the spherical images and finalised. Free movement within the corridor area can be achieved.

For each individual room with a number of objects causing occlusion (marked by spherical images number 6 to 9), the panoramic overview of the room can be achieved by one spherical image. If the visualisation algorithm developed for one spherical image is applied here, the user can observe the room with movement up to a certain range and sees the correct image result.

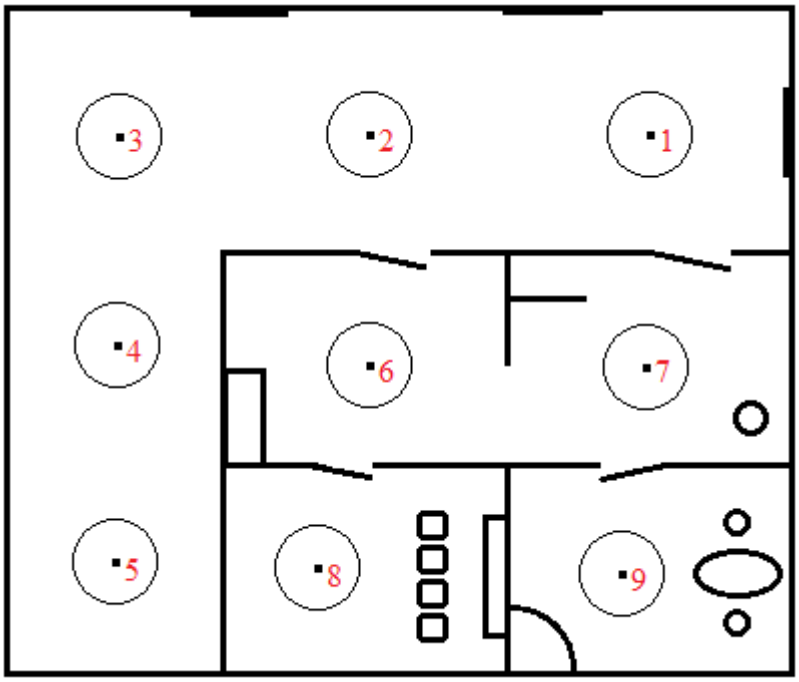


Figure 6-3 Indoor multi-room environment

When the user wishes to move from one room to another, a certain range near the doors of the virtual environment is defined. Collision detection (Ericson, 2005) is used when the user travels to the boundary of the door and it switches to another room if the user ‘pushes’ the door. If the multiscreen VR display system is used to visualise the result, the sudden change of the surroundings projected onto the screens could be disturbing. This could be improved if image morphing is applied to smooth the sharp change and the rendered intermediate results are displayed gradually.

With the combination of processing one and two view spherical images, the visualisation and navigation within a complex indoor environment could be modelled and simulated.

REFERENCES

ADELSON, E. H. & BERGEN, J. R. 1991. The Plenoptic Function and the Elements of Early Vision. *Computational Models of Visual Processing.* MIT Press.

AYACHE, N. & LUSTMAN, F. 1991. Trinocular Stereo Vision for Robotics. *IEEE Trans. Pattern Anal. Mach. Intell.*, 13, 73-85.

BAKSTEIN, H. & PAJDLA, T. Panoramic mosaicing with a 180° field of view lens. Omnidirectional Vision, 2002. Proceedings. Third Workshop on, 2002. 60-67.

BAKSTEIN, H. & PAJDLA, T. Rendering novel views from a set of omnidirectional mosaic images. Computer Vision and Pattern Recognition Workshop, 2003. CVPRW '03. Conference on, 16-22 June 2003. p74.

BENOSMAN, R. & KANG, S. B. (eds.) 2001. *Panoramic vision: sensors, theory, and applications*: Springer-Verlag New York, Inc.

BLUNDELL, B. 2008. *An introduction to computer graphics and creative 3-D environments*, London, Springer.

BOURKE, P. 1998. *The shortest line between two lines in 3D* [Online]. Available: http://paulbourke.net/geometry/lineline3d/ [Accessed 15th July 2011].

BRADLEY, D., BRUNTON, A., FIALA, M. & ROTH, G. 2005. Image based navigation in real environments using panoramas. *In IEEE International Workshop on Haptic Audio Visual Environments and their Applications.*

BURSCHKA, D., HAGER, G. D., DODDS, Z., JAGERSAND, M., COBZAS, D. & YEREX, K. 2003. Recent Methods for Image-Based Modeling and Rendering. *Proceedings of the IEEE Virtual Reality 2003.* IEEE Computer Society.

CAPEL, D. 2004. *Image mosaicing and super-resolution*, London, Springer.

CHAN, Y.-F., FOK, M.-H., FU, C.-W., HENG, P.-A. & WONG, T.-T. A panoramic-based walkthrough system using real photos. Computer Graphics and Applications, 1999. Proceedings. Seventh Pacific Conference on, 1999. 231-240.

CHEN, S. E. 1995. QuickTime VR: an image-based approach to virtual environment navigation. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques.* ACM.

CHEN, S. E. & WILLIAMS, L. 1993. View interpolation for image synthesis. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques.* Anaheim, CA. ACM.

CLAUS, D. & FITZGIBBON, A. W. A Rational Function Lens Distortion Model for General Cameras. *In:* ANDREW, W. F., ed., 2005. 213-219.

CRUZ-NEIRA, C., SANDIN, D. J. & DEFANTI, T. A. 1993. Surround-screen projection-based virtual reality: the design and implementation of the CAVE.

Proceedings of the 20th annual conference on Computer graphics and interactive techniques. Anaheim, CA. ACM.

CRUZ-NEIRA, C., SANDIN, D. J., DEFANTI, T. A., KENYON, R. V. & HART, J. C. 1992. The CAVE: audio visual experience automatic virtual environment. *Commun. ACM,* 35, 64-72.

CYGANEK, B. & SIEBERT, J. P. 2009. *An introduction to 3D computer vision techniques and algorithms,* Chichester, U.K., J. Wiley & Sons.

DREW, H. 2005. The fundamentals of photography. Lausanne, Switzerland: AVA Academia.

ERICSON, C. 2005. *Real-time Collision Detection,* Morgan Kaufmann.

FERNANDES, K. J., RAJA, V. & EYRE, J. 2003. Cybersphere: the fully immersive spherical projection system. *Commun. ACM,* 46, 141-146.

FIALA, M. Immersive panoramic imagery. Computer and Robot Vision, 2005. Proceedings. The 2nd Canadian Conference on, 9-11 May 2005. 386-391.

FIALA, M. & ROTH, G. 2005. Automatic Alignment and Graph Map Building of Panoramas. *HAVE 2005 - IEEE International Conference on Haptic Audio Visual Environments and their Applications.* Ottawa, Ontario, Canada.

FISCHLER, M. A. & BOLLES, R. C. 1981. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM,* 24, 381-395.

FU, C.-W., WONG, T.-T. & HENG, P. A. 1999. Computing Visibility for Triangulated Panoramas. *In Proceedings of Eurographics Rendering Workshop 1999, Eurographics.* Springer-Verlag.

FUJIKI, J., TORII, A. & AKAHO, S. 2007. Epipolar Geometry Via Rectification of Spherical Images. *In:* GAGALOWICZ, A. & PHILIPS, W. (eds.) *Computer Vision/Computer Graphics Collaboration Techniques.* Springer Berlin / Heidelberg.

FULLVIEW. 1999. *FullView® Panoramic Cameras* [Online]. Available: http://www.fullview.com/ [Accessed 15th July 2011].

FUSIELLO, A., TRUCCO, E. & VERRI, A. 2000. A compact algorithm for rectification of stereo pairs. *Mach. Vision Appl.,* 12, 16-22.

GLASBEY, C. A. & MARDIA, K. V. 1998. A review of image-warping methods. *Journal of Applied Statistics,* 25, 155-171.

GONZALEZ, R. C., WOODS, R. E. & EDDINS, S. L. 2004. *Digital Image processing using MATLAB,* Upper Saddle River, N. J., Pearson Prentice Hall.

GOOGLE. 2011. *Google Maps with Street View* [Online]. Available: http://maps.google.com/help/maps/streetview/ [Accessed 15th July 2011].

GORTLER, S. J., GRZESZCZUK, R., SZELISKI, R. & COHEN, M. F. 1996. The lumigraph. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* ACM.

GOSHTASBY, A. 1988. Registration of images with geometric distortions. *Geoscience and Remote Sensing, IEEE Transactions on,* 26, 60-64.

GROSS, H. (ed.) 2005. *Handbook of optical systems*: Weinheim, Wiley-VCH, c2005-.

GUTTMANN, A. J. (ed.) 2009. *Polygons, polyominoes and polycubes,* Dordrecht: Springer, with Canopus Academic Publishing Ltd. .

HARDER, R. L. & DESMARAIS, R. N. 1972. Interpolation using surface splines. *Journal of Aircraft,* 9, 189-191.

HARRIS, C. & STEPHENS, M. A Combined Corner and Edge Detection. Proceedings of The Fourth Alvey Vision Conference, 1988. 147-151.

HARTLEY, R. & ZISSERMAN, A. 2003. *Multiple view geometry in computer vision,* Cambridge; New York, Cambridge University Press.

HIROTA, T., NAGAHARA, H. & YACHIDA, M. 2006. Calibration of Rotating Line Camera for Spherical Imaging. *In:* NARAYANAN, P., NAYAR, S. & SHUM, H.-Y. (eds.) *Computer Vision – ACCV 2006.* Springer Berlin / Heidelberg.

HORRY, Y., ANJYO, K. & ARAI, K. Tour into the picture: using spidery mesh interface to make animation from a single image ACM SIGGRAPH 97 Conference Proceedings, 1997. pp. 225-232.

HUGIN, D. 2004. *Hugin - Panorama photo stitcher* [Online]. Available: http://hugin.sourceforge.net/ [Accessed 15th July 2011].

IMMERVISION. 2010. *ImmerVision Enables® 360°* [Online]. Available: http://www.immervision.com [Accessed July 15th 2011].

INTERSENSE 2004. Product Manual for use with the IS-900 SimTracker & VETracker.

JABI, W. Visualizing and Investigating Architectural Space Using Spherical Panoramic Imaging. ACSA Technology Conference, 2000 Cambridge, Massachusetts. Proceedings of the 2000 ACSA Technology Conference.

JÄHNE, B. 1997. *Digital image processing : concepts, algorithms, and scientific applications,* Berlin; London, Springer.

JUHO, K. 2006. A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* 28, 1335-1340.

KANG, H. W., PYO, S. H., ANJYO, K.-I. & SHIN, S. Y. 2001. Tour Into the Picture using a Vanishing Line and its Extension to Panoramic Images. *Computer Graphics Forum,* 20, 132-141.

KANG, S. B. & SZELISKI, R. 1997. 3-D Scene Data Recovery Using Omnidirectional Multibaseline Stereo. *Int. J. Comput. Vision,* 25, 167-183.

KITCHENS, S. A. 1998. *The QuickTime VR book : creating immersive imaging on your desktop,* Berkeley, Calif., Peachpit Press.

KUMLER, J. J. 2000. Fish-eye lens designs and their relative performance. *Proceedings of SPIE,* 360-369.

LAVEAU, S. & FAUGERAS, O. D. 3-D scene representation as a collection of images. Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision & Image Processing., Proceedings of the 12th IAPR International Conference on, 9-13 Oct 1994. 689-691 vol.1.

LEVOY, M. & HANRAHAN, P. 1996. Light field rendering. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques.* ACM.

LHUILLIER, M. & LONG, Q. 2003. Image-based rendering by joint view triangulation. *Circuits and Systems for Video Technology, IEEE Transactions on,* 13, 1051-1063.

LI, Y.-M. & ZHANG, J. X. 1998. *Least Squares Plane* [Online]. Available: http://www.infogoaround.org/JBook/LSQ_Plane.html [Accessed 15th July 2011].

LOWE, D. G. 1999. Object Recognition from Local Scale-Invariant Features. *Proceedings of the International Conference on Computer Vision-Volume 2 - Volume 2.* IEEE Computer Society.

MANFROTTO, L. 2010. *Manfrotto 055 MAG FIBER TRIPOD* [Online]. Available: http://www.manfrotto.com/product/0/055MF3/_/055_MAG_FIBER_TRIPOD_3_SCT [Accessed 15th July 2011].

MÁRQUEZ, J. J. 2002. An Introduction to Virtual Reality. Humans and Automation Seminar.

MCMILLAN, L. 1997. *An Image-Based Approach To Three-Dimensional Computer Graphics.* Ph.D. Dessertation, University of North Carolina.

MEDIA, I. 2011. *Dodeca® 2360 Camera System* [Online]. Available: http://www.immersivemedia.com/products/index.shtml [Accessed 15th July 2011].

MENDONCA, P. R. S. & CIPOLLA, R. 1999. A simple technique for self-calibration. *Proceedings 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Cat No PR00149,* 500-505.

MERRIAM-WEBSTER 2007. *Merriam-Webster's dictionary and thesaurus,* Springfield, Mass., Merriam-Webster; Colchester : TBS [distributor].

MINDS-EYE-VIEW. 2009. *IPIX 360-degree Video and Photography Products and Solutions* [Online]. Available: http://www.ipix.com [Accessed 15th July 2011].

MONTAGE. *Montage - An Astronomical Image Mosaic Engine* [Online]. Available: http://montage.ipac.caltech.edu/ [Accessed 15th July 2011].

NIKON, C. 2011. *Nikon Imaging Products - AF Fisheye-Nikkor 16mm f/2.8D* [Online]. Available: http://imaging.nikon.com/lineup/lens/specoalpurpose/fisheye/af_fisheye16mmf_28d/index.htm [Accessed 15th July 2011].

NIXON, M. S. & AGUADO, A. S. 2007. *Feature Extraction & Image Processing for Computer Vision*, ACADEMIC PRESS

OH, B. M., CHEN, M., DORSEY, J. & DURAND, F. 2001. Image-based modeling and photo editing. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques.* ACM.

PANOSCAN. *Panoscan Panoramic Camera* [Online]. Available: http://www.panoscan.com/ [Accessed 15th July 2011].

PARIAN, J. A. 2006. Panoramic Imaging: Techniques, Sensor Modeling and Applications. *International Summer School "Digital Recording and 3D Modeling".* Aghios Nikolaos, Crete, Greece: ISPRS Commission VI Special Interest Group "Technology Transfer Caravan".

PARIAN, J. A. 2007. *Sensor Modeling, Calibration and Point Positioning with Terrestrial Panoramic Cameras.* Doctor of Sciences, SWISS FEDERAL INSTITUTE OF TECHNOLOGY (ETH) ZURICH.

PARIAN, J. A. & GRUEN, A. 2003. A Sensor Model for Panoramic Cameras. *In:* KAHMEN, G. A. (ed.) *6th Conference on Optical 3D Measurement Techniques.* Zurich, Swizerland.

PARIAN, J. A. & GRUEN, A. 2004. An advanced sensor model for panoramic cameras. *International archives of Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXV, Part B Brown, D,* 37.

PHOTOGRAPHY, M. R. Lumix G 8mm f/3.5 Fisheye. *In:* 8MM_VS_9MM_FISHEYE_ILLUSTRATION.PNG (ed.). http://m43photo.blogspot.com/.

RAY, S. F. 2002. *Applied photographic optics: lenses and optical systems for photography, film, video, electronic and digital imaging*, Focal Press.

REINHARD, E. 2006. *High dynamic range imaging : acquisition, display, and image-based lighting,* Amsterdam; London, Elsevier. Morgan Kaufmann Publishers.

SCHEIBE, K., KORSITZKY, H., REULKE, R., SCHEELE, M. & SOLBRIG, M. 2001. EYESCAN - A High Resolution Digital Panoramic Camera. *Proceedings of the International Workshop on Robot Vision.* Springer-Verlag.

SCHNEIDER, D. & MAAS, H.-G. 2003. Geometric Modelling and Calibration of a High Resolution Panoramic Camera. *Optical 3-D Measurement Techniques VI,* II, 122-129.

SCHNEIDER, D. & MAAS, H.-G. Application and accuracy potential of a strict geometric model for rotating line cameras. Panoramic Photogrammetry Workshop, 2004 Dresden, Germany. International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, Part 5/W16.

SHADE, J. 2002. Approximating the Plenoptic Function. *UW CSE Technical Report.*

SHAPIRO, L. G. & STOCKMAN, G. C. 2001. *Computer vision,* Upper Saddle River, N.J., Prentice Hall ; London : Prentice-Hall International.

SHUM, H.-Y., CHAN, S.-C. & KANG, S. B. 2007. *Image-based rendering,* [United States], Springer.

SHUM, H.-Y. & HE, L.-W. 1999. Rendering with concentric mosaics. *Proceedings of the 26th annual conference on Computer graphics and interactive techniques.* ACM Press/Addison-Wesley Publishing Co.

SHUM, H.-Y., NG, K.-T. & CHAN, S.-C. 2005. A virtual reality system using the concentric mosaic: construction, rendering, and data compression. *Multimedia, IEEE Transactions on,* 7, 85-95.

SHUM, H. & KANG, S. A Review of Image-based Rendering Techniques Proc. IEEE/SPIE Conference on Visual Communication and Image Processing (VCIP), 2000. 2-13.

SHUMAKER, R. 2011. *Virtual and mixed reality - new trends : international conference, Virtual and Mixed Reality 2011, held as part of HCI International 2011, Orlando, FL, USA, July 9-14, 2011 : proceedings,* Heidelberg, Springer.

SPHEROCAM Image of SpheroCam HDR. *In:* SPHERON_STUDIO-0280355.JPG (ed.).

SPHERON-VR. *SpheronVR Technologies* [Online]. Available: http://www.spheron.com/ [Accessed 15th July 2011].

SPHERON-VR. 2004. *SpheronVR User Manual* [Online]. Available: http://www.spheron.com.

SPHERONVR Image based lighting with SpheroCam HDR, Product Brochure. Waldfischbach-Burgalben, Germany.

SVOBODA, T. & PAJDLA, T. 2002. *Epipolar Geometry for Central Catadioptric Cameras* [Online].

SZELISKI, R. 1996. Video mosaics for virtual environments. *Computer Graphics and Applications, IEEE,* 16, 22-30.

SZELISKI, R. & SHUM, H.-Y. 1997. Creating full view panoramic image mosaics and environment maps. *Proceedings of the 24th annual conference on Computer graphics and interactive techniques.* ACM Press/Addison-Wesley Publishing Co.

TAKAHASHI, T., KAWASAKI, H., IKEUCHI, K. & SAKAUCHI, M. Arbitrary view position and direction rendering for large-scale scenes. Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, 2000. 296-303 vol.2.

TRUCCO, E. & VERRI, A. 1998. *Introductory techniques for 3-D computer vision,* Upper Saddle River, NJ, Prentice Hall.

TSAI, R. 1987. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *Robotics and Automation, IEEE Journal of,* 3, 323-344.

VIRTUALWORLDLETS-BOOM. *VR Interfaces: Binocular Omni Orientation Monitor* [Online]. Available: http://www.virtualworldlets.net/Shop/ProductsDisplay/VRInterface.php?ID=12 [Accessed 15th July 2011].

VIRTUALWORLDLETS-HMD. *WVN Virtual Dictionary: HMD* [Online]. Available: http://www.virtualworldlets.net/Resources/Dictionary.php?Term=HMD [Accessed 15th July 2011].

WATKINS, C. D., SADUN, A. & MARENKA, S. R. 1993. *Modern image processing : warping, morphing, and classical techniques,* Boston ; London, Academic.

WATT, A. H. 2000. *3D computer graphics,* Harlow, Addison-Wesley.

WEI, G.-Q. & MA, S. D. 1994. Implicit and explicit camera calibration: theory and experiments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 16, 469-480.

ZHANG, Z. 2000. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on,* 22, 1330-1334.

Appendix A CAMERA CORRECTION DATA

Fisheye Lens


```
<?xml version="1.0" encoding="UTF-8" ?>
- <XElementList.XLens>
- <XLens id="Nikon 16mm f/2.8D AF Nikkor Fisheye 412441">
<LensClass value="Nikon|16mm f/2.8D AF Nikkor Fisheye" />
<Manufacturer value="Nikon" />
<Category value="Special / Fisheye" />
<ModelName value="16mm f/2.8D AF Nikkor Fisheye" />
<NominalFocalLength value="16.0" />
<LS value="1:2.8" />
<Fisheye value="true" />
<ViewAngle value="180.0" />
<MinimumFStop value="2.8" />
<MaximumFStop value="22" />
<Diameter value="63.0" />
<Weight value="290" />
<Name value="Nikon 16mm f/2.8D AF Nikkor Fisheye 412441" />
<SerialNumber value="412441" />
<Comment value="" />
<DateCreated value="2006-03-16" />
<GUID value="1416996B-53388D84-DFD91F31-4582FD39" />
<DistanceSetting value="+INF" />
</XLens>
</XElementList.XLens>
```


Optical System


```
<?xml version="1.0" encoding="UTF-8" ?>
- <XElementList.XOpticalSystemProperties>
- <XOpticalSystemProperties id="01-0000-0B65-287A-7C|Nikon|16mm f/2.8D AF Nikkor Fisheye|412441" member="OpticalSystemProperties">
- <XElementList.XOpticalSystemProperty.XCameraGeometry member="CameraGeometry">
- <XOpticalSystemProperty.XCameraGeometry>
- <XSamplePoints.triple.float.uint member="FisheyeFunction">
<MaxDiscrepancy value="0.000757109 0.000757109 0.000757109" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="15" value="2.90725 2.90865 2.90836" />
<XSimpleElement.triple.float.uint id="231" value="2.76354 2.76456 2.76432" />
<XSimpleElement.triple.float.uint id="453" value="2.62674 2.62755 2.62723" />
<XSimpleElement.triple.float.uint id="930" value="2.35397 2.35443 2.35417" />
<XSimpleElement.triple.float.uint id="1132" value="2.24304 2.2434 2.24319" />
<XSimpleElement.triple.float.uint id="1336" value="2.13388 2.13417 2.13399" />
<XSimpleElement.triple.float.uint id="1461" value="2.06718 2.06744 2.06727" />
<XSimpleElement.triple.float.uint id="1704" value="1.93965 1.93986 1.9397" />
<XSimpleElement.triple.float.uint id="1911" value="1.83162 1.83177 1.83161" />
<XSimpleElement.triple.float.uint id="2008" value="1.78165 1.78177 1.78161" />
<XSimpleElement.triple.float.uint id="2095" value="1.73638 1.73647 1.73631" />
<XSimpleElement.triple.float.uint id="2136" value="1.71552 1.7156 1.71544" />
<XSimpleElement.triple.float.uint id="2265" value="1.64849 1.64852 1.64836" />
<XSimpleElement.triple.float.uint id="2307" value="1.62717 1.62718 1.62702" />
```

<XSimpleElement.triple.float.uint id="2450" value="1.55312 1.55307 1.55292" />
<XSimpleElement.triple.float.uint id="2605" value="1.47347 1.47335 1.4732" />
<XSimpleElement.triple.float.uint id="2791" value="1.37694 1.37673 1.37658" />
<XSimpleElement.triple.float.uint id="2889" value="1.32652 1.32625 1.32611" />
<XSimpleElement.triple.float.uint id="3017" value="1.25965 1.25932 1.25918" />
<XSimpleElement.triple.float.uint id="3156" value="1.18739 1.18698 1.18684" />
<XSimpleElement.triple.float.uint id="3394" value="1.06202 1.0615 1.06137" />
<XSimpleElement.triple.float.uint id="3814" value="0.8358 0.835048 0.834925" />
<XSimpleElement.triple.float.uint id="4079" value="0.688357 0.687488 0.687404" />
<XSimpleElement.triple.float.uint id="4300" value="0.560683 0.559726 0.559715" />
<XSimpleElement.triple.float.uint id="4447" value="0.472759 0.471736 0.471788" />
<XSimpleElement.triple.float.uint id="4487" value="0.447872 0.446819 0.446883" />
<XSimpleElement.triple.float.uint id="4648" value="0.346323 0.345046 0.345109" />
<XSimpleElement.triple.float.uint id="4786" value="0.253195 0.251679 0.251746" />
<XSimpleElement.triple.float.uint id="4931" value="0.147162 0.145388 0.145603" />
<XSimpleElement.triple.float.uint id="5084" value="0.0192368 0.0165138 0.0172067" />
<DataSource value="STARGATE" />
</XSamplePoints.triple.float.uint>
<DistanceSetting value="+INF" />
<CameraServiceRevisionLevel value="0" />
<UpsideDown value="true" />
<SensorPixelSize value="8e-06" />
<NorthPolePixel value="5100" />
<SensorInterlineDistance value="5.9e-05" />
<NorthPoleDisplacement value="1e-05" />
<SouthPoleDisplacement value="-0.003" />
<Comment value="_matow 2006-03-16" />
</XOpticalSystemProperty.XCameraGeometry>
- <XOpticalSystemProperty.XCameraGeometry>
- <XSamplePoints.triple.float.uint member="FisheyeFunction">
<MaxDiscrepancy value="0.000365576 0.000365576 0.000365576" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="5229" value="0.00349921 6.97408e-05 0" />
<XSimpleElement.triple.float.uint id="5205" value="0.0305143 0.0274775 0.027492" />
<XSimpleElement.triple.float.uint id="5141" value="0.095311 0.092795 0.0927647" />
<XSimpleElement.triple.float.uint id="5098" value="0.134939 0.13272 0.132907" />
<XSimpleElement.triple.float.uint id="5010" value="0.209113 0.207164 0.207146" />
<XSimpleElement.triple.float.uint id="4848" value="0.330262 0.328667 0.328785" />
<XSimpleElement.triple.float.uint id="4551" value="0.525067 0.523985 0.524015" />
<XSimpleElement.triple.float.uint id="4285" value="0.684472 0.683556 0.683556" />
<XSimpleElement.triple.float.uint id="3489" value="1.12431 1.12374 1.1236" />

<XSimpleElement.triple.float.uint id="1748" value="2.0323 2.03249 2.03233" />
<XSimpleElement.triple.float.uint id="1026" value="2.4177 2.41814 2.4179" />
<XSimpleElement.triple.float.uint id="433" value="2.75556 2.75634 2.75603" />
<XSimpleElement.triple.float.uint id="164" value="2.92347 2.92454 2.92433" />
<XSimpleElement.triple.float.uint id="7" value="3.03013 3.03156 3.03133" />
</XSamplePoints.triple.float.uint>
<DistanceSetting value="1.5" />
<CameraServiceRevisionLevel value="0" />
<UpsideDown value="true" />
<SensorPixelSize value="8e-06" />
<NorthPolePixel value="5103" />
<SensorInterlineDistance value="5.9e-05" />
<NorthPoleDisplacement value="0" />
<SouthPoleDisplacement value="-0.003" />
<Comment value="_matow 2006-03-16" />
</XOpticalSystemProperty.XCameraGeometry>
- <XOpticalSystemProperty.XCameraGeometry>
- <XSamplePoints.triple.float.uint member="FisheyeFunction">
<MaxDiscrepancy value="0.000348911 0.000348911 0.000348911" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="5233" value="0.00357693 0 0.000397816" />
<XSimpleElement.triple.float.uint id="5198" value="0.0425106 0.0395936 0.0395685" />
<XSimpleElement.triple.float.uint id="5107" value="0.130845 0.128617 0.12866" />
<XSimpleElement.triple.float.uint id="4998" value="0.222453 0.220567 0.220606" />
<XSimpleElement.triple.float.uint id="4851" value="0.331706 0.330101 0.330256" />
<XSimpleElement.triple.float.uint id="4567" value="0.518413 0.517329 0.517385" />
<XSimpleElement.triple.float.uint id="4286" value="0.686993 0.686074 0.686111" />
<XSimpleElement.triple.float.uint id="3491" value="1.12588 1.12527 1.12516" />
<XSimpleElement.triple.float.uint id="1749" value="2.03319 2.03337 2.03321" />
<XSimpleElement.triple.float.uint id="1031" value="2.41593 2.41636 2.41611" />
<XSimpleElement.triple.float.uint id="435" value="2.75502 2.75571 2.75537" />
<XSimpleElement.triple.float.uint id="159" value="2.92707 2.92818 2.92784" />
<XSimpleElement.triple.float.uint id="7" value="3.03037 3.03172 3.03144" />
</XSamplePoints.triple.float.uint>
<DistanceSetting value="1" />
<CameraServiceRevisionLevel value="0" />
<UpsideDown value="true" />
<SensorPixelSize value="8e-06" />
<NorthPolePixel value="5106" />
<SensorInterlineDistance value="5.9e-05" />
<NorthPoleDisplacement value="0" />
<SouthPoleDisplacement value="-0.003" />
<Comment value="_matow 2006-03-16" />
</XOpticalSystemProperty.XCameraGeometry>
- <XOpticalSystemProperty.XCameraGeometry>
- <XSamplePoints.triple.float.uint member="FisheyeFunction">
<MaxDiscrepancy value="0.000345542 0.000345542 0.000345542" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="5245" value="0.00323271 0 0.000523148" />

```
<XSimpleElement.triple.float.uint id="5119" value="0.129551 0.127401 0.12761" />
<XSimpleElement.triple.float.uint id="5047" value="0.191254 0.189356 0.189382" />
<XSimpleElement.triple.float.uint id="4863" value="0.330076 0.328604 0.328699" />
<XSimpleElement.triple.float.uint id="4572" value="0.52072 0.519682 0.519873" />
<XSimpleElement.triple.float.uint id="4297" value="0.685153 0.684213 0.684313" />
<XSimpleElement.triple.float.uint id="3499" value="1.1244 1.12363 1.12357" />
<XSimpleElement.triple.float.uint id="1753" value="2.03024 2.03049 2.03026" />
<XSimpleElement.triple.float.uint id="1058" value="2.39929 2.39971 2.3995" />
<XSimpleElement.triple.float.uint id="485" value="2.72271 2.72337 2.72301" />
<XSimpleElement.triple.float.uint id="168" value="2.91857 2.91959 2.91926" />
<XSimpleElement.triple.float.uint id="7" value="3.02771 3.02895 3.02863" />
</XSamplePoints.triple.float.uint>
<DistanceSetting value="0.5" />
<CameraServiceRevisionLevel value="0.001" />
<UpsideDown value="true" />
<SensorPixelSize value="8e-06" />
<NorthPolePixel value="5114" />
<SensorInterlineDistance value="5.9e-05" />
<NorthPoleDisplacement value="0.005" />
<SouthPoleDisplacement value="-0.003" />
<Comment value="_matow 2006-03-16" />
</XOpticalSystemProperty.XCameraGeometry>
</XElementList.XOpticalSystemProperty.XCameraGeometry>
- <XElementList.XOpticalSystemProperty.XLensCameraVignette member="Vignette">
- <XOpticalSystemProperty.XLensCameraVignette>
<MaxDiscrepancy value="263.93 263.93 263.93" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="5112" value="1.40286 1.40286 1.40286" />
<XSimpleElement.triple.float.uint id="5060" value="1.28862 1.28862 1.28862" />
<XSimpleElement.triple.float.uint id="4956" value="1.23034 1.23034 1.23034" />
<XSimpleElement.triple.float.uint id="4819" value="1.18956 1.18956 1.18956" />
<XSimpleElement.triple.float.uint id="4414" value="1.11674 1.11674 1.11674" />
<XSimpleElement.triple.float.uint id="3807" value="1.07268 1.07268 1.07268" />
<XSimpleElement.triple.float.uint id="3598" value="1.06081 1.06081 1.06081" />
<XSimpleElement.triple.float.uint id="3463" value="1.04909 1.04909 1.04909" />
<XSimpleElement.triple.float.uint id="3381" value="1.04978 1.04978 1.04978" />
<XSimpleElement.triple.float.uint id="3263" value="1.03707 1.03707 1.03707" />
<XSimpleElement.triple.float.uint id="3077" value="1.03891 1.03891 1.03891" />
<XSimpleElement.triple.float.uint id="2793" value="1.03137 1.03137 1.03137" />
<XSimpleElement.triple.float.uint id="2690" value="1.03249 1.03249 1.03249" />
<XSimpleElement.triple.float.uint id="2343" value="1.03378 1.03378 1.03378" />
<XSimpleElement.triple.float.uint id="1917" value="1.04506 1.04506 1.04506" />
<XSimpleElement.triple.float.uint id="1760" value="1.04102 1.04102 1.04102" />
<XSimpleElement.triple.float.uint id="1667" value="1.05201 1.05201 1.05201" />
<XSimpleElement.triple.float.uint id="991" value="1.09362 1.09362 1.09362" />
<XSimpleElement.triple.float.uint id="471" value="1.15379 1.15379 1.15379" />
<XSimpleElement.triple.float.uint id="57" value="1.22553 1.22553 1.22553" />
<ApertureSetting value="11" />
<CameraServiceRevisionLevel value="0" />
</XOpticalSystemProperty.XLensCameraVignette>
```

```
- <XOpticalSystemProperty.XLensCameraVignette>
  <MaxDiscrepancy value="190.464 190.464 190.464" />
  <InterpolationMethod value="CUBIC_SPLINE" />
  <XSimpleElement.triple.float.uint id="5093" value="1.39683 1.39683 1.39683" />
  <XSimpleElement.triple.float.uint id="5044" value="1.2826 1.2826 1.2826" />
  <XSimpleElement.triple.float.uint id="4952" value="1.23041 1.23041 1.23041" />
  <XSimpleElement.triple.float.uint id="4810" value="1.18691 1.18691 1.18691" />
  <XSimpleElement.triple.float.uint id="4446" value="1.11893 1.11893 1.11893" />
  <XSimpleElement.triple.float.uint id="3800" value="1.06994 1.06994 1.06994" />
  <XSimpleElement.triple.float.uint id="3636" value="1.05885 1.05885 1.05885" />
  <XSimpleElement.triple.float.uint id="3469" value="1.04605 1.04605 1.04605" />
  <XSimpleElement.triple.float.uint id="3385" value="1.04846 1.04846 1.04846" />
  <XSimpleElement.triple.float.uint id="3257" value="1.03559 1.03559 1.03559" />
  <XSimpleElement.triple.float.uint id="3179" value="1.03857 1.03857 1.03857" />
  <XSimpleElement.triple.float.uint id="3053" value="1.03685 1.03685 1.03685" />
  <XSimpleElement.triple.float.uint id="2769" value="1.03161 1.03161 1.03161" />
  <XSimpleElement.triple.float.uint id="2333" value="1.03486 1.03486 1.03486" />
  <XSimpleElement.triple.float.uint id="2104" value="1.03929 1.03929 1.03929" />
  <XSimpleElement.triple.float.uint id="1920" value="1.03738 1.03738 1.03738" />
  <XSimpleElement.triple.float.uint id="1763" value="1.03449 1.03449 1.03449" />
  <XSimpleElement.triple.float.uint id="1671" value="1.04473 1.04473 1.04473" />
  <XSimpleElement.triple.float.uint id="57" value="1.21442 1.21442 1.21442" />
  <ApertureSetting value="8" />
  <CameraServiceRevisionLevel value="0" />
    </XOpticalSystemProperty.XLensCameraVignette>
- <XOpticalSystemProperty.XLensCameraVignette>
  <MaxDiscrepancy value="333.145 333.145 333.145" />
  <InterpolationMethod value="CUBIC_SPLINE" />
  <XSimpleElement.triple.float.uint id="5174" value="2.64534 2.64534 2.64534" />
  <XSimpleElement.triple.float.uint id="5073" value="1.63197 1.63197 1.63197" />
  <XSimpleElement.triple.float.uint id="4977" value="1.41452 1.41452 1.41452" />
  <XSimpleElement.triple.float.uint id="4879" value="1.30457 1.30457 1.30457" />
  <XSimpleElement.triple.float.uint id="4442" value="1.12118 1.12118 1.12118" />
  <XSimpleElement.triple.float.uint id="4275" value="1.09719 1.09719 1.09719" />
  <XSimpleElement.triple.float.uint id="4071" value="1.08311 1.08311 1.08311" />
  <XSimpleElement.triple.float.uint id="3640" value="1.05447 1.05447 1.05447" />
  <XSimpleElement.triple.float.uint id="3467" value="1.04053 1.04053 1.04053" />
  <XSimpleElement.triple.float.uint id="3377" value="1.04419 1.04419 1.04419" />
  <XSimpleElement.triple.float.uint id="3262" value="1.0343 1.0343 1.0343" />
  <XSimpleElement.triple.float.uint id="3058" value="1.03501 1.03501 1.03501" />
  <XSimpleElement.triple.float.uint id="2714" value="1.03094 1.03094 1.03094" />
  <XSimpleElement.triple.float.uint id="2339" value="1.03722 1.03722 1.03722" />
  <XSimpleElement.triple.float.uint id="1909" value="1.04055 1.04055 1.04055" />
  <XSimpleElement.triple.float.uint id="1762" value="1.0367 1.0367 1.0367" />
  <XSimpleElement.triple.float.uint id="1672" value="1.04768 1.04768 1.04768" />
  <XSimpleElement.triple.float.uint id="728" value="1.11731 1.11731 1.11731" />
  <XSimpleElement.triple.float.uint id="453" value="1.15575 1.15575 1.15575" />
  <XSimpleElement.triple.float.uint id="57" value="1.25178 1.25178 1.25178" />
  <ApertureSetting value="5.6" />
  <CameraServiceRevisionLevel value="0" />
    </XOpticalSystemProperty.XLensCameraVignette>
- <XOpticalSystemProperty.XLensCameraVignette>
  <MaxDiscrepancy value="458.193 458.193 458.193" />
  <InterpolationMethod value="CUBIC_SPLINE" />
  <XSimpleElement.triple.float.uint id="5175" value="5.03114 5.03114 5.03114" />
```

```
<XSimpleElement.triple.float.uint id="4970" value="2.35836 2.35836 2.35836" />
<XSimpleElement.triple.float.uint id="4875" value="2.06912 2.06912 2.06912" />
<XSimpleElement.triple.float.uint id="4758" value="1.81019 1.81019 1.81019" />
<XSimpleElement.triple.float.uint id="4428" value="1.4228 1.4228 1.4228" />
<XSimpleElement.triple.float.uint id="4048" value="1.23599 1.23599 1.23599" />
<XSimpleElement.triple.float.uint id="3874" value="1.18433 1.18433 1.18433" />
<XSimpleElement.triple.float.uint id="3736" value="1.18076 1.18076 1.18076" />
<XSimpleElement.triple.float.uint id="3436" value="1.12056 1.12056 1.12056" />
<XSimpleElement.triple.float.uint id="3310" value="1.12006 1.12006 1.12006" />
<XSimpleElement.triple.float.uint id="3191" value="1.1143 1.1143 1.1143" />
<XSimpleElement.triple.float.uint id="2917" value="1.08348 1.08348 1.08348" />
<XSimpleElement.triple.float.uint id="2727" value="1.07792 1.07792 1.07792" />
<XSimpleElement.triple.float.uint id="2133" value="1.05399 1.05399 1.05399" />
<XSimpleElement.triple.float.uint id="1883" value="1.04887 1.04887 1.04887" />
<XSimpleElement.triple.float.uint id="1747" value="1.04228 1.04228 1.04228" />
<XSimpleElement.triple.float.uint id="1654" value="1.04949 1.04949 1.04949" />
<XSimpleElement.triple.float.uint id="1108" value="1.07846 1.07846 1.07846" />
<XSimpleElement.triple.float.uint id="741" value="1.13363 1.13363 1.13363" />
<XSimpleElement.triple.float.uint id="247" value="1.39449 1.39449 1.39449" />
<XSimpleElement.triple.float.uint id="57" value="1.58129 1.58129 1.58129" />
<ApertureSetting value="4" />
<CameraServiceRevisionLevel value="0" />
  </XOpticalSystemProperty.XLensCameraVignette>
- <XOpticalSystemProperty.XLensCameraVignette>
<MaxDiscrepancy value="302.431 302.431 302.431" />
<InterpolationMethod value="CUBIC_SPLINE" />
<XSimpleElement.triple.float.uint id="5158" value="7.01699 7.01699 7.01699" />
<XSimpleElement.triple.float.uint id="4978" value="3.83005 3.83005 3.83005" />
<XSimpleElement.triple.float.uint id="4878" value="3.22607 3.22607 3.22607" />
<XSimpleElement.triple.float.uint id="4742" value="2.72048 2.72048 2.72048" />
<XSimpleElement.triple.float.uint id="4430" value="2.05694 2.05694 2.05694" />
<XSimpleElement.triple.float.uint id="3770" value="1.49567 1.49567 1.49567" />
<XSimpleElement.triple.float.uint id="3425" value="1.3135 1.3135 1.3135" />
<XSimpleElement.triple.float.uint id="3328" value="1.28388 1.28388 1.28388" />
<XSimpleElement.triple.float.uint id="2801" value="1.12224 1.12224 1.12224" />
<XSimpleElement.triple.float.uint id="2634" value="1.08369 1.08369 1.08369" />
<XSimpleElement.triple.float.uint id="2532" value="1.0571 1.0571 1.0571" />
<XSimpleElement.triple.float.uint id="2448" value="1.05533 1.05533 1.05533" />
<XSimpleElement.triple.float.uint id="2273" value="1.03854 1.03854 1.03854" />
<XSimpleElement.triple.float.uint id="2189" value="1.04667 1.04667 1.04667" />
<XSimpleElement.triple.float.uint id="2027" value="1.06041 1.06041 1.06041" />
<XSimpleElement.triple.float.uint id="1741" value="1.09428 1.09428 1.09428" />
<XSimpleElement.triple.float.uint id="1613" value="1.13362 1.13362 1.13362" />
<XSimpleElement.triple.float.uint id="1267" value="1.23752 1.23752 1.23752" />
<XSimpleElement.triple.float.uint id="964" value="1.382 1.382 1.382" />
<XSimpleElement.triple.float.uint id="424" value="1.80088 1.80088 1.80088" />
<XSimpleElement.triple.float.uint id="57" value="2.37963 2.37963 2.37963" />
<ApertureSetting value="2.8" />
<CameraServiceRevisionLevel value="0" />
  </XOpticalSystemProperty.XLensCameraVignette>
  </XElementList.XOpticalSystemProperty.XLensCameraVignette>
  </XOpticalSystemProperties>
  </XElementList.XOpticalSystemProperties>
```

Appendix B FRAMES FROM NAVIGATION MOVIE

A movie is made to demonstrate the navigation within a virtual environment mapped with one spherical image. The virtual camera captures frames through a small viewing window when being placed at different positions and orientations. The navigation route (in a plan view) is shown in Figure B-1 with arrows indicating the camera's viewing direction. The capture starts at the centre of the sphere with the camera looking straight ahead. It then moves sideways to the right, backwards in a diagonal way, towards the front again diagonally with its head turning to the left while moving, and finally turns 360 degrees at its finishing point. The frames of the movie are shown at a rate of 1/5 FPS (frames per second) in Figure B-2. Note the camera motion also contains looking up and down, which is not shown on the 2D route map but obvious on the frames.

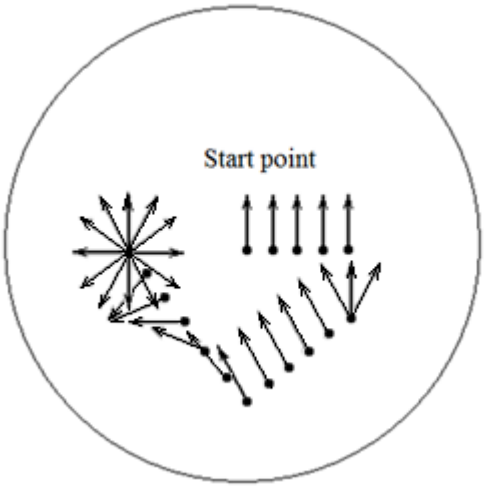


Figure B-1 Virtual camera route (plan view)

Figure B-2 Frames from navigation movie

Appendix C PUBLISHED PAPERS

Distortion Correction for Immersive Navigation in Spherical Image Environment, International Conference on CyberWorlds 09, Sep 2009, pp. 96-101

Calibration of Rotating Line Spherical Camera based on Checkerboard Pattern on Multiple Planes and its Accuracy Assessment, British Machine Vision Conference (BMVC10) Postgraduate Workshop, Sep 2010