A System for Real-time Panorama Generation and Display in Tele-immersive Applications

Wai-Kwan Tang Tien-Tsin Wong Pheng-Ann Heng

wktang@cse.cuhk.edu.hk ttwong@acm.org pheng@cse.cuhk.edu.hk

Department of Computer Science & Engineering,

The Chinese University of Hong Kong

Wai-Kwan Tang

Dept. of Computer Science & Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong.

Email: wktang@cse.cuhk.edu.hk

Tien-Tsin Wong (* Contact author)

Dept. of Computer Science & Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong.

Tel: +852-26098433 Fax: +852-26035024 Email: ttwong@acm.org

Pheng-Ann Heng

Dept. of Computer Science & Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong.

Tel: +852-26098424 Fax: +852-26035024 Email: pheng@cse.cuhk.edu.hk

Abstract

Wide field-of-view is necessary for many industrial applications, such as air traffic control, large vehicle driving and navigation. Unfortunately, the supporting structure/frame in most systems usually blocks part of the view, results in "blind spot" and raises the risk to the pilot. In this paper, we introduce a video-based tele-immersive system, called the *Immersive Cockpit*. It captures live videos from the working site and recreates an immersive environment at the remote site where the pilot situates. It immerses the pilot at the remote site with a panoramic view of the environment, hence improves interactivity and safety. The design goals of our system are *real-time*, *live*, *low-cost* and *scalable*. We stitch multiple video streams captured from ordinary CCD cameras to generate a panoramic video. To avoid being blocked by the supporting frame, we allow a flexible placement of cameras. This approach trades the accuracy of the generated panoramic image for a larger field-of-view. To reduce the computation, parameters for stitching are determined once during the system initialization. The panoramic video is presented on an immersive display which covers the field-of-view of the viewer. We discuss how to correctly present the panoramic video on this non-planar immersive display screen by sweet spot relocation. We also present the result and the performance evaluation of the system.

Keywords

immersive display, augmented reality, image mosaicing, video streaming, real-time panorama

I. Introduction

Immersion increases the perceived level of reality and helps interactivity in many multimedia applications, ranging from video conferencing to entertainment industry. A wide field-of-view (FOV) is needed to immerse the user into the virtual/reconstructed environment. For high level of fidelity, real-time provision of immersive video is necessary. Real-time video streaming is possible, but it is limited to a restricted field-of-view without the use of a specialized camera. It is still challenging to develop a tele-immersive application that acquires and presents panoramic video in a *real-time*, *live*, *low-cost* and *scalable* fashion.

The feature of wide FOV is necessary in many applications. For instance, an air traffic control tower with almost 360° FOV (Figure 1(b)) is ideal for traffic coordinators inside to respond quickly during an emergency. Hence the wide FOV improves safety. However, the FOV is often limited by the supporting frame of the building (Figure 1(a)). Drivers of large-scale vehicles (e.g. excavator in Figure 2(a)) also need a complete picture of the working environment in order to work efficiently and safely. Again, the FOV is reduced by the supporting frame (Figure 2(b)). In some cases, the working site is hazardous to the driver. It would be ideal if the driver sits in a safe place and controls remotely as if he/she was located in the actual site without being obscured by the supporting frame (Figures 2(a)-(d)).

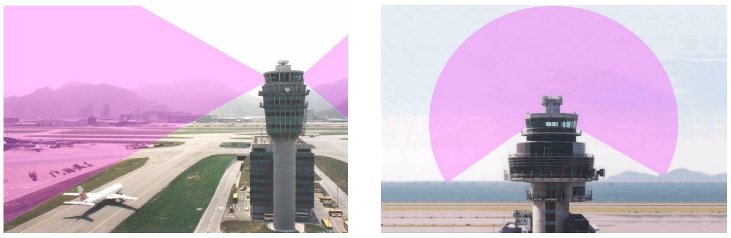


Fig. 1. Air traffic control tower. (a) FOV is limited by the supporting frame. An ideal case is to have a wide FOV as illustrated by the shaded region in (b).

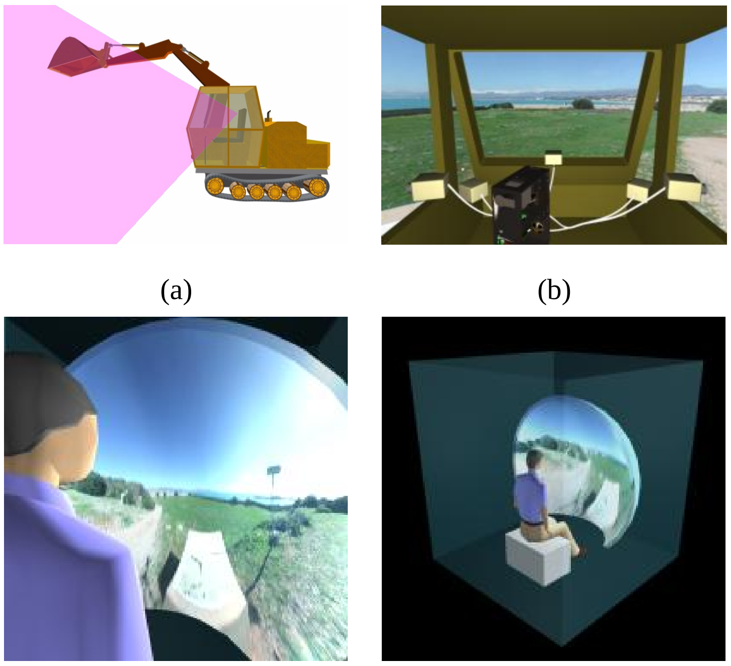


Fig. 2. (a) Potential industrial application of the Immersive Cockpit. (b) Flexible camera placement avoids occlusion of supporting structure and offers wide FOV (c) & (d).

Our goal is to develop a low-cost, workable, and scalable tele-immersion system in order to demonstrate a system framework which allows the generation of panoramic video stream in real-time. The video-based immersive system we developed is called the *Immersive Cockpit*. It generates panoramic video by applying the image mosaicing techniques in still imagery. We use groups of low-cost CCD video cameras (without using expensive specialized cameras), each pointing at different direction, to capture the scene. Their views are slightly overlapped in order to facilitate the stitching of these videos to form the panoramic video. Live video streams from the cameras are fed into our mosaicing engine to create live panoramic video. Unlike the previous systems [23], [7], [21], [40] which restrict the placement of cameras in a small region in order to minimize the parallax between camera centers, we allow a flexible placement of cameras to avoid the occlusion due to the supporting frame (see the camera placement in Figure 2(b)). This approach may introduce error into the composed panoramic video, but enlarge the FOV. In other words, we trade the accuracy of panorama for larger FOV. The panoramic video is created in real-time and projected onto a large hemispherical display. Due to the large FOV coverage, users are immersed in the reconstructed environment just as situated in the actual environment. The goal of developing the Immersive Cockpit is to allow user to be situated anywhere to work remotely as if he/she was located in the actual site.

The development of the Immersive Cockpit faces challenges in various fields, ranging from computer vision, image processing, video compression, networking, high performance computing and computer graphics. The system is built partially based on off-the-shelf components. Our contribution is mainly the system integration of different technologies stemmed from different research fields. The remainder of this paper presents how we handle different challenges. It is organized as follows. Section II describes some of the related works. Section III gives the architectural overview of our Immersive Cockpit. Section IV describes how our system acquires the video streams of the scene. In Section V, we explain how the collection of video streams are stitched to form a panoramic FOV video. Section VI describes how this panoramic video is correctly displayed on an immersive display. The results are followed in Section VII. Finally, conclusions are drawn in Section VIII.

II. Related Work

Creating virtual environment is not new. Most of the existing systems fall into one of the three categories: either geometry-based, image-based and hybrid systems. Geometry-based virtual reality system uses geometrical objects to represent the scenes. Flight simulator for training air pilot is a typical example. Since real scene can be arbitrarily complex, modeling real scene may result in huge amount of data which cannot be rendered in real-time. Besides, the modeling cost is prohibitively high. Instead, scenes can be represented using the image-based approach [18], [29]. The reconstructed environment is realistic and the modeling process can be avoided. Hybrid systems make use of both image and geometric model. Augmented reality systems belong to this category.

The proposed Immersive Cockpit belongs to category of image-based system. Three main challenges arise in this area: image acquisition, image registration and display.

A. Image Acquisition

In acquiring a scene, ordinary video cameras can only capture a limited FOV [37]. By using a cluster of cameras [22], [39], [13], [12], wide FOV image can be obtained. These cameras introduce further problems such as the requirement of a common optical center. Instead of using ordinary ones, cameras with fish-eye lens can be used [46]. Furthermore, specially designed cameras are developed. Nayar [23] developed an omni-directional camera which captures at video rate, a hemispherical field of view as seen from a single point. Majumder *et al.* [20] developed a camera cluster with a single camera center. Baldwin *et al.* [3] used a single camera to capture the image reflected from a conic mirror.

Most previous acquisition systems require a cluster of cameras being mounted on a framework and/or special devices (such as fish-eye lens or conic mirror) being installed. In our system, multiple ordinary cameras are adopted and no special device is needed. The placement of cameras is flexible and not mounted on any specially designed framework in order to avoid the occlusion due to the supporting frame (Figure 2(b)). However the relative position and orientation among the cameras should be fixed during the capture.

B. Image Registration

Since we do not fix the cameras on any specially designed framework, we rely on the image registration techniques to generate the panorama. Most existing techniques fall into three main categories. The phase correlation approach bases on the image properties in frequency domain. Firstly proposed by Kuglin and Hines [19], the algorithm uses 2D Fourier transform and computes the displacement between two images from the phase of their cross power spectrum. This algorithm is scene-independent and accurate to within one pixel for image differing by a pure 2D translation. Instead of finding the displacement, other methods [33], [8] determine the rotation and scaling for image registration. These algorithms, however, are quite sensitive to noise and also require the overlap extent to occupy a significant portion of the images (e.g. at least 50%).

The second approach is transformation recovery [7], [21], [40], [42]. It is a more intuitive approach. The method makes use of the fact that two spatially neighboring images are related to each other by a homography transformation. By recovering the homography transformation, images can be stitched together. To recover the homography transformation, feature based methods [7], [21] rely on the detection of image features, with which the camera motion can be computed. In the absence of distinctive features, automatic feature extraction methods usually fail. Another way to recover the homography transformation [15] is to iteratively adjust the camera motion parameters. Szeliski [40] and Gonzalez *et al.* [14] proposed the use of Levenberg-Marquardt minimization to find the transformation matrix. Szeliski and Shum [38], [42] further formulated the 2D mosaicing problem by assuming the camera rotates around a fixed point. In our system, we applied their method to stitch video frames from different cameras.

The third approach is the video sweeping [28], [14], [34]. It creates panoramic mosaics by combining the scene information from the frames in a video sequence. The motion of the camera is recovered and thus the information of the video frames can be properly combined together. Peleg and Herman [28] used a manifold projection which simulates the sweeping of the scene with a plane using an one dimensional sensor array. Their method works fine for a single video strip only but usually fails for the case of multiple video strips. Sawhney *et al.* [34] improved the method by using topology inference and local-to-global alignment.

C. Immersive Display

For immersive presentation of data, it comes mainly in two forms. The first is the head mount display (HMD). The second is the very large screen [31], [35], [30], typically projected onto a wall and referred to as a spatially immersive display [43], [6]. HMDs have a number of deficiencies such as separating the users from their familiar environment, low resolution, bulky, and heavy. Since Cruz-Neira *et al.* developed CAVE [6], the focus moves to the construction of display system with multiple screens and projectors. The problem becomes finding the relationship between screens and projectors so that the whole set of devices can be considered as a single logical display unit [6], [4], [1], [44], [9], [25]. With multiple randomly placed projectors, Raskar *et al.* [30] tackled the problem of aligning the projected images so that they appear as a seamless one. Screens are commonly planar in shape. Raskar *et al.* [32] projected the image onto non-planar screens. This approach allows the use of virtually any wall or structure of a room to act as the display screen. In our system, we adopt a hemispherical display screen which covers about 180° FOV of the viewer.

III. System Overview

The Immersive Cockpit is an integration of a number of modules. Figure 3 illustrates the interaction between different modules. The video capture module captures the real-world scenery as digital video. The video streaming module allows the captured video data to be transmitted across the network. The stitching and rendering module combines multiple video frames at one time instance to form a single panoramic frame. Finally, the display module presents the panoramic video to the viewer on a hemispherical display. The following sections describe each module in details.

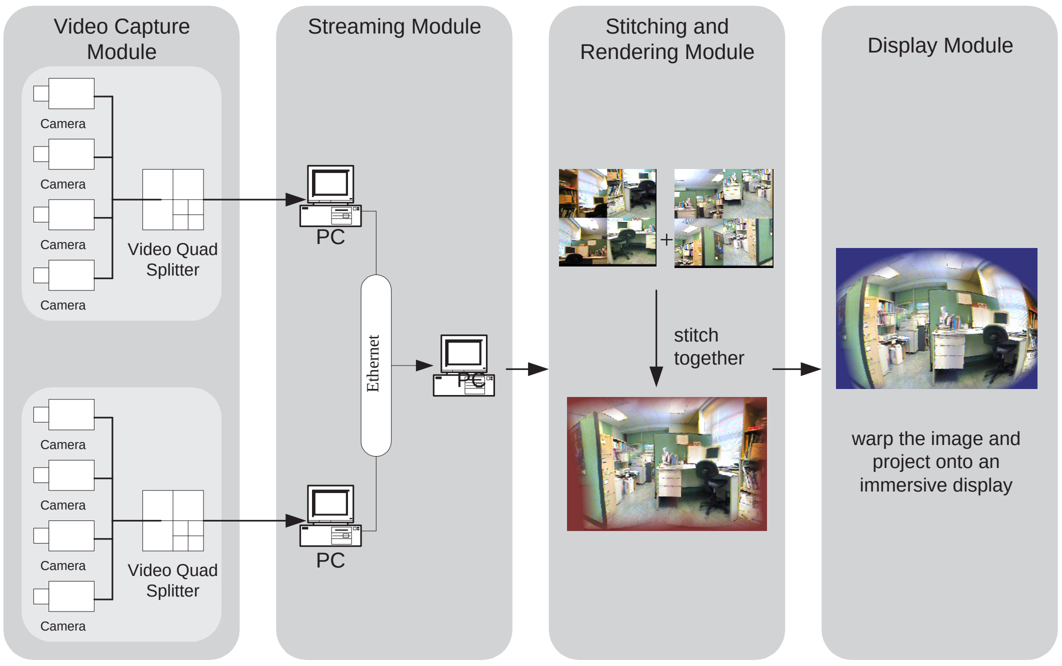


Fig. 3. System configuration

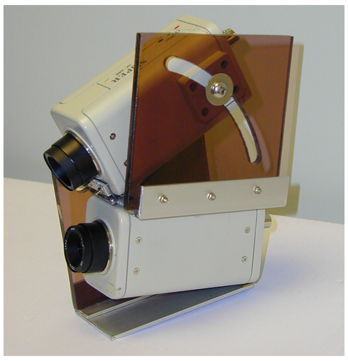
IV. Video Acquisition

The video capture module captures the scene with as large FOV as possible. Capturing devices such as digital camcorder or CCD camera should be used. Instead of using sophisticated cameras as in [39], [20], we use off-the-shelf components in developing our system. This has numerous advantages including higher flexibility and greater system scalability.

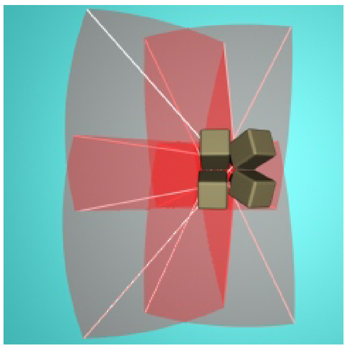
A. Camera Placement

Unless using special equipment [3], [46], it is not possible to capture the whole 360° scene (spherical, not just cylindrical) with a single camera. To capture video streams of 360° FOV, our system uses multiple ordinary CCD cameras. The video cameras are placed so as to enlarge the FOV and avoid the occlusion due to the supporting frame. One requirement of our system is scalability, *i.e.* when more cameras are used, we get image with larger FOV. Since our display unit is of hemispherical shape, capturing 180° is sufficient for our need. Our system can be easily extended to capture 360° FOV by adding more cameras.

Theoretically, these cameras should be placed close together so that their centers of projection (COPs) coincide. This arrangement of cameras facilitates the mosaicing process. However, in practice, this coincidence requirement can be relaxed depending on the actual applications. Outdoor scene capture



(a)



(b)

Fig. 4. (a) Basic capture unit: 2 CCD cameras connected with an angle-adjustable frame. (b) The views of the cameras should overlap in certain degrees.

(with no nearby object) relaxes this coincidence constraint. Even in the indoor scene capture, relaxation is possible if the user accepts small error in the final panoramic images. Moreover, we want to avoid the occlusion due to the supporting frame, a setup with all camera centers being coincident cannot avoid occlusion due to the nearby obstacle. Instead, our system requires all camera centers should be *loosely* coincident, *i.e.* all cameras should be pointing roughly from the same point. Interested readers are referred to Appendix for the details of this parallax problem. The Appendix also serves as a guideline for camera placement. Another requirement is that there should be about 20% of overlap between the views of two adjacent cameras in order to facilitate the image mosaicing (Figure 4(b)).

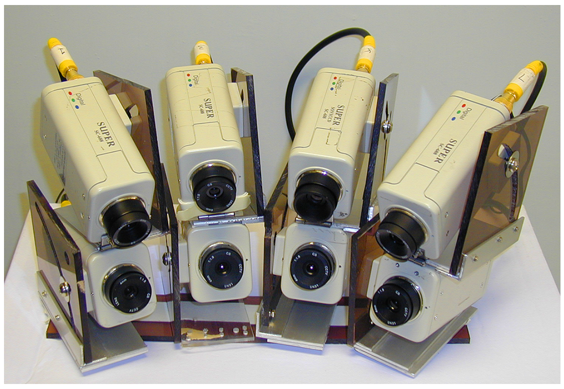


Fig. 5. Camera placement for indoor scene capture

The basic capture unit is a group of two cameras mounted on an angle-adjustable framework as shown in Figure 4(a). The upper camera points downward while the lower one points upward in order to reduce the distance between the COPs of two cameras (Figure 4(a)). Note that we mount two cameras on a framework, not because our system requires so, but solely due to convenience.

For indoor scene where objects are near, the distance between COPs should be as small as possible. Figure 5 shows the camera placement for indoor scene capture. For outdoor scene, cameras can be placed farther apart without causing serious problems in the stitching process. Figure 14 shows the floor-plan of our camera placement for outdoor scene capture (more details in Section VII). With eight cameras, our system covers approximately a view with horizontal FOV of 150° and vertical FOV of 110°. To enlarge the coverage, more cameras can be added.

B. *Video Capturing*

During capture, the output of 4 cameras (2 basic capture units) are connected to a video merger which combines 4 videos into 1 video (Figure 6). The combined video is then fed to the video grabber installed in a PC. This combination degrades the video quality. The reason of using a video merger is because normally a PC can handle only one input video stream without relying on some expensive specialized video capture boards. For an eight camera cluster, this would require a farm of eight PCs. Using a video merger can greatly reduce to using two PCs only, though sacrificing the video quality. With the combined video stream, video compression and streaming can be carried out in a more cost-effective manner. Figure 6 shows an example. Four 320×240 video streams are combined into a single 640×480 video stream.

C. *Video Streaming*

The captured video is compressed in real-time and streamed across the network to the target display module. Since we transmit multiple video streams to the stitching and rendering module, it is important

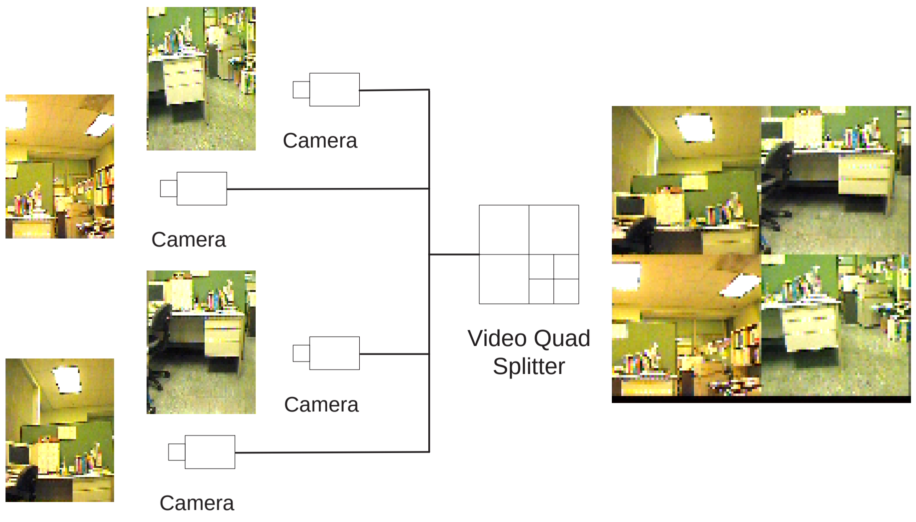


Fig. 6. The video merger combines 4 video streams

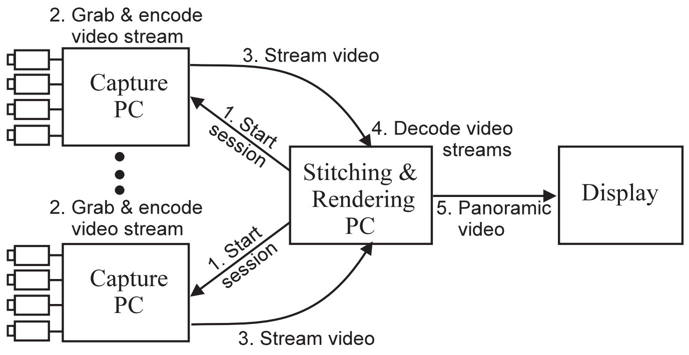


Fig. 7. Steps of capture

to synchronize the input video streams. Otherwise, the stitching and rendering module may wrongly use the frames of different time instance to compose the panoramic frame and introduce discontinuity into the resultant panoramic frame. We synchronize the clock of each video capture PC with the network time protocol (NTP). NTP is a protocol which can accurately distribute network time to the accuracy of millisecond. Video usually has a frame rate of 25 to 30 fps, which is in the order of ten of milliseconds. Hence NTP offers sufficient accuracy for our need. During capture, we add a time-stamp to each frame so that the stitching and rendering module can determine the right set of frames at any particular time instance.

Our delivery model for the video streams follows the simple peer-to-peer design. This consists of the following steps (Figure 7):

1. Before the streaming starts, messages are sent to the capturing PCs to notify the beginning of the capture session.
2. Video are grabbed and encoded by the software video encoder.
3. The encoded data is time-stamped and sent across the network to the receiver side.
4. Upon receiving the encoded frames, they are decompressed. Time-stamps are used for recovering the frame sequence and synchronizing frames from different video streams.
5. The synchronized video frames are passed to the stitching and rendering module to generate the panoramic video.

Different video streaming models [27], [11], [24], [36], [2] suit for different applications. Although we currently use the peer-to-peer model, the system can be easily extended to the broadcast basis which allows multiple users to experience the same virtual environment simultaneously.

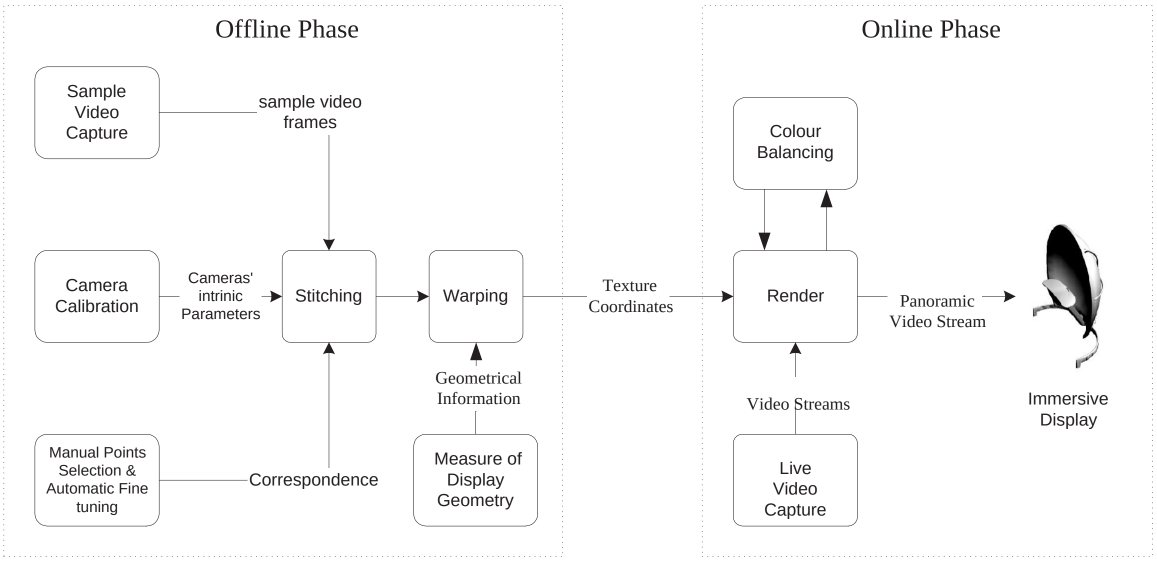


Fig. 8. Processing Pipeline

V. Panorama Construction

Image mosaicing is the task of combining a collection of images with small FOV to obtain an image of larger FOV. In our system, we combine a collection of continuous video streams (fixed viewing direction) with small FOV to generate a continuous panoramic video stream of large FOV. We called this process *continuous video mosaicing*. Note that we do not use the terminology *video mosaic* as it has been used by other researchers [14], [41] to refer the construction of a static panoramic image from a video stream.

Instead of running the image mosaic algorithm once per video frame, we apply the mosaicing algorithm once and re-use the mosaic parameters for all the subsequent frames. In this way, the image mosaicing can be done in the offline phase. Figure 8 shows the processing pipeline. The processing pipeline can be divided into offline and online phases. In the offline phase, all necessary pre-computations are done. This includes the computation of lens distortion parameters, intrinsic parameters, extrinsic parameters, and warping parameters due to sweet spot relocation (explained shortly). All these parameters are then implicitly stored as texture coordinates attached to the vertices of a triangular mesh representing the panorama. In the online phase, minimal work is carried out to create and display the panoramic video stream. The subsequent video frames from multiple CCD cameras are simply treated as textures and mapped onto the triangular mesh using standard graphics hardware to synthesize the panoramic video frames. Hence, no computation of those parameters is necessary.

The offline phase includes the following steps:

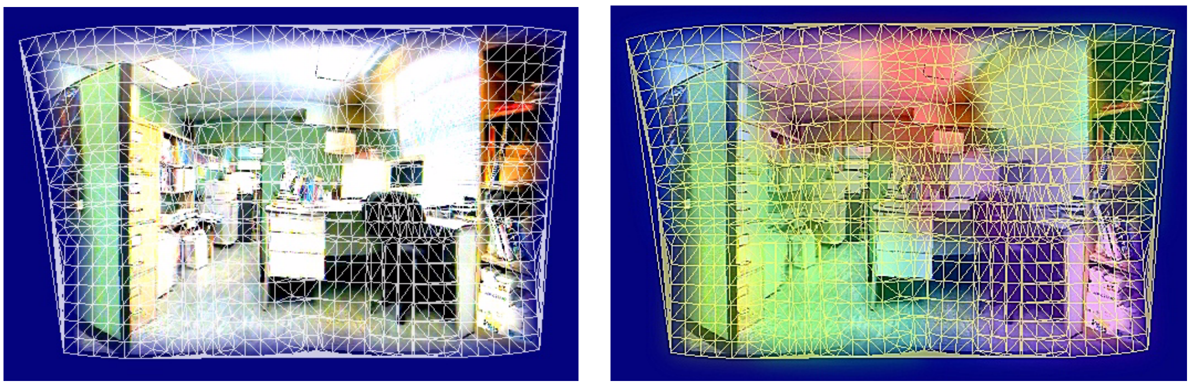
1. *Calibration of Cameras* - Each camera is calibrated separately for computing its intrinsic parameters

and lens distortion parameters.

2. *Finding Correspondences* - Since the overlapping region among different views is limited (we try to minimize the number of cameras used), automatic correspondence determination methods usually cannot reliably function. Instead, correspondences are manually identified. Automatic refinement is done to obtain a better correspondence matches.

3. *Mapping Video Frames onto Panorama* - Using the obtained parameters and the geometry of display screen, we compute a triangulated panorama and the texture coordinates of its vertices. These texture coordinates actually tell where on the screen the pixels from the video frames should be drawn. Once the texture coordinates are computed, they are repeatedly used for stitching subsequent video frames.

During video capture, very little amount of computing resource is available. The major task in the online phase is the *real-time* composition of panoramic frames. In order to speed up the panoramic composition process, we take advantage of the current commodity graphics accelerators which render millions of texture-mapped triangles in real-time. Upon receiving all video frames, they are used as the texture maps. Using the precomputed triangulated panorama and its texture coordinates, the video frames are mapped onto the triangular mesh and blended together to form a panoramic frame in a real-time fashion. Figure 9 shows the triangulated panorama. Each camera contributes part of the panorama. To visualize their contributions, we color-coded the contributions in Figure 9(b).



(a) (b)

Fig. 9. Panoramic frame is composed by texture mapping and blending. (a) The composed panorama and (b) contributions from different cameras visualized by color-coding.

A. Assumptions

When stitching the video frames, we make the following assumptions:

1. The COPs of all cameras are at the same location.

2. The cameras' relative positions and orientations are fixed.

3. There are sufficient regions of overlapping between adjacent camera views for registration and alignment.

4. The input video streams are synchronized.

Normal cameras cannot be placed at the same location as they physically occupy certain amount of space. Thus, the first assumption is normally invalid unless using specially designed cameras and framework [39], [13]. The violation of this single-COP assumption makes the image mosaicing difficult. However, within a tolerable error, the standard image mosaicing algorithm can still be applied without noticeable artifact. This tolerance depends on the distance from camera to the closest object in the scene (see Appendix). Hence outdoor scene has a higher tolerance than that of the indoor scene.

In our application, the relative positions and orientations of cameras do not change. To avoid any potential turbulence such as perturbation due to wind, cameras can be mounted firmly on a stable base. Moreover, as cameras are manually positioned in the setup phase, it is always possible to arrange in such a way that there are sufficient degree of image overlapping between adjacent cameras. Our flexible camera placement allows cameras be placed to avoid obstacle from blocking the view. Thus, the second and the third assumptions are guaranteed. For the synchronization of video streams, as mentioned before, we use the Network Time Protocol and timestamping.

B. Camera Calibration

Several methods for camera calibration have been proposed in previous literatures. Readers are referred to [10] [45] for detail derivation. The model we employed is as follows: an ideal camera can be thought as a perspective projection which maps a 3D point $P_w = (x_w, y_w, z_w)$ in camera space to a 2D point $P_s = (x_s, y_s)$ on the image. This ideal pin-hole camera model can be described by Equation 1,

$$\begin{pmatrix} x_s \\ y_s \\ 1 \end{pmatrix} = \begin{pmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_n \\ y_n \\ 1 \end{pmatrix} \tag{1}$$

where $P_n = (x_n, y_n) = (x_w/z_w, y_w/z_w)$.

Real cameras usually deviate from this ideal model. Camera distortion has to be taken into account when dealing with the real scene capture. This distortion can be decomposed into three major components: shift of the optical center, radial distortion and de-centering distortion. Although it may not be apparent when viewing a single image, the distortion may cause noticeable mismatch when stitching multiple images.

The internal camera model we employ is similar to that used by Heikkila and Silven [16]. With the introduction of distortion, we get the distorted point $\tilde{P}_n = (\tilde{x}_n, \tilde{y}_n)$ which is defined as follow:

$$\begin{aligned} \begin{pmatrix} \tilde{x}_n \\ \tilde{y}_n \end{pmatrix} &= (1 + k_1 r^2 + k_2 r^4 + k_5 r^6) \begin{pmatrix} x_n \\ y_n \end{pmatrix} \\ &+ \begin{pmatrix} 2k_3 x_n y_n + k_4(r^2 + 2x_n^2) \\ k_3(r^2 + 2y_n^2) + 2k_4 x_n y_n \end{pmatrix} \end{aligned} \quad (2)$$

where $r^2 = x_n^2 + y_n^2$ and k_i are the distortion coefficients. The distorted point $\tilde{P}_n$ in image space can be computed similarly using Equation 1 with P_n replaced by $\tilde{P}_n$. Finding the undistorted point P_s from a given distorted point $\tilde{P}_s$ (distorted point from the camera) requires the inverse mapping of distortion. We solve this by an iterative numerical method.

To calibrate the cameras, we employ the camera calibration toolbox of Open Source Computer Vision Library (OpenCV) (http://sourceforge.net/projects/opencvlibrary/). We place a checkerboard in front of each camera and capture several sample images. The recognized corners in the checkerboard images and the known geometry of the checkerboard allow us to feed the parameters required in Equations 1 and 2.

C. Homography

For any two cameras with a common optical center, their images can be related by a 2D projective transformation, or homography. Several approaches have been proposed, interested readers are referred to [40], [38] for in-depth derivation. For completeness, we summarize the key idea in this section. This homography maps pixel (x, y) of image I to pixel (x', y') of image I'. This 3×3 homography matrix $\mathbf{H}$ has nine unknowns but with eight degree of freedoms.

$$\mathbf{H} = \begin{pmatrix} m_0 & m_1 & m_2 \\ m_3 & m_4 & m_5 \\ m_6 & m_7 & m_8 \end{pmatrix} \quad (3)$$

By applying the constraint $m_8 = 1$, the homography can be rewritten to the following form:

$$x' = \frac{m_0 x + m_1 y + m_2}{m_6 x + m_7 y + 1} \qquad y' = \frac{m_3 x + m_4 y + m_5}{m_6 x + m_7 y + 1}$$

These eight unknowns can be solved without using any 3D information from the scene. With a minimum of four pairs of image correspondences, we can determine all unknowns [40], [38]. In practice, more correspondence pairs are identified to minimize the error. With n correspondence points, we form

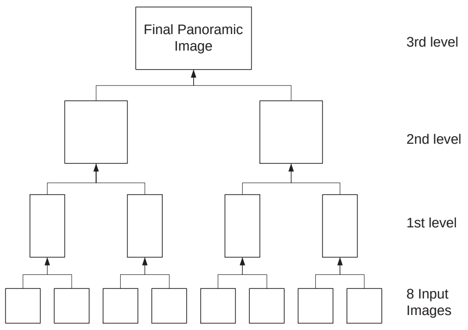


Fig. 10. Stitching hierarchy

a system of $2n$ equations.

$$\begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -y_1x_1' \\ 0 & 0 & 0 & x_1 & y_1 & 1 & -x_1y_1' & -y_1y_1' \\ x_2 & y_2 & 1 & 0 & 0 & 0 & -x_2x_2' & -y_2x_2' \\ 0 & 0 & 0 & x_2 & y_2 & 1 & -x_2y_2' & -y_2y_2' \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n & y_n & 1 & 0 & 0 & 0 & -x_nx_n' & -y_nx_n' \\ 0 & 0 & 0 & x_n & y_n & 1 & -x_ny_n' & -y_ny_n' \end{pmatrix} \begin{pmatrix} m_0 \\ m_1 \\ m_2 \\ m_3 \\ m_4 \\ m_5 \\ m_6 \\ m_7 \end{pmatrix} = \begin{pmatrix} x_1' \\ y_1' \\ x_2' \\ y_2' \\ \vdots \\ x_n' \\ y_n' \end{pmatrix} \tag{4}$$

Solving the system for m_i, the homography can be obtained. When the system is over-determined, we find a solution which introduces the least error. We achieve this by using singular value decomposition (SVD). It registers images reasonably well, provided that image correspondences are correct. The method depends heavily on the accuracy of image correspondences. When errors exist, the homography computed may not well register images. Therefore, a fine adjustment step is carried out. It minimizes the following sum of the squared intensity errors of the image pairs.

$$E = \sum \left[I'(x', y') - I(x, y)\right]^2 \tag{5}$$

where $I(x, y)$ and $I'(x', y')$ are the values of corresponding pixel pairs in images I and I' respectively. The error function sums all pixels within the overlapping region of images I and I'. We employ the Levenberg-Marquardt method [40], [38] to perform this minimization. With this minimization, the computed homography is further improved.

D. Stitching Multiple Sources

So far we only consider the case of fusing two input images. For N input images, the same method is recursively applied. In the first round, adjacent images are combined, resulting in $N/2$ images. Recursively, the adjacent images from the newly formed $N/2$ images are combined to give $N/4$ images. This is done repeatedly until one single mosaic is resulted. This hierarchical approach is illustrated in Figure 10.

After registering all input images, the images are composed into a complete mosaic. However, there are inevitably errors in the registration process. A simple linear blending function is used to smoothly blend the images together, resulting in a seamless mosaic. This blending is done in real-time using the commodity graphics accelerators. For better composition, one may opt to use multilevel blending method described by Burt and Adelson [5], in the expense of higher computation requirement.

VI. Immersive Projection

A. Immersive Display

A large display is essential to fool the human so as to provide a feeling of immersion in the virtual/reconstructed environment. As sight contributes around 70% of human perception [17], fooling the sense of sight is the main concern for immersion. This feeling of immersion can be achieved by surrounding the whole FOV of users. Normal people have a horizontal FOV of around 150° and vertical FOV of around 110°. This is also the degree of FOV we capture.

In our system, we employ a hemispherical display called the VisionStation [1]. It consists of a LCD projector equipped with the F-Theta lens (a fish-eye-like lens) and a human-sized hemispherical screen. The large hemispherical screen engulfs the viewer and offers a horizontal FOV of 180° and vertical FOV of 135° (see Figure 11).

B. Sweet Spot Relocation

Sweet spot refers to an ideal viewing position where a correct image can be observed. Obviously, this spot should be located at the center of projection (COP) of the projector. However, this position is normally not available for a viewer since the projector itself occupies space. Once the viewer moves away from the sweet spot, he/she observes a distorted image. To overcome this problem, the sweet spot should be relocated to a designated viewing position. In other words, we should warp the image so that the viewer can observe a distortion-free image at the designated viewing position.

To relocate the sweet spot, we need to know the position of the projector P_p, the position of the viewer P_e (sweet spot), and the geometry of the display screen. Then, rays are cast from the sweet spot towards

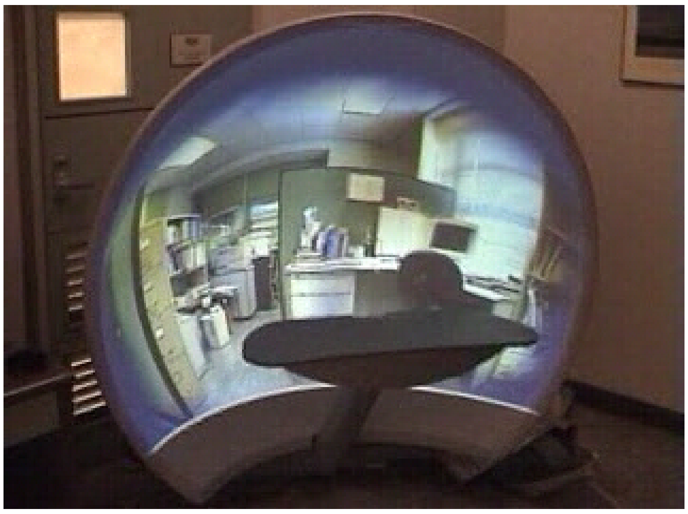


Fig. 11. Panoramic display of indoor scene on the hemispherical screen.

the display screen. For a point u on the image plane associated with the sweet spot, we project it onto the screen and obtain an intersection point P_t. The projector sees P_t as v on the projector image plane. Therefore, a relationship can be established between u and v (Figure 12). By warping the pixel from u to v, we pre-distort the image so that the image looks correct as viewed from the sweet spot. For display of simple geometry, an analytical solution of this warping can be derived. For general display surface which can be partially concave and partially convex, analytical solution may not be available. Numerical method is the only choice in such case.

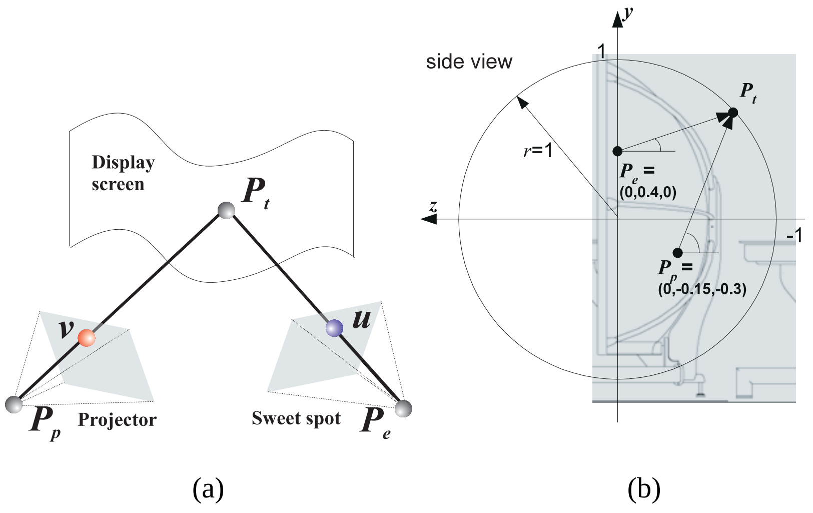


Fig. 12. Sweet spot relocation

To display the distortion-free image on the VisionStation, we compute the image warping by ray casting. Firstly, we associate an image plane with the sweet spot. This image plane should be oriented to face the display screen. For each pixel u on this image plane, we determine its corresponding point v on the projector image plane. A ray is cast from P_e via u towards the projection screen along the direction R_e. The ray hits the projection screen at P_t. Thus,

$$P_t = P_e + tR_e \tag{6}$$

where t is a scalar.

The VisionStation display screen is considered as a unit sphere with center at the original (Figure 12(b)), therefore,

$$\begin{aligned} |P_t| &= 1 \\ |R_e|^2 t^2 + 2P_e \cdot R_e t + |P_e|^2 - 1 &= 0 \end{aligned} \tag{7}$$

Solving Equation 7, we obtain two solutions of t. We discard the one not on the projection screen (z-coordinate of $P_t > 0$ in our model).

Next, we determine point v on the projector image plane where P_t is projected onto. A ray is cast from P_p towards P_t. Therefore,

$$P_t = P_p + sR_p \tag{8}$$

where s is a scalar and R_p is the unit vector from P_p towards P_t.
Combining with Equation 6, we get

$$R_p = \frac{P_e - P_p + tR_e}{|P_e - P_p + tR_e|} \tag{9}$$

The VisionStation projector employs a F-Theta lens, which is a fish-eye-like lens. Its projection is no longer perspective. Instead, the relationship between the point v and the vector R_p can be described by the following equation,

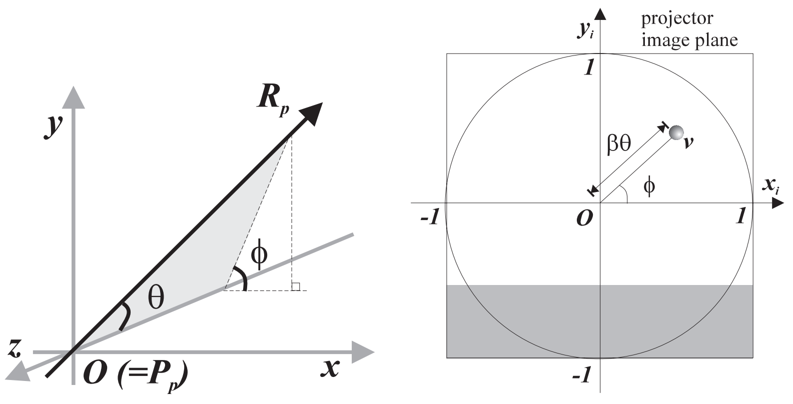
$$l = \beta\theta \tag{10}$$

where l is the distance between v and the image center, β is a proportionality constant and θ is the angle between the associated ray (R_p) and the principle projection axis (see Figure 13 for the illustration of notations). In other words, if the light ray R_p deviates θ from the principle axis, it intersects the image plane at a point v with distance l from the image center.

By simple trigonometry, we get

$$\phi = \tan^{-1}\left(\frac{y_p}{x_p}\right) \qquad \theta = \cos^{-1}\left(R_p \cdot -\hat{Z}\right) \tag{11}$$

where $R_p = (x_p, y_p, z_p)$ and $-\hat{Z}$ is the negative z-axis of the projector coordinate system.



(a) (b)

Fig. 13. The non-perspective projection provides a full coverage of the hemispherical display. (a) & (b) illustrates the meaning of notations used in our derivation.

Using Equations 10 and 11 we obtain the image coordinate of v

$$v = (x_v, y_v) = (\beta\theta \cos\phi, \beta\theta \sin\phi). \tag{12}$$

This warping from u to v is encoded in the texture coordinates computed during the offline phase.

In actual implementation, we have to obtain several measurements. These measurements include the physical geometry of the hemispherical screen and the physical viewpoint of the user. All the steps carried out in the sweet spot relocation are performed off-line.

VII. Implementation and Results

Our current implementation of the Immersive Cockpit uses eight color CCD cameras which are connected to two video quad mergers. Two PCs (Pentium III 500 MHz) digitize the signals and send the video across the network to a remote site for panoramic display. The receiver side contains a Pentium III 600 MHz PC connecting to an immersive display, the VisionStation. The computers are connected through an 100 Mbps Ethernet network. Since we do not rely on any high-end workstation, the cost of our system is not high. Readers are referred to the following webpage for the supplementary video

`http://www.cse.cuhk.edu.hk/∼ttwong/papers/cockpit/cockpit.html`

A. Scenes from Static Cameras

Indoor scene contains nearby objects. Hence, cameras should be placed as close as possible in order to minimize the error. Figure 5 shows the arrangement of cameras. Figure 16(a) shows all 8 captured views. The stitched panorama is shown in Figure 16(b). This panorama is further warped for sweet spot relocation (Figure 16(c)) so that it looks correct when displayed on the hemispherical display screen.

Figure 11 shows the warped panorama displayed on the VisionStation. There are minor artifacts in the final panoramic video. When an object approaches the cameras, there is a slight ghosting artifact due to the violation of single-COP assumption.

For the outdoor case, objects are expected to be far away. The cameras can be positioned farther apart without significant error due to parallax. In our experiment, we capture the environment surrounding our Engineering building. The capture is taken place on the 10th floor of the building. Figure 14 shows the floor-plan of our camera arrangement. The cameras are arranged into two groups which are approximately 3 meters apart. Even with this large displacement, the video streams can be combined with little visual artifact (Figure 16(d)-(f)). For instance, if the maximum displacement between optical center of cameras are 20cm, a distance of more than 112.4 meters can be considered as far away (see appendix). Hence, our mosaicing engine can be used without introducing significant visual artifacts. In the outdoor example illustrated in Figure 14, the minimum distance is 1.684 kilometers.

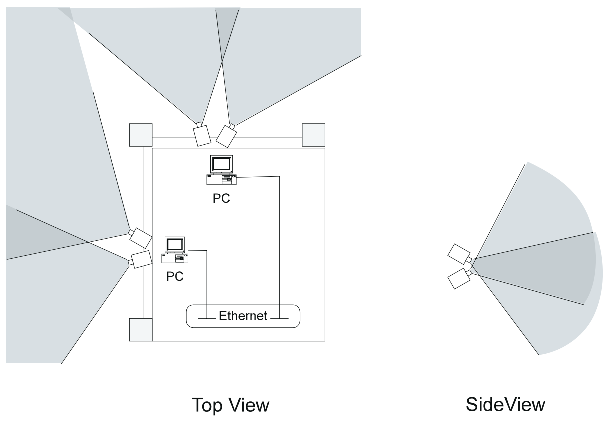


Fig. 14. Arrangement of cameras for outdoor scene capture

B. *Scenes from Dynamic Cameras*

To demonstrate the ability of our system in the application of navigation, we also capture the walk-through sequences of two outdoor scenes using dynamic cameras. We use the same camera configuration as in the indoor static case. The camera set is moved as a whole. The relative position and orientation among cameras remain unchanged during capture. Figures 17(a)-(d) show four frames from the cafeteria example in which the camera set moves from left to right. Another example, the bridge, is shown in Figures 17(e)-(h). In this case, the camera set moves forward.

Source	Latency (milliseconds)
Video compression	15 - 30 ms
Video decompression	15 - 30 ms
Network buffering	120 ms
Network latency	10 ms
Rendering	$<$30 ms
Total	*$<$ 220 ms*

TABLE I

LATENCY

Video Codec	avg. fps	compression ratio
Motion JPEG	24.256	16.63
MPEG-4	8.479	113.10
Indeo	1.250	34.04

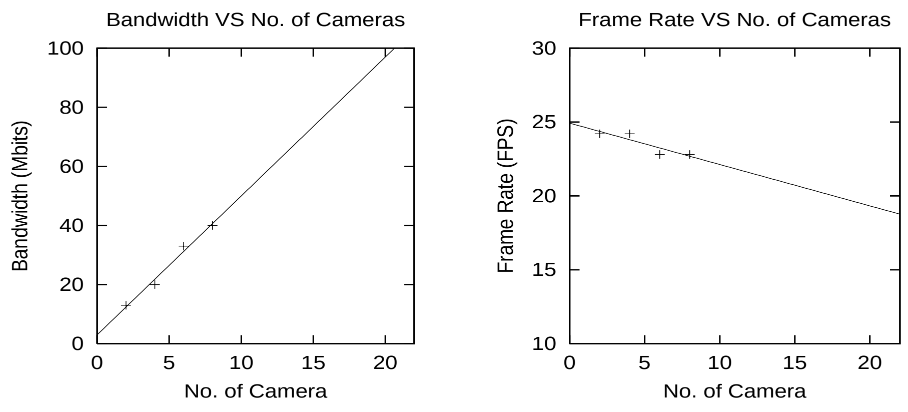
TABLE II

PERFORMANCE OF THREE DIFFERENT VIDEO CODECS

C. Performance

Latency is an important issue in any tele-immersive applications. Our current implementation assumes the distance between the cameras and the remote viewer is within the same city. Our system can attain a latency of less than 220 milliseconds. Table I shows the breakdown of latency in our system. The latency is contributed by multiple components including, video encoding, network buffering, network latency and rendering.

Table II shows the performance against different video codecs. Under our high bandwidth network (100Mbps), motion JPEG outperforms other compression methods. Park and Kenyon [26] pointed out that a maximum latency of 100 milliseconds is required for tele-operation. Our current implementation is suitable for passive viewing but may not be suitable for tele-operation. To satisfy such latency requirement, we need to reduce the time for network buffering. Currently, we assume the network is very noisy and keep a large buffer. Another bottleneck is the speed of video codec. Note that we are currently using low-end PCs (Pentium III 500MHz) and software video codec. The performance can be significantly improved if we use high-end PCs and install the hardware video codec.



(a) (b)

Fig. 15. (a) Bandwidth vs. No. of camera. (b) Frame rate vs. No. of camera.

D. Scalability

Our Immersive Cockpit is designed to be scalable. The number of video cameras for scene capture can be increased if necessary. By using 8 video cameras, we are able to cover almost the whole viewing hemisphere, resulting in about 150° horizontal FOV and about 110° vertical FOV. We investigate the system by varying the number of video cameras used. In our current arrangement, each basic capture unit (two cameras) captures a vertical band of the scene. Each addition of a basic capture unit increases the horizontal FOV by approximately 35°. Hence, about eleven units are enough to cover the whole 360° FOV.

We have also measured the bandwidth usage against the number of cameras (Figure 15(a)). The relationship is about linear. Our current configuration still does not utilize the whole available bandwidth. There is still room to be scaled up. More cameras can be added. With a network bandwidth of 100Mbps, more than 20 cameras can be supported. If the Ethernet does not offer such high efficiency, we can increase the compression ratio to accommodate more video streams (as a trade-off of image quality) in the same network.

Figure 15(b) shows the average frame rate against the number of cameras. There is a slight drop of frame rate when more cameras are added. Using software codec, we still obtain a real-time performance. For additional cameras, the frame rate is expected to drop. The frame rate of our system is mostly guaranteed to be above 20 fps.

The optimal image resolution depends on applications. An application requiring large FOV and far-away object monitoring, as in the case of air traffic control tower, may require high-resolution video. On the other hand, an application requiring 180° FOV and close-object monitoring, such as the case of vehicle driving, can accept lower resolution video.

E. Limitations

The generated panoramic video is sometimes not sharp enough. In some cases, ghosting effect is observable. Moreover, there is a problem of discontinuity in the color tone. These are mainly due to the following reasons:

1. The video streams may not be well stitched due to the errors in selecting feature points for correspondence. Another possible problem is the violation of far-away-scene assumption as some objects are too close to the cameras.
2. During the run-time, the depth of objects may change and cause stitching errors.
3. Ordinary CCD video cameras are usually equipped with a built-in automatic gain control (AGC) which automatically adjusts the brightness as well as the contrast. Different parts of the scene have different lighting conditions, hence different cameras (pointing at different parts of the scene) may have different gain values. This results in varying tone across the stitched images. This tone problem can be solved by using video cameras with computer-controllable exposure unit. Note that the exposure of all cameras should be adjusted in phase in order to maintain the tone consistency.
4. In capturing the video signals, the video merger shrinks the resolution by half in both width and height hence reduces the image quality.
5. The video codec degrades the video quality for compression. Setting a lower compression ratio can certainly improve the image quality, but with an increase of requested network bandwidth.

VIII. Conclusion

In this paper, we present our system, the Immersive Cockpit, for the tele-immersive reproduction of the remote environment. Panoramic video stream can be generated on-the-fly without using specially-designed video capture devices. By employing common CCD cameras, we reconstruct the remote environment on the immersive display.

In the future, we will extend our system to include 3D sound and remote control functions. This will be especially useful for industrial tele-immersive applications and usage in remote exploration. Currently, we use eight cameras which cover around 150° FOV. We will increase the possible viewing direction by adding more cameras to the system.

ACKNOWLEDGMENT

We would like to thank Keung-Tat Lee, Yim-Pan Chui, Wai-Man Pang and Chung-King Ma (Oscar) and Lui-Yuen Lai for their help throughout the experiment. This project is supported by the Research Grants Council of the Hong Kong Special Administrative Region, under CUHK Direct Grant for Research (Project No. 2050283) and RGC Co-operative Research Centres (CRC) Scheme (Project No. CRC 4/98).

REFERENCES

[1] Alternate Realities Corporation. http://www.elumens.com/.

[2] D. Anderson, S. Tzou, R. Wahbe, R. Govindan, and M. Andrews. Support for live digital audio and video. In *Proceedings of 10th International Conference on Distributed Computing Systems*, pages 54–61, Paris, France, May 1990.

[3] J. Baldwin, A. Basu, and H. Zhang. Panoramic video with predictive windows for telepresence applications. In *Proc. 1999 IEEE International Conference on Robotics and Automation*, Detroit, U.S.A., May 1999.

[4] D. Browning, Cruz Neira, C. Sandin, and Tom DeFanti. The cave automatic virtual environment: Projection-based virtual environments and disability. In *Proceedings of the First Annual International Conference, Virtual Reality and People with Disabilities*, January 1993.

[5] Peter J. Burt and Edward H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.

[6] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the CAVE. In *Proceedings of SIGGRAPH 93*, pages 135–142, August 1993.

[7] P. Dani and S. Chaudhuri. Automated assembling of images: Image montage preparation. In *Pattern Recognition*, volume 28, pages 431–445, March 1995.

[8] J. Davis. Mosaics of scenes with moving objects. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[9] Fakespace Systems. http://www.fakespacesystems.com/.

[10] O. Faugeras. *Three-Dimensional Computer Vision*. MIT Press, 1993.

[11] Domenico Ferrari. Design and applications of a delay jitter control scheme for packet-switching internetworks. In *Proceedings of the second International Conference on Network and Operating System Support for Digital Audio and Video*, 1991.

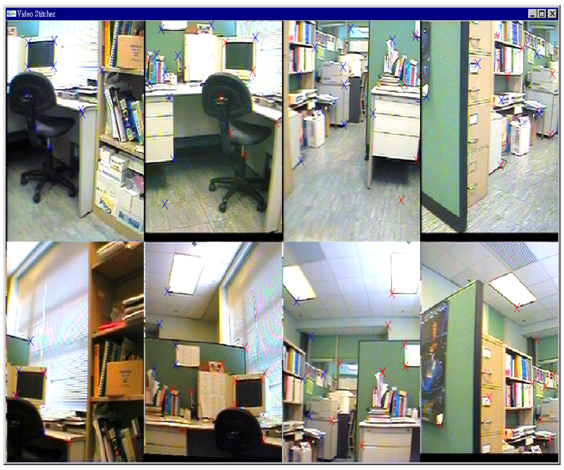
[12] J. Foote and D. Kimber. FlyCam: Practical panoramic video and automatic camera control. In *Proceedings of IEEE International Conference on Multimedia and Expo*, volume 3, pages 1419–1422, 2000.

[13] Joshua Gluckman, Shree K. Nayar, and Keith J. Thoresz. Real-time omnidirectional and panoramic stereo. In *Proceedings of the 198 DARPA Image Understanding Workshop*, California, USA, November 1998.

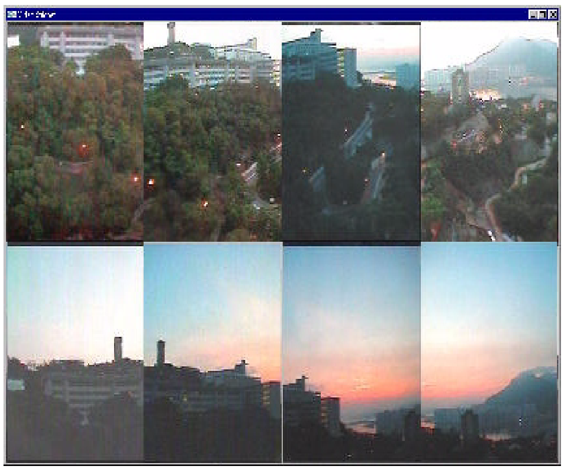
[14] Manuel Guillen Gonzalez, Phil Holifield, and Martin Varley. Improved video mosaic construction by accumulated alignment error distribution. In *Proceedings of the British Machine Vision Conference*, 1998.

[15] Sevket Gümüştekin and Richard W. Hall. Mosaic image generation on a flattened gaussian sphere. In *Proceedings of IEEE Workshop on Applications of Computer Vision*, pages 50–55, 1996.

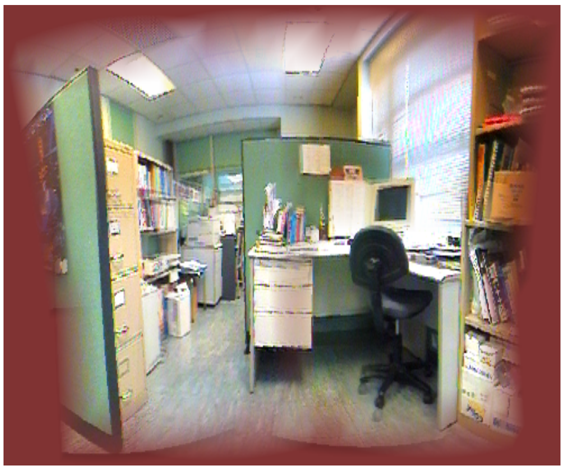
[16] Janne Heikkila and Olli Silven. A four-step camera calibration procedure with implicit image correction. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, 1997.



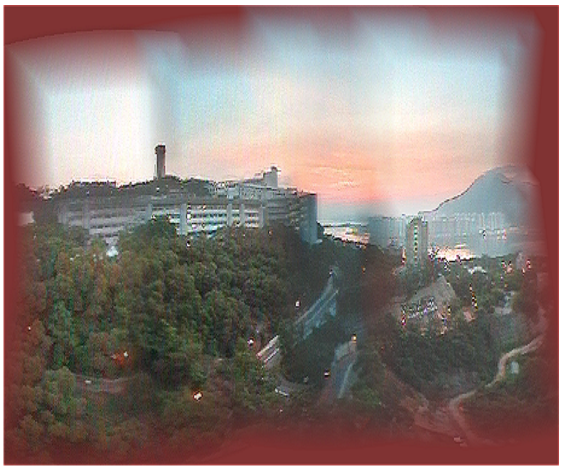
(a) Captured frames of indoor scene



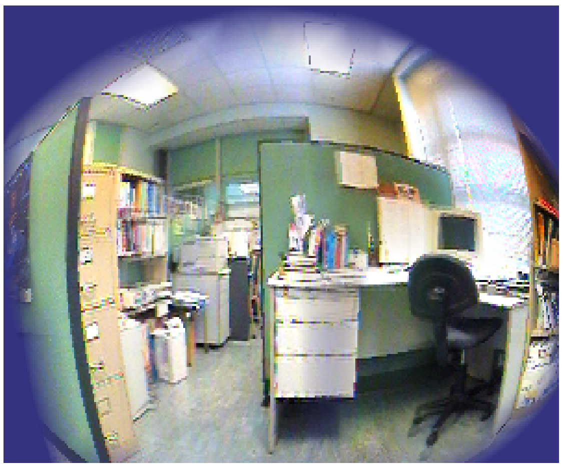
(d) Captured frames of outdoor scene



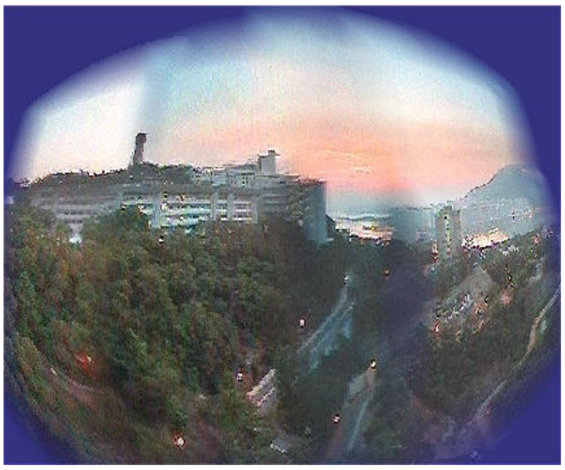
(b) Stitched indoor scene



(e) Stitched outdoor scene



(c) Warped indoor scene



(f) Warped outdoor scene

Fig. 16. (a)-(c): Indoor scene example. (d)-(f):Outdoor scene example.

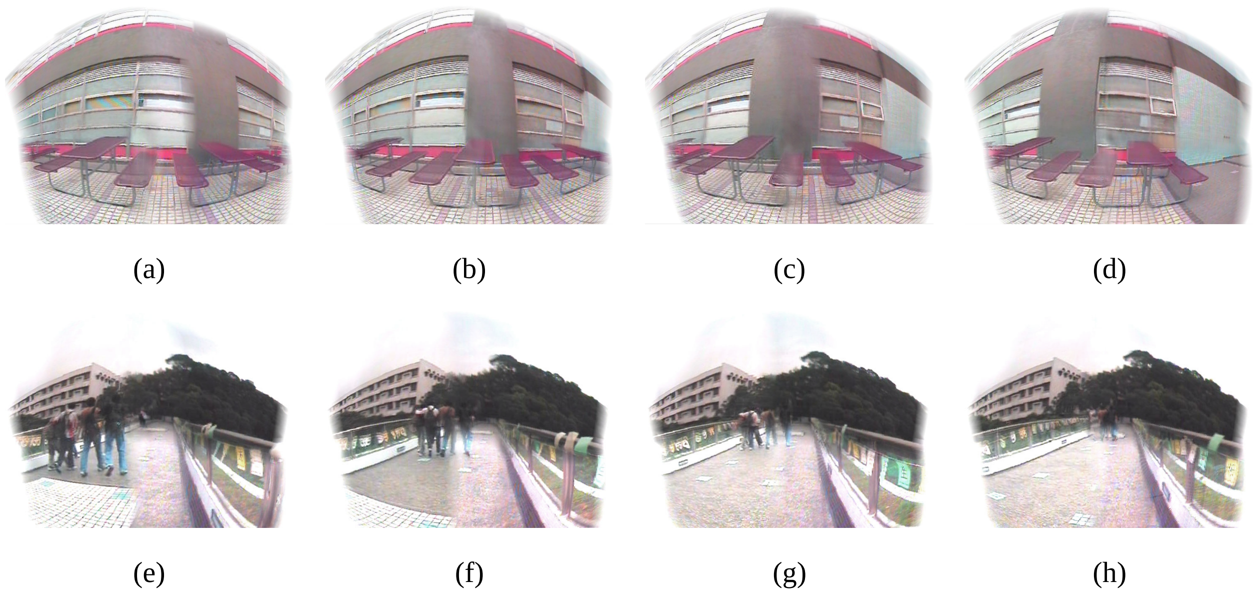


Fig. 17. Frames extracted from two panoramic videos with moving cameras. (a)-(d): The cafeteria example. (e)-(h): The bridge example.

[17] M. L. Heliig and El Cine del Futuro. The cinema of the future. *Presence*, 1(3):279–294, 1992.

[18] Michitaka Hirose, Tetsuro Ogi, and Toshio Yamada. Integrating live video for immersive environments. *IEEE MultiMedia*, 6(3):14–22, July–September 1999.

[19] C. D. Kuglin and D. C. Hines. The phase correlation image alignment method. In *Conference on Cybernetics and Society*, pages 163–165, September 1975.

[20] A. Majumder, G. Meenakshisundaram, W. B. Seales, and H. Fuchs. Immersive teleconferencing: A new algorithm to generate seamless panoramic imagery. In *Proceedings of ACM Multimedia*, pages 169–178, November 1999.

[21] D. L. Milgram. Adaptive techniques for photo mosaicing. *IEEE Transactions in Computers*, C(26):1175–1180, 1977.

[22] Jane Mulligan and Kostas Daniilidis. View-independent scene acquisition for tele-presence. In *Proceedings of International Symposium on Augmented Reality*, October 2000.

[23] Shree K. Nayar. Omnidirectional video camera. In *Proceedings of the 1997 DARPA Image Understanding Workshop*, May 1997.

[24] Network Working Group. Real time streaming protocol (RTSP). Request for Comments 2326, April 1998.

[25] Panoram Technolgies, Inc. http://www.panoramtech.com/.

[26] Kyoung Shin Park and Robert V. Kenyon. Effects of network characteristics on human performance in a collaborative virtual environment. In *Proceedings of IEEE VR '99*, pages 104–111, Houston, TX, March 1999.

[27] C. Partridge. Isochronous applications do not require jitter-controlled networks. Request for Comments 1257, September 1991.

[28] Shmuel Peleg and Joshua Herman. Panoramic mosaics by manifold projection. In *Proceedings of Computer Vision and Pattern Recognition*, pages 338–343, June 1997.

[29] Venkata N. Peri and Shree K. Nayar. Generation of perspective and panoramic video from omnidirectional video. In *Proceedings of DARPA Image Understanding Workshop*, pages 243–245, 1997.

[30] R. Raskar, M. S. Brown, R. Yang, W. C. Chen, G. Welch, H. Towles, B. Seales, and H. Fuchs. Multi-projector displays using camera-based registration. In *Proceedings of IEEE Visualization 1999*, pages 161–168, San Fransisco, CA, October 24-29 1999.

[31] Ramesh Raskar. Immersive planar display using roughly aligned projectors. In *Proceedings of IEEE Virtual Reality 2000*, March 18–22 2000.

[32] Ramesh Raskar, Greg Welch, Matt Cutts, Adam Lake, Lev Stesin, and Henry Fuchs. The office of the future: A unified approach to image-based modeling and spatially immersive displays. In *Proceedings of SIGGRAPH 98*, pages 179–188, July 1998.

[33] B. S. Reddy and B. N. Chatterji. An FFT-based technique for translation, rotation, and scale-invariant image registration. *IEEE Transactions on Image Processing*, 5(8), 1996.

[34] Harpreet S. Sawhney, Steve Hsu, and Rakesh Kumar. Robust video mosaicing through topology inference and local to global alignment. In *Proceedings of the European Conference on Computer Vision*, 1998.

[35] Daniel R. Schikore, Richard A. Fischer, Randall Frank, Ross Gaunt, John Hobson, and Brad Whitlock. High-resolution multiprojector display wall. *IEEE Computer Graphics and Applications*, 20(4):38–44, July/August 2000.

[36] Prashant J. Shenoy, Pawan Goyal, and Harrick M. Vin. Issues in multimedia server design. *Computing Surveys*, 27(4):636–639, 1995.

[37] Heung-Yeung Shum and Li-Wei He. Rendering with concentric mosaics. In *Proceedings of SIGGRAPH 99*, pages 299–306, August 1999.

[38] Heung Yeung Shum and Richard Szeliski. Panoramic image mosaics. Technical Report MST-TR-97-23, Microsoft Research, 1997.

[39] Rahul Swaminathan and Shree K. Nayar. Polycameras: Camera clusters for wide angle imaging. Technical Report CUCS-012-99, Columbia University Technical Report, New York, 1999.

[40] Richard Szeliski. Image mosaicing for tele-reality applications. Technical Report CRL 94/2, DEC Cambridge Research Lab, May 1994.

[41] Richard Szeliski. Video mosaics for virtual environments. *Virtual Reality*, pages 22–30, March 1996.

[42] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and texture-mapped models. In *Proceedings of SIGGRAPH 97*, pages 251–258, August 1997.

[43] The PowerWall Team. Powerwall. http://www.lcse.umn.edu/research/powerwall/powerwall.html.

[44] Trimension Systems Ltd. http://www.trimension-inc.com/.

[45] R. Tsai. A Versatile Camera Calibration Technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras. *IEEE Journal of Robotics and Automation*, pages 323–344, 1987.

[46] Yalin Xiong and Ken Turkowski. Creating image-based VR using a self-calibrating fisheye lens. In *Proceedings of CVPR*, 1997.

Appendix

Parallax is the difference in apparent direction of an object as seen from two different viewpoints. If two cameras see a 3D point with parallax less than half of the minimum angle subtended by a pixel, then the projected pixel coordinates will deviate less than half a pixel as if the cameras share a common optical center. This can be formulate as the following inequality:

$$A_p < A_e/2 \tag{13}$$

where A_e is the angle subtended by a pixel and A_p is the angle of parallax. By simple trigonometry (Figure 18), we have

$$\tan\frac{A_p}{2} \approx \frac{D_e/2}{D} \tag{14}$$

where D_e is the distance between the optical centers of camera C_1 and C_2 and D is the distance from the object. Hence,

$$\frac{D_e/2}{D} < \tan\frac{A_e}{4} \tag{15}$$

As A_e is usually very small, $\tan(A_e/4)$ is approximately equal to $A_e/4$. Thus,

$$D > 2D_e/A_e \tag{16}$$

In our current setup for outdoor scene, D_e is approximately 3 meters and A_e is 0.00356 radian. Hence the minimum distance D from the closest object should be at least 1.684 kilometers in order to keep the error within a pixel.

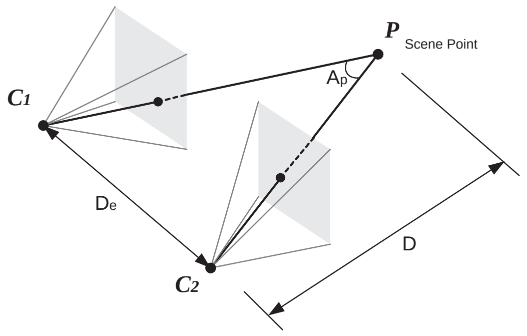


Fig. 18. Parallax of two cameras