Cubic-Panorama Image Dataset Analysis for Storage and Transmission

by

Saeed Salehi Doolabi

PhD Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
In partial fulfillment of the requirements
For the Ph.D. degree in
Electrical Engineering

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

This thesis involves systems for virtual presence in remote locations, a field referred to as telepresence. Recent image-based representations such as Google map's street view provide a familiar example. Several areas of research are open; such image-based representations are huge in size and the necessity to compress data efficiently for storage is inevitable. On the other hand, users are usually located in remote areas, and thus efficient transmission of the visual information is another issue of great importance.

In this work, real-world images are used in preference to computer graphics representations, mainly due to the photorealism that they provide as well as to avoid the high computational cost required for simulating large-scale environments. The cubic format is selected for panoramas in this thesis. A major feature of the captured cubic-panoramic image datasets in this work is the assumption of static scenes, and major issues of the system are compression efficiency and random access for storage, as well as computational complexity for transmission upon remote users' requests.

First, in order to enable smooth navigation across different view-points, a method for aligning cubic-panorama image datasets by using the geometry of the scene is proposed and tested. Feature detection and camera calibration are incorporated and unlike the existing method, which is limited to a pair of panoramas, our approach is applicable to datasets with a large number of panoramic images, with no need for extra numerical estimation.

Second, the problem of cubic-panorama image dataset compression is addressed in a number of ways. Two state-of-the-art approaches, namely the standardized scheme of H.264 and a wavelet-based codec named Dirac, are used and compared for the application of virtual navigation in image based representations of real world environments. Different frame prediction structures and group of pictures lengths are investigated and compared for this new type of visual data. At this stage, based on the obtained results, an efficient prediction structure and bitstream syntax using features of the data as well as satisfying major requirements of the system are proposed.

Third, we have proposed novel methods to address the important issue of disparity estimation. A client-server based scheme is assumed and a remote user is assumed to seek information at each navigation step. Considering the compression stage, a fast method that uses our previous work on the geometry of the scene as well as the proposed prediction structure together with the cubic format of panoramas is used to estimate disparity vectors efficiently.

Considering the transmission stage, a new transcoding scheme is introduced and a number of different frame-format conversion scenarios are addressed towards the goal of free navigation. Different types of navigation scenarios including forward or backward navigation, as well as user pan, tilt, and zoom are addressed. In all the aforementioned cases, results are compared both visually through error images and videos as well as using the objective measures. Altogether free navigation within the captured panoramic image datasets will be facilitated using our work and it can be incorporated in state-of-the-art of emerging cubic-panorama image dataset compression/transmission schemes.

Acknowledgements

I would like to take this opportunity to thank my PhD supervisor, professor Eric Dubois, for his constant support as well as helpful and in time suggestions throughout these years. Also I would like to thank all the committee members of my comprehensive exam as well as the thesis proposal and defense session, in addition to all my colleagues in the VIVA lab who have provided a friendly environment for research. I would like to extend my gratitude to the kind staff of the engineering department at the University of Ottawa and my dear friends in the city of Ottawa. Last but not least, special thanks to my dear family which without their kind supports the completion of this work would not be imaginable.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

There exist numerous applications [1, 2, 3, 4] in which a user desires to navigate virtually in a distant real-world environment. Numerous examples can be envisioned: Tourists may wish to remotely feel the presence in a historic environment. Passengers looking for a hotel in a city other than theirs may desire to know what it would feel like walking in the building before making reservations. Not everybody may be able to visit a museum of interest in a distant location. A customer may want to buy a house in a city different from his hometown and looks forward to a sense of walking in the new house in advance. New students applying for universities abroad may want to get a taste of their new campus from home. Similar demands may exist when an employee looks for a new job in a remote company. Not everybody has the time, budget, and means to take a space tour and walk on the moon, but images can be captured in high resolution; A movie is being shot in an artistically decorated scenery and the director hopes he could have captured the environment itself such that people could not only watch the movie, but also be able to virtually walk within the scenery freely after watching it and reproduce actors experiences themselves. The manager of a newly built sports complex may want to provide customers with a virtual version of the facilities available for his marketing purposes. Every year more than two million people attend the religious gathering of hajj and need to practice the rituals in advance and here again a virtual practice session can be a great help.

Almost everyone has had such an experience at least once when playing computer games. A lot of advancements have been achieved in virtual navigation in a virtual environment e.g. 3D games and 3D animated movies. However, conventional geometry-based image rendering has its own drawbacks. They rely on geometric model building

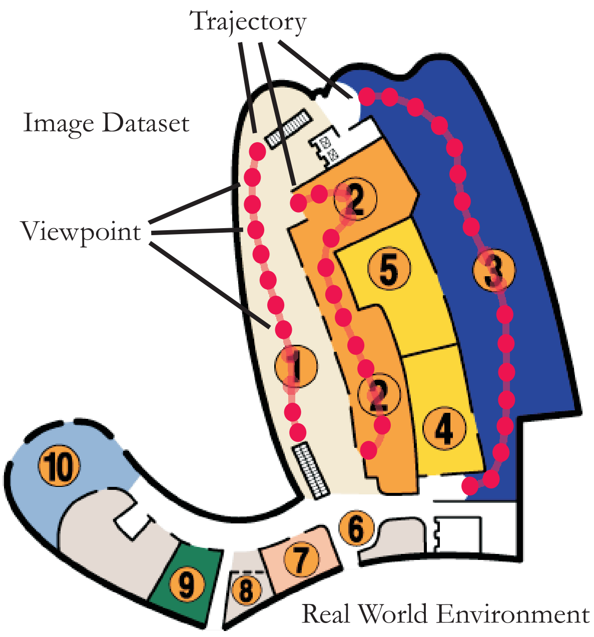


Figure 1.1: An image dataset consisting of a collection of basis images in a particular image format as a discrete image representation of a real-world environment

to synthesize novel views and building such models is a complicated task and computationally expensive. They also depend on the complexity of the environment. If a good sense of reality is sought, then some features like the photometric properties of lighting and shading of the environments should be incorporated, which makes the whole process more expensive. Also synthesizing novel views from numerous geometric models is so time consuming that a trade off has to be made in order to balance the rendered image quality and the rendering speed. While it is difficult to achieve photo-realistic quality even using the state-of-the-art geometry-based image rendering techniques, image based rendering (IBR), to which more attention has been paid recently, directly takes image inputs to synthesize novel views. IBR just relies on the basis images to synthesize novel images and hence the rendering speed is high and not dependent on the complexity of the environment. Therefore IBR is an effective way to represent the real-world environments and provides a promising approach to generate photo-realistic novel-view images.

Having captured a single omnidirectional panoramic image from a single viewpoint, the user will be able to view the environment in any desired direction within a desired field of view. If a series of such high-resolution omnidirectional images is taken and stored properly as shown in Figure 1.1, then the user is able not only to look around, but also

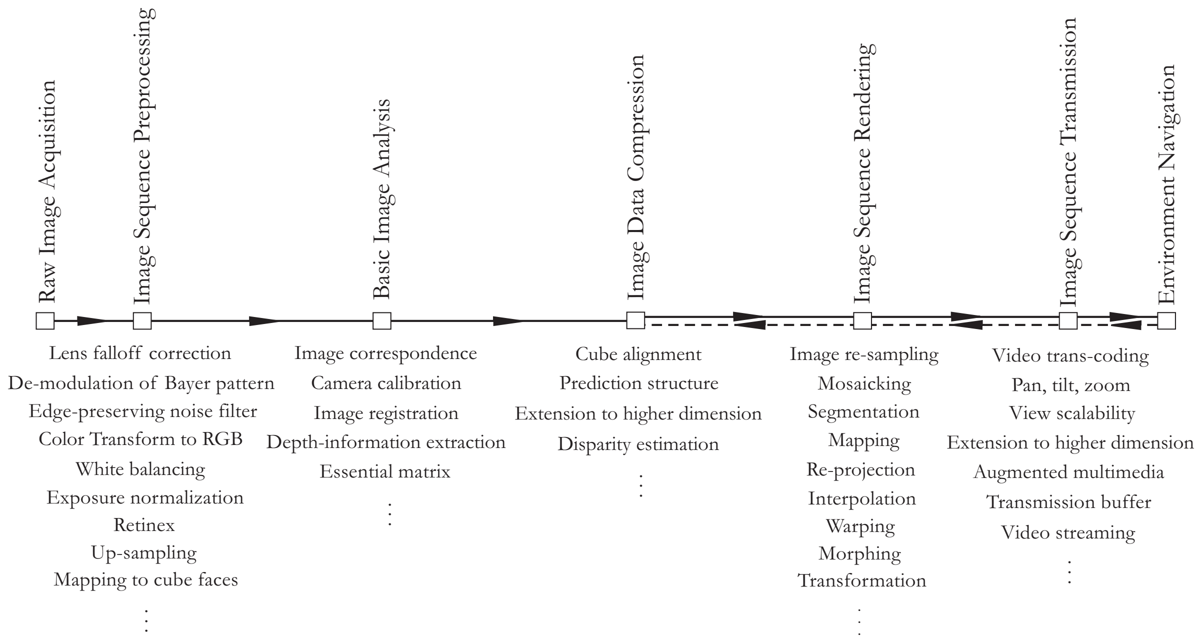


Figure 1.2: An image-based virtual environment navigation system with its main components including image dataset compression and transmission

to step forward or backward and get a feeling of walking through the environment [2]. From the end-to-end system viewpoint, the user at a remote location issues a command seeking a new viewing frame and the server in response transmits the next view of the generated virtual video.

In practice, such panoramic image datasets should be compressed efficiently. As it will be explained later, the set of images considered in our work has specific features which can be exploited during the process of compression for storage as well as transmission. Dealing with these features distinguish our study from a conventional video processing paradigm. Secondly, in real time according to user commands the compressed data, i.e. bitstream, should be transcoded from the storage format to a proper video format in an efficient way to be transmitted through the communication channel. This task constitutes another area of interest for us. Finally at the receiver side, reconstructed frames are generated and remote virtual navigation will be realized.

The system architecture consists of various stages, namely, image acquisition; processing and analysis; dataset compression for storage; image sequence rendering and transmission to a remote destination; and finally virtual environment navigation as shown in

Figure 1.2. The first section of this chapter tries to shed light on how these stages work in more detail. Our colleagues in the VIVA lab have been working on the aforementioned stages of this application [1, 5, 6, 7, 8, 9, 10] as part of the NSERC funded NAVIRE project [11]. The main focus of our research will be the analysis of such cubic-panorama image datasets toward achieving efficient compression as well as transmission, keeping the goal of free navigation in mind all the time, while cubic-panorama image dataset preprocessing stage is addressed as starting point.

Since this is a brand new application, no off-the-shelf solution is available in the market to be purchased and little research exists on the specific problems. The rest of this chapter is organized as follows: in section one, the general architecture of the system from image acquisition to virtual navigation is explained in more detail. In section two, specific features of the image dataset that can be considered before compression and transmission are explained. In the last section of this chapter, requirements and challenges which are considered throughout our research are explained. These challenges are addressed during this thesis.

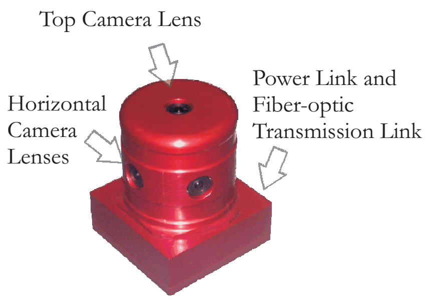
1.1 Image-Based Virtual Environment Navigation System

Different practical image-based virtual environment navigation systems providing different exploration capabilities have been developed [1, 2, 3, 4]. The system that has been used in our research is explained below with more emphasis on compression and transmission stages [1, 11].

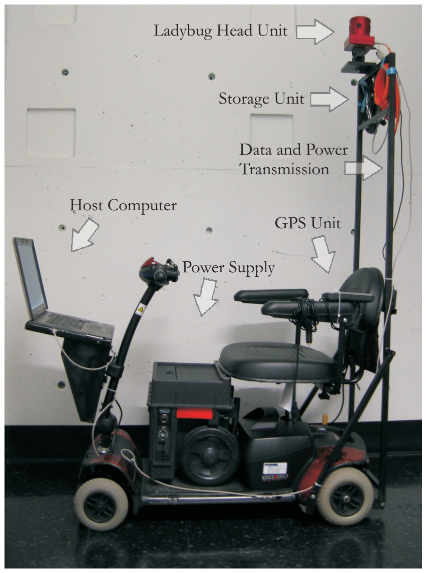
1.1.1 Raw-Image Acquisition

The raw-image acquisition process generates the source input for the navigation system. Camera orientation and motion trajectory should be well planned to capture an efficient representation of the environment. Over-sampling the environment increases the burden and size of raw image sequence by unnecessary overlapping [12] while under-sampling causes aliasing and image distortion. Spectral analysis [13] and plenoptic sampling theories [14, 15] provide guidelines to avoid either of the above issues.

Such a system can use a single camera [16], multiple cameras [1], or a camera array combined with a data recording device for storage and a portable computer for controlling



(a) the multisensor omnidirectional Ladybug camera head



(b) the imaging system

Figure 1.3: Cubic-panorama raw-image sequence acquisition

the acquisition process, and an on-board power source for providing a steady power supply.

In our system, as shown in Figure 1.3, a Ladybug camera from Point Grey Research is used. This camera consists of six ICX204AQ color CCD image sensors and has six high quality micro lenses with the focal length of 2.5mm. One lens on the top of the camera head unit points up and five lenses pointing horizontally are assembled in a horizontal circle. The camera head is pre-calibrated by the manufacturer to enable high-quality image processing. The lens settings, e.g., the iris and the focus, are fixed to keep the cameras calibrated during the acquisition process. This camera has been utilized in several applications [1, 3, 17]. The storage unit involves a number of large-capacity hard drives to record the uncompressed raw image sequences with huge sizes, and the host computer is used to control the acquisition procedure and to process and retrieve the recorded image sequences.

The multi-sensor camera head outputs 6 simultaneously sampled color filter array (CFA) images of the Bayer raw image signal space at each shot instant of a sequential acquisition process, Figure 1.4. The CCD sensors capture the five horizontal views in landscape orientation with resolution 768×1024 per view. The view overlap of adjacent lenses is controlled within a small amount of 5 to 10 pixels. Therefore, the horizontal

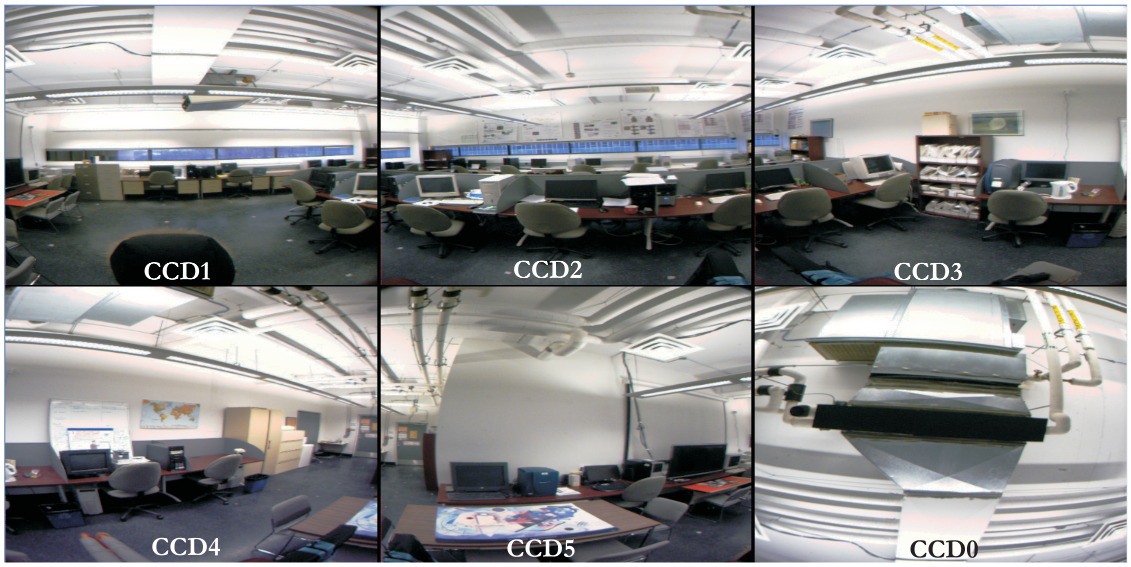


Figure 1.4: An example of demosaicked images from the Ladybug camera

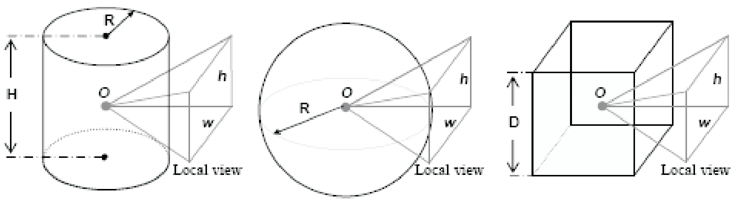
circumference is covered by approximately 3800 pixels. Totally the six CCD sensors capture approximately 4.7M effective pixels at each shot. The uncompressed streaming raw Bayer CFA image data is transmitted to the storage unit through a fiber-optic cable at an adjustable rate of 3.75, 7.5, 15 or 30 frames per second.

We assume that no explicit geometry data from the environment needs to be acquired in this stage. In section 2.1 image base rendering techniques and their classification depending on captured or estimated geometry data will be explained. Cubic-panorama image dataset has specific features that can be exploited efficiently for later storage and transmission stages. These features will be explained in section 1.2.

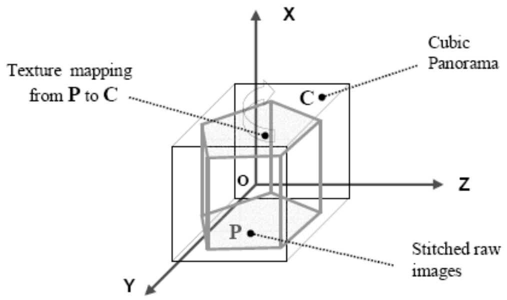
1.1.2 Image Sequence Preprocessing

Image sequence preprocessing performs operations such as image signal correction, noise filtering, image format conversion, image reorganization, etc [18, 19] on raw images to prepare images for the subsequent image processing stage.

As shown in Figure 1.5(a) there are three main categories of panoramas, namely: cylindrical [2, 20, 21], spherical [22, 23, 24, 25], and cubic panoramas [26, 27, 28]. The cylindrical representation can be easily mapped onto a 2D surface but has restrictions in rendering views when the viewer is facing either up or down. The spherical format has



(a) The cylindrical, the spherical, and the cubic panorama



(b) Texture mapping from six stitched raw images to a cubic panorama consisting of six side images

Figure 1.5: Three main categories of panoramas and cubic panorama generation

no bias regarding the viewing directions but has no uniform sampling density and results in problems at its north and south poles. Cubic panoramas on the other hand facilitate applying planar mapping tools, on side images, which are widely incorporated in practical applications and using them makes our algorithms most conveniently compatible with the existing literature including standard image and frame formats. Therefore cubic format for panoramas is adopted and applied throughout this work.

Basic image preprocessing operations include gamma correction, white balance, noise filtering, and image format conversion to change the format of raw image signals to the standard image format required by image dataset compression. Output images from sensors are usually Bayer color filter array (CFA) images with one color component at each sample position. The pixel values for the two other color components at each sample position are interpolated to obtain RGB format, where a demosaicking process is required. After R, G, and B components are interpolated, a group of six raw full RGB images can be used to create a basis panorama which consists of six side images $\mathcal{I}_{\mathcal{B},k,j}$ ($\forall j \in \{d, u, l, r, b, f\}$) where the subscripts represent the *down, up, left, right, back,*,

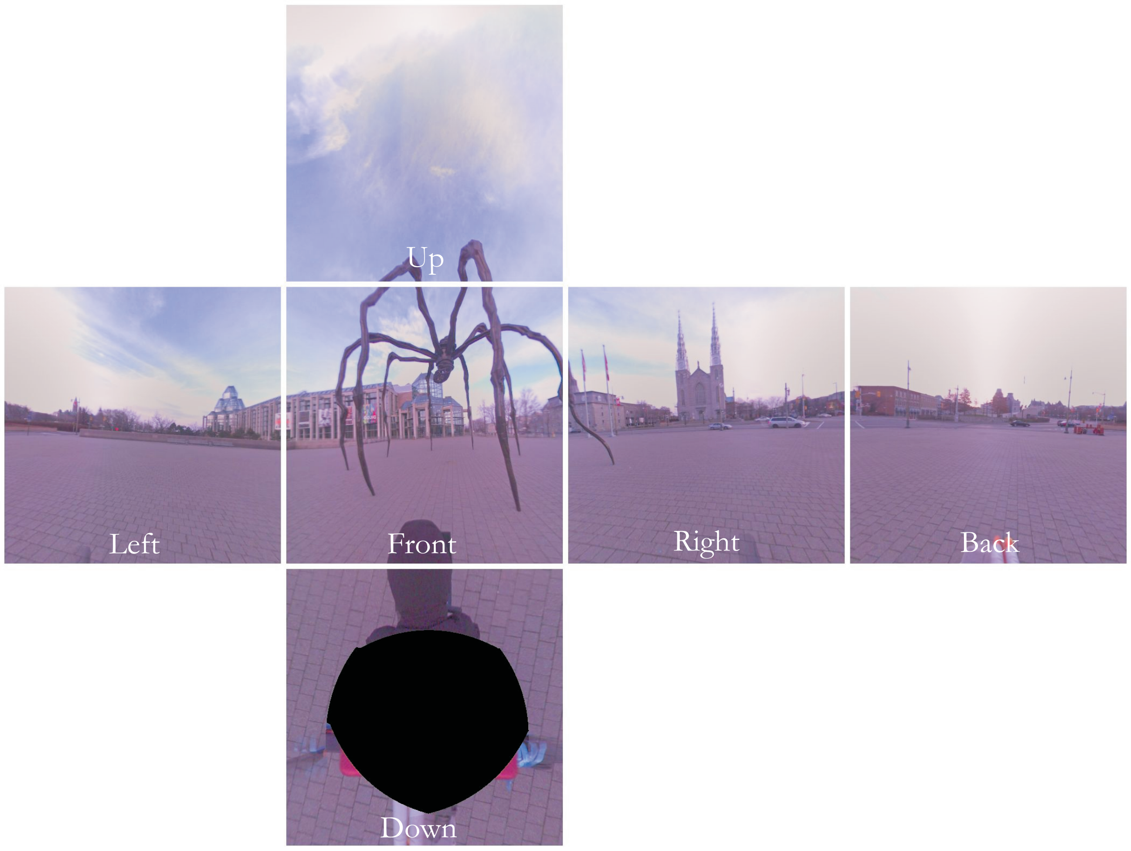


Figure 1.6: An example of a set of six resulting side images composing a cubic panorama unfolded in a flattened out pattern

and *front* side image respectively, see Figure 1.5(b) borrowed from [10]. The camera set is pre-calibrated and values are retained during the whole process of acquisition. The original overlap of 5 to 10 pixels between adjacent raw images is fused by using a standard blending technique in OpenGL [1] or the more sophisticated multi-perspective plane sweep technique [3]. Then by locating the geometric center of a cube **C** at the same common projection center of the group of the six raw images, the six faces of **C** are projected into the surface **P** of the stitched raw images and texture mapping is performed from raw images of resolution 768×1024 onto the surface of **C** (Figure 1.5(b)) with six side images of resolution 512×512 or 1024×1024. Each of these side views has 90 degrees of *FOV* (Field Of View) in both horizontal and vertical directions. There is a blind area in the bottom view of the cubic panorama due to the lack of a lens facing down in the camera set.

Finally in order to reduce the color data correlation of the tristimulus values, image transformation from RGB format to YUV 4:2:0 format is carried out. This will result in a more efficient representation of cubic-panorama image dataset by down-sampling the chroma components by a factor of 2 in each spatial direction. Here Y represents the luminance component and U and V are down-sampled chrominance components. This conversion doesn't affect the visual quality of the images as perceived by human eyes because the human visual system (HVS) is less sensitive to high frequency chrominance signals than luminance signals. This conversion is in line with the fact that the YUV format is taken as standard color format for video sequence processing and compression. A typical unfolded cubic-panorama is shown in Figure 1.6.

1.1.3 Basic Image Analysis

Basic image analysis may include image correspondence, camera calibration, image registration, depth-information extraction and so on. Resultant auxiliary data can be used to assist the subsequent image compression, image sequence rendering and virtual navigation.

Image correspondence across two or more images plays an important role in image processing and video compression applications. Region-based correspondence estimation techniques are applied to all image samples and provide a dense disparity map, but they tend to break down where there is lack of texture or a depth discontinuity occurs. On the other hand, feature based methods provide more reliable matching, but correspondence between sparse sets of image features is achieved with extra computational cost due to the extra computations for feature definition and detection. Hybrid methods combining the two approaches also exist in the literature. A review of image correspondence techniques is provided in section 2.4.

With camera calibration the intrinsic and extrinsic camera parameters may be determined and used in the subsequent compression stage. As we will see in chapters four and five, image correspondence, plus some camera calibration parameters such as focal length, play a key issue in video compression in general and especially in our research where images taken from different view-points are concatenated to form a video sequence. Since all required data is available offline, a large number of auxiliary data and side information can be calculated, and used later in real time when the rendered views are being transmitted. Finally, notice that since scene geometry is unknown to us, as we will see in section 2.1, our research is associated with the category of image based rendering with no geometry.

1.1.4 Image Dataset Compression

A cubic-panorama image dataset should be compressed efficiently to provide certain requirements of storage, random access, and complexity. These requirements for dataset compression, one of the major concerns of this thesis, are explained in section 1.3 in more detail.

Three major approaches include vector quantization (VQ) methods, wavelet-based methods, and methods based on standardized compression schemes that will be reviewed in section 2.2.

Since each panorama can be considered as six different views taken at each view location, parallels can be made with recently developed multi-view video coding (MVC) application, therefore a quick introduction is given in section 2.6.

As mentioned earlier, analysis of the cubic-panorama image datasets for the compression stage, as well as the transmission stage which will be explained later, constitute the main focus of our work.

1.1.5 Image Sequence Rendering

The aim of image sequence rendering is to synthesize novel-view images by taking primitive inputs from compressed image datasets and generating rendered image sequences based on IBR techniques. Each user has a limited field of view and needs part of the whole set of cube faces of each panorama to render a view at any time instant and at an arbitrary viewing location. Free view point rendering can be achieved by means of image re-sampling, mosaicking, segmentation, or more generally mapping, re-projection, interpolation, or more specifically warping, morphing, transformation and so on [8, 9, 29].

Some IBR techniques need environmental geometric data while there is a tendency to apply IBR without using geometric data mainly due to ease of image acquisition process. In this case, more basis images with greater image correlation are required to avoid image aliasing.

As explained in [30], the viewing space can be restrained in order to simplify the image representation. Another concern in this stage is to serve users with rendered images at a certain level of quality, providing smooth transition among neighbor views.

We assume that rendered views are generated based on user commands at the receiver by decoding the bitstream at the storage unit and applying linear mapping on part of the panorama faces which are within the field of view of the user according to its viewing direction. This process along with the next stage, i.e., image sequence transmission can also be considered as a video transcoding stage.

1.1.6 Image Sequence Transmission

As already mentioned, according to the user commands, part of the stored data must be decoded and converted to proper bitstream format to be transmitted through the communication channel. Standard video coding schemes can satisfy the requirements for compressing the rendered image sequence very well. This notion benefits from the compatibility that the standard provides with other third-party developed software or hardware.

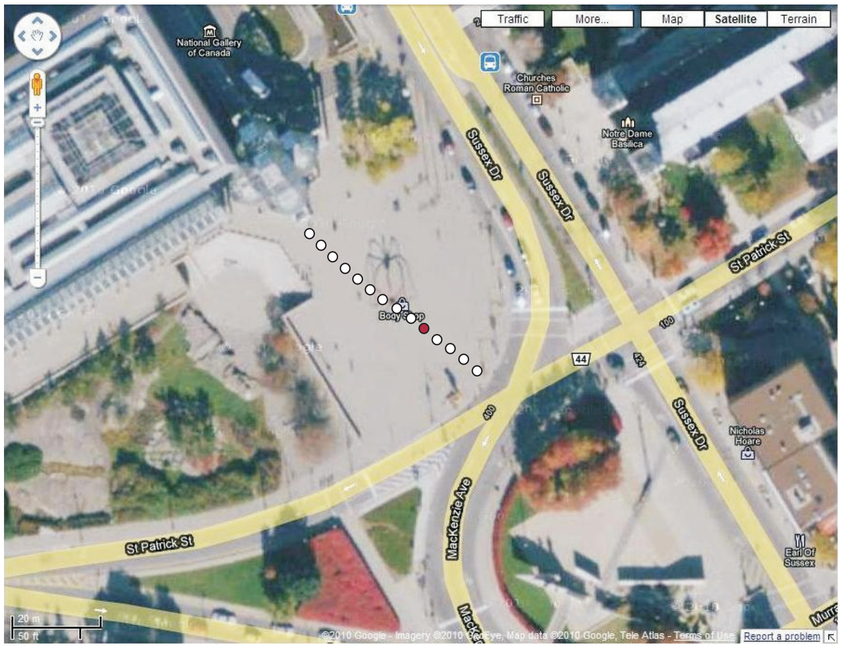
A quick review of image and video compression standards and techniques is provided in section 2.3. Notice that as we will see later, motion/disparity estimation and compensation play a very important role here.

In addition to the streaming output of rendered image sequences, some auxiliary information and augmented multimedia contents can be combined in the transmitter multiplexer. Packet techniques can be utilized to map the multiplexed source data to the transmission layer. Error resilience tools can also be applied for error-prone communications like streaming over Internet and wireless mobile communications, which is not studied in this thesis. A transmission buffer can also be designed to regulate the output bit-stream rate to fit the band-width limitations of the communications channels.

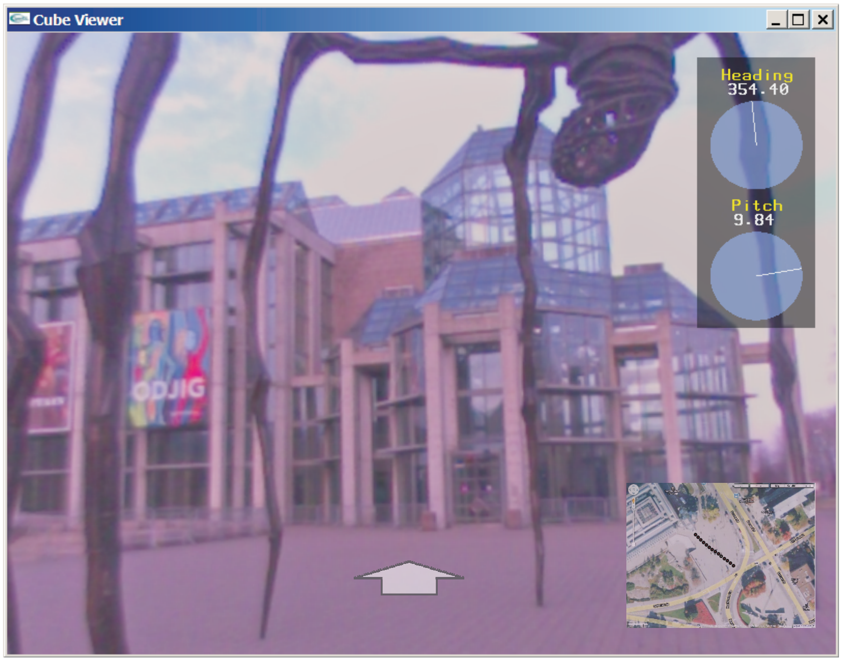
Since this is very close to the idea of video transcoding, this concept and different kinds of application scenarios is explained in section 2.5.

1.1.7 Virtual Environment Navigation

A user should be able to navigate in the environment freely by interactively requesting new data at the receiver side. Through proper interfaces, such as mouse, joystick or a keyboard, control signals are issued to the transmitter side in real-time. With each view point and view direction, data corresponding to the requested novel view is extracted from the storage bitstream, converted through a transcoding process and transmitted through the communication channel. Figure 1.7(a) shows an orientation-location map. A screen shot of the NAVIRE viewer [31] is shown in Figure 1.7(b).



(a) Graph of panorama locations



(b) Display interface of the cubic panorama viewer

Figure 1.7: Graph of panorama locations and the interface

1.2 Features of Image Datasets

In this section features of the image dataset used for virtual navigation are described from the view point of image dataset analysis for compression and transmission. These features, if exploited well, can enable us to meet the requirements for image dataset compression and image sequence transmission explained in the subsequent section.

1.2.1 Image Dataset as Samples of Simplified Plenoptic Function

Mathematically, IBR techniques can be investigated by making use of the plenoptic function [30] representing the intensity distribution of light rays in a three dimensional space. In its most general form, the plenoptic function is a seven-dimensional function which can be expressed as:

$$\mathcal{L}_P = f_P(P_x, P_y, P_z, \theta, \phi, \lambda, t), \tag{1.1}$$

representing the light ray intensity $\mathcal{L}_P$ of any wavelength λ, at any position (P_x, P_y, P_z), towards any direction (θ, ϕ) and at any time t in a three-dimensional environment space (Figure 1.8). An IBR technique can be considered as a method for determining the values of the plenoptic function corresponding to required orientations and viewpoints.

The plenoptic function in general corresponds to a complicated high-dimensional multi-variable data structure, which makes image dataset compression more difficult to deal with. As explained in [32] there are six commonly assumptions that can be used in order to restrain the viewing space and reduce the amount of data respectively. Some of those assumptions can be applied here and are explained below:

First, the wavelength variable can be removed from the plenoptic function by simplifying the wavelength dimension into three channels, i.e., the red, green, and blue. Subsequently, each color component can be investigated separately:

$$\mathcal{L}_P \to \mathcal{L}_{P6} = f_{P6}(P_x, P_y, P_z, \theta, \phi, t) \tag{1.2}$$

Second, the time variable t can be removed from the expression because the environment under discussion is assumed to be static.

$$\mathcal{L}_P \to \mathcal{L}_{P5} = f_{P5}(P_x, P_y, P_z, \theta, \phi) \tag{1.3}$$

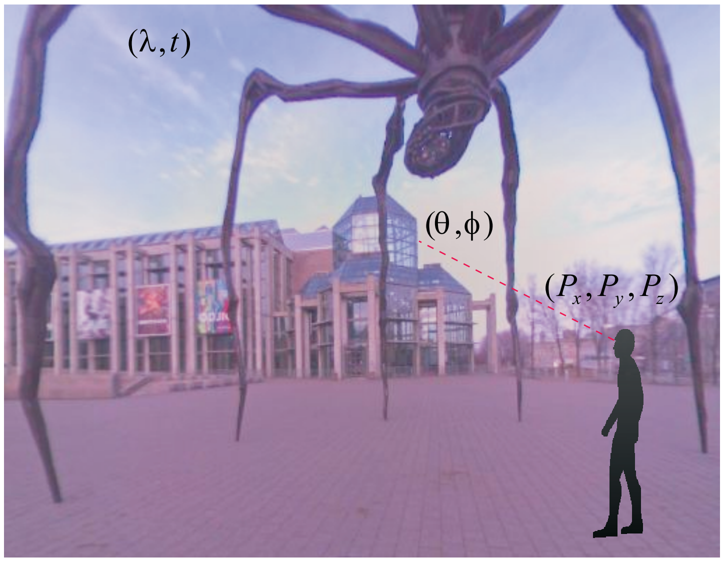


Figure 1.8: Generalized seven-dimensional plenoptic function representation of light ray distribution in a three-dimensional space

Instead of moving in the 3D space, the viewer is constrained to be on a surface, e.g. the ground plane. The plenoptic function can then be reduced one dimension as the viewers space location becomes 2D.

$$\mathcal{L}_P \to \mathcal{L}_{P4} = f_{P4}(P_x, P_y, \theta, \phi) \tag{1.4}$$

If the viewer moves along a certain trajectory, that is the viewer can move forward or backward along the trajectory, but s/he can not move off the trajectory, the function will reduce one dimension.

$$\mathcal{L}_P \to \mathcal{L}_{P3} = f_{P3}(\text{Trajectory}, \theta, \phi) \tag{1.5}$$

This 3D plenoptic function scenario will be studied in our work and the second last 4D plenoptic function will be considered as an extension scenario. Although restraining the viewing space reduces the amount of data in each image dataset, the navigation flexibility will be limited proportionately. Therefore we need to use other features of image datasets for reducing the size and efficient representation of image datasets.

1.2.2 Large Number of High-Resolution Basis Images

High sampling rates are required to avoid aliasing when capturing the environment data. In order to completely cover the represented environments and to be able to synthesize any novel views at arbitrary viewpoints in the authorized navigation space, even higher rates are required. The image dataset size is further increased in the case of representing a large environment with a large navigation space and enabling good quality zoom in image rendering based on high-resolution basis images. This results in a large number of high-resolution basis images. The increased amount of data can be regarded as the expense incurred to avoid the trouble of model building for photo-realistic rendering.

One can think of a minimal number of basis images by properly considering the sampling theory and spectral analysis but even with minimum number of basis images, the typical image datasets are still large in size and involve a huge amount of data. As a result compression of the image datasets is unavoidable in order to reduce the size of such image datasets and make the image-based virtual environment navigation plausible in practice.

1.2.3 High Local and Cross-Image Redundancies

Thinking of basis images as the samples of simplified plenoptic functions, there exist enormous redundancies in all dimensions. When the viewing path is restricted to a predefined trajectory, image datasets exhibit redundancies in 3 dimensions, while if the viewer trajectory is only restricted to a planar surface, these redundancies exist in 4 dimensions. When explaining the image based rendering techniques in section 2.1 other types of constraints and the resultant datasets with different dimensionality will be mentioned.

In other words, all the basis images in the image datasets are image samples of the same environment obtained from different viewing locations. This leads to tremendous cross-image redundancy and due to this feature, high compression ratios are expected. Note that the compression efficiency increases with the dimensionality of the plenoptic function representation.

1.2.4 Static Scene, Certain Pattern in Cross Image Displacement

In a generic video sequence, image motion is caused by movements of the objects captured in the frame sequence as well as the camera motion when capturing the frames, resulting in a complicated motion vector distribution. As mentioned before, in our study we assume that the scene is static, that means there will be no moving object in the environment to be captured. As a result motion, or better here to say displacement, is caused only by camera motion. Although the depth distribution of the environment generates some variations in the displacement distribution vector map, the dominant cross image displacement constitutes a certain pattern corresponding to the camera motion pattern. This cross-image displacement feature can be utilized efficiently to decrease computational complexity or equivalently improve the compression performance significantly.

1.2.5 Side Information

Image datasets are sometimes accompanied by certain types of side information, such as camera pose or location, global motion parameters, etc. Such information can be used for the purpose of exploiting image coherency and efficient compression of image datasets with extra considerations taken into account for embedding and perhaps compression of the side information.

1.3 Requirements for Image Dataset Compression

Requirements for image dataset compression can be summarized as high compression ratio, high quality, random access, selective decoding, view scalability, low decoding complexity, fast transcoding, memory constraints, and extendability to higher dimensions.

1.3.1 Compression versus Quality

High efficiency for image dataset compression is the main objective and fundamental requirement to make image-based virtual environment navigation available in practical applications. High-resolution basis images with high local and cross-image redundancies have the potential for achieving high compression efficiency. The other challenge, besides achieving high compression ratio, is to obtain high compression efficiency and simultaneously to satisfy other requirements for compressing the image dataset.

The coded basis images should be kept in uniform and high image quality as these images will be used as input images to synthesize novel-view images and may be displayed as still images for interactive virtual environment navigation. In conventional video sequence coding, bit-rate control techniques are applied to adjust the coding parameters generating coded images with different image qualities related to the image display order while in image-based virtual environment navigation the access to basis images and the generation of synthesized images are randomly controlled by user and there is no fixed order for displaying them.

The human visual system (HVS) also generally has a smaller distortion tolerance to still images than time-variant image sequences. As a result since both the basis images and the synthesized images can be required to display as still images at the navigation stage, basis images and synthesized images are expected to be of high quality.

Here the fact that all data is available offline prior to the compression stage is important to notice. It means an asymmetric codec scheme with higher encoding complexity and lower decoding complexity will be most useful. That is because all the encoding processing will be done offline where not a highly restricting time constraint exist. On the other hand, side information obtained in the encoding process can be exploited in real time when decoding is being performed to speed up the view generation process.

1.3.2 Random Access, Selective Decoding, and View Scalability

The generated bitstream at the storage unit needs to be highly randomly accessible. In conventional video coding schemes, random access is usually provided at some intra coded frames while subsequent frames are being predicted from the previous ones. Hence to reconstruct a frame, all preceding frames up to the intra coded frame should be decoded first. This may cause no serious harm because in standard video sequences, the order of encoding, decoding, and display are usually known in advance, but in image based virtual environment navigation no such predefined frame sequence order exists. The user has the ability to navigate the environment freely in an interactive manner. Not only s/he may start the navigation at any desired frame, but s/he might want to navigate in a direction which is not known in advance. Notice that the time variable t in standard video compression scenario is replaced with view location in a virtual navigation system. The above mentioned requirement is similar to the problem of reverse browsing of standard video sequence.

Another important issue is random access within each panoramic basis image. We know that the field of view of human eyes is limited, and a user at each time instant needs to access only part of all data available within a panoramic image basis. This requirement is called selective decoding and the cubic format of panoramic images can help in achieving this requirement down to the cube face level.

One other requirement is that different users might prefer different virtual walk-through speeds. In other words, starting navigation at a given frame and a certain direction, a user may choose normal speed, or wish to navigate with faster or slower speeds. The latter is associated with frame interpolation and novel view generation while the former is associated with view scalability. This is analogous to temporal scalability in standard video sequence coding.

1.3.3 Decoding Complexity, Transcoding, and Memory Constraints

In practice captured panoramic image datasets will be encoded once and stored in the storage unit. On the other hand, the resultant bitstream will be reused several times, accurately speaking, anytime a new user asks for access to stored data. As a result, the encoder is allowed to be of higher complexity while the decoder should be kept at lowest possible complexity, especially for some applications targeting real-time IBR without hardware assistance.

Since the user commands a new viewing location and view direction each time, the overall delay for decoding stored data and putting the generated views together to be passed through the channel should be small. Decoding an existing bitstream followed by encoding the video sequence with new coding parameters is called video transcoding which will be surveyed in section 2.5.

The decoding speed is affected by the random access mechanism, the selective-decoding ability as well as the decoding complexity. Usually fast decoding requires that the average decoding time should not exceed a prescribed value, e.g. 0.03 or 0.04 second for real-time video sequence display.

In comparison with the application at hand, if real-time image sequence display is required for the synthesized image sequences, it means that the basis image decoding time and the interactive image sequence rendering time put together should not exceed the prescribed time limit. Thus less time will be left for the decoding process itself compared to the conventional video decoding. As a result fast decoding becomes a more challenging requirement in the design of image dataset compression schemes.

Memory requirements is another issue which should be taken into account. The number of frames which are required to be stored in the server local memory to reconstruct some desired frame will introduce limitations both in terms of introduced delay and local memory requirements and this should be kept in mind when designing a group of pictures (GOP) format.

1.3.4 Extendability to Higher Dimensions

As mentioned before, the 3D plenoptic function scenario will be studied where a predefined trajectory is assumed to exist in the image acquisition process up to the rendering stage. The higher 4D scenario, where no such predefined trajectory exists in the 2D ground plane, or even a 5D dataset scenario can be envisioned where the user trajectory involves moving above the ground plane. Any compression scheme which is finally adopted for storage should be conveniently extendable to higher dimensional signal spaces. In the ultimate scenario the user won't be limited to a predefined trajectory and would be able to navigate any location within the ground plane for which captured basis images are available. All above mentioned compression, random access, decoding speed, etc., is better to be dealt with keeping this extendability along with other requirements in mind.

1.4 Requirements for Image Sequence Transmission

Requirements for image sequence transmission consist of efficient image dataset compression, high quality, users' variable spatial resolution capabilities, repeatability, memory constraints, and error resiliency.

1.4.1 Compression versus Quality

A video compression standard such as H.264 AVC can potentially be used to generate a virtual camera sequence by extracting required data from the bitstream to produce the real-time video camera, according to the user navigational commands.

Here the compression efficiency as well as the computational complexity are the most important requirements for transmission as it was in compression for storage case. Note that the image quality and compression efficiency of the virtual video sequence to be transmitted through the channel will be directly affected by the quality and efficiency of the stored bitstream in the storage unit. Much analysis of the dataset can be done offline and should be used to efficiently and quickly generate such rendered video sequence.

1.4.2 Different Spatial Capabilities

Since this is assumed to be a navigation system used by different users, one point that should be considered is that viewers may have different capabilities in terms of spatial resolution of the desired virtual camera. In this case, the decoded data should be processed quickly to be able to generate a virtual camera sequence of the desired spatial resolution, which might be of the similar quality as the reference stored image basis dataset, or of lower spatial resolution. Notice that a transcoding-like scheme is being applied, therefore we don't name this requirement spatial scalability. Any virtual camera sequence is unique and will be used by a single user and hence no spatial scalability is necessarily required at the bitstream level. But a single user's capability may vary from mobile phone displays up to high definition television, and this requirement can be taken into account.

1.4.3 Repeatability, Memory Constraints, and Error Resiliency

Unlike conventional video sequence standards, the user might want to repeat navigation of part of the environment more than once. Examples are when the user turns back and virtually walks in the opposite direction. In such cases, the information which has been already sent to the user might need to be exploited again and there should be no need to retransmit old data again. Addressing this requirement can save the bandwidth usage to a great extent.

Another example is when the user stops walking and starts looking around while standing virtually at the same spot. Here s/he even might look back after a short time again at the same direction as he was looking originally when s/he arrived to that spot. In such cases, a cache memory at the receiver can be devised to provide this flexibility. In order to meet the above requirement, local memory constraints at the receiver should be taken into account.

Another advantage of such an idea is to provide error resiliency. If for some reason part of data is lost through the communication channel, the existing information at the receiver side about neighbor viewing locations or the same location but in different view directions can be used to conceal the loss occurred in the transmission stage.

1.5 Summary and the Road Map

In this chapter an image based virtual environment navigation system and its components, from early raw image acquisition stage to the final environment navigation stage, was explained first. Each of these parts consist of many subsections and need separate care and research on their own and our goal was to provide the big picture for the reader of this introduction. Subsequently, features of the image datasets which will be exploited throughout this thesis were explained, followed by the major analysis requirements for the compression and transmission stages.

In chapter two a survey of different existing concepts and existing solutions in the area of image and video processing which are related to our work will be presented.

Here proposed approaches will be summarized without going through details. As already mentioned, image dataset analysis for storage and transmission constitutes the main concern of this thesis. We have noticed that before the compression stage, some preprocessing techniques are required and should be applied on the basis images, mainly because the captured images, although already preprocessed for improved quality, are not aligned in relation to each other. When a video sequence is created by concatenating a series of such images, in a manner which will be explained later, the transition among the viewpoints may not necessarily be smooth and a jitter effect might be visible.

In the third chapter we present a novel method to alleviate this problem, that is a reliable algorithm for cubic-panoramic image dataset alignment as the first contribution of our work. A quantitative measure for the panoramic image dataset alignment is introduced and exploited in order to test the performance of the proposed algorithm. The presented approach will be used for higher dimension, that is the case of navigation on a plane rather than a predefined trajectory.

In chapter four a comprehensive comparison is made between the existing compression schemes by running simulations on already preprocessed panoramic image datasets. First, the standardized compression scheme, namely H.264 advanced video coding (AVC), is applied to compress the captured image datasets, then results are compared with that of a state of the art wavelet-based implementation, namely Dirac codec [33] which is one of the main rivals of the standardized method. Different measures will be used for evaluation and comparison, namely Peak Signal to Noise Ratio (PSNR), Structural Similarity

(SSIM), and subjectively through image frames and constructed videos.

After this stage, the best scheme is adopted and, within this framework, the efficient prediction structure among frames, equivalently the cube faces, is sought by examining different possible frame types and structures. As it will be shown, different Group Of Picture (GOP) structures as well as different GOP sizes are tested and compared to each other in order to find the optimum solution. Then, the bitstream syntax and the prediction structure is extended to the higher dimension and eventually the final proposed bitstream syntax is presented there. Results up to this point will be presented in chapter four.

In chapter five novel methods are proposed, mainly in order to reduce the computational complexity in disparity estimation. Those ideas include the following:

- Finding global displacement parameters among panoramic images and applying the epipolar constraint in order to improve the computational complexity of the compression scheme.

- Exploiting a video transcoding like scheme to link the stored data to the communication channel and respond to the requests for new views issued by the remote user. Computational complexity and random access would be the main requirement at his stage.

- Reducing the computational complexity or data size, by exploiting the available information in case of pan, tilt, and zoom requests, i.e., user navigating within a single viewpoint.

Our results are obtained using image datasets captured on campus and in the city. Specifically, twelve image datasets with different characteristics were captured and are tested throughout this work. These images are captured indoor and outdoor, on a predefined trajectory and on a rectangular grid, on a single straight line and on a group of connected trajectories, on a linear path and on a semicircular path, etc. The outcome of our comprehensive image acquisition stage can be considered as a database for future testing and comparison purposes. Results obtained by testing the proposed approaches on the above-mentioned captured data are presented at the end of corresponding chapters respectively.

More details will be provided in the corresponding sections in chapters three, four, and five where the proposed methods and the corresponding results are presented respectively. Finally, chapter six is dedicated to the conclusion and future directions.

Chapter 2

Background

In this chapter we present the background, state-of-the-art, and the required definitions which may be referred to and used later throughout this thesis. The topics covered are varied and each subject is explained quite briefly in order to avoid the loss of the big picture. In the first section image-based rendering techniques are introduced as the starting point of our literature review. In the subsequent section, different approaches toward compression of the image datasets are summarized and compared briefly. Since standardized compression schemes are of interest in our research, the existing image and video compression [34, 35] standards are listed and reviewed in the subsequent section. Image matching techniques constitute an essential block in video processing, besides many other image and video processing applications, hence these techniques are categorized in the subsequent section. Then, different possible video transcoding scenarios, which will be referred to later in the transmission stage of the virtual navigation system, are summarized. Finally, multi-view video coding as a recent area of research, that can be considered as the extended version of the standard single-view video coding, is addressed through the introduction of a few existing research papers that are most relevant in one way or another. At the end of this chapter, the reader should have obtained the required knowledge regarding the context and the existing background on which our work is built up.

2.1 Image-Based Rendering Techniques

We start this section by providing the definition of image-based rendering. According to [32]:

IBR	Dim.	Observe Space	Rendering Method	Ref.
Panoramas	2D	fixed point	image warping and re-projection	[2] [22] [26]
Concentric mosaics	3D	circular area	interpolation of image slits based on basis image columns	[16]
Light field rendering	4D	rectangular area	re-sampling and assembling the basis images	[36]
Plenoptic stitching	4D	unobstructed area	stitching image columns derived from recoded omni-directional images	[37]

Table 2.1: Representative IBR techniques without geometric data

Image-Based Rendering Given a continuous plenoptic function that describes a scene, image-based rendering is a process of two stages - sampling and rendering. In the sampling stage, samples are taken from the plenoptic function for representation and storage. In the rendering stage, the continuous plenoptic function is reconstructed from the captured samples.

Although studying image-based rendering techniques on its own is not the major issue of concern in this thesis, basic acquaintance with the subject will help the reader to better place our work within the existing literature. As it was explained it the previous chapter, the plenoptic function is originally a function of seven continuous variables. However, as mentioned in [32], up to 6 assumptions might be made to restrain the viewing space. Restraining the viewing space is one way to make the image-based rendering data manageable. Introducing source descriptors is an alternative way and includes *scene geometry*, the *texture map*, the *surface reflection model*, etc. These latter methods can reduce the overall number of necessary light rays to be captured by using the correspondence among the light rays. Here we will not go through more details regarding the source descriptors, but we will rather summarize the 6 assumptions which can be used to increasingly restrain the viewing space:

Assumption 1 Wavelength dimension may be simplified into three channels, i.e., the red, green and blue channels;

Assumption 2 The air may be assumed to be transparent, in other words, we won't need to record the radiance of a light ray at different positions along its path, as they are all the same;

Assumption 3 The scene may be static, hence the time dimension can be dropped;

Assumption 4 The viewer position may be constrained to lie on a surface;

Assumption 5 The viewer may move along a certain trajectory;

Assumption 6 The viewer may have a fixed position.

As it was shown in the previous chapter, we used assumptions 1, 3, 4, and/or 5 and ended up in the specific 3D or 4D representations that we will be working on throughout this thesis. A different set of viewing space constraints will result in different representations, including *light field* [36], *lumigraph* [38], and *concentric mosaics (CM)*, to name a few.

2.1.1 Rendering With No Geometry

The above-mentioned IBR representations fall into the category of rendering with no geometry [39]. This is the category of interest for us, since it is the most practical assumption that can be considered so long as the ease of the image acquisition stage in the virtual navigation system is emphasized. As shown in [10], representative IBR techniques without geometry data are summarized in Table 2.1.
The two other rendering categories include rendering with implicit geometry and rendering with explicit geometry:

2.1.2 Rendering With Implicit Geometry

This is a class of techniques that rely on positional correspondences across a small number of images in rendering new views. The term *implicit* is used to emphasize the fact that geometry is not directly available; rather 3D information is computed only using the usual projection calculation. New views are computed based on direct manipulation of these positional correspondences, which are usually point features. They include *view interpolation*, *view morphing* [40], and *transfer methods*.

2.1.3 Rendering With Explicit Geometry

In this class of techniques, the representation has direct 3D information encoded in it, either in the form of depth along known lines of sight, or 3D coordinates. These techniques include *3D warping*, *layered depth image rendering* [41], and *view-dependent texture mapping*.

Sampling and *Compression* constitutes the two remaining areas of significance in the area of image-based rendering, beside the above-mentioned topic of *Representation*. The main question in *Sampling* is the following: "How many samples do we need for anti-aliasing reconstruction?". However, *Compression* is the topic of interest from our point of view and will be looked into in the following section.

2.2 Image Dataset Compression

In general, there are two major methods to reduce the data size of image-based representations. The first one is to reduce the dimensionality, often by limiting viewpoints or sacrificing some realism, as it was explained in the previous sections. The second approach, which is more classical, is to exploit the high correlation, or alternatively redundancy, within the specific representation using waveform coding or other model-based techniques. The second approach itself can be classified into three broad categories [39], namely:

1. Pixel-based methods;
2. Disparity compensation/prediction (DCP) methods;
3. Model-based/model-aided methods.

In pixel-based methods, the correlation among adjacent image pixels is exploited using traditional techniques such as vector quantization (VQ) and transform coding. On the other hand, wavelet-based schemes and standardized compression schemes are two examples of the DCP methods and will be described below. Finally, model-based approaches, the area which is out of the scope of this thesis, recover the geometry of the objects or scene in coding the observed images. In this approach, the models and other information such as prediction residuals or view-dependent texture maps are then encoded.

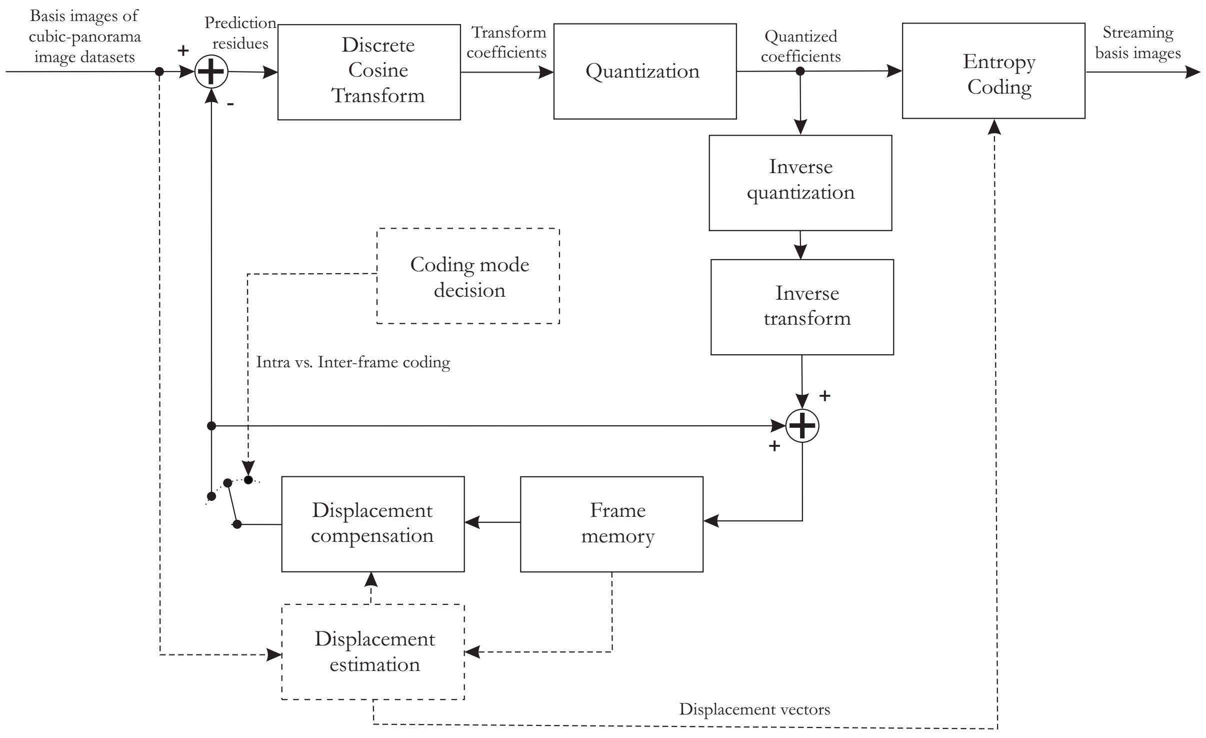


Figure 2.1: Block diagram of a standardized compression scheme

2.2.1 Vector Quantization Based Schemes

The vector quantization based schemes [36, 42], like other pixel-based methods, utilize the correlation only among adjacent image pixels and very little geometry information is used. VQ based schemes are easy to implement and in some cases the random access problem is usually less complicated. However, their compression performance is limited compared to the other approaches.

2.2.2 Standardized Compression Schemes

In the DCP methods, scene geometry is utilized implicitly by exploiting the disparity among neighbor frames/images, to achieve better compression performance. The term *Disparity* refers to the relative displacement of pixels in images taken in adjacent physical locations. Disparity, which has been utilized in coding stereoscopic and multi-view images, is analogous to the concept of *motion* of objects in classical standardized video coding.

Standardized compression schemes [43, 44] are examples of the so called DCP meth-

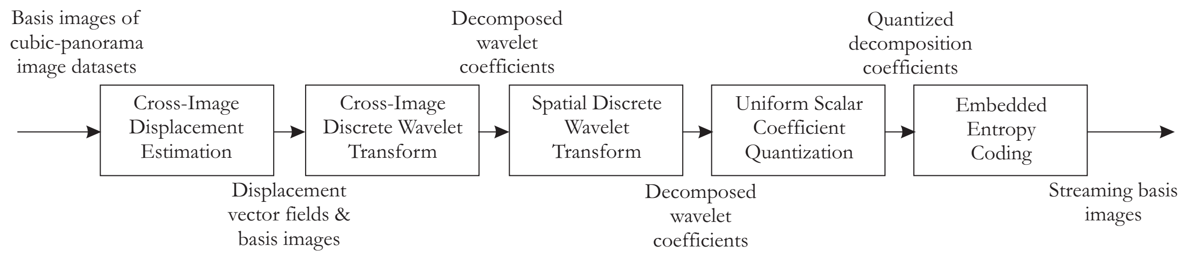

Basis images of cubic-panorama image datasets
Cross-Image Displacement Estimation
Displacement vector fields & basis images
Cross-Image Discrete Wavelet Transform
Decomposed wavelet coefficients
Spatial Discrete Wavelet Transform
Decomposed wavelet coefficients
Uniform Scalar Coefficient Quantization
Quantized decomposition coefficients
Embedded Entropy Coding
Streaming basis images

Figure 2.2: Block diagram of a typical wavelet-based scheme

ods. In Figure 2.1 we have shown the block diagram of a standardized video compression scheme, which has been adopted to be applicable for compression of our image-based environment representations. The feedback loop is responsible for exploiting the aforementioned disparities among adjacent images in order to facilitate achieving a high compression ratio. The discrete cosine transform (DCT) module utilizes redundancies among adjacent pixels inside the basis image. The Quantization block [45] exploits another type of inherent redundancies by properly quantizing the amplitudes of the DCT coefficients. Finally, the function of the entropy coding block is to exploit entropy redundancies, and these building blocks altogether are aimed to produce an output bitstream which is much smaller in size as compared to the input image dataset. The concept of motion estimation/compensation is replaced with disparity estimation/compensation, Motion Vector (MV) with Disparity Vector (DV), frame with basis image, and so on.

The standardized video compression scheme has been called block based hybrid coding in the literature because motion or disparity among neighbor frames/images are estimated on image blocks (usually of size 4×4 or 8×8) and also because a combination of the DCT and motion/disparity estimation are used to exploit both inter and intra frame redundancies in realizing an efficient video or image dataset compression algorithm.

2.2.3 Wavelet-Based Schemes

Wavelet-based schemes [46, 47] are alternative DCP based solutions to the problem of the image dataset compression. Compared to the standardized methods, two major differences can be pointed out: First, the redundancy among neighbor images is exploited using discrete wavelet transform (DWT), which is called Motion Compensated Temporal Filtering (MCTF) in case of conventional video coding, as compared to the block based disparity estimation which works based on discrete cosine transform. Second, no explicit

Compression Scheme	Compression Efficiency	Structure Complexity	Scheme Characteristics	References
VQ-based scheme	Low	Low	Simple fast decoding, easy random access and selective decoding	[16], [36], [42], [48]
Standardized base scheme	High	Moderate	well-developed, widely-applied coding techniques	[43], [44], [49], [50], [51], [52]
Wavelet-transform-based scheme	High	High	temporal, spatial, and quality scalabilities	[46], [47], [53], [54]

Table 2.2: Comparison of the coding schemes for image dataset compression used for image-based virtual environment navigation

feedback loop is used as redundancies are exploited once and among a predefined number of pictures (basis images) as compared to the case for the block based hybrid coding method where disparities are estimated among two usually adjacent neighbor frames and the procedure is repeated each time a new frame is to be encoded.

As compared to the block based hybrid coding scheme, the DCT block is replaced with spatial DWT block and the feedback loop is replaced with the cross-image displacement estimation and cross-image DWT (see Figure 2.2). An interesting feature of the wavelet-based scheme is the video scalability characteristic, most importantly spatial scalability and temporal scalability (the latter can be called across-viewpoint scalability for the modified case). Therefore, *embedded* entropy coding has become plausible which means partial decoding of the output bitstream, depending on the user requirements and channel capacity, will result in low resolution versions of the original data.

However, since more than two frames are used in the cross-image disparity estimation, the cross-image displacement estimation and the cross-image DWT will cause the problem of unconnected pixels which should be addressed properly, and existing techniques known as lifted motion-compensated wavelet transform are potentially applicable. Nevertheless, unlike standardized compression schemes, this method has not been standardized yet; instead, different solutions and research papers each responding to specific requirements have been proposed. The compression efficiency and structure complexity of the three above mentioned method are compared in brief in Table 2.2 [10] along with the scheme characteristics and relevant references.

At the end of this section, notice than among the three type of panoramas, i.e. cubic, cylindrical, and spherical, in the latter case, applying the harmonic transforms and convolutions takes a different form due to different sampling nature of the spherical-panoramic images. Our concern in this thesis is the cubic format. For a study on spherical harmonic transforms and convolutions the reader may refer to [23].

2.3 Image and Video Compression Standards

As it has been shown so far, many image dataset compression methods in the literature benefit from existing standardized video coding structures. In this section a quick review of the image and video compression standards from JPEG to H.264/MPEG-4 AVC [55] as well as the emerging standard is provided:

2.3.1 JPEG and JPEG2000

Standardized coding of still pictures was first introduced in mid 1980s. A collaborative work between International Telecommunication Union (ITU-T) and International Standards Organization (ISO) resulted in the well known JPEG standard [56]. It is a DCT based algorithm. After the quantization stage, coefficients are variable length coded using Huffman method for entropy coding. The standard has become very flexible. As an example, the encoder is parameterizable hence the desired rate distortion trade-offs can be determined based on the application. There are applications in which the JPEG standard is used in video coding, which are called motion JPEG. However, temporal redundancy across frames is not exploited.

At the turn of the new millennium another standard called JPEG2000 [57, 58, 59] was introduced in response to increasing demands for multimedia, internet, and a variety of digital imagery standards. It is a wavelet based standard providing better compression ratio and scalability features. Three major wavelet-based image coding techniques developed in the literature over time are Embedded Zero-tree Wavelet (EZW) encoding [60], Set Partitioning in Hierarchical Trees (SPIHT) [61], and Embedded Block Coding with Optimal Truncation (EBCOT) [62], the latter being used in JPEG2000. At last, it is worth to mention that, in a manner similar to motion JPEG, motion JPEG2000 has also been used in video applications.

2.3.2 H.261

H.261 [63] is the first widely used video codec initially made for video conferencing applications. The concept of hybrid codec uses DCT transformed coefficient to exploit spatial redundancy along with temporal prediction to exploit temporal redundancy of data to be encoded. Residual error is DCT-transformed, quantized, and entropy coded by a Variable Length Coding (VLC) module. Each quantized block is zigzag scanned before entropy coding (Huffman coding is used). A loop filter is inserted in the prediction loop in order to reduce the blocking effect. A buffer is used for rate control before the encoded data is transmitted through the channel. Different macro-block types are defined based on motion compensation or no motion compensation decision and inter or intra mode decision. It is interesting to notice that motion estimation module was originally optional in this standard.

2.3.3 MPEG-1

MPEG-1 [64] was introduced in response to the industry need for an efficient way of storing visual information on storage media other than the conventional analogue Video Cassette Recorders (VCR). The concept of bidirectional prediction and B pictures was introduced for the first time to provide higher compression ratio, hence preprocessing and picture reordering became necessary. Since B pictures are disposable by nature, temporal scalability and fast play was achieved. Motion estimation was considered more seriously, especially motion estimation with half a pixel precision. Rate control and adaptive quantization among I, P, and B frames were addressed.

2.3.4 MPEG-2

MPEG-2 [65] was initially introduced for broadcasting purposes. Therefore different channel and decoder capabilities should have been met. The concept of level and profile was introduced to categorize different requirements. Interlaced video coding mode was introduced in addition to the conventional progressive coding mode which resulted in a new combination of field/frame prediction. Due to different users and channel capacities, the concept of scalability was introduced including temporal, spatial, SNR and hybrid scalability.

2.3.5 H.263

In H.263 [66] the goal was to achieve low bit rate communications due to huge amount of data and channel bandwidth limitations at the time. Coefficients and motion vectors were coded differently. Some optional modes were introduced in order to increase the compression ratio. Advanced motion estimation/compensation was applied including unrestricted motion vector, four motion vectors per macro-block, overlapped motion compensation all appreciating importance of motion compensation more seriously. Also this scheme added the de-blocking filter [67] and motion estimation/compensation with spatial transforms. B pictures were treated differently resulting in the concept of PB frames. Advanced variable length coding was used including arithmetic coding, reversible variable length coding, re-synchronization markers, and advanced intra/inter VLC. Protection against error was also important, including forward error correction, back channel, data partitioning, error detection by post-processing, intra and inter-frame error concealment, loss concealment, and selection of best estimated motion vector. Scalability was extended to multi-layers. Buffer regulation was also addressed.

2.3.6 H.264/MPEG-4 AVC

H.264/MPEG-4 AVC [68, 69, 70, 71, 72, 73] was considered as the solution to future needs to cover long term requirements of video compression. The rate distortion performance was much higher than the existing standards and about 50 percent bit saving were achieved as compared to H.263 standard. Major differences are listed here:

- Introduction of integer transform;
- Much larger number of motion compensation block sizes;
- Higher precision of motion estimation;
- Multiple reference frames;
- Introduction of SI/SP frames [74, 75];
- De-blocking filter as part of core H.264;
- Hierarchical B Pictures [76].

For ITU-T recommendation for H.264 see [77], for H.264/AVC reference software manual see [78], and for text description of joint model reference encoding methods see [79].

2.3.7 HEVC

At the time of submitting this thesis, we noticed that the new generation of video compression standard known as High Efficiency Video Coding (HEVC) [80] is being prepared as the newest video coding standard of the ITU-T Video Coding Experts Group and the ISO/IEC Moving Picture Experts Group. An increasing diversity of services, the growing popularity of HD video, and the emergence of beyond-HD formats (e.g., $4k \times 2k$ or $8k \times 4k$ resolution) are creating even stronger needs for coding efficiency superior to H.264/MPEG-4 AVC's capabilities. The need is even stronger when higher resolution is accompanied by stereo or multiview capture and display. Moreover, the traffic caused by video applications targeting mobile devices and tablet PCs, as well as the transmission needs for video-on-demand services, are imposing severe challenges on today's networks. An increased desire for higher quality and resolutions is also arising in mobile applications.

HEVC has been designed to address essentially all existing applications of H.264/MPEG-4 AVC and to particularly focus on two key issues: increased video resolution and increased use of parallel processing architectures. The first edition of the HEVC standard is expected to be finalized in January 2013. Additional work is planned to extend the standard to support several additional application scenarios, including extended-range uses with enhanced precision and color format support, scalable video coding, and 3D/stereo/multiview video coding. In ISO/IEC, the HEVC standard will become MPEG-H Part 2 (ISO/IEC 23008-2) and in ITU-T it is likely to become ITU-T Recommendation H.265.

2.4 Image Matching Techniques

As it was shown before, disparity estimation is an integral part in any image dataset compression scheme beside many other applications such as computational stereo [81]. Computational stereo refers to the problem of determining three-dimensional structure of a scene from two or more images taken from distinct viewpoints and has many applications, e.g. in turning images into 3D models [82]. In such a problem, matching pixels in one image with their corresponding pixels in the other image plays an important role because disparities can reveal useful information about the depth of the real world points matched between the two images. As we will see later, for static scenes the search range can be reduced to one dimension or equivalently, the search can be limited on a scan-line called the *epipolar* line. Obtaining disparities among images implies information about the depth and geometry of the scene which is useful in computational stereo especially in order to reduce the computational complexity. In an analogous way, if match points are properly found, high compression ratios can be achieved in an image dataset compression application.

All of the existing correspondence methods use specific constraints when attempting to match pixels in one image with their corresponding pixels in the other image. Constraints defined on a small number of pixels surrounding a pixel of interest are referred to as *local* constraints and similarly constraints defined on some scan-lines or on the entire image are loosely referred to as *global* constraints. Local and global constraints in most of the applications usually rely on two views, however there are applications where more than two views are used for finding correspondences.

Local methods can be very efficient, but they are sensitive to locally ambiguous regions in images, e.g. occlusion regions or regions with uniform texture. On the other hand, global methods can be less sensitive to these problems since global constraints provide additional support for regions difficult to match locally. However, the global methods are computationally more expensive. According to [81] local methods in short include the following:

Block Matching Search for maximum match score or minimum error over small region, typically using variants of cross-correlation or robust rank metrics.

Gradient-Based Optimization Minimize a functional, typically the sum of squared differences, over a small region.

Feature matching Match dependable features rather than intensities themselves.

On the other hand, global methods can be categorized as below:

Dynamic Programming Determine the disparity surface for a scan-line as the best path between two sequences of ordered primitives.

Intrinsic Curves Map epipolar scan-lines to intrinsic curve space to convert the search problem to a nearest-neighbors lookup problem.

Graph Cuts Determine the disparity surface as the minimum cut of the maximum flow in a graph.

Nonlinear Diffusion Aggregate support by applying a local diffusion process.

Belief Propagation Solve for disparities via message passing in a belief network.

Correspondence Methods Deform a model of the scene based on an objective function.

We will hear more about local methods later in this thesis, especially block matching and feature matching approaches. The block matching method is adopted in standardized compression schemes such as H.264. These methods seek to estimate disparity at a point in one image by comparing a small region about that point with a series of small regions extracted from the other image. Usually, three classes of metrics are used for block matching: correlation, intensity differences, and rank metrics.

Normalized Cross Correlation (NCC) is the standard statistical method for determining similarity. Its normalization, both in the mean and variance, makes it relatively insensitive to radiometric gain and bias.

The Sum of Squared Differences (SSD) metric is computationally simpler than cross correlation, and it can be normalized as well:

$$\mathrm{SSD}(d_1, d_2) = \sum_{n_1,n_2} \left(\mathcal{I}_{\mathcal{B},c}[n_1, n_2] - \mathcal{I}_{\mathcal{B},r}[n_1 + d_1, n_2 + d_2]\right)^2 \tag{2.1}$$

In addition to NCC and SSD, many variations of each with different normalization schemes have been used. One popular example is the Sum of Absolute Differences (SAD), which is often used for computational efficiency:

$$\mathrm{SAD}(d_1, d_2) = \sum_{n_1,n_2} \left|\mathcal{I}_{\mathcal{B},c}[n_1, n_2] - \mathcal{I}_{\mathcal{B},r}[n_1 + d_1, n_2 + d_2]\right| \tag{2.2}$$

In addition to the above-mentioned correspondence methods, methods for *occlusion* and *real-time implementations* are important issues that one will face in computational stereo. Much of the stereo research in the last decade has focused on detecting and measuring occlusion regions in stereo imagery and recovering accurate depth estimates for these regions. The occlusion problem in stereovision refers to the fact that some points in the scene are visible to one camera but not the other, due to the scene and camera geometries. In [81] three classes of algorithms for handling occlusion are reviewed. They

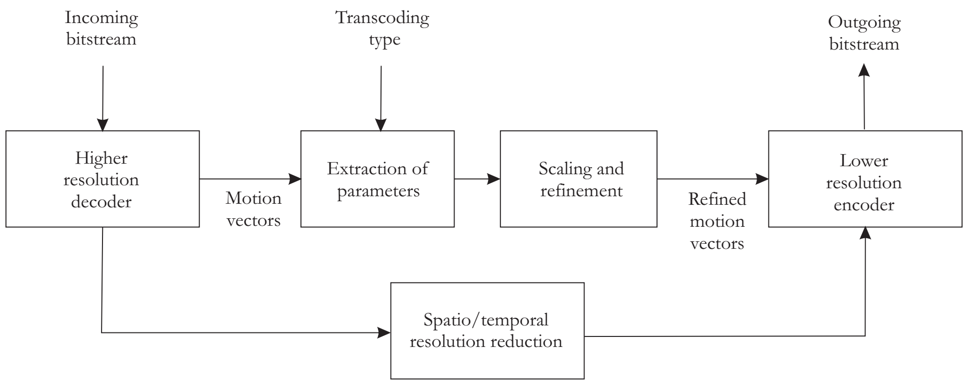


Figure 2.3: Block diagram of a heterogeneous video transcoder

include: methods that detect occlusion, methods that reduce sensitivity to occlusion, and methods that model the occlusion geometry.

Real-time stereo implementation is also another issue of importance which is covered in [81]. It is interesting to notice that although several approaches have been proposed for the problem of disparity estimation in the literature, simple methods working based on block matching are still adopted in most of the real time implementations due to their minimal computational complexity.

2.5 Video Transcoding

Video transcoding, Figure 2.3, is the operation of converting a video from one format into another format. A format is defined by characteristics such as the bit rate, frame rate, spatial resolution, coding syntax, and content. One of the earliest application of transcoding is to adapt the bit rate of a pre-compressed video stream to a channel bandwidth. For example a TV program may be originally compressed at a high bit rate for studio applications, but later needs to be transmitted over a channel at a much lower rate. As the reader might have figured out, there is an analogy here again between conventional video transcoding and similar concepts in image dataset compression and transmission. As it is mentioned in [83] video transcoding techniques can be categorized into the following types:

Bitrate Transcoding Three transcoding architectures for the bit-rate transcoding are reviewed in [83]: open-loop transcoder, cascade pixel-domain transcoders, and DCT-domain transcoders. The open-loop transcoders are computationally efficient. However, they suffer from the drift problem. The two other approaches have been proposed to alleviate the problem.

Spatial and Temporal Transcoding The heterogeneity of communication networks and network access terminals often demand the conversion of compressed video not only in the bit rates, but also in the spatial/temporal resolutions. One of the challenging tasks in spatial/temporal transcoding is how to efficiently reestimate or map the target motion vectors from the input motion vectors.

Standards Transcoding In many application, video coded in one coding standard may need to be converted to another standard besides the changes in bit rate and resolution. In [83] two examples are made to illustrate how the information obtained from the input video sequence may be used to help the standards transcoding process. The first one is MPEG-2 to MPEG-4 simple profile transcoding and the other is MPEG-2 to MPEG-4 advanced simple profile transcoding. As we will see later, this type of video transcoding will be useful in image dataset compression and transmission, specifically when data need to be extracted from the storage unit and sent to the user through the communication channel.

Transcoding Quality Optimization In video transcoding many useful statistics such as the quantization step size, coding modes, coded bits of each macroblock and frame, and motion vectors, can be easily obtained from the input video bitstream to help the second pass encoding. Therefore, it is possible for the transcoder to achieve better video quality than the direct one-pass encoding using the original source. According to [83] technologies related to video quality optimization fit into three categories, namely: requantization, rate control, and mode decision.

Information Insertion Transcoding Generally speaking, any operation that changes the content of a compressed video stream may be regarded as video transcoding. Two information insertion examples are discussed in [83]. First, logo or watermark insertion where video watermarks and company logos are inserted into the compressed bitstream for copyright protection. Second, error-resilience transcoding where a video transcoder can be placed in a network node connected to a high-loss network to insert error resilience features into the video bitstream to achieve reliable video transmission over wireless channels.

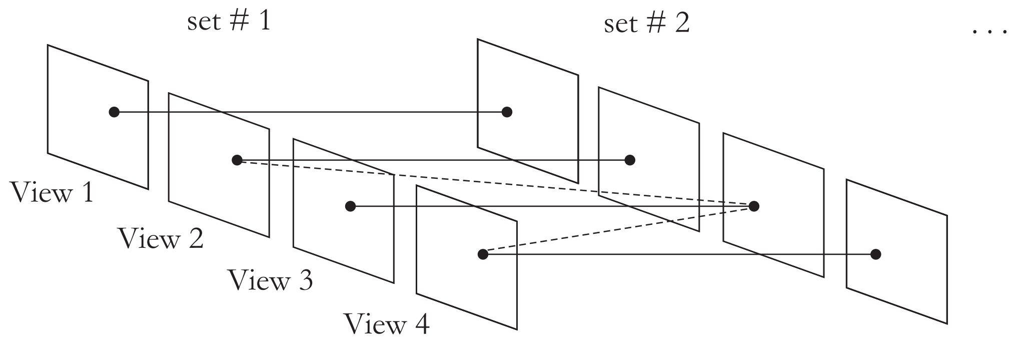


Figure 2.4: Multiview video captured by synchronized cameras from different viewpoints

2.6 Multi-view Video Coding

Multiview video captured by synchronized cameras, from different viewpoints comprises rich 3D information of a scene and is widely used as a signal of new types of visual media such as 3DTV and Free viewpoint TV (FTV). However, it results in a tremendous amount of data depending on the number of cameras. Thus, efficient compression of multiview video is a key enabling factor for 3DTV and FTV applications. Multi-view Video Coding (MVC) explores the interview statistical dependencies in addition to the temporal ones. A major research topic in this direction is the prediction structure for example, combined inter-view/temporal predictions are developed by several worldwide groups. The MVC was released by the Joint Video Team (JVT) in 2008 as an extension of H.264/AVC (Amendment 4).

Hur *et al.* in their paper [84] explain how to compensate the illumination mismatches of the cameras. Using a mean removed SAD together with the conventional SAD during the disparity vector estimation for a macroblock, illumination can be well compensated with a small increase of the encoding complexity.

Merkle *et al.* present multiview video coding with optimized temporal and inter-view prediction structures for efficient compression based on the H.264/AVC video coding standard in their paper [85]. The idea is to exploit the statistical dependencies from both temporal and inter-view reference pictures for motion compensated prediction. The result shows that the prediction with temporal reference picture is very efficient, whereas for about 20% blocks in a picture the prediction with reference pictures from adjacent views is more efficient.

San *et al.* present in their paper [86] how incorporation of knowledge about scene geometry can improve disparity estimation and coding efficiency of MVC. Principles of camera projection and epipolar geometry are efficiently exploited to improve inter-view prediction.

A scalable MVC coding structure especially suited for interactive real-time viewing is presented in the paper [87] by Kurutepe *et al.*, where the base layer videos of all views are compressed with high ratio, any two views in the enhancement layer are coded by similar method, and the view selection is user driven. The enhancement layer provides higher quality under a bit allocation policy for the base layer and for the enhancement layer.

Analogies can be made between MVC and the specific type of image dataset which is under study in this thesis. For example, at first sight, the six faces of a cubic panorama can be considered virtually as captured views by some synchronized cameras. However, since these virtual cameras are facing towards different directions, not much correlation exist between any pair of such views and they do not share much visual information, shown as dotted lines in Figure 2.4. But so long as the extended scenario of the 4D signal, i.e., panoramic images captured on a rectangular grid, is considered, similarities can be found with the idea of MVC already explained in this section. Notice that, however, there exists another major difference, i.e., in MVC we are dealing with different views and different time instances, while in our captured image datasets, no explicit time dimension exists and we are always dealing with different views in either direction. Nevertheless, we found it useful to finish our background chapter by introducing the idea of MVC and a few relevant research papers.

2.7 Summary

To sum up briefly, in this chapter we first introduced image based rendering and reviewed the existing IBR techniques. Subsequently, existing image dataset compression schemes were reviewed. Then, an introduction to image and video compression standards and image matching techniques were presented. Afterwards, existing video transcoding paradigms were explained followed by a quick survey of relevant issues in multiview video coding. In the following chapters we will propose a number of ideas addressing image dataset analysis and preprocessing of image datasets for storage and transmission.

Chapter 3

Image Dataset Acquisition, Analysis, and Alignment

Panoramic images are a new type of visual data that provide many new possibilities as compared to the classic planar images. Tele-presence and virtual navigation are examples of such interesting applications. In order to enable smooth navigation across different view points, we propose a method for aligning cubic-panorama image datasets by using the concept of *epipolar* geometry. Unlike the existing method which is limited and applicable to only one pair of panoramas, our approach is applicable on image datasets with larger number of panoramic images. It will be shown that using our method no extra numerical estimation is required. We also introduce a measure in order to enable objective evaluations.

3.1 Introduction

Due to advancements in communication technologies, panoramic image datasets are going to be used widely in upcoming applications such as tele-presence and we expect that this will open up a new venue in the area of multimedia signal processing and communication in the near future [1]. As explained before, in this work real images are preferred to computer generated ones due to the lower costs and further realism they can provide. This work is part of the *NAVIRE* project at the University of Ottawa which aims at developing the necessary technology to allow a user to virtually walk through in an image-based representation of a remote environment.

For the processing and storage of panoramic images, several representations have been proposed, including cylindrical and spherical. This work focuses on a cubic representation of the image panoramas. The idea has been motivated and used by Bradley *et al.* [1] and Apple in their QuickTime panorama viewer [2]. Cubic panoramas offer numerous advantages that make them attractive for our study: storage and rendering are facilitated, they can be equivalently handled as set of perspective images, there exist intrinsic relationships between the faces, implicit calibration information is available, etc.

Image dataset analysis and alignment are important issues and will be addressed in this chapter. Image dataset alignment is required because usually the adjacent cubic panoramas in the available image datasets, captured either indoor or outdoor, are not aligned mainly due to the distance between the image capturing positions and camera displacement, as compared to the conventional video where usually more than 15 frames per second are captured. An alignment procedure will be required regardless of how much attention has been taken into consideration by the camera operator and due to unknown extrinsic camera parameters; if the original captured image dataset is used in virtual navigation, some undesired effects will be perceived by the user of the navigation system in real time. As a result, in a practical setting, extrinsic camera parameters should be estimated and used in a reliable manner in order to align the captured panoramic images and facilitate seamless virtual navigation within the real world environment.

Our goal is to apply an image rectification algorithm on datasets with a large number of panoramic images, i.e. more than only a pair of them, in order to enable seamless virtual navigation in a remote real-world environment. In estimating virtual camera parameters, two approaches exist, namely the fundamental matrix and the essential matrix. The latter has the advantage of exploiting the entire set of correspondences among the two adjacent panoramas, hence providing a more reliable and compact solution. However, the existing methods are not accurate, unable to facilitate seamless navigation on image datasets with a large number of panoramic images, and no objective measure for evaluating the results is presented. Here we will use the essential matrix and alleviate the shortcomings of the existing work in the literature [7].

Finally, to see a work on alignment of cylindrical-panoramas, the reader is referred to [20]. An alignment correction method is proposed in their work based on the dense disparity map between panoramas. However, unlike the approach that we are going to use, this is a featureless and un-calibrated solution to the alignment of closely taken panoramic snapshots.

This chapter is organized as the following: in the next section we present a quick review of the image dataset acquisition and preprocessing stages. In the third section, we introduce the existing work. In section four, we introduce our proposed solution to the problem, and in section five, we present the experimental results of the proposed algorithm applied on a large number of captured panoramic image datasets. Finally we summarize the chapter in section six.

3.2 Image Dataset Acquisition and Preprocessing

In this section we will briefly give an introduction to the acquisition and preprocessing stages of the project which come before the analysis stage discussed in this chapter.

3.2.1 Raw-Image Acquisition

The raw-image acquisition process generates the source input for the navigation system. A Ladybug camera from Point Grey Research is used. This camera consists of six ICX204AQ color CCD image sensors and has six high quality micro lenses with the focal length of 2.5mm. One lens on the top of the camera head unit points up and five lenses pointing horizontally are assembled in a horizontal circle. If panoramic images are captured on a linear trajectory, the resultant image dataset will constitute a 3D signal and if images are captured on a rectangular grid, then we will refer to the acquired image dataset as a 4D signal.

3.2.2 Image Dataset Preprocessing

Basic image preprocessing operations include gamma correction, white balance, noise filtering, and image format conversion to change the format of raw image signals to the standard image format required by image dataset compression [19]. After the preprocessing stage, a group of six raw full RGB images can be used to create a basis panorama which consists of six side images $\mathcal{I}_{\mathcal{B},k,j}$ $(\forall j \in \{d, u, l, r, b, f\})$ where the subscripts represent the *down, up, left, right, back,*, and *front* side image respectively. Each of these side views has a 90 degree of FOV in both horizontal and vertical directions. There is a blind area in the bottom view of the cubic panorama due to the lack of a lens facing down in the camera set. We need to align the cubic panoramas to a common world coordinate system at this point.

3.3 Existing Work

The essential matrix applied to cubic panoramas has been discussed by Fiala and Roth in [5]. They proposed a method for adjacent panorama alignment in which they assume that the translational component can be neglected. Later in [7] a generalized solution of the alignment problem was presented that does not neglect the translational vector. Two methods were introduced, namely Fundamental matrix **F** and Essential matrix **E**. A set of feature correspondences are found and used to estimate the associated matrix. In the case of using the fundamental matrix, each cube face is dealt independently of other faces and correspondences are considered among pairs of cube faces in the two panoramic images to be aligned. On the other hand, the method estimating the essential matrix considers the feature points in the 3D space independently of the face they are projected on. Experimental results in [7] confirm that the two approaches are equivalent. However, we believe that in a practical application, the method based on the fundamental matrix will fail as the feature points might be distributed, for example, on two different faces of the current and reference cubic-panoramas. As a result we adopted and use the essential matrix estimation. The standard 8-point algorithm discussed in [88] can be used to estimate the essential matrix. The estimated essential matrix is then used in [7] to estimate the unity translational vector, i.e., a unity vector starting from the center of the reference cube and pointing towards the center of the current cube, using the SVD (Singular Value Decomposition) of the Essential Matrix **E**. Fig. 3.1(a) (*left*) illustrates a pair of the original cubic panoramas before processing and Fig. 3.1(b) (*left*) shows the result when the two panoramas are aligned with the unity translational vector. Full alignment has not been achieved yet since this approach does not take into account rotation around the axis connecting the two centers. As a result, a minimization problem is formulated and solved as the final step in [7].

3.4 Proposed Method

Our proposed method tries to address the shortcomings of the approach explained above. In review, we identified three major problems with that approach, as listed below:

Problem of Approximation As already mentioned, this approach is unable to compensate for the third rotational parameter around the central axis and a numerical method is used instead.

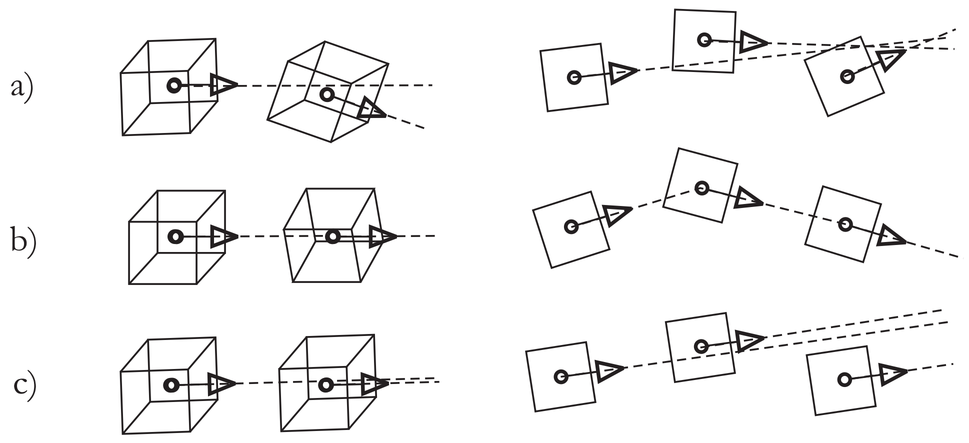


Figure 3.1: (a) Original data, (b) Existing method using translation vector T, and (c) Proposed method using rotation matrix R for cubic-panorama image dataset alignment

Problem of Application The major problem is that this approach cannot be used in practice on datasets with a large number of panoramic images. This notion is illustrated in Fig. 3.1. The figures on the right hand side depict the top view of panoramic images emphasizing the *front* face by a small arrow. If one aims to align the original cubes in (a), the result will be something similar to what is shown in (b). This is because the first two cubes can be aligned (ignoring the third angle problem) but when it comes to the alignment of the third cube, the already aligned second cube will align itself now with the third cube, hence at the end no overall improvement will be achieved.

Problem of Evaluation As the last shortcoming, no quantitative measure is introduced in [7] to evaluate the level of alignment in a given image dataset before and after the alignment process.

In order to address the first (and the second) problem, having estimated the essential matrix, instead of the Translation vector we estimate the Rotation matrix between the pair of panoramas subsequently.

Here we provide details regarding method for estimating the essential matrix first: As was explained in earlier chapters, the captured raw images are mapped onto a cubic panorama format similar to what is shown in Figure 3.2. Now if two such cubic panoramas are considered, the global image displacement associated with the camera motion can be modeled with a parameter set consisting of six translational and rotational global displacement model parameters $\mathbb{P}_M = \{T_x, T_y, T_z, \theta, \psi, \varphi\}$ (see Figure 3.2 and [89]). At

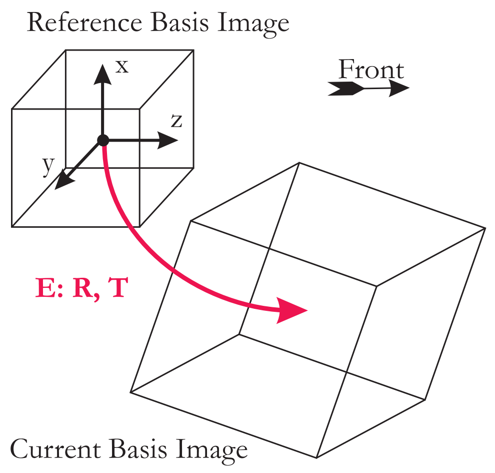


Figure 3.2: Essential matrix, Rotation matrix, and Translation vector

this point the reader might be able to guess how cube alignment can be achieved. If the rotational and translational parameters are estimated properly, the current basis image can be aligned with the reference basis image by either compensating changes made due to rotation or using the translation vector for alignment.

At this point, we will formulate the idea, specifically the approach that we are going to use to estimate the translation and rotational parameters. The coordinates of a static scene point with respect to the coordinate system with its origin fixed at the focus of the camera hold the relationship [90]

$$(x_r, y_r, z_r)^T = \mathbf{R} \cdot (x_c, y_c, z_c)^T + \mathbf{T}, \tag{3.1}$$

where (x_r, y_r, z_r) denotes the coordinates of a scene point before the camera motion, and (x_c, y_c, z_c) denoted its coordinate after the camera motion. The rotation matrix $\mathbf{R}$ can be represented with respect to the rotation angles θ, ψ, φ as shown in Figure 3.3 [90]

$$\mathbf{R} = (r_{ij})_{33} = \begin{pmatrix} \cos\varphi\cos\psi & \cos\varphi\sin\psi\sin\theta - \sin\varphi\cos\theta & \cos\varphi\sin\psi\cos\theta + \sin\varphi\sin\theta \\ \sin\varphi\cos\psi & \sin\varphi\sin\psi\sin\theta + \cos\varphi\cos\theta & \sin\varphi\sin\psi\cos\theta - \cos\varphi\sin\theta \\ -\sin\psi & \cos\psi\sin\theta & \cos\psi\cos\theta \end{pmatrix} \tag{3.2}$$

The model parameters $\mathbf{R}$ and $\mathbf{T}$ can be obtained by replacing the coordinates of image correspondence points into the following equation [91]

$$(x_r, y_r, z_r) \cdot \mathbf{E} \cdot (x_c, y_c, z_c)^T = 0, \tag{3.3}$$

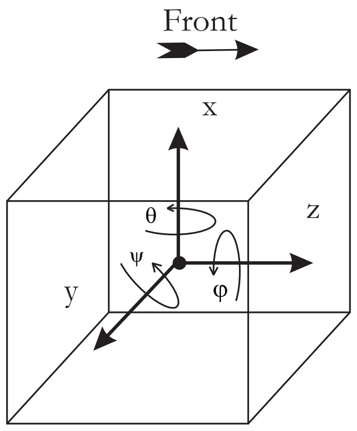


Figure 3.3: Three rotational parameters θ, ψ, and φ in $\mathbf{R}$

where the essential matrix $\mathbf{E}$, expressed with the components of the translation vector $\mathbf{T}$ and the elements of the rotation matrix $\mathbf{R}$, is given by

$$\mathbf{E} = (e_{ij})_{33} = \begin{pmatrix} T_z \cdot r_{21} - T_y \cdot r_{31} & T_z \cdot r_{22} - T_y \cdot r_{32} & T_z \cdot r_{23} - T_y \cdot r_{33} \\ T_x \cdot r_{31} - T_z \cdot r_{11} & T_x \cdot r_{32} - T_z \cdot r_{12} & T_x \cdot r_{33} - T_z \cdot r_{13} \\ T_y \cdot r_{11} - T_x \cdot r_{21} & T_y \cdot r_{12} - T_x \cdot r_{22} & T_y \cdot r_{13} - T_x \cdot r_{23} \end{pmatrix} \tag{3.4}$$

If we can estimate the essential matrix reliably, then finding the translation vector and the rotation matrix would be the following step. The translation column vector $\mathbf{T} = (T_x, T_y, T_z)^T = \alpha \cdot (\Delta x, \Delta y, \Delta z)^T$ can be extracted from the estimated essential matrix $\mathbf{E}$ by solving the homogeneous equations corresponding to $\mathbf{E}^T\mathbf{T} = 0$ [92] while $\Delta x, \Delta y, \Delta z$ satisfy the normalization condition

$$(\Delta x)^2 + (\Delta y)^2 + (\Delta z)^2 = 1 \tag{3.5}$$

The rotation parameters of the global displacement model are derived from the essential matrix $\mathbf{E}$ with the singular value decomposition (SVD) given by:

$$\mathbf{E} = \mathbf{U} \cdot \Xi \cdot \mathbf{V}^T, \tag{3.6}$$

where, Ξ is a diagonal matrix of the same dimension as $\mathbf{E}$ with nonnegative diagonal elements in decreasing order; $\mathbf{U}$ and $\mathbf{V}$ are unitary matrices. Then, the desired rotation matrix is obtained by

$$\mathbf{R} = \mathbf{U} \cdot \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & s \end{pmatrix} \cdot \mathbf{V}^T, \text{ or } \mathbf{R} = \mathbf{U} \cdot \begin{pmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & s \end{pmatrix} \cdot \mathbf{V}^T, \tag{3.7}$$

where $s = \det(\mathbf{U}) \cdot \det(\mathbf{V}) = \pm 1$. The rotation angles θ, ψ, φ can be determined upon request by using equation 3.2 directly from the rotation matrix $\mathbf{R}$.

As we see, estimating the essential matrix plays an important role in cubic-panorama adjustment as well as displacement estimation which will be discussed later. To estimate the essential matrix from equation 3.3 we need at least eight correspondences between the reference and the current basis images. An image matching process is performed to obtain a number of correspondence pairs in the reference image and the predicted current image respectively. A sufficient number of correspondence pairs are needed to obtain robust model parameter estimates. A variety of image matching techniques can be employed based on their effectiveness and efficiency in performing feature point selection, detection and matching.

Determination of the essential matrix $\mathbf{E}$ is a typical optimization process of estimation error minimization. By replacing the coordinates of the correspondence pairs into the image coordinate transformation equation, the sum of the squared errors S_E is obtained. A linear least-squares estimation algorithm is applied by solving a group of equations resulting from $\frac{\partial}{\partial e_{ij}} S_E = 0$ ($i \in \{1, 2, 3\}$, $j \in \{1, 2, 3\}$) to determine the values of the elements in the essential matrix $\mathbf{E}$.

In [7] two approaches are introduced for finding the essential matrix between two cubic panoramas, one of which works by estimating the fundamental matrix between each pair of cube faces. The global rotational and translational parameters for each cube face are estimated separately and then the six set of parameters, associated with the six cube faces, are merged to achieve an overall set of parameters $\mathbb{P}_M = \{T_x, T_y, T_z, \theta, \psi, \varphi\}$. This approach has its own drawbacks since, unlike what is done in [7], here we are dealing with a large number of cubic panoramas and we expect to face situations where not enough image correspondences will be obtained between two corresponding faces of the current and reference basis images. In other words, in some cubic-panorama image datasets there might be occasions were required feature points appear across different cube faces.

On the other hand, the second approach tries to estimate the essential matrix instead of the fundamental matrix, a matrix that corresponds to the whole cubic panorama and not only individual faces. Also, in cases where feature points in the space are mapped

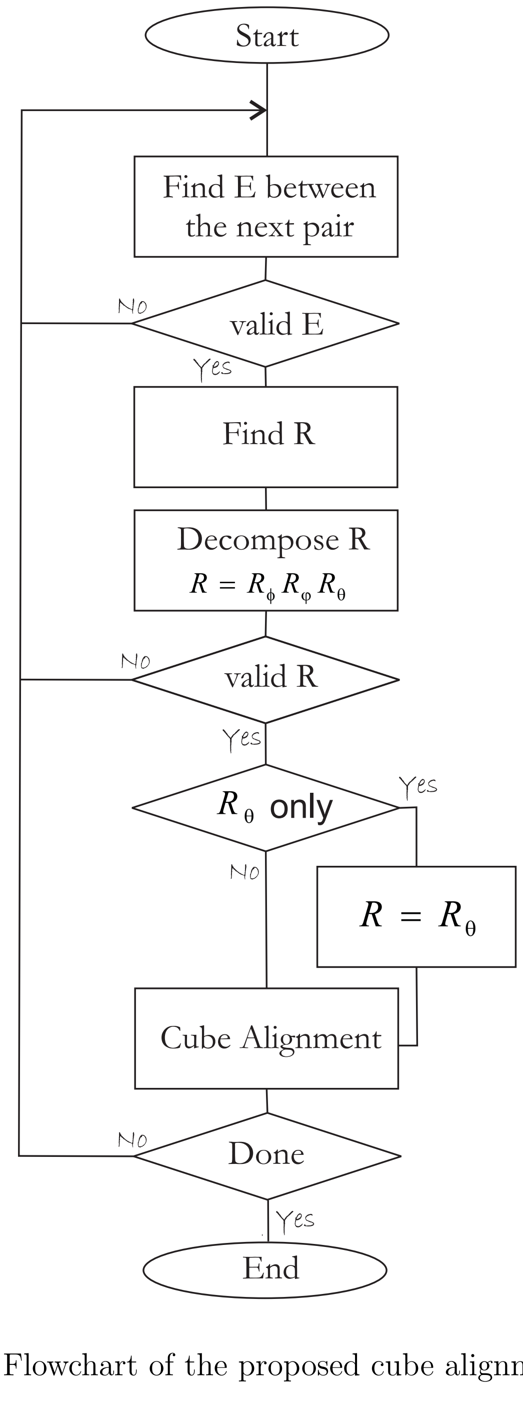


Figure 3.4: Flowchart of the proposed cube alignment method

onto different faces of the two cubes, the first approach, i.e., using the fundamental matrix, would not be capable of detecting and matching such feature points, but using the second approach, i.e., the essential matrix, will have no limitation in this regard. This is mainly because the essential matrix works on coordinates in the 3D space while the fundamental matrix work on 2D image coordinates.

Image coordinates can be converted to the 3D coordinates on the cube surface by a simple matrix operation $(x, y, z)^T = \mathbf{T}_j \cdot (X_j, Y_j, F)^T$ where $\mathbf{T}_j$ is the transformation matrix associated with each cube face ($\forall j \in \{u, b, l, f, r, d\}$), where the subscripts represent the *up, back, left, front, right,* and *down* side image respectively, F is the camera focal length, and L is the size of a cube:

$$\mathbf{T}_d = \begin{pmatrix} 0 & 0 & -\frac{L}{2} \\ 0 & +1 & -\frac{L}{2} \\ -1 & 0 & +\frac{L}{2} \end{pmatrix}, \mathbf{T}_u = \begin{pmatrix} 0 & 0 & +\frac{L}{2} \\ 0 & +1 & -\frac{L}{2} \\ +1 & 0 & -\frac{L}{2} \end{pmatrix}, \mathbf{T}_l = \begin{pmatrix} -1 & 0 & +\frac{L}{2} \\ 0 & 0 & -\frac{L}{2} \\ 0 & +1 & -\frac{L}{2} \end{pmatrix} \tag{3.8}$$

$$\mathbf{T}_r = \begin{pmatrix} -1 & 0 & +\frac{L}{2} \\ 0 & 0 & +\frac{L}{2} \\ 0 & -1 & +\frac{L}{2} \end{pmatrix}, \mathbf{T}_b = \begin{pmatrix} -1 & 0 & +\frac{L}{2} \\ 0 & -1 & +\frac{L}{2} \\ 0 & 0 & -\frac{L}{2} \end{pmatrix}, \mathbf{T}_f = \begin{pmatrix} -1 & 0 & +\frac{L}{2} \\ 0 & +1 & -\frac{L}{2} \\ 0 & 0 & +\frac{L}{2} \end{pmatrix} \tag{3.9}$$

The flowchart of a new cube alignment method is shown in more detail in Figure 3.4. At the first step, the essential matrix between the first (or the subsequent) pair of panoramas is calculated. This step needs further explanations which will be provided soon. As we will see later, there might be occasions where a reliable essential matrix can not be achieved between a pair of basis images. In such scenarios, which did not happen so frequently in practice, the cube alignment process for that pair will be skipped. But if a valid essential matrix can be found, then the rotation matrix would be extracted in the subsequent step. Then, the rotation matrix can be decomposed into three matrices associated with the three rotational angles,

$$\mathbf{R} = \mathbf{R}_\varphi \cdot \mathbf{R}_\psi \cdot \mathbf{R}_\theta \tag{3.10}$$

There are many occasions in which we know that the captured image dataset is originally acquired in such a manner that the rotational parameters are not solely random numbers, in other words, we can guess an upper limit for the rotational angles. This observation will provide a second chance for us to test the validity of our parameter estimation algorithm by comparing the amplitude of the obtained angles with some

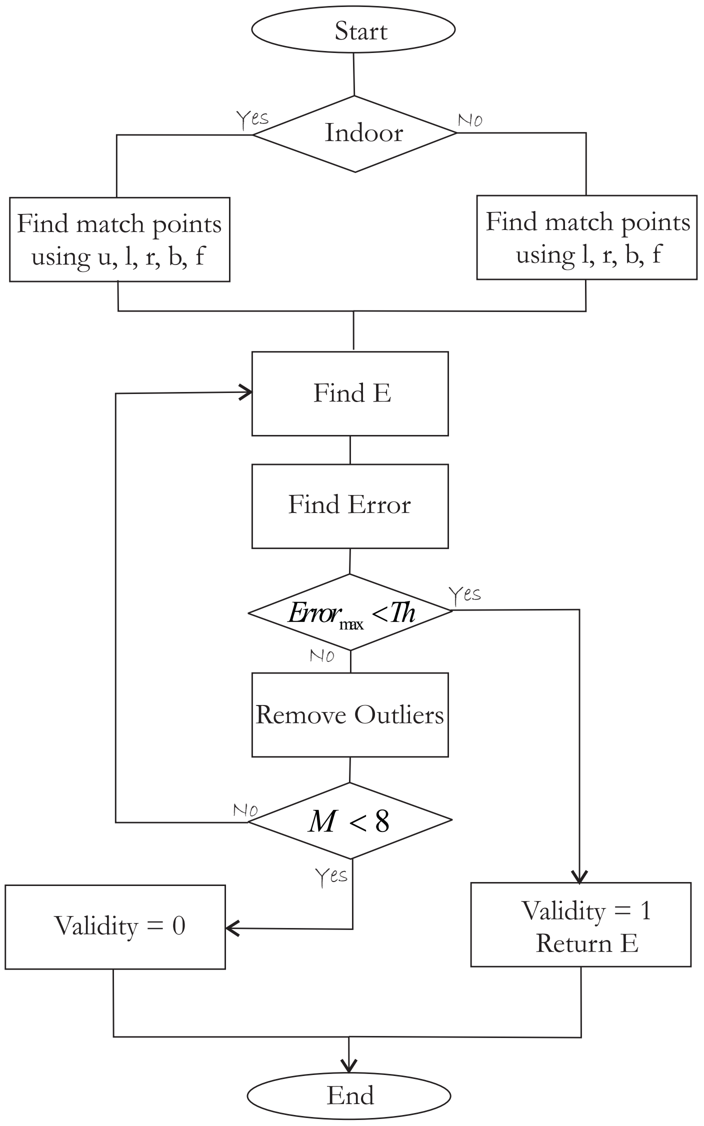


Figure 3.5: Finding the Essential matrix subsection

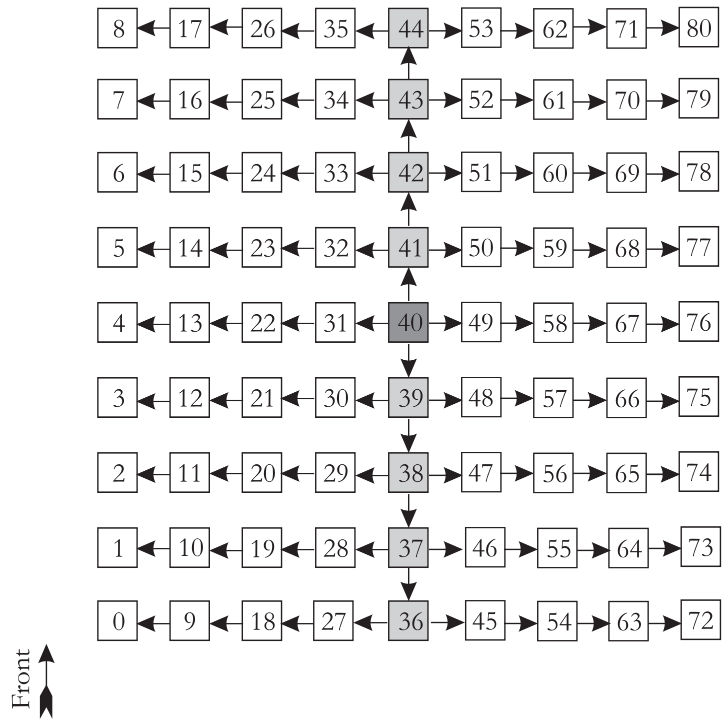


Figure 3.6: Alignment structure for 4D image datasets

predefined threshold values. Following the comparisons, if the obtained angles are bigger than the predefined threshold values, we will skip the current iteration looking forward to the next pair of basis images.

At this point, we have the option to decide if we are going to use all the three angles for cube alignment or only the first angle θ would suffice. This option exists because, in some practical applications, the camera rotations might be mainly limited to the θ angle only, the assumption that may apply when the ground plane is flat, most probably in indoor environments. Nevertheless, the final decision is up to the author of the system either to use all the estimated rotational parameters or only the θ value. At this point, the pair of cubic-panoramas should be aligned by accessing the raw acquired images and mapping them again onto a cube format now considering the modifying rotation matrix. The original raw images should necessarily be accessed and used at this stage, since rotation of a cubic panorama is usually a lossy procedure.

Finally, Figure 3.5 shows how we can find the essential matrix iteratively. First, based on whether dealing with an indoor or an outdoor environment, we use different cube faces for image correspondence. The *down* face is never used for feature matching since not much visual information is available, due to lack of a capturing camera lens facing downwards. Also in outdoor environments, the *up* face is also ignored in feature matching, because it usually does not convey much useful data or feature points as it is facing upwards, i.e., towards the open sky.

Then, the essential matrix is estimated and outlier removal is involved in the model parameter estimation process to obtain more accurate parameter estimates through discarding 10% of the correspondence pairs that have bigger coordinate differences than others at each iteration step. The essential matrix is recalculated after the outlier removal until the maximum error is smaller than a predefined threshold value or the number of remaining correspondences M is smaller than the minimum required number of 8. In the former case a reliable essential matrix has been estimated while in the latter, the algorithm has failed. The essential matrix estimation block returns a flag indicating whether the estimated essential matrix is valid in addition to the essential matrix itself.

Results will look like what is depicted in Fig. 3.1(c) for the 3D view of two adjacent panoramas (*left*) and top view of three or higher number of panoramas (*right*). As can be seen, the two panoramas can be fully aligned using our method (*left*) and also all the three (or more) panoramas can be aligned towards the same direction (*right*).

An alignment structure that can be used to align basis images captured on a rectangular grid is also shown in Figure 3.6. The alignment algorithm is performed first on the middle column (in light grey) starting from the middle basis image, i.e., basis image number 40 (in dark grey). Middle column and middle basis image are chosen to minimize the error propagation effects, if there is any. Once the middle column is aligned, each row of basis image (in white) are aligned using the corresponding middle basis image as reference.

Finally, in order to evaluate the performance of the proposed algorithm, we introduced a measure for panoramic image dataset alignment which is defined as below:

$$\mathcal{E}_{\text{RMS}} = \sqrt{\frac{1}{K-1}\sum_{k=1}^{K-1} \text{avg}\{(\mathbf{R}_k - I_{33}) \circ (\mathbf{R}_k - I_{33})\}}, \tag{3.11}$$

where K is the total number of basis images in the image dataset, $\mathbf{R}_k$ is the rotation matrix between $\mathcal{I}_{\mathcal{B},k}$ and $\mathcal{I}_{\mathcal{B},k+1}$, and I_{33} is a 3 by 3 identity matrix. The $\circ$ operator denotes the Hadamard product and the *avg* operator calculates the average of the elements in the input 3 by 3 matrix.

Proposed solution to the problem of cubic panorama alignment is tested on a variety of test sequences. Experimental results will be presented in the following sections.

No.	Name	Location	In./Out.	Size	Dim.
1	CHPL	Tabaret Hall, 1st floor, Chapel	Indoor	65	3D
2	TBT1	Tabaret Hall, 1st floor			
3	TBT2	Tabaret Hall, 2nd floor			
4	LAB	CBY Building, 4th floor, VIVA Lab			
5	NGC	National Gallery of Canada, Outside	Outdoor		
6	CBY	Colonel By Building, Outside			
7	CHPL	Tabaret Hall, 1st floor, Chapel	Indoor	9 x 9 = 81	4D
8	TBT1	Tabaret Hall, 1st floor, Lobby			
9	LAB	CBY Building, 4th floor, VIVA Lab			
10	LBY	CBY Building, 1st floor, Lobby			
11	NGC	National Gallery of Canada, Outside	Outdoor		
12	CBY	Colonel By Building, Outside			

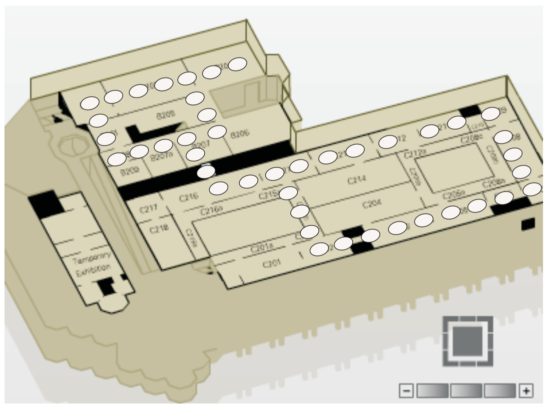
Table 3.1: Summary of captured image datasets

3.5 Experimental Results

In this section experimental results associated with the proposed approach in this chapter will be presented. They can be categorized as the following:

- Raw image acquisition;
- Image dataset alignment;

Two different scenarios will be considered for image dataset acquisition. The first possibility is to acquire panoramic images on a connected graph like what is illustrated in Figure 3.7(a). Here each white circle represents one basis image. Captured panoramas should be indexed properly to facilitate the ease of random access. In a practical application however, capturing more than one set of panoramic images might be required, i.e., data will be captured on a number of trajectories which might be connected at some intersecting viewpoints. Such image datasets constitute a 3D signal as it was explained in the introduction chapter. On the other hand, as shown in Figure 3.7(b), panoramic images can also be captured on a rectangular grid, instead of some predefined trajectories, where the resultant image dataset would be a 4D signal and free navigation will be facilitated so long as the user is restricted to the ground plane. Even more complicated


(a) 2D connected graph of panorama locations resulting in a 3D image dataset

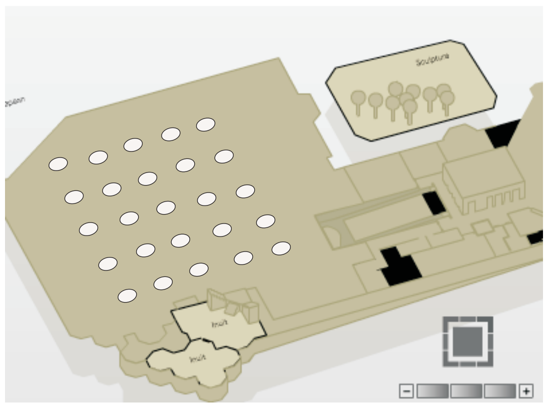
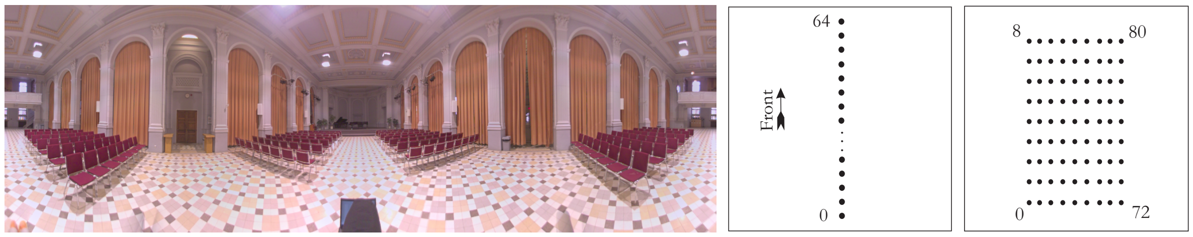

(b) A rectangular grid resulting in a 4D image dataset

Figure 3.7: Two image acquisition scenarios

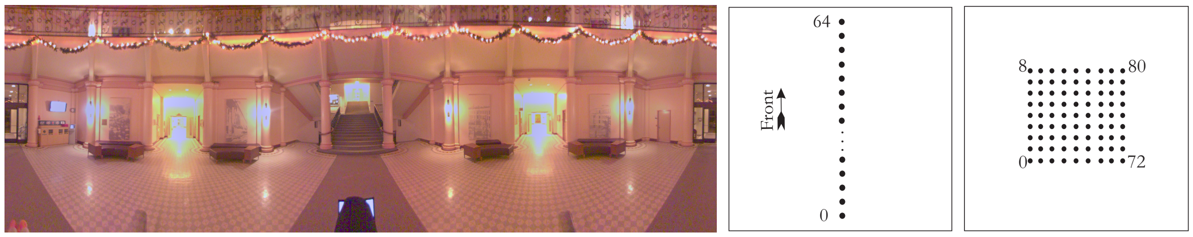
scenarios could be envisioned, for example a 5D image dataset where the user would be able to navigate vertically off the ground level. Although this case will not be studied in this thesis, obtained results can be easily extended to higher dimensions. Later in chapter four, a bitstream syntax for image dataset storage in the application layer will be presented towards organizing the huge data in the storage unit.

At the raw image acquisition stage, twelve image datasets were captured, i.e., $J_0 = 12$ and $K_0 = 1$, where J_0 is the number of real world environments where the image datasets are captured and K_0 is the number of segment within each image dataset as it will be shown in chapter four. In order to maximize the diversity, various indoor and outdoor locations were chosen for image acquisition in the city of Ottawa in the Fall of 2009. They including National Gallery of Canada (NGC), Tabaret Hall building, Colonel By (CBY) building. Tabaret Hall building itself includes the first floor (TBT1), the second floor (TBT2), and the meeting room or Chapel (CHPL). Colonel By building includes the Lobby (LBY) located at the first floor, VIVA lab (LAB) located at the fourth floor, and the outside space. These information are summarized in Table 3.1.

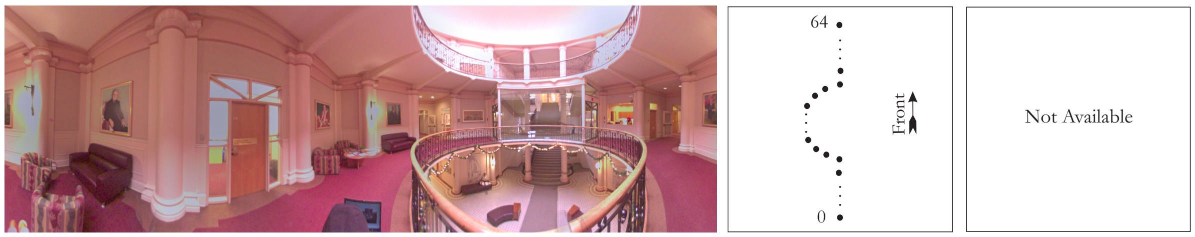
The first six image datasets consist of 65 basis images, indexed from 0 to 64, captured on a predefined trajectory (thus compromising 3D image datasets), while the last six image datasets are captured on a 9 by 9 rectangular grid, i.e., $L_0 = 9$, resulting in 81 basis images (thus compromising 4D image datasets). In each case, the first four image datasets are captured indoors, while the last two are captured outdoors. Abbreviated names are used for ease of later referencing.



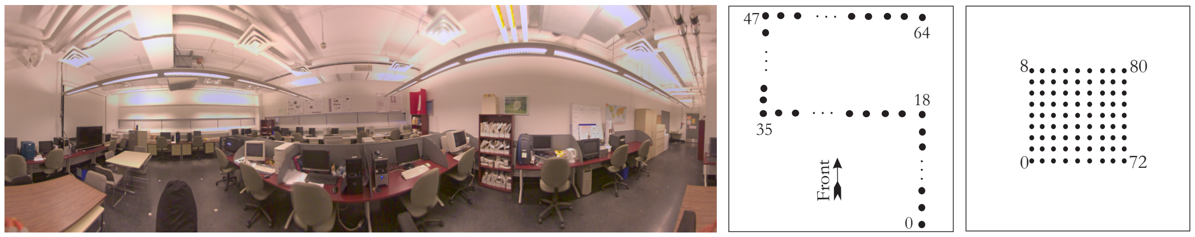
(a) CHPL



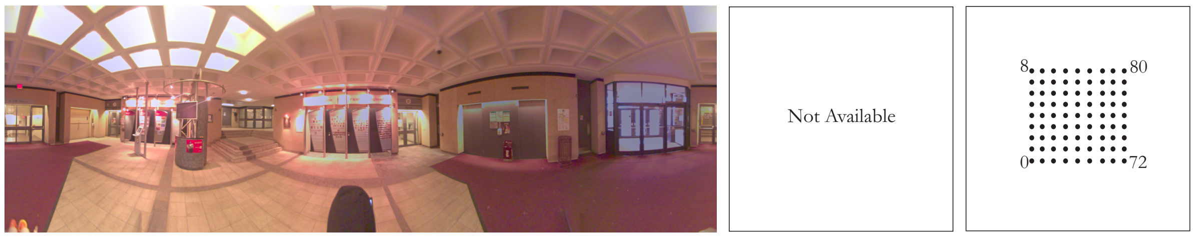
(b) TBT1



(c) TBT2

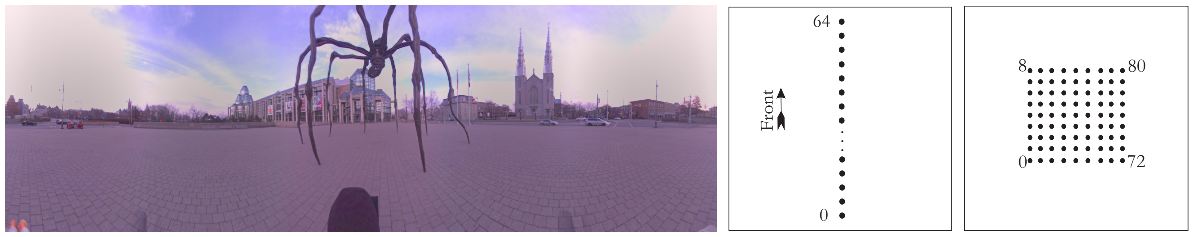


(d) LAB

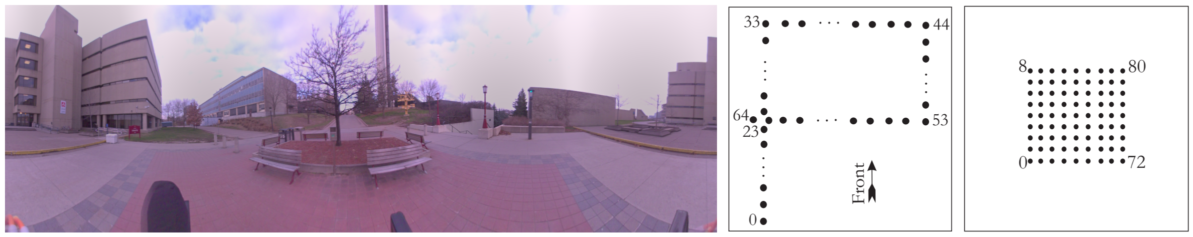


(e) LBY

Figure 3.8: Captured data (Indoor)



(a) NGC



(b) CBY

Figure 3.9: Captured data (Outdoor)

Application of the captured image datasets is not necessarily limited to the work done in this thesis and they can be used as test data in many other applications any time in the future. Figures 3.8 and 3.9 show a snap-shot of the image acquisition locations (depicted in the cylindrical format) for indoor and outdoor environments respectively. On the right hand side, the top view of the image capturing trajectories is depicted as well as the specific indexing method for both 3D and 4D image datasets. These image datasets will be used as the input data later in this chapter for testing and comparing different algorithms.

In this section we will apply the method previously proposed in this chapter for aligning the adjacent cubic panoramas. The flowcharts already shown in Figures 3.4 and 3.5 will be used. We use the value of $Th = 3$ pixels, i.e., the threshold for the maximum error in feature point image coordinates which is calculated based on the estimated essential matrix. If the maximum error is higher than this threshold value, then 10% of the outliers will be removed and the essential matrix will be estimated again. This process will be repeated until either the maximum error will be less than Th or the number of remaining feature points will become less than 8, i.e., the minimum required number of feature points in the eight-point algorithm [88] for estimating the essential matrix.

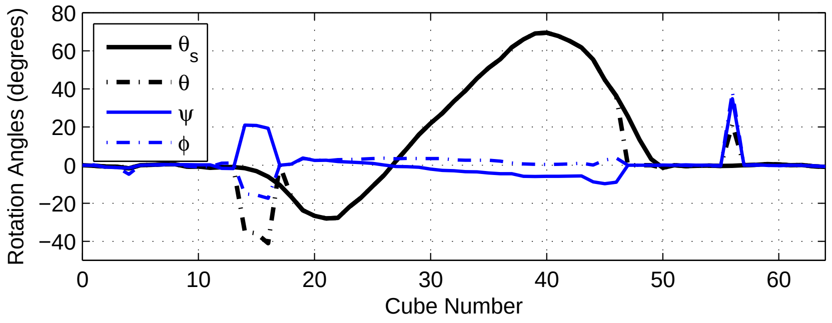


Figure 3.10: Alignment results for TBT2 image dataset

At this point we will explain further practical details regarding processing of the TBT2 image dataset as a very good example. For this image dataset, we preferred to estimate the $\mathbf{R} = \mathbf{R}_\theta$ values for image dataset alignment. We also used the threshold values of $(90°, 10°, 10°)$ for $(|\theta|, |\psi|, |\varphi|)$ which, as shown in Figure 3.10, means that in basis images number 14, 15, 16, and 56, compensation for the rotation matrix is skipped because of the inaccuracies in estimating the rotational parameters. After compensating for the available rotation matrices in this image dataset, we faced a gap between the successfully aligned basis images at two segments of the trajectory, namely around frames 15 as well as 47, due to some non-existing essential or rotation matrices. In this example, since we knew that the initial image capturing procedure was a quite smooth one, we found ourselves eligible to use the spline interpolation for the missing alignment parameters, specifically the missing θ values. Finally, we ended up to the θ_s plot shown in Figure 3.10.

Simulations were made on all the captured image datasets and test results for the first six image datasets are presented in Figure 3.12. For 4D signals similar results were obtained.

We normalized all the six calculated original Root Mean Square Error (RMSE) values $\mathcal{E}_{\text{RMS}}$ by the maximum obtained value among the six image datasets, i.e., that of TBT2, in order to be able to make comparisons more easily. Results are shown in figure 3.12(a). As can be seen, the maximum normalized RMSE (value of 1) belongs to the TBT2 image dataset, mainly due to the special type of the trajectory as shown in Figure 3.8(c). After TBT2, the two image datasets captured outdoors, i.e. CBY and NGC, possess the

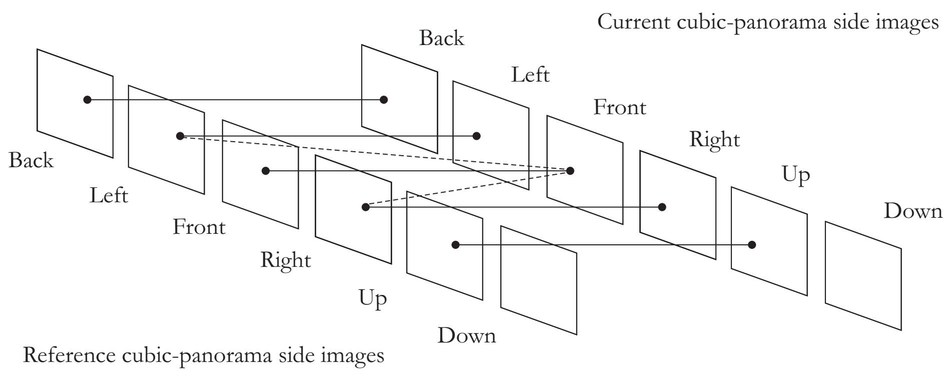

Current cubic-panorama side images
Back
Left
Front
Right
Up
Down
Back
Left
Front
Right
Up
Down
Reference cubic-panorama side images

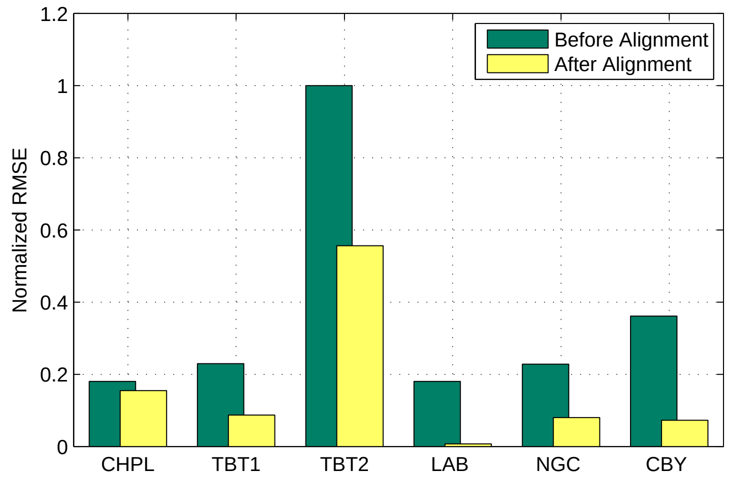
Figure 3.11: Feature matching scenario between faces of two adjacent cubic panoramas

highest $\mathcal{E}_{\text{RMS}}$ value. In the same figure, normalized RMSE values after image dataset alignment are shown. In CHPL, TBT1, and TBT2 image datasets, only $\mathbf{R} = \mathbf{R}_\theta$ is used, while in LAB, NGC, and CBY image datasets, $\mathbf{R} = \mathbf{R}_\varphi \cdot \mathbf{R}_\psi \cdot \mathbf{R}_\theta$ is exploited for image dataset alignment. Figure 3.12(b) shows the percentage of the number of basis images that have been successfully aligned as well as the percentage in RMSE decrease. As can be seen, in TBT2 image dataset, which is taken in a relatively complicated environment on a nonlinear trajectory, at least 80 percent of the basis images have been aligned successfully. For the rest, almost all the basis images have been aligned with success. In all of the aforementioned experiments, subjective quality of the resultant image datasets was also satisfactory.[1]

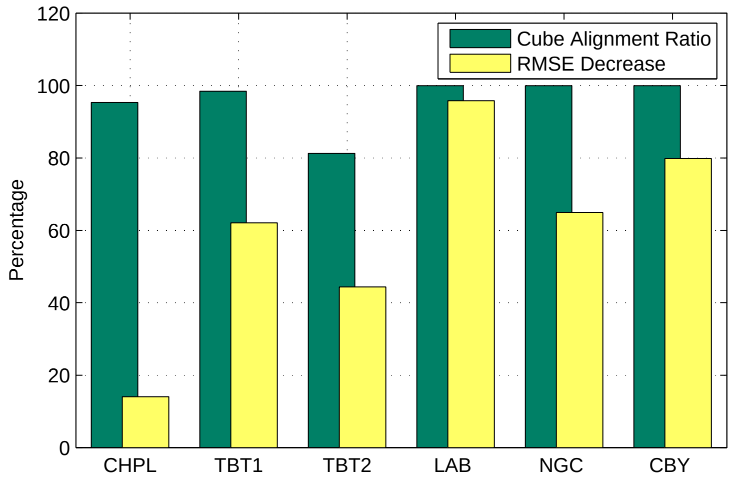
A combination of Matlab programming and existing image acquisition system in the lab was used [18, 19]. Notice that in feature matching, a required task in estimating the essential matrix from equation 3.3, a Scale-Invariant Feature Transform (SIFT) method was used[2]. In TBT2 image dataset, search for feature points was not restricted to the associated side image in the reference basis image only, but side images to the left and right of the associated cube face in the reference basis image were also exploited in feature point search by using the matrix transformations of 3.8 and 3.9, see dotted lines in Figure 3.11. We needed to do so mainly due to the specific nature of the trajectory in TBT2 image dataset.

[1]Video accompanying this thesis can be downloaded from the University of Ottawa thesis repository at www.ruor.uottawa.ca. It includes alignment results for NGC and CBY cubic-panorama image datasets.

[2]Visit http://people.cs.ubc.ca/ lowe/keypoints/ to access the SIFT keypoint detector that has been used in our work



(a) Normalized RMSE before and after alignment



(b) Cube alignment success ratio and RMSE decrease

Figure 3.12: Cube alignment performance evaluation

3.6 Summary

In this chapter we presented a novel image dataset alignment algorithm for cubic panoramas enabling seamless transition in acquired real world environments [93]. Our method is accurate and not restricted to only a pair of panoramic images and also does not add to the computational complexity, hence can be applied in practice on datasets with a large number of panoramic images. Our objective measure also confirmed the expected improvements numerically. Subjectively, our processed image datasets possess no navigational jitter effects as observed in the original data. This method can be used as a first step in many applications such as cubic panorama interpolation, view synthesis, disparity estimation, etc. Methods used in this chapter to estimate the essential matrix will be used in chapter five when the problem of disparity estimation will be addressed. In the next chapter main issues regarding the storage of data including compression standards and schemes, prediction structure, and the bitstream syntax will be addressed.

Chapter 4

Cubic-Panorama Image Dataset Compression

In this chapter we address issues regarding cubic panorama image dataset compression. Two state-of-the-art approaches, namely H.264/MPEG4 AVC and Dirac video codec, are used and compared for the application of virtual navigation in image based representations of real world environments. Different prediction structures and Group Of Pictures (GOP) sizes are investigated and compared on this new type of visual data. Based on the obtained results, as well as the requirements of the system, an efficient prediction structure and bitstream syntax are proposed.

4.1 Introduction

In the previous chapter we proposed an image rectification algorithm to be used on a large number of captured cubic-panorama images, in order to provide a seamless virtual navigation in a remote real world environment [93]. Followed by the aforementioned image dataset analysis stage our visual data will be ready for compression. We will apply and compare the two major existing approaches from the literature first, i.e., a standardized method based on H.264/MPEG4 AVC and an existing wavelet-based scheme called Dirac. Then, the problem of indexing will be addressed considering the compression efficiency, random access, and other requirements of our application. Advantages of using B frames are shown in cases where captured datasets are not uniformly distributed on a rectangular grid. Based on the aforementioned considerations an appropriate bitstream syntax will be introduced for cubic-panorama image dataset compression.

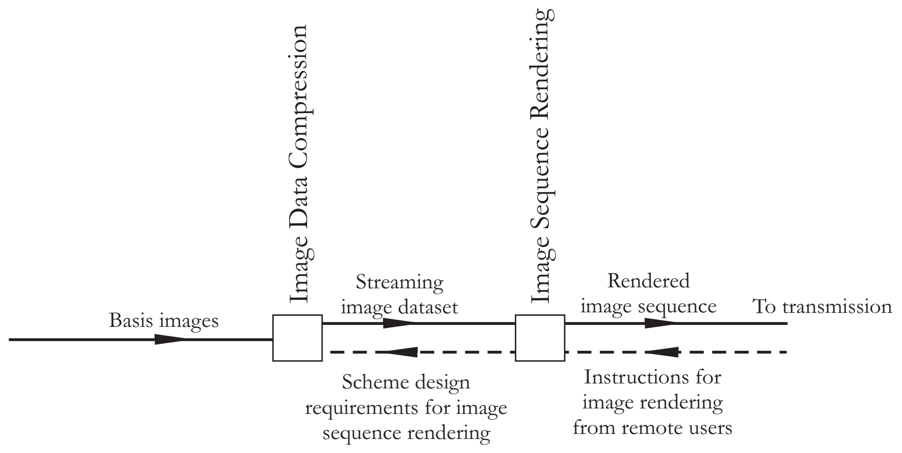


Figure 4.1: Diagram of client server based approach to the problem

The rest of this chapter is organized as the following: in the next section we will present the background. Our proposed scenario including details on prediction structure and bitstream syntax will be presented in section three. In section four experimental results that complement our scheme will be presented. Finally we will conclude in section five.

A diagram of a client server based approach to the problem is illustrated in Figure 4.1. Here the desired subsection of the general virtual navigation system is drawn again with additional details. In this setup, image dataset compression stage of the navigation system receives the captured basis images as the input data and processes them in order to generate an output bitstream, relatively small in size, that will be referred to later each time when a remotely located user requests access to parts of the captured image dataset. Instructions for image rendering will be issued by the remotely located user and commands will be transmitted backwards to the image sequence rendering stage. Subsequently, scheme design requirements for rendering novel views are sent in a backward channel to the storage unit so that, depending on the rendering scheme, required data will be sent forward to the rendering unit. The novel view will be prepared at this point for transmission through the communication channel.

4.2 Background

Among many existing coding schemes and frame prediction formats, we will try to adopt the best possible structure which would be able to satisfy the image dataset compression requirements introduced in the first chapter. A review of image dataset compression schemes was presented in section 2.2 and it was mentioned that three major approaches exist in the literature. They include vector quantization based methods [36, 42], wavelet-based schemes [46, 47] and standardized based approaches [43, 44].

Vector quantization based methods are simple in implementation with low structural complexity, decoding is simple and fast, and random access and selective decoding is provided. However, compression efficiency cannot be high since no inter-frame redundancy is exploited throughout the image dataset compression procedure. Therefore these approaches will not be useful for compression of our captured panoramic image datasets which are usually huge in size and demand very high compression ratios.

On the other hand, wavelet-based methods are able to provide high compression ratios due to the efficient use of wavelet transform both in spatial and temporal directions in conventional video coding. Wavelet-based approaches are also able to provide temporal, spatial, and quality scalability. However the structural complexity of such methods are usually high.

Standardized method for image dataset compression is depicted in Figure 2.1. One advantage of standardized-based approaches is that users, usually located at various remote places, will be able to use the compressed image dataset data quite easily if they only have access to the widely used standard decoders. Also, although scalability [94] is a significant feature of the wavelet-based methods, we will see later that since the encoding of data is performed offline and independent of the transmission stage, we can achieve a high range of output video qualities by choosing proper quantization parameters in real time. Similarly, at the transcoding stage and in real time, the compressed image dataset can be decoded and encoded into any desired spatial resolution. The last scalability feature is named view scalability, analogous to temporal scalability in conventional video coding, where user navigation with different walking speeds is facilitated. Similar to quality and spatial scalability, the view scalability requirement can be addressed in the video transcoding stage too.

The only work that exists on compression of cubic-panorama image datasets in the literature [10] uses Motion Compensated Temporal Filtering (MCTF) which is a wavelet based approach. On the other hand we have the standardized video compression scheme H.264/MPEG4 AVC [68] which is a hybrid video codec. Recently, a new video codec called Dirac[1] was introduced and was used internally by the BBC to transmit High Defi-

[1]For more details and to download and use the latest version of the Dirac video codec visit http://www.diracvideo.org

nition Television (HDTV) pictures at the Beijing Olympics in 2008. A wavelet transform is applied spatially in order to reduce the spatial redundancy. Notice that although, in contrast with the MCTF, this method does not apply wavelet filters in the temporal direction and uses feedback loop for temporal prediction similar to what is shown in figure 2.1, we name it a wavelet-based scheme for ease of reference. The algorithms in the Dirac specification have been designed with the intention to provide a competitive performance as compared to state-of-the-art international standards. Whether they succeeded is an open question. At least one comparison exists which used implementations from the second quarter of 2008 [95]. It shows x264 scoring higher than Dirac for classic videos, however it is somewhat out of date. A study on the performances of the Dirac codec, dated from August 2009 [96], finds that the quality obtained on SDTV is inferior to the H.264 output. The study mentions HD in its conclusion, but the numbers on HD are missing.

In all the aforementioned comparisons, standard video sequences are used as data for testing. In this work, we use the captured and aligned cubic-panorama image datasets and compare the performance of the two approaches on this new type of visual data. After choosing the appropriate video codec, we search for the best frame prediction structure and Group Of Pictures (GOP) size by running a number of experiments on our image datasets. For similar existing work on frame type and GOP size which studies classic video sequences, the reader is referred to [76] where hierarchical B pictures are analyzed.

In the state-of-the-art video coding standard, H.264/AVC, different frame types are defined. Intra frames I, Figure 4.2(a), are not predicted from neighbor frames in a video sequence, hence their encoding is fast but very high compression ratios can not be achieved. On the other hand P frames are those frames that use one neighbor frame for prediction, as an example, a typical frame structure for group of picture size of 4 is shown in Figure 4.2(b).

In B frames more than one frame is allowed to be used as reference for prediction as shown in Figure 4.2(c). Another type of B frames are introduced in the literature, known as hierarchical B frames, that are shown in Figure 4.2(d). In this type of prediction structure, B frames at lower levels of a pyramid can be used as reference for some other B frame at higher levels of the pyramid.

P frames possess higher compression efficiency as compared to the I frames in video compression. However, the need to decode a series of past frames before decoding the current P frame, makes them not to be the choice for the image dataset compression

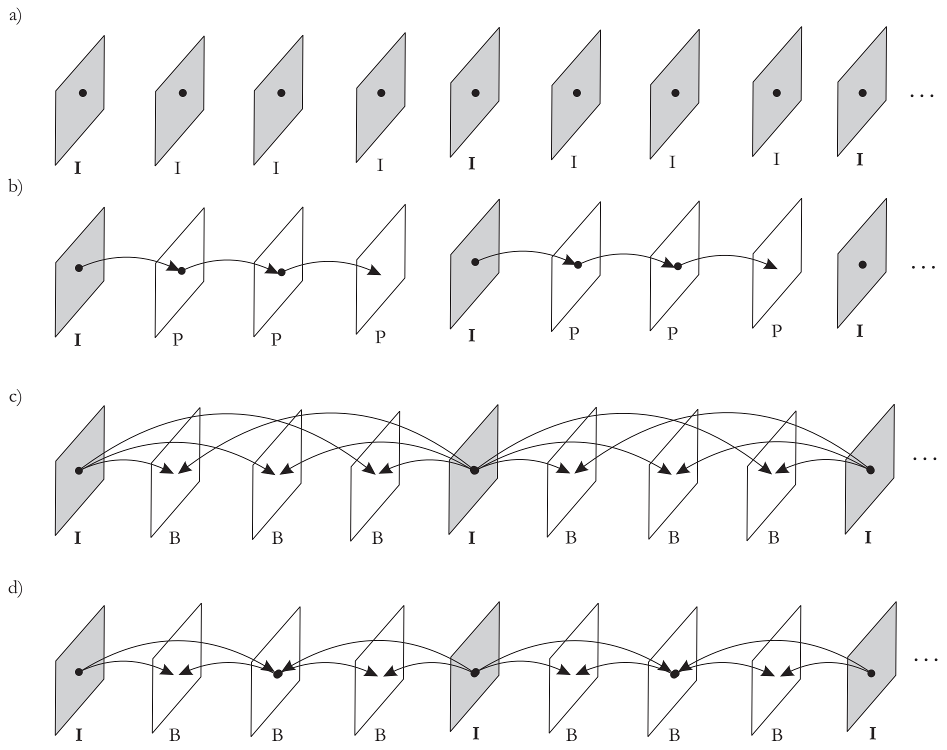


Figure 4.2: Different frame types (GOP size $N_0 = 4$): a) I, b) P, c) B, and d) hierarchical B pictures

application under study. In [76] it has been shown that hierarchical B pictures achieve efficient performance for coding conventional video sequences as compared to the other frame types and also compared to the motion compensated temporal filtering approach which is a wavelet-based scheme. Similarly, in [85] hierarchical B pictures are adopted in MVC for temporal frame prediction structure while regular B frames are used for cross-view frame prediction.

Nevertheless, decoding a B frame at higher layer introduces some delay, since it requires decoding a number of other reference frames, and this is against the random access requirement explained in the first chapter. As a result, although hierarchical B pictures achieve slightly better rate-distortion performance as compared to that of the B frames, due to the aforementioned requirements we eventually adopt the B frame structure of Figure 4.2(c) for image dataset compression. As can be seen in the figure, in order to be able to decode a B frame, only two I frames need to be decoded and if the intra frames are decoded one time, similar information can be reused to decode other neighbor B frames and new I frames need to be decoded only once a new GOP is to be

Frame type	I	P	B	hierarchical B
Number	0	3	2	3

Table 4.1: Maximum number of frames to be decoded to access a frame (GOP size $N_0 = 4$)

accessed.

Usually when B frames are used, the encoding order will not be the same as the display order and image sequence reordering in both encoding and decoding stages should be taken into consideration. For example in Figure 4.2(c) the two intra frames need to be encoded/decoded first and then it will be time to encode/decode the B frames in between. In any case since the encoding procedure will be performed offline and not in real time, the introduced encoding delay will not cause any problem. The maximum number of frames to be decoded to be able to access a frame in Figure 4.2 is 0, 3, 2, and 3 for I, P, B, and hierarchical B frames respectively. As can be seen in table 4.1, after intraframe coding, which would not meet the rate-distortion requirement, proposed structure of B frames needs the minimum required number of frames to be decoded in order to randomly access one B frame in the middle of the image sequence, a feature that is in line with the random access requirement described in the introduction.

To be able to decode one B frame, two I frames need to be decoded first, which means that the memory requirement is much less compared to the case of hierarchical B frames. Also unlike the hierarchical B frame structure, the proposed B frame structure is easily extendable and applicable to image dataset signals with higher dimensionality. Also, unlike P frames, B frames facilitate view scalability, that is another desirable requirement in image dataset compression.

Another parameter that needs to be decided is the size of GOP. Video sequences with different GOP sizes are depicted in Figure 4.3. First is the intra-frame I coding where each frame is compressed independently, hence the GOP size can be assumed to be one, i.e., each GOP consists of a single I frame. GOP values of 2, 4, and 8 are shown subsequently, which can be denoted by GOP2, GOP4, and GOP8 respectively. Although in [76] GOP16 and GOP32 are also taken into account, we will not consider those long structures here. Because in conventional video sequences, frames are captured in time, where usually 15 to 30 frames are captured during a second, as a result, a very high correlation will exist among a larger number of neighboring frames. Therefore, in the

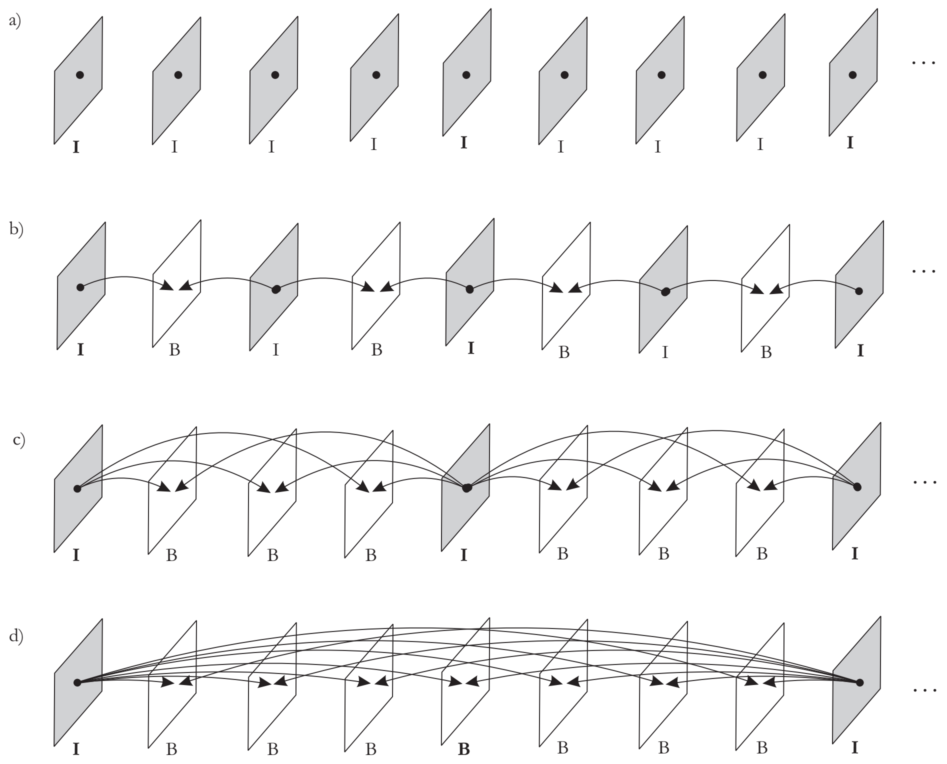


Figure 4.3: Different GOP sizes (B pictures): a) $N_0 = 1$, b) $N_0 = 2$, c) $N_0 = 4$, and d) $N_0 = 8$

image datasets under study, although the scene is assumed to be static and no moving objects exist in the environment, since the spacing between the image capturing spots is sometimes in the range of up to few meters, the rate-distortion performance starts to deteriorate at GOP8 and higher. This will be shown later in the simulation results section. It is noteworthy to mention here that the case of GOP2 is used for studio postproduction, high quality video for storage, and video distribution [55]. We propose GOP2 as well as GOP4, especially for image datasets with higher cross-correlation among neighbor basis images, in image dataset compression.

To study the performance of different frame structures, several experiments have been performed on captured image datasets. Experimental results will be shown later in detail.

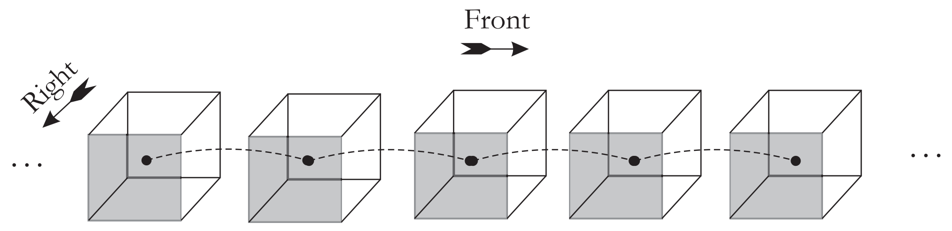


Figure 4.4: An example of the virtual video constructed by concatenating the right faces of the neighbor cubes

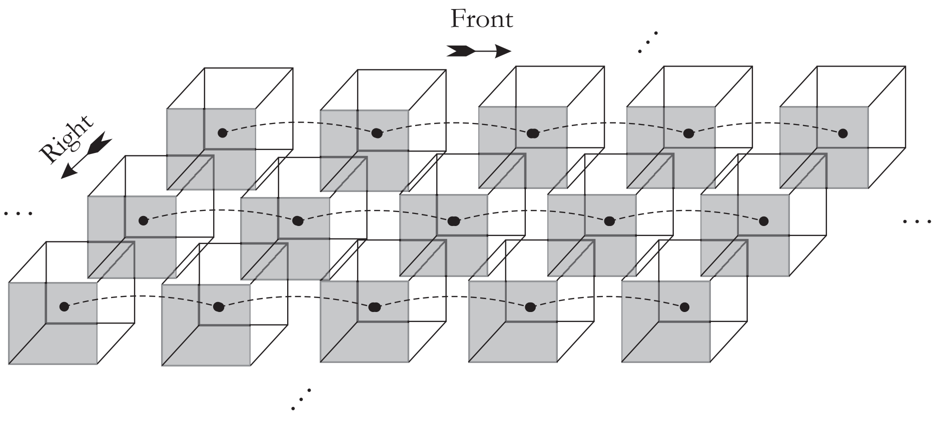
4.3 Architecture of the Proposed Compression Scheme

In the existing cubic panoramic image datasets, each basis image consists of six side images, i.e., *down, up, left, right, back*, and *front* side images. When images are captured on a linear trajectory, we are dealing with a 3D image dataset as in Figure 4.4 and when they are captured on a rectangular grid we have 4D image datasets as in Figure 4.5. For analysis and alignment of such image datasets visit our earlier work [93] explained in the previous chapter.

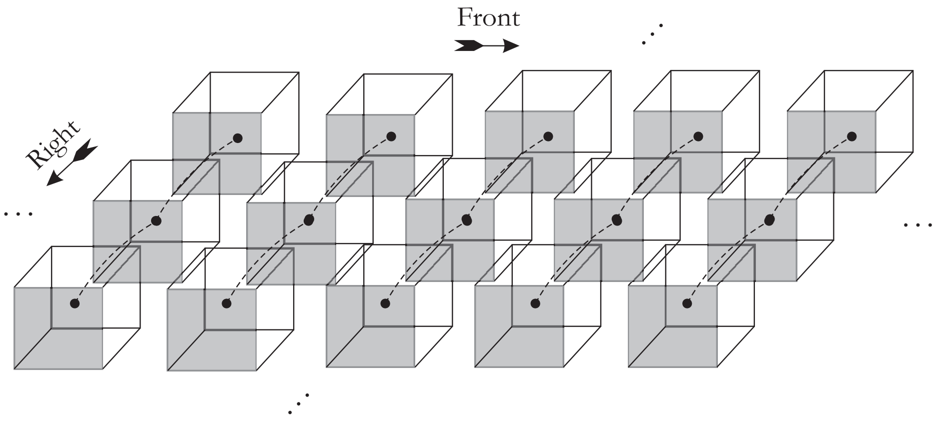
4.3.1 Image Acquisition and Indexing Scenarios

In an existing solution [10], the whole basis image including all of the side images, is regarded as one single piece of information and a 3D indexing method is used to provide access to the side images and required information within each basis image. However, through an approach that we propose in this section, each image dataset is decomposed into six video sequences and after that each video sequence is processed separately. A typical video sequence generated as an example by concatenating all the *right* side images is shown in grey in Figure 4.4. The advantage of such approach is that it will facilitate the selective decoding requirement down to the cube face level. That means to access (decode) one face of the cube, one does not necessarily need to decode other faces of that cube.

Now consider the case where images are captured on a rectangular grid where the outcome will be a 4D image dataset as shown in Figure 4.5. Here we will have options for the *right* and *left* side images. Video sequences, for the *right* side images for example, can be constructed by concatenating the frames on the rectangular grid of images in a column-wise manner instead of the already introduced row-wise approach for 3D datasets. Our simulations made on all *right* side images of TBT1 and NGC 4D image datasets, see



(a) without using the advantage of the new dimension



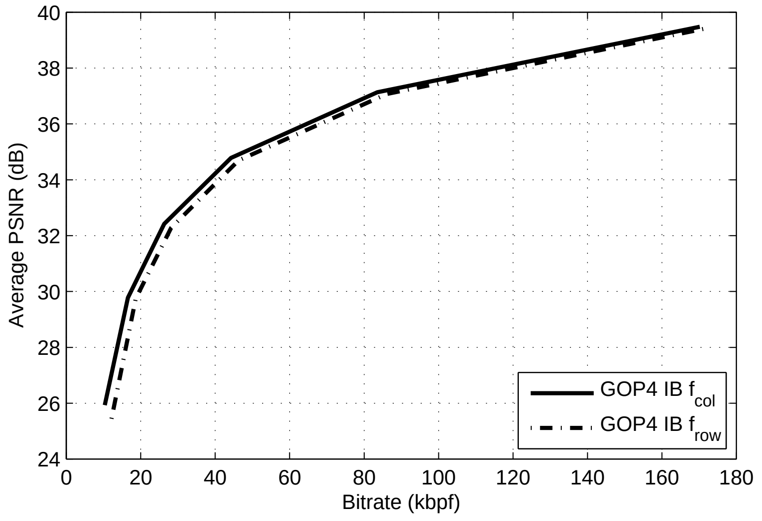
(b) proposed structure for prediction

Figure 4.5: 4D cubic-panorama image datasets, two different frame prediction scenarios

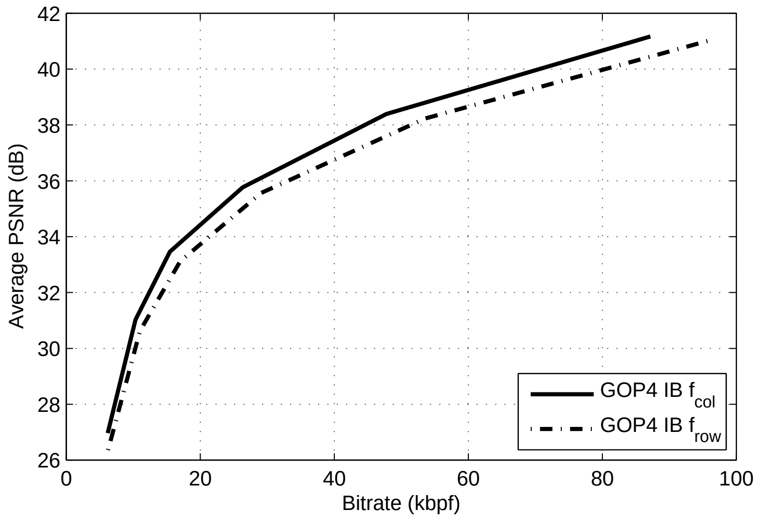
Figure 4.6, show that the former structure would result in better overall rate-distortion performance because, for example, for the *right* side images higher correlation exist among frames in each column as compared to that of *right* side images of each row. As a result, for the *down, up, back*, and *front* side images we use the row-wise prediction structure as in Figure 4.4, while for the *right* and *left* side images we prefer a prediction structure like what is shown in Figure 4.5(b).

4.3.2 Performance Comparison and Requirements

Performance of the two state-of-the-art video coding schemes, i.e. H.264 and Dirac video codec will be compared in the next section. Our results show that standardized H.264



(a) TBT1



(b) NGC

Figure 4.6: 4D cubic-panorama image datasets, comparing two frame prediction scenarios using PSNR measure, experiments made on *right* side images of TBT1 and NGC image datasets

outperforms the Dirac video codec, when tested on cubic-panorama image datasets. In the state-of-the-art video coding standard, H.264/AVC, different frame types are defined. They include I, P, B, and hierarchical B pictures. Rate-distortion performance of various prediction structures will be compared in the next section.

We prefer B frames for prediction in image dataset compression due to the following reasons and requirements:

Problem of Occlusion Parts of an object in the scene might not be visible in a neighboring frame while when two frames on both sides are used as reference for prediction, this will become a very rare possibility;

Random Access Requirement Only two intra frames need to be decoded before we can access a typical B frame, unlike the case of P or hierarchical B frames;

Compression Efficiency Since two references are used for prediction, higher compression ratios can be achieved as compared to the case when single frame is used;

View Scalability The view scalability feature can be facilitated similar to the idea of temporal scalability in the paradigm of classic video coding;

Boundary Problem If an object/feature is not visible in one neighboring frame due to camera displacement, most probably this object/feature exists in the neighboring frame on the opposite side;

Memory Constraints Unlike hierarchical B pictures, there are only two layers of frames, hence extra memory for storing additional reference frames will not be required.

Another advantage of using B frames is that they can be removed from the structure without disturbing the existing prediction structure. 3D image rendering applications are also special case of the image datasets explained before when we have only two columns/rows of basis images on the aforementioned rectangular grid.

The proposed overall prediction structure is shown in Figure 4.7 for GOP4. To be able to extend the existing prediction structure, i.e., 3D image datasets, we had to shift the structure of the image dataset at each row or column by one frame, so that each B frame will have two I frames on both sides, either left and right, or top and bottom. More specifically, in Figure 4.7 each square represents a top view of a basis image, i.e., each square side represents one side image, namely *left, back, right* and *front* counterclockwise

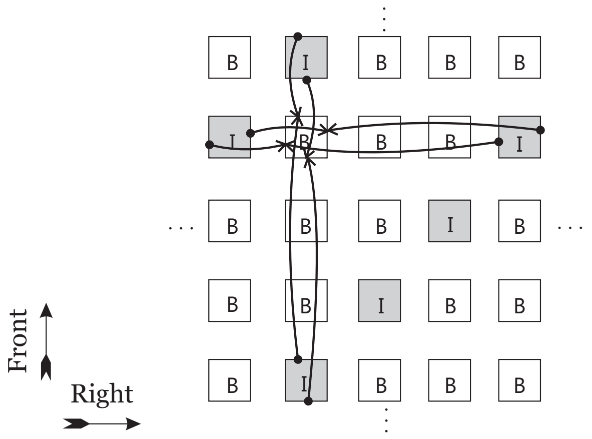


Figure 4.7: Top view of the proposed GOP structure for 4D image datasets

starting from the side image on the left. Any *back* and *front* side images in a basis image of type B is predicted with the corresponding *front* and *back* side images in the two I type basis images located to the top and bottom of the current B type basis image respectively. In a similar manner, any *left* and *right* side image is predicted with the corresponding *left* and *right* side images of the two I type basis images located to the left and right of the current B type basis image respectively. Finally, a similar structure is used for the *up* and *down* side images. As can be seen, a B type basis image uses four basis images of type I in prediction, two of them to the top and to the bottom and the other two, to the left and to the right of the current B type basis image. In order to decode one B type side image, as before, at most two side images of type I need to be decoded beforehand. At the borders of the image dataset rectangular grid, B frames can be replaced with P frames, since at those positions only one frame for prediction is available.

4.3.3 Missing Data, Nonuniform Datasets, Stereoscopic Panoramas

As can be seen in Figure 4.8 one advantage of using B frames is that they can be removed from the structure without disturbing the existing prediction structure. Figure 4.8(a) shows a scenario where some basis images are missing or when captured data were not originally uniform. Stereoscopic applications are also special case of the image datasets explained before when we have only two columns of basis images on the aforementioned rectangular grid. At least two prediction structures are possible and depicted in Fig-

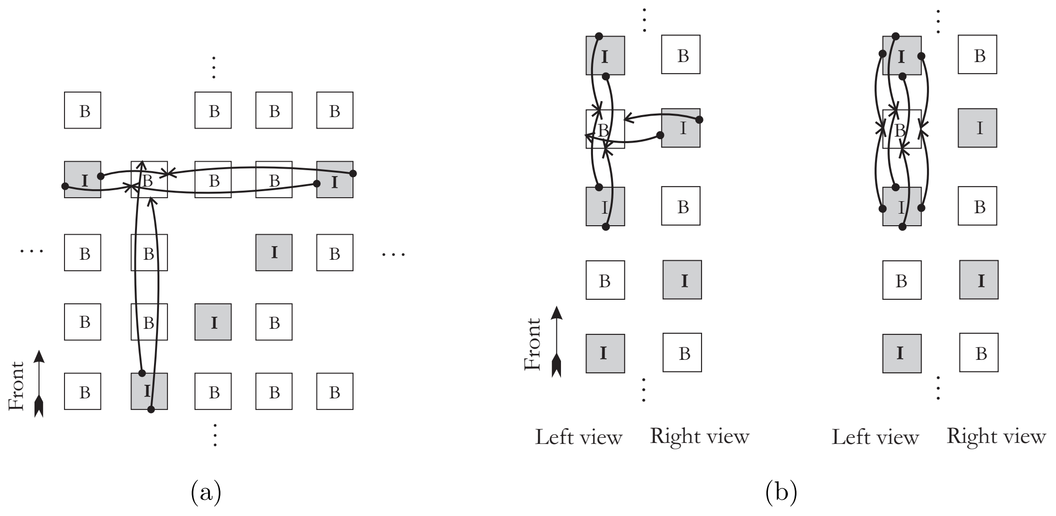


Figure 4.8: (a) Usefulness of *B* frames in case of nonuniform image datasets or missing data (b) Possible prediction structures for stereoscopic image rendering applications

ure 4.8(b), on the left the one based on what is proposed in the previous sections and on the right one in which the left views can be decoded for regular applications when stereoscopic display is not required or available. The one on the left is more efficient as far as compression efficiency is considered while the one on the right provides scalability between regular and stereoscopic applications. In order to see examples of work on cylindrical and spherical stereoscopic panoramas the reader is referred to visit [21] and [24] respectively. In [21] a technique is developed for efficient acquisition and rendering of omni-stereoscopic images based on sampling the scene with clusters of three panoramic images arranged in a controlled geometric pattern and in [24] a multi-scale approach for high resolution stereoscopic planar images are proposed first and then extended to be applicable to spherical-panoramas.

4.3.4 Bitstream Syntax

Now we present the final bitstream syntax for storage and compression of very large sized image datasets captured in extensive environments. First of all, as it was shown in Figure 3.7 in the previous chapter, two image capturing scenarios or even a combination of the two, i.e., a combination of Figures 3.7(a) and 3.7(b) might be used in practice. Also, in larger environments we might need to capture more than one set of image datasets to be able to cover the whole area for virtual navigation. Furthermore, in the storage unit, we

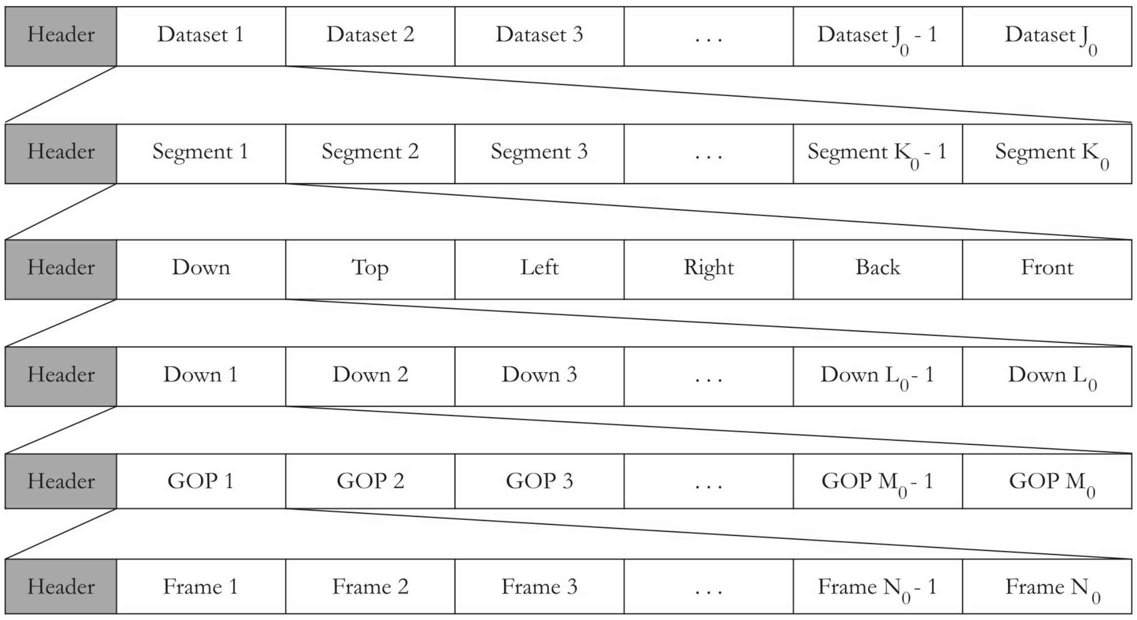


Figure 4.9: Proposed bitstream structure

will usually need to store information captured at different environments. The number of real world environments where images are captured is denoted by J_0.

The overall structure of data, at the application layer, is shown in Figure 4.9. An overall header, on top left side of the figure (in gray), will facilitate access to each panoramic image dataset. As was explained earlier, each image dataset consists of a number of trajectories or rectangular grids or a combination, where basis images are captured. We refer to each such unit of information as a *segment* inside an image dataset of a given environment and assume K_0 such segments exist within an image dataset. Each segment itself consists of six parts, namely, *down, up, left, right, back*, and *front* side image sequences. For the case of 4D signals, data at each side image is divided further into a number of rows or columns of data, indicated by L_0, see Figure 4.3. For 3D image datasets we will have $L_0 = 1$. Each side image sequence consists of a number of group of pictures, this number is indicated by M_0 in the picture. Finally, each GOP consists of N_0 distinct frames, GOP size, and frames are the smallest access units. Headers are shown in gray; they contain required side information and facilitate random access to the smaller units of data.

4.4 Experimental Results

Simulations will be made to compare the rate distortion performance of the standardized scheme with that of the wavelet-based approach applied on our captured image datasets as the input data. They include codec comparison, GOP size, and GOP structure analysis. It is important to mention that before generating the input video sequence for each side image, all images in the image dataset should be converted to the YUV 4:2:0 format because this format is usually used in standardized video compression schemes; also doing so will achieve some initial compression gain after down-sampling the two color components of images. We use the fact that human eye is not as sensitive to color data and this will not affect the perceived quality of the existing image datasets. Results are obtained by experimenting on 3D cubic-panorama image datasets and similar results are expected for the case of 4D cubic-panorama image datasets when the structure of figure 4.7 is used.

4.4.1 Codec Comparison

In this section we will compare the performance of the standardized based approach, H.264, with that of the wavelet-based method, Dirac. We will compare the rate distortion performance of the two approaches here, using two measures in estimating the distortion, namely: Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM) measure introduced in [97]. PSNR is the most commonly used measure while SSIM produces more reliable results when comparing two different video codecs.

In order to find the PSNR, one should calculate the Root Mean Square (RMS) error $\mathcal{E}_{\text{RMS}}$ first:

$$\mathcal{E}_{\text{MS}} = \frac{1}{N_1 N_2} \sum_{n_1, n_2} (\mathcal{I}_{\mathcal{B},k}[n_1, n_2] - \hat{\mathcal{I}}_{\mathcal{B},k}[n_1, n_2])^2 \tag{4.1}$$

Subsequently, PSNR can be obtained using the following equation:

$$\text{PSNR} = 10 \log_{10} \frac{(\mathcal{I}_{\mathcal{B},\max} - \mathcal{I}_{\mathcal{B},\min})^2}{\mathcal{E}_{\text{MS}}} \tag{4.2}$$

We use the PSNR value corresponding to the Y component of each frame as a measure reflecting the visual quality of the whole frame. Also we use the average PSNR which means average of the frame PSNR values over the whole sequence of side images. However, there is another alternative, called the overall PSNR, which finds the average

MSE over the whole image sequence first and then finds the PSNR based on the obtained average MSE value. For the image datasets that we are here dealing with, both measures would be suitable and we pick the average PSNR.

On the other hand, we have the SSIM[2] which is usually calculated and averaged over small blocks or windows w_i of size 8×8.

$$\mathrm{SSIM}(w_1, w_2) = \frac{(2\mu_{w_1}\mu_{w_2} + c_1)(2\sigma_{w_1 w_2} + c_2)}{(\mu_{w_1}^2 + \mu_{w_2}^2 + c_1)(\sigma_{w_1}^2 + \sigma_{w_2}^2 + c_2)}, \tag{4.3}$$

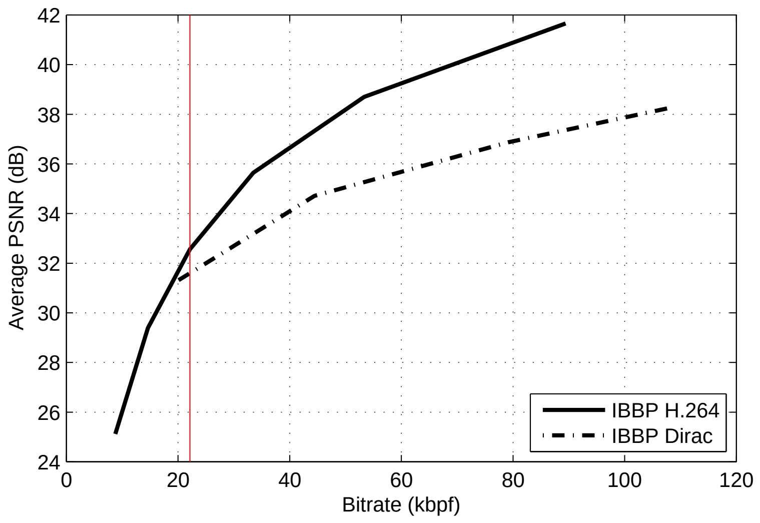
where μ_{w_i} is the average of w_i, $\sigma_{w_i}^2$ is the variance of w_i, $\sigma_{w_1 w_2}$ is the covariance of w_1 and w_2, and c_1, c_2 are variables to stabilize the division with weak denominator [97].

Experiments have been made on *front* side images of the LAB and CBY image datasets, as indoor and outdoor examples respectively. The existing reference softwares for both H.264/AVC[3] and Dirac video codec were used by finding and applying appropriate parameters in each software manual and by creating and applying appropriate input image sequences in each experiment. As it can be seen in the associated trajectories of Figures 3.8 and 3.9, different types of displacements are involved in the two image datasets and as a result they will constitute good examples of our image datasets for testing and comparison purposes. A combination of I, P, and B frames is used for the GOP structure denoted by $IBBP$, which is the most commonly used structure in conventional video sequence encoding. Rate distortion curves are shown in Figure 4.10, PSNR (in dB) versus bitrate (in kilobits per frame). As can be seen, the H.264/AVC based approach outperforms the Dirac video codec and the difference even increases at higher bitrates. Similar results using the SSIM measure is also depicted in Figure 4.11 where a more constant difference is visible throughout various bitrates. Figure 4.12 also shows the SSIM versus Compression Ratio (CR) values $\mathcal{R}_C$.

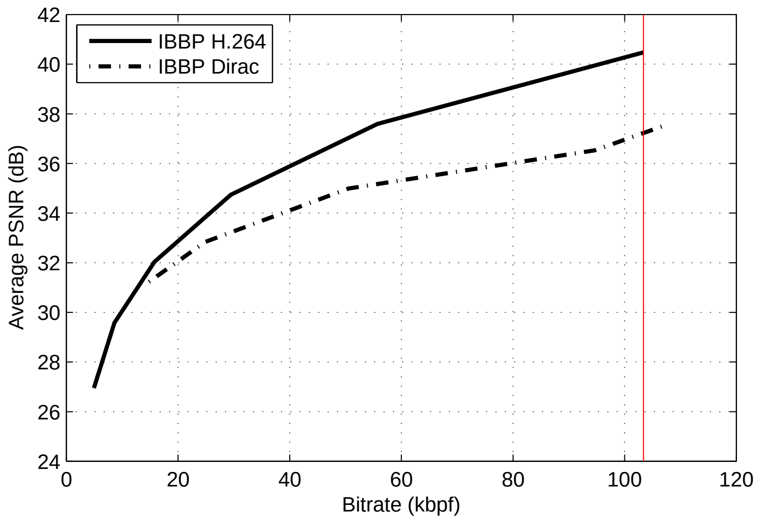
In order to take a closer look at the simulation results, average frame PSNR versus frame number for low bit rates corresponding to the vertical bar in figure 4.10(a) in the LAB image dataset is illustrated in figure 4.13(a). Average frame PSNR versus frame number for high bit rates corresponding to the vertical bar in figure 4.10(b) in the CBY image dataset is illustrated in figure 4.13(b). Notice that since GOP length of 12 is used in this experiment, usually big local maxima, i.e. frames 0, 12, 24, and so on, correspond

[2]Here is the link to the video quality measurement tool that was used in out work: http://compression.ru/video/quality_measure/video_measurement_tool_en.html

[3]The latest H.264/AVC reference software and its documentations including software manual can be accessed and downloaded here: http://iphome.hhi.de/suehring/tml/

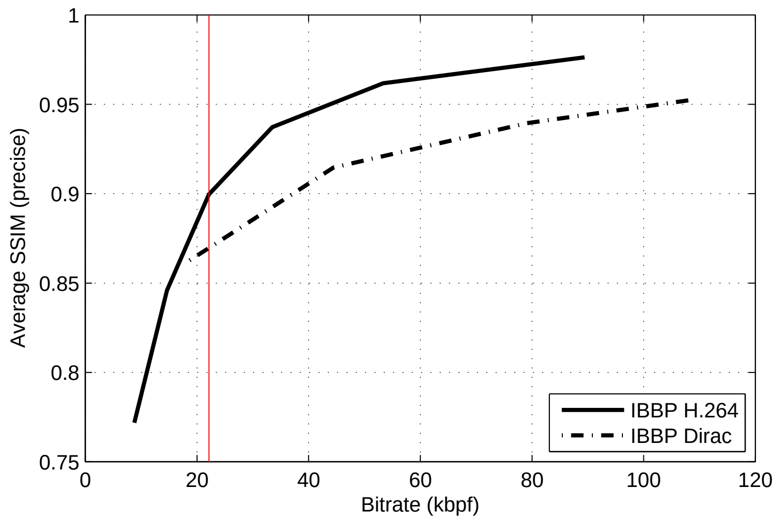


(a) LAB image dataset *front*

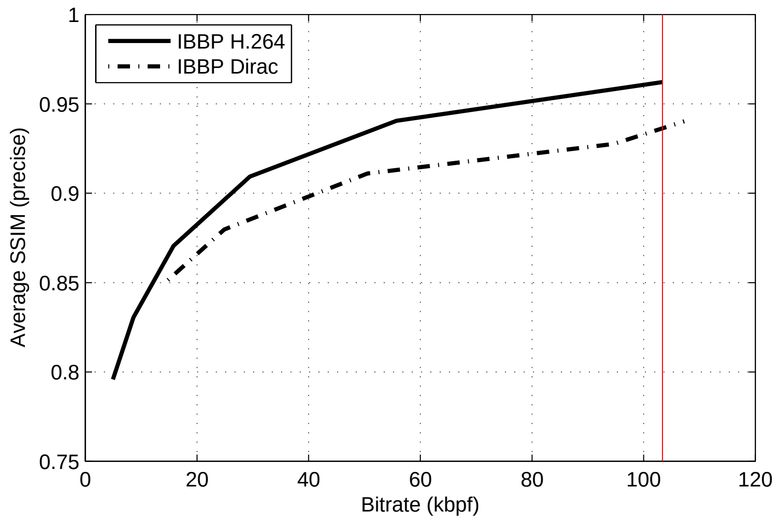


(b) CBY image dataset *front*

Figure 4.10: PSNR performance comparison between H.264 and Dirac video codec (versus bitrate)

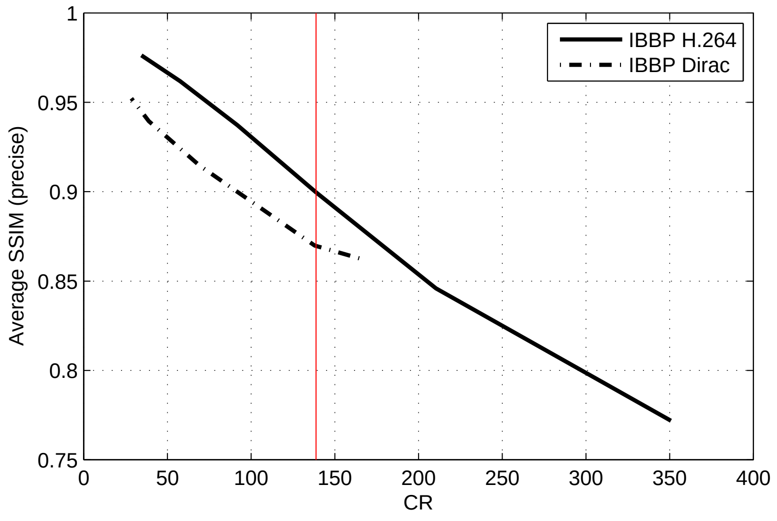


(a) LAB image dataset *front*

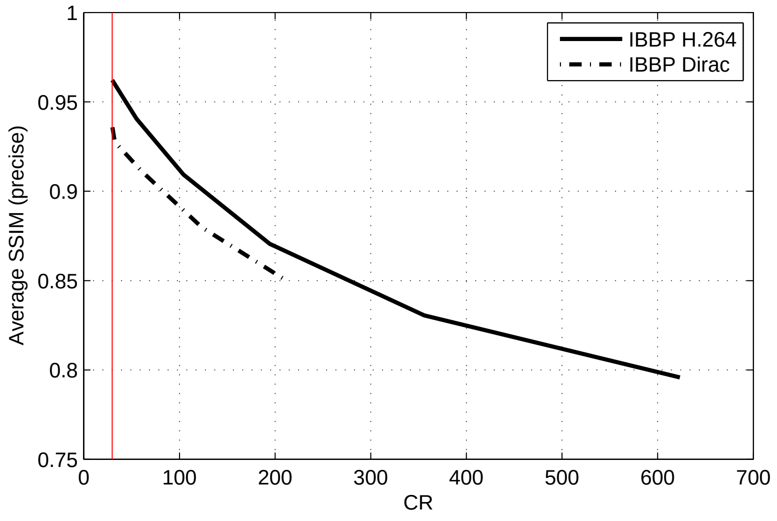


(b) CBY image dataset *front*

Figure 4.11: SSIM performance comparison between H.264 and Dirac video codec (versus bitrate)

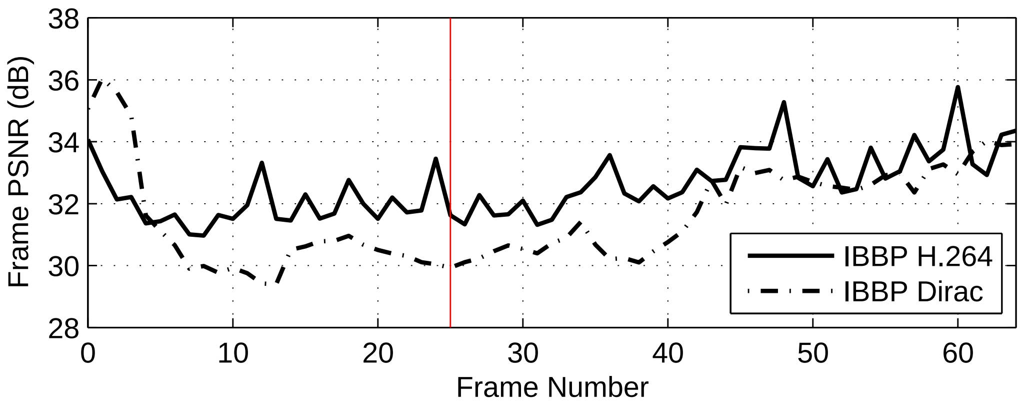


(a) LAB image dataset *front*

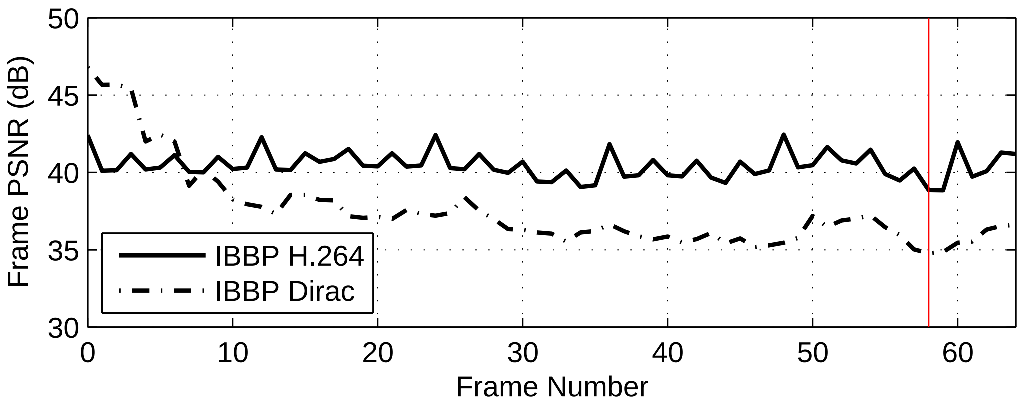


(b) CBY image dataset *front*

Figure 4.12: SSIM performance Comparison between H.264 and Dirac video codec (versus Compression Ratio)

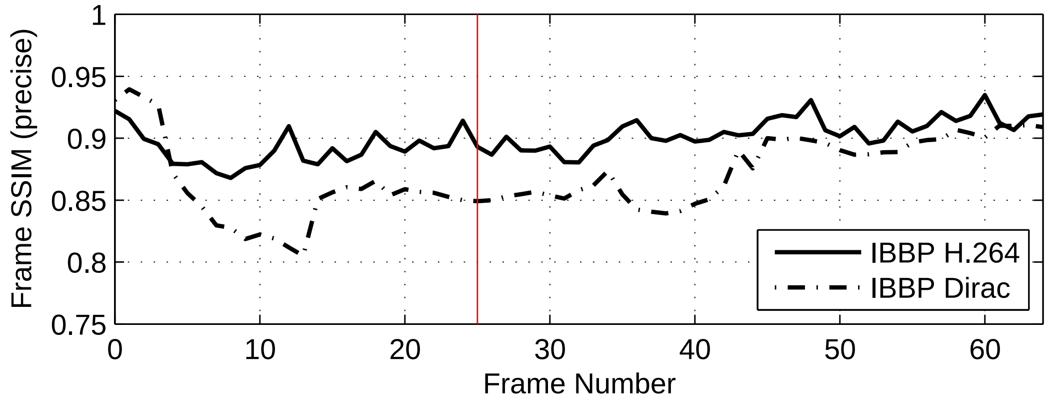


(a) LAB image dataset *front*

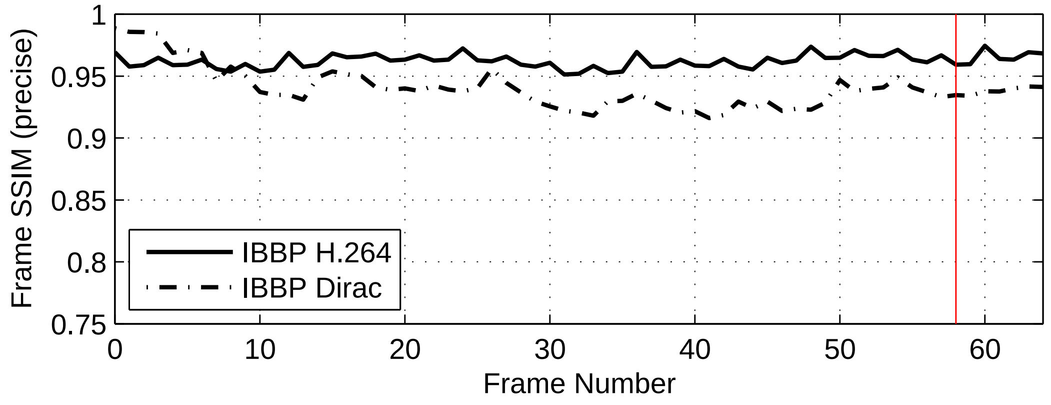


(b) CBY image dataset *front*

Figure 4.13: Frame PSNR versus frame number



(a) LAB image dataset *front*



(b) CBY image dataset *front*

Figure 4.14: Frame SSIM versus frame number

to I pictures and small local maxima, i.e. frames 3, 6, 9, and so on, correspond to P picture.

Similarly, average frame SSIM versus frame number for low bit rates corresponding to the vertical bar in figure 4.11(a) in the LAB image dataset is illustrated in figure 4.14(a). Average frame SSIM versus frame number for high bit rates corresponding to the vertical bar in figure 4.11(b) in the CBY image dataset is illustrated in figure 4.14(b).

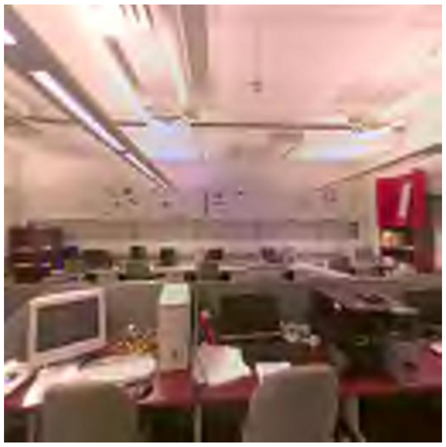
Finally, sample results are illustrated and can be compared subjectively for a single frame (side images corresponding to the vertical bars in figure 4.13 or 4.14) in each image dataset in Figures 4.15 and 4.16. Differences in the visual quality are visible in various areas in each frame. They include: chairs and set of computers in the foreground as well as the background in the LAB image sequence, and details in branches of the trees, at the foot of the tree, and on the bench and its texture at the foreground in the CBY image sequence.[4]

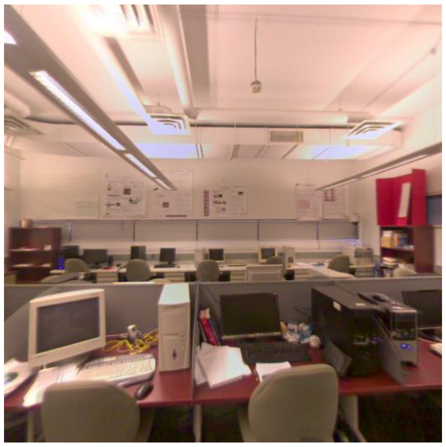
Altogether, standardized approach of H.264/AVC, proposed to be used in the compression stage of the virtual navigation system, outperforms the wavelet-based method of Dirac video codec both objectively throughout a range of bitrates, as verified through PSNR as well as SSIM calculations and subjectively through presenting typical side image examples as well as generated video sequences attached to this thesis.

4.4.2 GOP Size

Now we evaluate and compare the performance of different GOP sizes, namely $N_0 = 1$, 2, 4, and 8. Simulation results are shown in figure 4.17. As can be seen here, the case of $N_0 = 1$ possesses the lowest performance because this is an intra frame video coding scheme exploiting no cross-view redundancy in compression. As can be seen, $N_0 = 2$ and $N_0 = 4$ have comparable results, while the quality drops when N_0 reaches the value of 8. This is mainly because less correlation exists among frames located farther away inside any image sequence. We observe that, $N_0 = 2$ and $N_0 = 4$ will be appropriate values for the GOP size in the prediction structure. Figure 4.18 shows frame bits versus frame number for similar quality corresponding to the horizontal bar in figure 4.17.

[4]Video accompanying this thesis can be downloaded from the University of Ottawa thesis repository at www.ruor.uottawa.ca. It includes compression results for LAB and CBY cubic-panorama image datasets.


(a) Dirac video codec


(b) Original side image

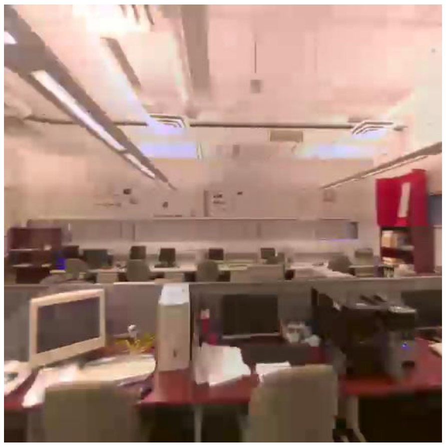
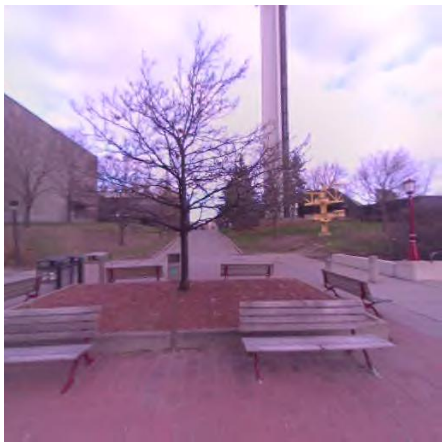
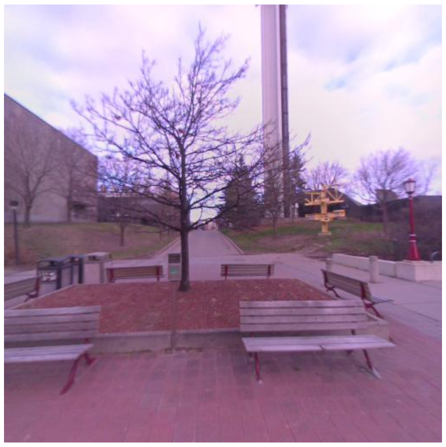

(c) H.264/AVC

Figure 4.15: Subjective comparison, LAB


(a) Dirac video codec


(b) Original side image

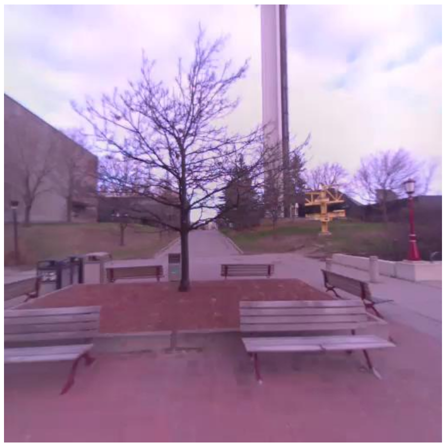
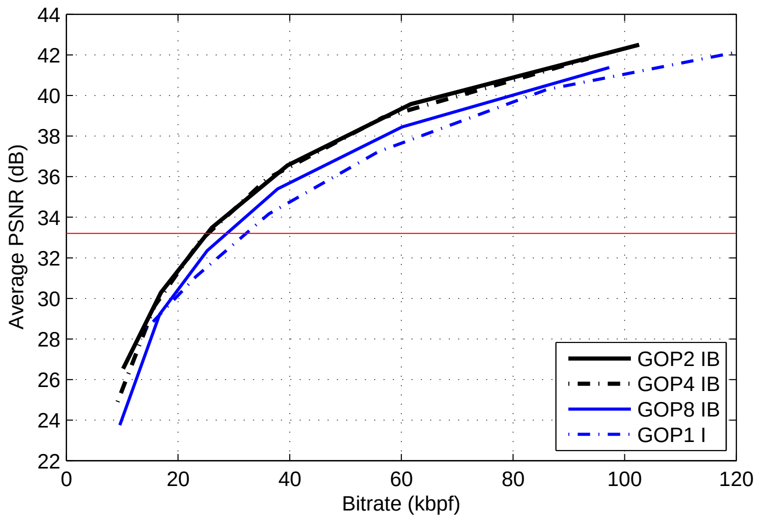
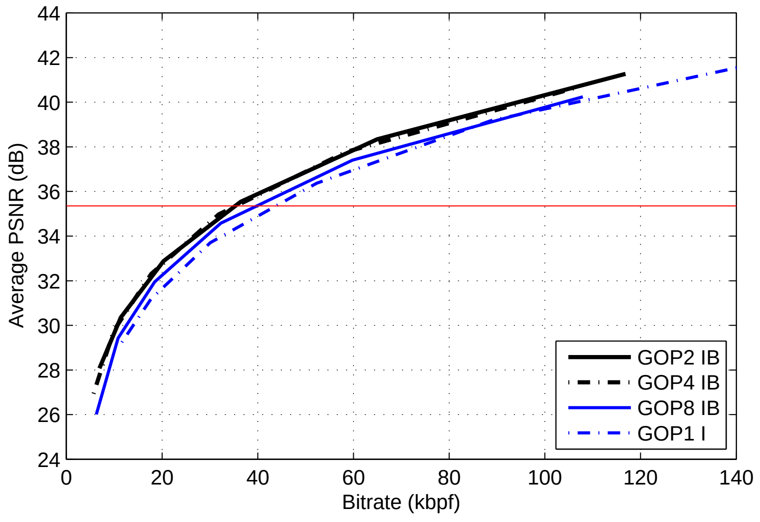

(c) H.264/AVC

Figure 4.16: Subjective comparison, CBY

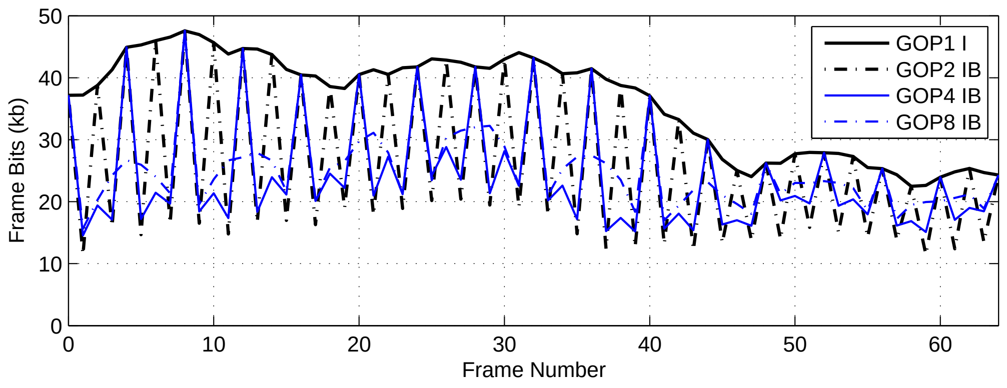


(a) LAB image dataset

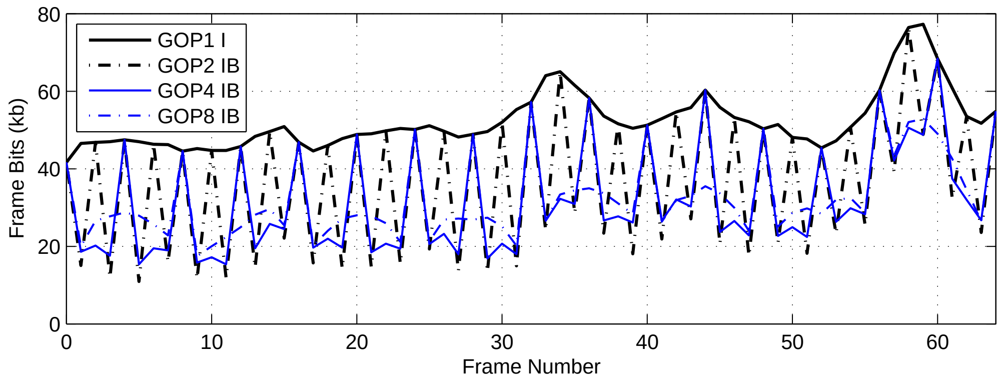


(b) CBY image dataset

Figure 4.17: Average PSNR versus bitrate, GOP size $N_0 = 1, 2, 4, 8$



(a) LAB image dataset



(b) CBY image dataset

Figure 4.18: Frame bits versus frame number (similar PSNR), GOP size $N_0 = 1, 2, 4, 8$

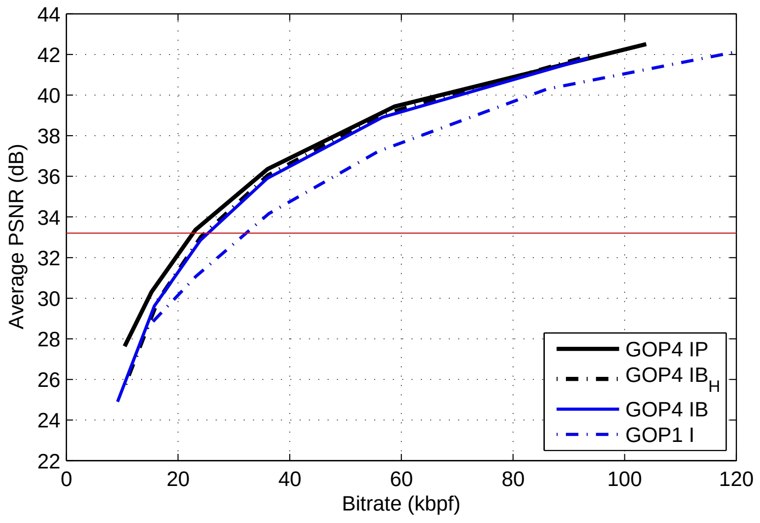
4.4.3 GOP Structure

Different frame types in our test image sequences are shown and compared in figure 4.19. As it can be seen here again, there is a distance between the case of intra frame coding $N_0 = 1$ and all the other inter frame schemes. P frame structure possesses a slightly better rate distortion performance at lower bitrates, while at higher bit rate it is the other way round. Also, although no direct comparison is provided in [76] between B and hierarchical B pictures for classic video sequences, here B pictures and hierarchical B pictures show a very similar performance for cubic-panorama image datasets captured in static environments. Therefore, we adopt the B frame structure due all the aforementioned advantages and requirements in the panoramic image data set compression.

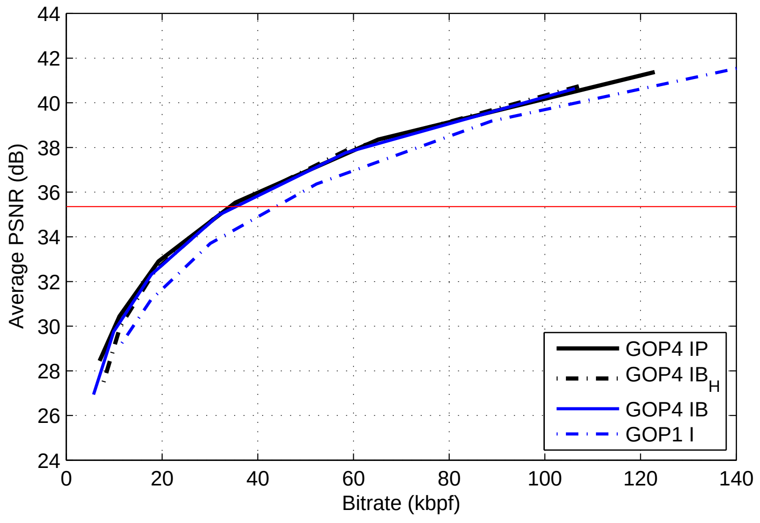
Finally, frame bits versus frame numbers, for a similar quality corresponding to the horizontal bar in figure 4.19, is shown in figure 4.20. In this figure, the difference between the number of bits required for encoding an intra I frame as compared to that of P, B, or hierarchical B frames is one interesting observation to notice.

4.5 Summary

In this chapter a novel compression structure suitable for cubic-panorama image datasets has been presented [98]. The advantage of the H.264 standard was verified against wavelet based methods for these new type of image datasets. It was shown that, as compared to the hierarchical B pictures or the $IBBP$ structure, effective use of B pictures makes our method effective in the case of panoramic image datasets captured nonuniformly on a rectangular grid in addition to fulfilling the compression/random access requirements of the system under study. Efficient prediction structure, especially the GOP size as well as the frame type and specific prediction structure was studied, keeping image dataset requirements, such as random access, in mind. In the next chapter, the idea of using the epipolar constraint in order to reduce the computational complexity of the encoding stage will be discussed. In addition, regarding the transmission stage, a new transcoding paradigm will be proposed as well as a method to deal with pan, tilt, and zoom in virtual navigation. Throughout the next chapter, our focus will be mainly on efficient disparity estimation.

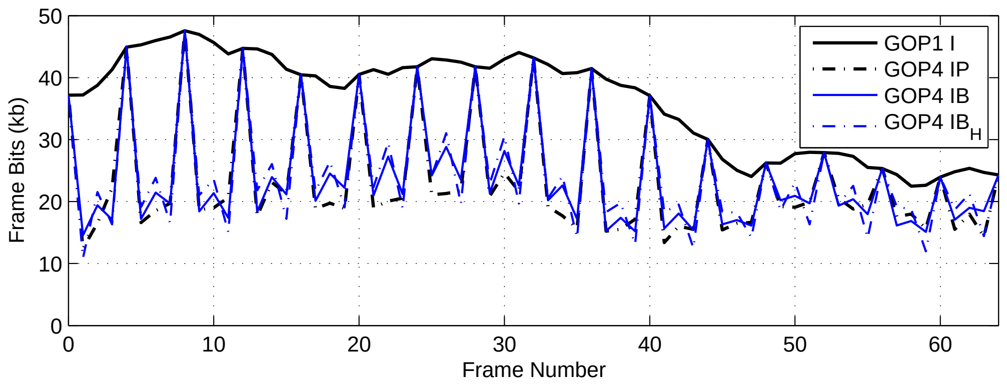


(a) LAB image dataset

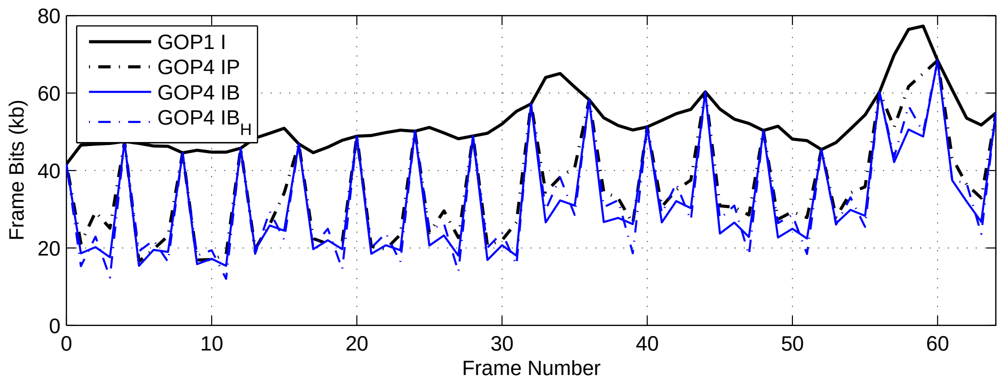


(b) CBY image dataset

Figure 4.19: Average PSNR versus bitrate, I, P, B, and hierarchical B GOP structures (GOP size $N_0 = 4$)



(a) LAB image dataset



(b) CBY image dataset

Figure 4.20: Frame bits versus frame number (similar PSNR), I, P, B, and hierarchical B GOP structures (GOP size $N_0 = 4$)

Chapter 5

Disparity Estimation for Storage and Transmission

In this chapter we address the problem of disparity estimation required for free navigation in the cubic-panorama image dataset compression stage as well as in the transmission stage. As for the storage case, a fast method that uses properties of the epipolar geometry, explained earlier in chapter three, together with the cubic format of panoramas is used to estimate disparity vectors efficiently. Assuming the use of $IBIB$ format, explained earlier in chapter four, the concept of forward and backward prediction is addressed and dealt with. As the transmission stage is considered, a new disparity vector transcoding-like scheme is introduced and a number of different frame-format conversion scenarios are addressed. Details on how to pick the best vector among disparity vector candidates is also explained. Different types of navigation including forward or backward motion, as well as pan, tilt, and zoom are explained separately. In all the above mentioned cases, results are compared both visually through disparity vector plots and error images as well as using the objective measure of PSNR versus time.

5.1 Introduction

In video coding, as well as in image dataset compression, motion or disparity estimation compromises 60-70% of the encoding complexity [99]. On the other hand, image datasets which we are dealing with possess specific features, as explained earlier in the introduction chapter, that can be exploited in order to decrease the encoding complexity to a great extent.

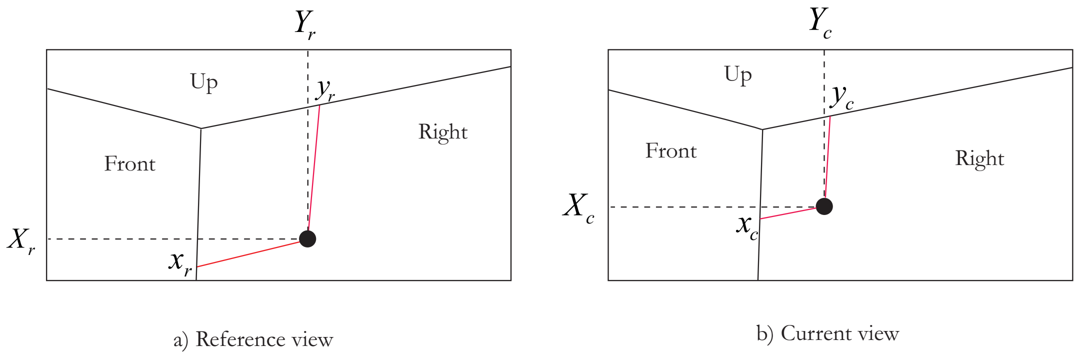


Figure 5.1: Disparity: current and reference rendered views when the user moves from one cubic-panorama to another

In an existing method for Multiview Image Coding (MIC) that works based on geometric prediction [86], the essential matrices between three frames are used, i.e., the current frame and the two previously encoded frames, in order to calculate the Disparity Vectors (DV) among the current frame and the reference frame (i.e., one of the previously encoded frames). The epipolar constraint and DV search need to be performed at the encoder and the decoder side. This method achieves some rate-distortion gain, by calculating the DVs at the decoder side, mainly at low bitrates because at this region disparity vectors dominate the residual error as compared to the high bitrates. However, encoding complexity increases 30% and the decoding complexity increases over 1000% when using full search algorithms and 100% when using fast search algorithms with some decrease in the coding efficiency. Nevertheless, since our goal first and foremost is to decrease the encoding complexity without increasing the decoding complexity, we will propose a method for disparity estimation for storage which works on two frames rather three frames used in the aforementioned existing work.

Subsequently, proposed methods for disparity estimation in the transmission stage can be divided into two categories: first, digital video transformation, and second, dealing with pan, tilt, and zoom. The block diagram of a heterogeneous video transcoder was shown in Figure 2.3. In chapter two, different existing types of video transcoding schemes in the literature were explained. They include: bitrate transcoding, spatial and temporal transcoding, standards transcoding, transcoding quality optimization, and information insertion transcoding [83]. In this chapter we propose a new type of video transcoding scheme which converts the input video sequence from cube image format, with six side images, into a planar image format that can be used for view rendering in the virtual

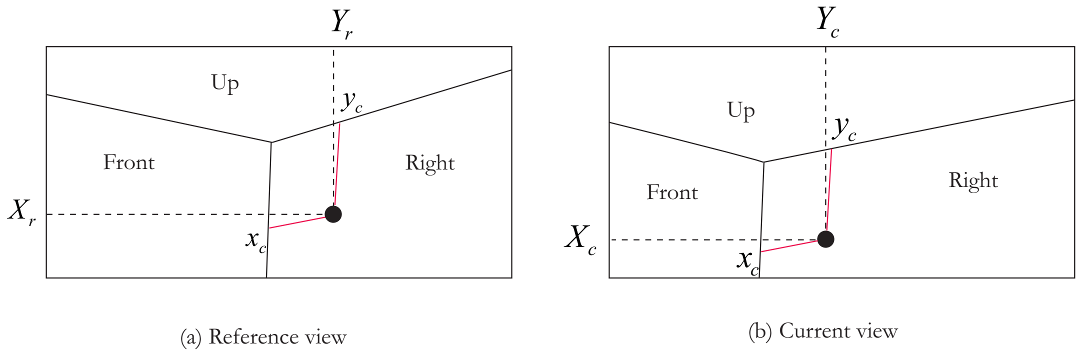


Figure 5.2: Disparity: current rendered views when the user does Pan, Tilt, and/or Zoom

navigation system. Our main concern is how to extract and reuse the existing disparity vectors previously calculated and stored. As it is shown in Figure 5.1(a), the applied view rendering technique uses a planar mapping approach and at each viewing location, renders a novel view based on the user's pan and tilt angles. In this scenario, in each view at most three faces of the current cubic basis image would be visible, e.g. *front, right* and *up* side images viewed from inside the cubic-panorama. Now, if the user moves forward to another viewing location, and possibly changes its pan and tilt angles too, then we will have a rendered view similar to what is depicted in Figure 5.1(b). Our aim is to extract up to three sets of disparity vectors for each rendered frame and generate a new set of disparity vectors suitable for new rendered views.

At the end, we will address the pan, tilt, and zoom scenario as shown in Figure 5.2. That is the case where the user is looking around but not changing the view point. A real navigation experience can be considered as the synthesis of these two types of experience. Eventually, our aim is to propose a method that enables the user to enter the navigation system at any desirable view point in the captured trajectory, be able to move in any direction of interest, look around to the left or right or top or bottom or zoom in and zoom out and turn back and move in the opposite direction. Altogether this will form a free navigation system.

5.2 Disparity Estimation for Storage

The approach that we present in this section can be considered as a mixture of global displacement estimation and local displacement estimation methods. We use the term global estimation because the essential matrix including the global translational and

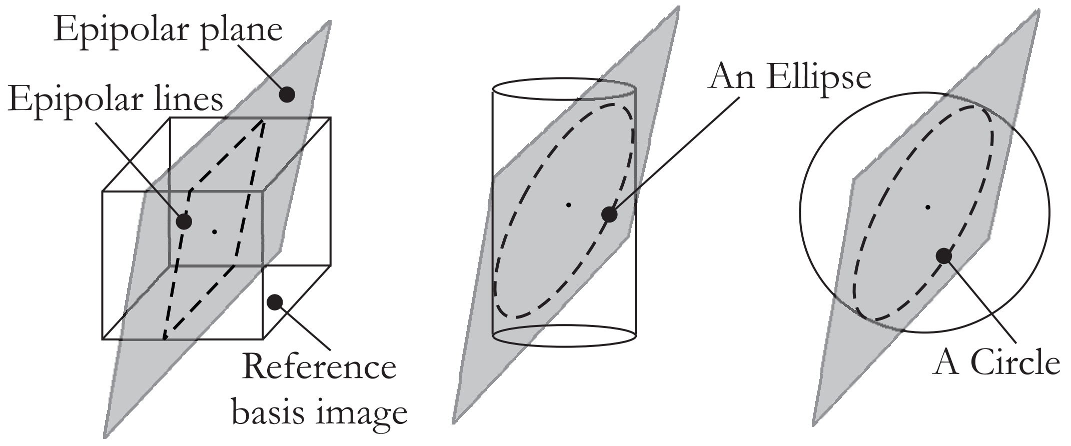


Figure 5.3: Epipolar curves (a Quadrilateral, an Eclipse, and a Circle) as the intersection of basis images (Cube, Cylinder, and Sphere respectively) with the epipolar plane

rotational parameters is involved, and also local estimation because further search on the epipolar line has to be made due to the depth variations of the objects in the environment.

Approaches that only rely on local displacement estimation, for example the block matching algorithm used in the standardized video codecs such as H.264/AVC, will not benefit from the existing knowledge of the image datasets, especially the fact that the scene is static and the existing information about the navigation direction. On the other hand, in global displacement estimation approaches, the scene depth is usually assumed to be uniform, or alternatively the camera translation is ignored [100].

5.2.1 Using the Epipolar Constraint

The relationship between the reference and current coordinates of a point in the 3D space was shown in chapter three in equation 3.3. Now if we know the coordinates of a point in the current basis image and insert the value in the equation, we obtain:

$$(x_r, y_r, z_r) \cdot \mathbf{E} \cdot p = (x_r, y_r, z_r) \cdot (a, b, c)^T = 0, \tag{5.1}$$

where $\mathbf{E} \cdot p$ is called the epipolar plane, a plane that goes through the center of the cube and is denoted by (a, b, c) as its normal. Therefore, the point on the reference basis image would lie somewhere at the intersection of the epipolar plane and the reference basis image, i.e., a cube in the 3D space, as shown in Figure 5.3.

This intersection forms a quadrilateral in the 3D space and since we know which face of the cube is being used for disparity estimation at any time, the search for the

Face	a_i	b_i	c_i
down	$-c$	$+b$	$g_d(-a)$
up	$+c$	$+b$	$g_u(+a)$
left	$-a$	$+c$	$g_l(-b)$
right	$-a$	$-c$	$g_r(+b)$
back	$-a$	$-b$	$g_b(-c)$
front	$-a$	$+b$	$g_f(+c)$

Table 5.1: Epipolar lines as intersections of cube and epipolar plane

disparity vector will be restricted to one dimension, i.e., on a line denoted as the epipolar line. Similarly, this constraint is known as the epipolar constraint in the literature. We will obtain the epipolar lines corresponding to each cube face using the relation between 3D reference coordinates and 2D side image coordinates $(x_r, y_r, z_r)^T = \mathbf{T}_j \cdot (X_j, Y_j, F)^T$ where F is the camera focal length (can be assumed 1 here) and transformation matrixes associated with each cube face given in equations 3.8 and 3.9 in chapter three. Results are summarized in Table 5.1. Different lines $l_i = (a_i, b_i, c_i)$ lie on the associated faces of the cube $(\forall i \in \{d, u, l, r, b, f\})$.

The functions g_i in Table 5.1 are defined as below:

$$g_i(m) = \frac{L}{2}(m - (a_i + b_i)) \tag{5.2}$$

Figure 5.4 shows the overall directions of displacement in a typical pair of cubic panoramas. Roughly speaking, the global displacement for *left*, *right*, *up*, and *down* faces are similar to vertical or horizontal displacements while in case of *front* and *back* faces we witness a zooming in or zooming out effect.

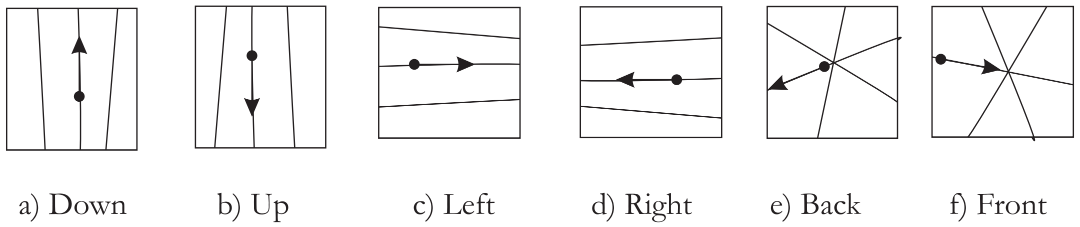


Figure 5.4: Overall directions of displacement on different cubic-panorama faces in a typical pair of cubic panoramas

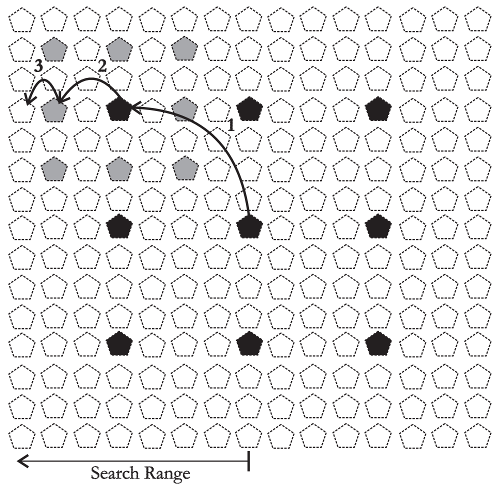


Figure 5.5: Three Step Search (TSS) algorithm, Search Range = ± 7

Before elaborating on our disparity search method, we explain the motion estimation methods commonly used in the video coding literature. In standardized video compression schemes, each frame is usually divided into small blocks, for example 8×8 pixels. For each block in the current frame, a Block Matching Algorithm (BMA) is applied in the reference frame, i.e., usually the previous one. Given a specific search range, Full Search Algorithm (FSA) is the most trivial one in which all possibilities are examined and the best match using a minimization measure is chosen. Minimization criteria include Sum of Squared Differences (SSD) and Sum of Absolute Differences (SAD) as explained earlier in chapter two. FSA method is computationally so exhaustive therefore a number of fast methods have been introduced. They include, but are not limited to, Three Step Search (TSS) and Cross Search Algorithm (CSA). Figure 5.5 shows how the TSS algorithm works. Given a Search Range of ± 7 pixels, the search is performed in three steps. All eight positions surrounding the coordinate with a step size of 4 are searched first. At each minimum position the search step size is halved and the next eight new positions are searched. This method searches 25 positions to locate the best match. If higher search ranges are desired, then one might start with bigger initial step size, for example 8. In that case it will be a Four Step Search (4SS) algorithm where the search range will be ± 15 and 33 positions are searched to locate the best match. In the same manner one can imagine Five Step Search (5SS), Six Step Search (6SS), Seven Step Search (7SS) algorithms and so on. Search range and computational complexity for each algorithm is summarized in the first part of table 5.2. Using fast estimation methods computational

Method	Complexity	Range
FSA	225	± 7
TSS	25	± 7
4SS	33	± 15
5SS	41	± 31
6SS	49	± 63
7SS	57	± 127
Disparity Estimation for Storage		
Dir	$N+4$	$\pm(4(N-1)+3)$
Dir + refinement	$N+12$	$\pm(4(N-1)+4)$
Disparity Estimation for Transmission		
Best	5	-
Median, Mean, and Rand	2	-
Best + refinement	13	-
Median, Mean, and Rand + refinement	10	-
Pan, Tilt, and Zoom		
PTZ	0	-

Table 5.2: Computational complexity and search range comparison

complexity reduces in a logarithmic scale.

Our method of search along the epipolar line is inspired by the above mentioned TSS algorithm. First, for center coordinates of each block on the current frame, we find the corresponding epipolar line using table 5.1. Then we find the nearest point on the epipolar line to the center coordinates of corresponding block. This point will be the reference point for our search and since we know the direction of displacement in advance, we need only to search on one side of the reference point on the epipolar line; see figure 5.6 for forward movement and the *right* reference face. A one dimensional three step search algorithm named Dir, referring to the directional nature of the search operation, is proposed. If N search operations are used in the first step, using initial step size of 4, we achieve search range of $\pm(4(N-1)+3)$. In this situation, $N+4$ search operations will be required to find the best match. If further refinement is desired for compensating computational errors, eight adjacent pixels around the found match can be searched and therefore the computational complexity will become $N+12$ and the search range increments 1. Search range and computational complexity for each algorithm is

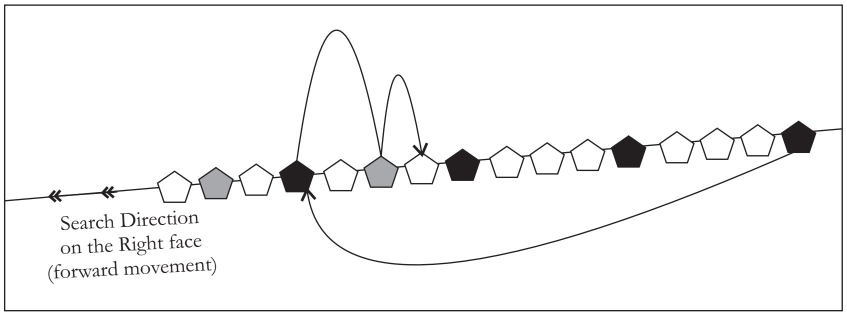


Figure 5.6: Proposed three step Directional search (Dir) on the epipolar line

summarized in the second part of table 5.2. We will refer again to this table later in the subsequent section.

Since we are working on panoramas in cubic format, there will be occasions where the search algorithm goes beyond one face boundary. Figure 5.7 depicts a good example for the case of *right* face. In case the search algorithm stops in the face it was started in, like the case indicated by scenario 1, obtained horizontal and vertical disparities are used directly. In case the best match found is beyond the borders, for example the case of scenario 2 which is on *front* face, we store the horizontal and vertical disparities as shown in the figure. Since in disparity estimation we are interested in two dimensional disparity vectors, we do not involve 3D coordinates of match points here. Similar awareness exists at the time of disparity vector extraction and if displacements go beyond the coordinates of the existing face, it means data on the corresponding neighbor face shall be used.

As shown in chapter four, the $IBIB$ is an appropriate format for storage of cubic-panorama image datasets. In addition to the bitrate efficiency, this structure facilitates entering into the captured sequence at any I frame and as we will show in more detail in the current chapter, the user will be able to walk freely in either direction which is very important. Introduction of B frames brings with it two types of prediction structures, namely forward and backward prediction as shown in figure 5.8. In other words, if the middle frame in the figure is the current frame, depending on which other frame is considered to be the reference frame, the direction of the search algorithm will vary. Considering the case of *front* face, in the forward prediction scenario, the search operation must be performed toward the epipolar center, usually in the middle of the cube face, while in the backward prediction structure, search operations must be performed away from the epipolar center. As a result, this forward or backward prediction choice will

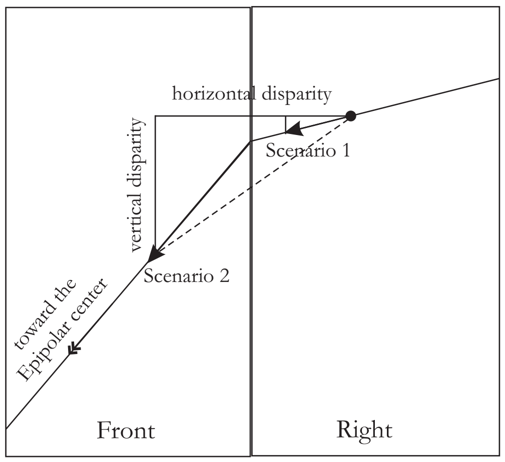


Figure 5.7: An example when the search operation goes beyond the side image boundary

show up in all the steps used in our work as we will see soon.

Flowcharts for disparity estimation corresponding to the *right* and *front* faces are shown in figures 5.9 and 5.10 respectively. At the beginning, the essential matrix E between the pair of cubic panoramas is calculated using methods explained in chapter three. First consider the *right* face. The intersection of the epipolar plane with the cubic shape is calculated and the intersection line is found using the information in table 5.1. Then the nearest point on this epipolar line to the search reference coordinates, i.e. the center of current 8×8 block, is found and used as the reference in the search algorithm. At this stage, if forward prediction is going on, the search operation is performed toward the left side, as shown in figure 5.6. Otherwise search operation is performed towards the right side. In either case, if the search operation goes beyond the current face, i.e., to the *front* face or the *back* face, the first step of search is continued there and the best match coordinates are updated accordingly. Otherwise the next two steps of search are done. At the end, in case refined results are desired, a refinement stage is also applied.

In the case of searching for disparities on the *front* face, after finding the nearest point on the epipolar line, the corresponding epipolar center is also calculated. In the case of forward prediction, the search operation is performed inwards, i.e., toward the epipolar center. Otherwise the search operation is performed outwards, i.e., away from the epipolar center and towards the face borders. In backward prediction, in case the search operation goes beyond the current face, the search operation is continued on the *left* (or the *right*) face and the search results are updated accordingly. The remaining

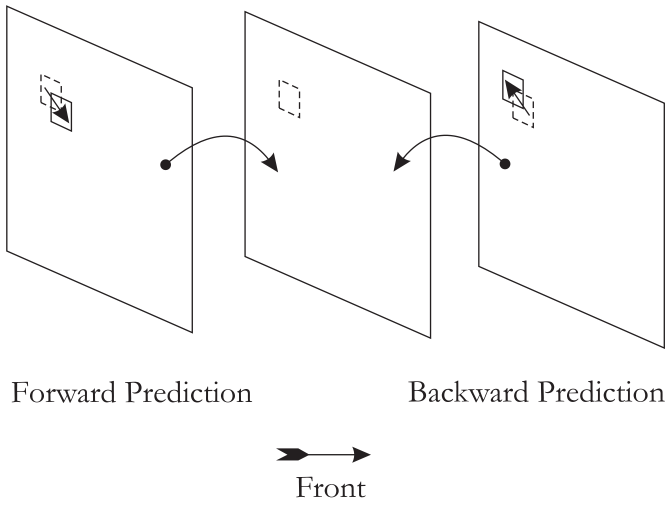


Figure 5.8: Forward and backward prediction scenario in B frames

steps are similar to the previous flowchart.

Finally, similar to the conventional block-based coding schemes, the disparity vector of each block is predicted as one of its neighboring disparity vectors, and then the difference between the true (or current) and predicted disparity vector is encoded using a variable-length coder. The median predictor is widely used in state-of-the-art coding schemes, see figure 5.11. The predictors are calculated separately for the horizontal and vertical components of the disparity vectors DV1, DV2, and DV3, see figure 5.12(a). For each component, the predictor is the median value of the three candidate predictors for this component:

$$pred_x = median(DV1_x, DV2_x, DV3_x) \tag{5.3}$$

$$pred_y = median(DV1_y, DV2_y, DV3_y) \tag{5.4}$$

The difference between the components of the current disparity vector and their predictions are variable length coded. The vector differences are defined by:

$$DVD_x = DV_x - pred_x \tag{5.5}$$

$$DVD_y = DV_y - pred_y \tag{5.6}$$

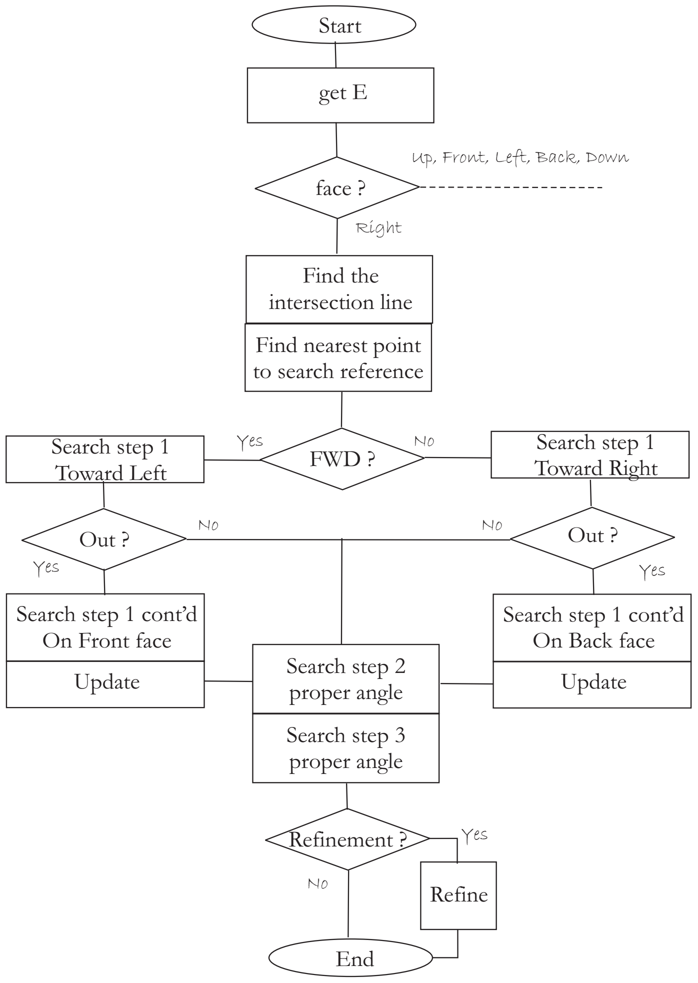

Start
get E
face ?
Up, Front, Left, Back, Down
Right
Find the intersection line
Find nearest point to search reference
Yes
FWD ?
No
Search step 1 Toward Left
Search step 1 Toward Right
Out ?
No
Yes
Out ?
No
Yes
Search step 1 cont'd On Front face
Update
Search step 1 cont'd On Back face
Update
Search step 2 proper angle
Search step 3 proper angle
Refinement ?
Yes
Refine
No
End

Figure 5.9: Flowchart for Dir method when the *right* face is concerned

In the special cases at the borders of the picture, the following decision rules are applied in order as follows:

1. The candidate predictor DV1 is set to zero if the corresponding block is outside the picture at the left side (figure 5.12(b)).

2. The candidate predictors DV2 and DV3 are set to DV1 if the corresponding blocks are outside the picture at the top (figure 5.12(c)).

3. The candidate predictor DV3 is set to zero if the corresponding block is outside the picture at the right side (figure 5.12(d)).

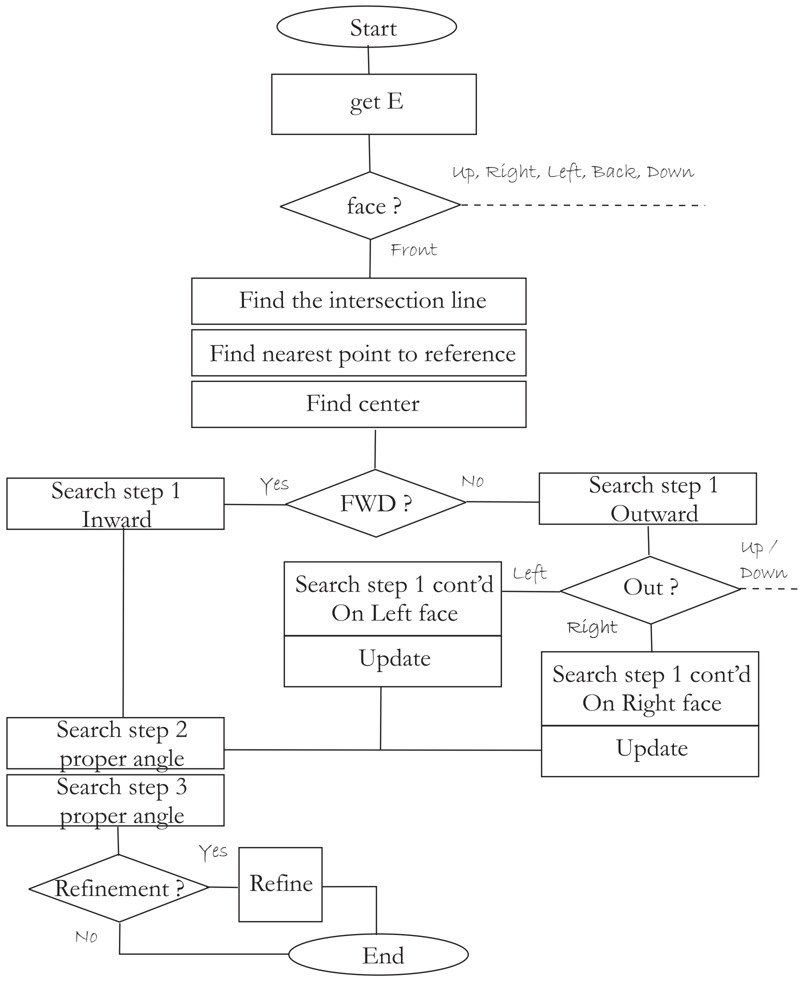

Start
get E
face ?
Up, Right, Left, Back, Down
Front
Find the intersection line
Find nearest point to reference
Find center
FWD ?
Yes
No
Search step 1 Inward
Search step 1 Outward
Out ?
Left
Right
Up / Down
Search step 1 cont'd On Left face
Update
Search step 1 cont'd On Right face
Update
Search step 2 proper angle
Search step 3 proper angle
Refinement ?
Yes
No
Refine
End

Figure 5.10: Flowchart for Dir method when the *front* face is concerned

5.2.2 Experimental Results

A number of experiments are made on CHPL and LAB cubic-panorama image datasets as explained below. We use *front* face and *right* face of cubic-panoramas to include all types of displacement. Latest version of Matlab is used for experiments of this chapter. Our directional method named Dir is compared with Full Search Algorithm (FSA) and fast step search algorithms explained earlier in this chapter as well as the case where no disparity estimation is performed. When using the Three Step Search (TSS) or similar algorithms, disparity vectors are predicted using median predictor similar to what was explained in equations 5.3 and 5.4. In our Dir algorithm no prediction is required since for each block the reference for search is calculated based on epipolar line calculations

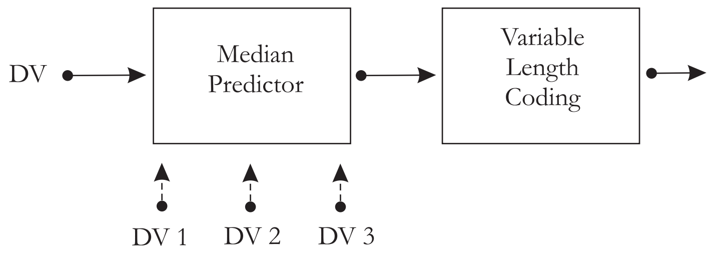


Figure 5.11: Encoding of disparity vectors using median predictor and variable length coding

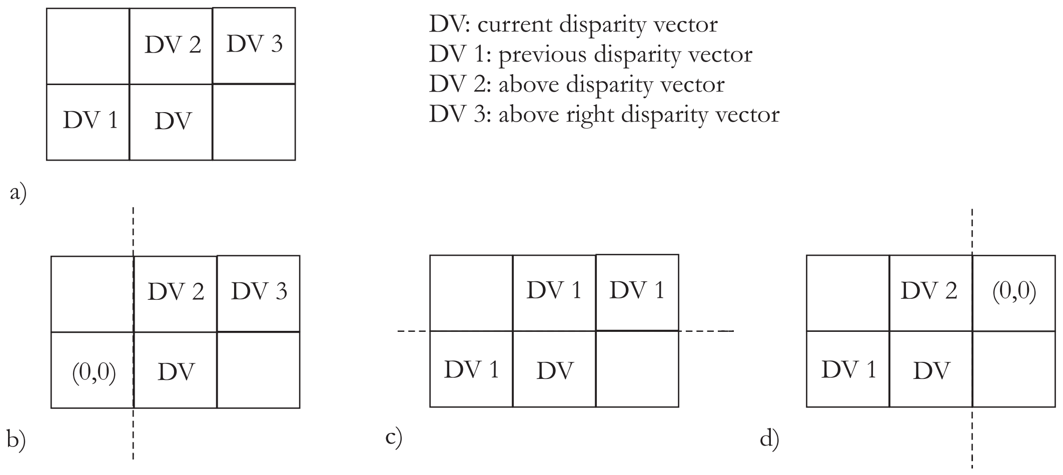
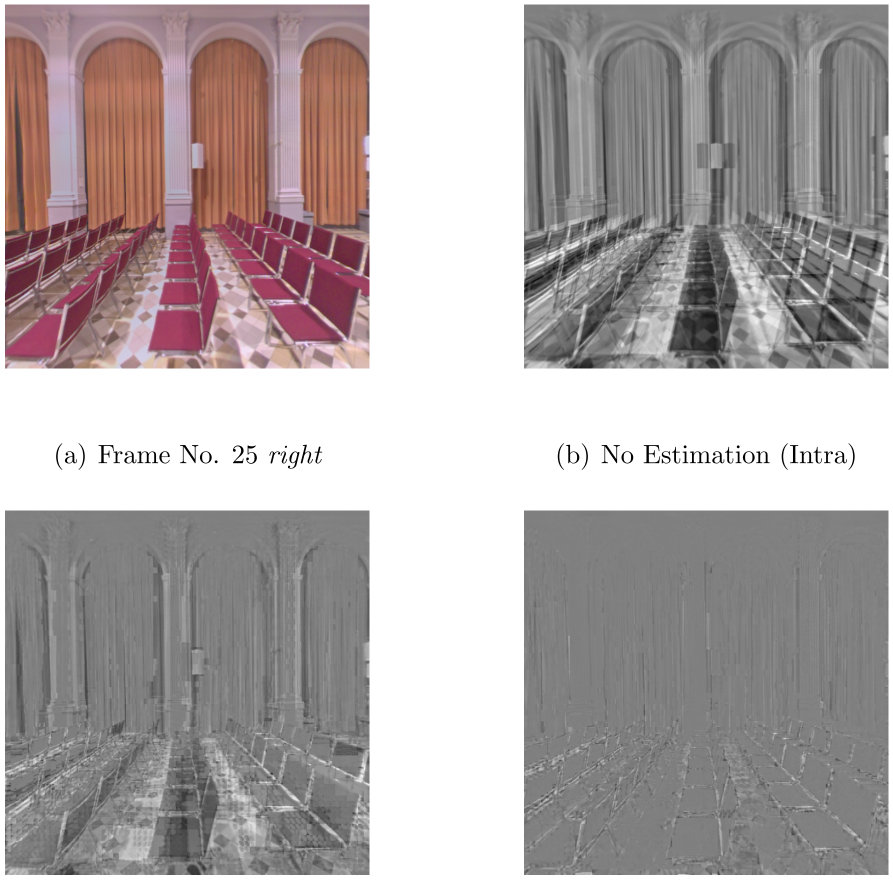

DV 2
DV 3
DV 1
DV
DV: current disparity vector
DV 1: previous disparity vector
DV 2: above disparity vector
DV 3: above right disparity vector
a)
DV 2
DV 3
(0,0)
DV
b)
DV 1
DV 1
DV 1
DV
c)
DV 2
(0,0)
DV 1
DV
d)

Figure 5.12: Disparity vector prediction structure at the borders of a side image

independently and we are even aware of the search direction. Disparity compensation error and associated PSNR outcomes are used in objective assessment of performances as well as the computational complexity or time in seconds. Assuming that $IBIB$ prediction structure is desired, the middle frame must be predicted using its immediate neighbors on both sides. In B frames, disparity compensation is performed forward and backward and the average disparity compensation error is calculated. In order to simulate an encoding procedure, obtained disparity vectors are eventually encoded using median predictor of figure 5.12 followed by the huffman variable length coding procedure that is available in Matlab using appropriate category based on search range and maximum absolute value of disparity vector differences.

To start, we pick three consecutive frames of cubic-panoramas, for example *right* faces number 24, 25, and 26 from CHPL image dataset. Frame number 25 is shown in figure



(a) Frame No. 25 *right*

(b) No Estimation (Intra)

(c) Three Step Search (TSS) algorithm

(d) Directional Search (Dir) algorithm

Figure 5.13: Residual error images (CHPL image dataset*right*, side images No. 24, 25, and 26)

5.13(a). Next, the essential matrix for pairs number 24 and 25 and also pairs number 25 and 26 is calculated using the algorithm explained in chapter three. We compare results with Intra and TSS methods. Intra simply means the case where no disparity estimation is performed and the error image is the difference between the current frame and the reference frame; since in B pictures we have two reference frames, the average of the two is used as the reference. This is shown in figure 5.13(b) and shows how the three frames under study differ without processing. For disparity estimation each frame is partitioned into 8×8 blocks. We use the luminance component of each image for disparity estimation. The TSS algorithm needs 25 operations for each block as shown in table 5.2 when search range ± 7 pixels is desired. To have a similar computational complexity in the Dir algorithm we set $N = 21$ so that the computational complexity $N + 4$ would be the same as in TSS algorithm. In this case the search range would be $\pm(4(N-1)+3) = 83$ pixels. At the borders we use the scheme explained in figure 5.7. No refinement is used at this point to maintain similar computational complexity.

Results for this experiment can be compared in figures 5.13(c) and (d). Notice that for better visualization, the value of 0.5 is added to disparity compensation error in each case. Notice also that in the CHPL sequence, disparity compensation error is higher where the chairs are placed due to different depths as well as on the walls where we have a large number of vertical columns. With similar computational complexity, a much better outcome is achieved using the proposed Dir method. Similar experiments were made on *right* faces of CHPL sequence as well as the *LAB* sequence which has a different visual content. Also FSA is used to see how results will improve when more computational complexity is allowed as compared to TSS algorithm. FSA increases the computational complexity drastically as shown in table 5.2 from 25 to 255 while our experiments show that negligible quality improvement is achieved and that is mainly due to the same short search range of ± 7 pixels. Instead, for a better comparison 4SS or higher step search algorithms are used later in this section.

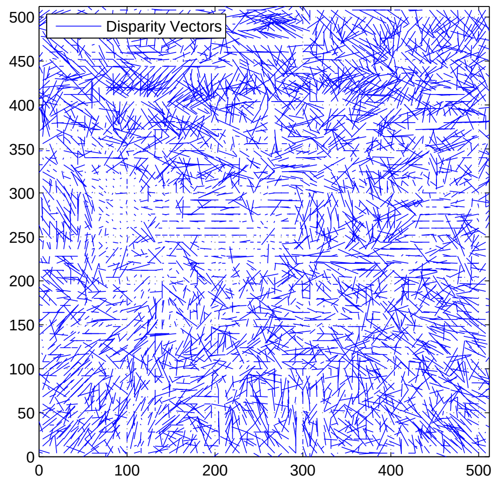
As an example estimated disparity vectors for indoor sequence LAB, faces number 36 and 37 are shown in figure 5.14. Specifically 7SS algorithm has computational complexity of 57 and search range of ± 127, i.e., the first step search algorithm with higher search range as compared to Dir. Obtained disparity vectors are depicted visually in figures 5.14 (a), (b), and (c) for 5SS, 6SS, and 7SS algorithms respectively and disparity vectors for Dir is shown in 5.14(d). Here again we see that even 7SS algorithm with higher search range can not grasp the nature of real disparity vectors.

Results regarding disparity estimation for storage are summarized in table 5.3. Experiments are made on all 65 *front* and *right* frames of CHPL and LAB image sequence. In one set of experiments using Dir, final refinement is also applied to make an improvement and also be able to compare the results using quality versus computational complexity curves. Time percentage required for estimating the essential matrix is also calculated. That shows the ratio of computational complexity required for global disparity estimation versus local search operations. These results are depicted as disparity compensation error PSNR versus spent time for each scenario in figures 5.15 and 5.16 in our four experiments. One interesting observation is that using step search at higher search ranges in CHPL sequence disparity compensation error increases after a certain point. That means although the overall search range is increasing, the accuracy of vector

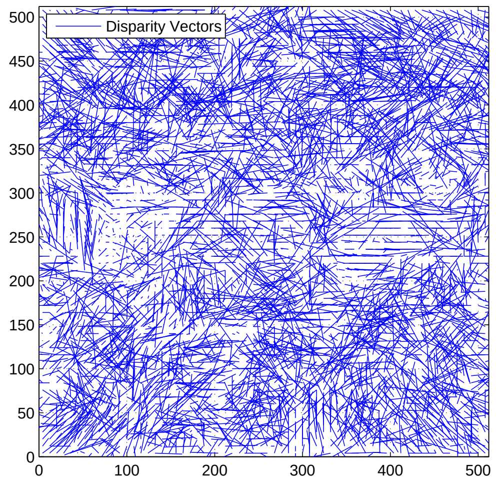
	CHPL		TBT1	
DE Method	*front*	*right*	*front*	*right*
Intra	20.7 dB	17.9 dB	19.9 dB	17.9 dB
TSS	26.6 dB	22.5 dB	23.7 dB	20.0 dB
4SS	28.8 dB	25.8 dB	26.2 dB	21.6 dB
5SS	29.5 dB	27.9 dB	28.6 dB	24.0 dB
6SS	29.0 dB	28.4 dB	29.8 dB	27.3 dB
7SS	28.4 dB	27.8 dB	30.2 dB	29.8 dB
Dir	30.8 dB	29.6 dB	29.1 dB	24.6 dB
Dir + refinement	31.4 dB	30.2 dB	29.7 dB	24.9 dB
Time				
TSS	932	947	938	963
4SS	1197	1217	1219	1247
5SS	1572	1577	1523	1508
6SS	1783	1823	1857	1789
7SS	2173	2118	2094	2110
Dir	1021	1037	1003	1034
Essential Matrix	10.7 %	10.4 %	8.6 %	8.5 %
Dir + refinement	1331	1351	1310	1357
Essential Matrix	8.2 %	8.0 %	6.6 %	6.5 %

Table 5.3: Disparity estimation for storage: Average PSNR and Time (in seconds)

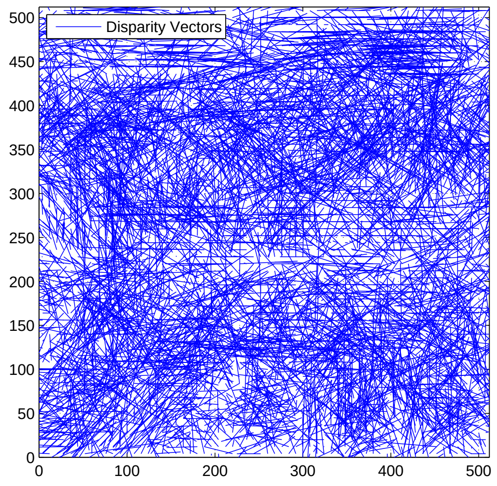
estimations are decreasing mainly due to large step sizes. Another observation is that overall and in both image datasets, *front* sequences result in less disparity compensation error as compared to the *right* sequences. That is mainly due to the larger disparities involved in side-wise displacement as compared to the case for forward displacement. It can be observed in these figures that for a given similar disparity compensation error, required disparity estimation time can be decreased to a great extent using our proposed smart search algorithm. For example consider horizontal bars in figures 5.16(b) and (a). It shows that, to maintain similar disparity compensation error, by using our proposed method around 34 to 40 percent of disparity estimation time will be saved respectively, resulting in similarly accurate disparity vectors.



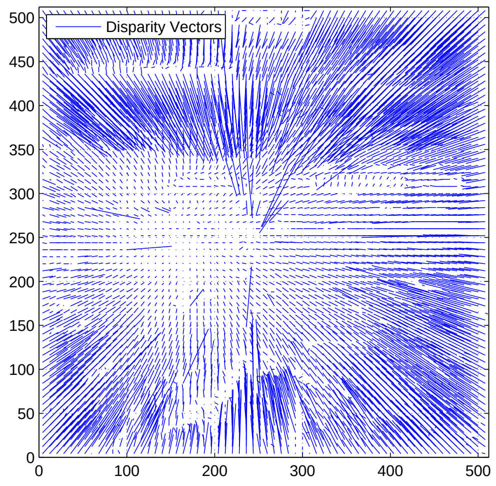
(a) DV *front* 5SS algorithm



(b) DV *front* 6SS algorithm

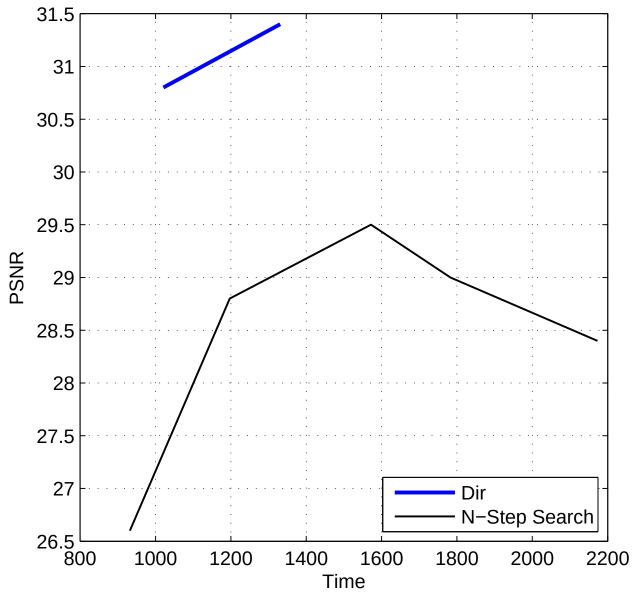


(c) DV *front* 7SS algorithm

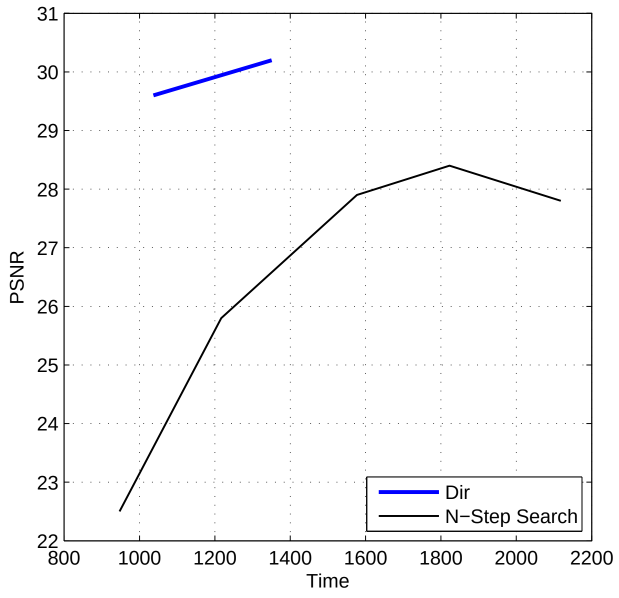


(d) DV *front* Dir algorithm

Figure 5.14: Estimated disparity vectors (for indoor image sequence LAB, side images No. 36 and 37)

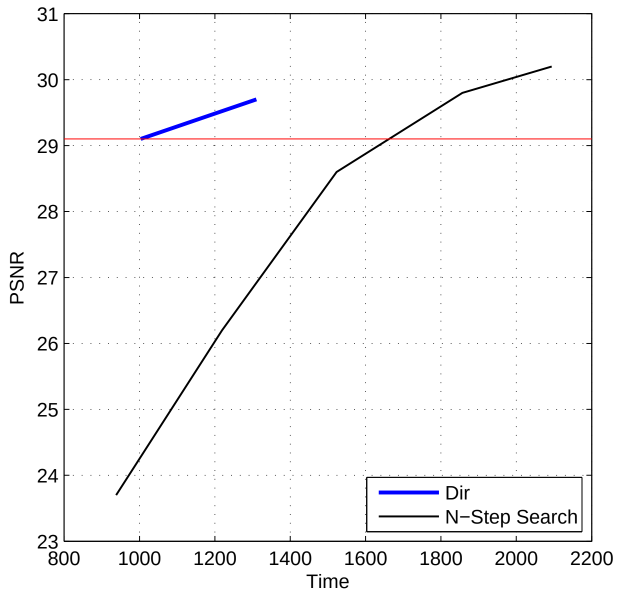


(a) *front* side images

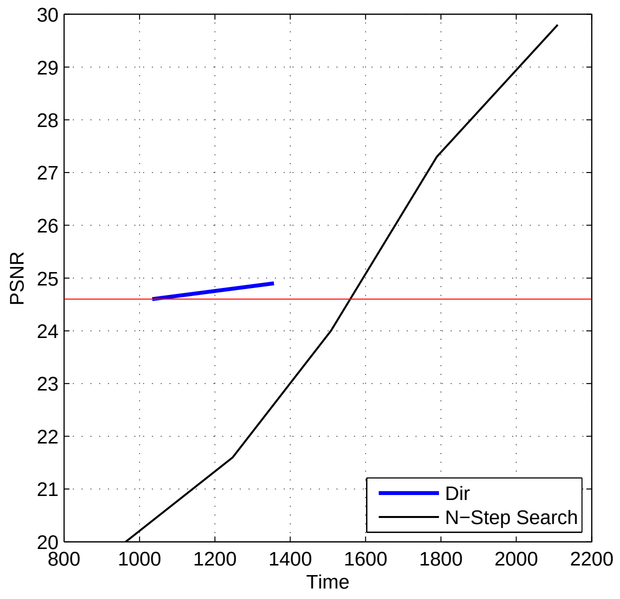


(b) *right* side images

Figure 5.15: Disparity estimation for storage: Residual error PSNR (dB) versus Time (in seconds), CHPL image dataset

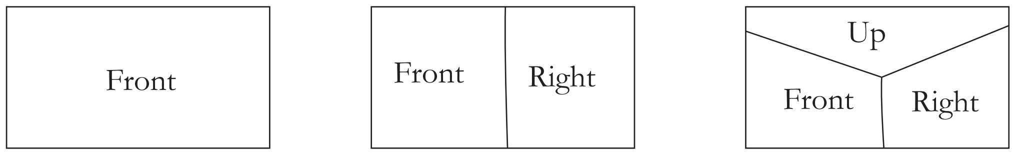


(a) *front* side images



(b) *right* side images

Figure 5.16: Disparity estimation for storage: Residual error PSNR (dB) versus Time (in seconds), TBT1 image dataset



a) One side image visible b) Two side images visible c) Three side images visible

Figure 5.17: Three different possibilities in view rendering

5.3 Disparity Estimation for Transmission

The results obtained in the previous section can be used in the cubic-panorama image dataset compression stage. Disparity vectors can be found faster and with more accuracy using data obtained after analyzing the image dataset. In real time, remote users are interested to interact with the stored image datasets by sending signals backward through the communication channel. In between, there exists a view rendering stage where, given the desired viewing position and angles, a planar view is generated by extracting appropriate information from the dataset. Three different possibilities are depicted in figure 5.17. The information to be extracted mainly includes stored residual error as well as the stored disparity vectors. In this section we propose a novel method for extracting and converting the existing disparity vectors efficiently. In other words, our objective is to reuse the existing data obtained in the storage stage, and adapt them to the new coordinate system and frame prediction structure to avoid extra computational complexities at the transmission stage.

Consider required rendered view as shown in figure 5.1 at the beginning of this chapter. A new set of disparity vectors is to be calculated, based on three sets of disparity vectors existing between each pair of cube faces at the storage unit. Figure 5.18 is a schematic plot of our proposed system. In addition to the disparity vector extraction and transformation it includes a disparity vector selection stage. That is required because cubic-panorama image datasets are assumed to be stored in $IBIB$ format while the user will be interested in $IPPP$ prediction structure. Standards transcoding have been used to convert the video frame structure using B frames (for storage) to the one using P frames (for transmission) in the literature. In [99] a set of candidate motion vectors, from the incoming bitstream, are extracted and used in the new video sequence format. Inspired from techniques used in their approach, we design a new disparity vector transformation scheme that can be embedded in the image sequence transmission

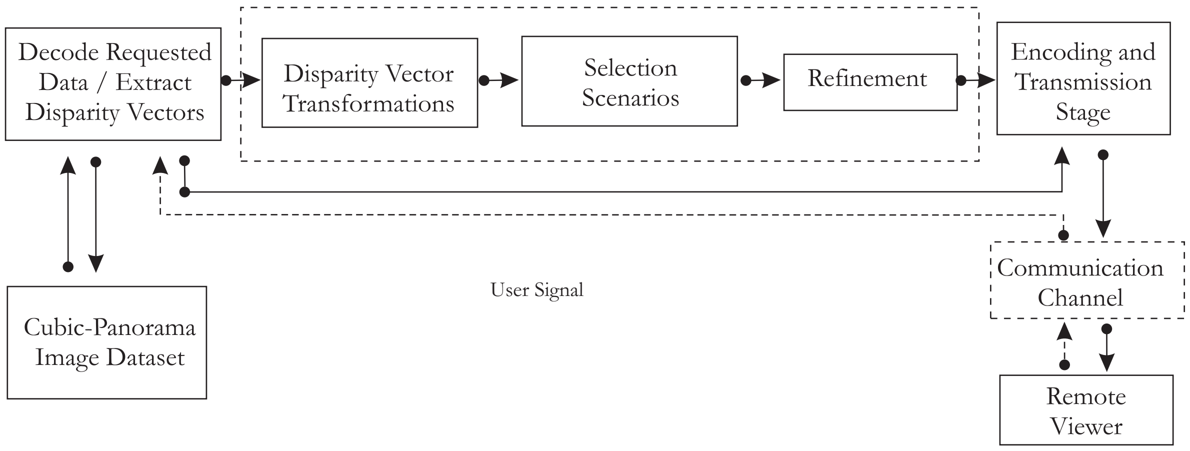

Decode Requested Data / Extract Disparity Vectors
Disparity Vector Transformations
Selection Scenarios
Refinement
Encoding and Transmission Stage
Cubic-Panorama Image Dataset
User Signal
Communication Channel
Remote Viewer

Figure 5.18: Proposed disparity vector extraction and conversion structure

stage of our project.

5.3.1 Introducing a New Video Transcoding Scheme

The disparity transformation stage can be summarized as follows. In order to find the image coordinates (X_r, Y_r) in the reference picture corresponding to the image coordinates (X_c, Y_c) in the current picture, one should take the following steps:

1. First find the coordinates of the point on the cube face in the 3D space (x_c, y_c) by applying an inverse mapping transformation.

2. Then, by using the existing information, such as the precalculated disparity vectors at the encoding stage performed on the source image dataset, new coordinates (x_r, y_r) in the 3D space on the reference basis image, which is in cube format, can be obtained.

3. Finally, by using a forward projection transformation, desired 2D image coordinates (X_r, Y_r) can be calculated. At this point, overall Disparity Vector (DV) can be obtained as below:

$$\text{DV} = (d_1, d_2) = (X_c, Y_c) - (X_r, Y_r) \tag{5.7}$$

In order to find all the required disparity vectors, this procedure should be repeated for all the smallest coding units, e.g. 4×4 or 8×8 blocks, at each frame. Using the

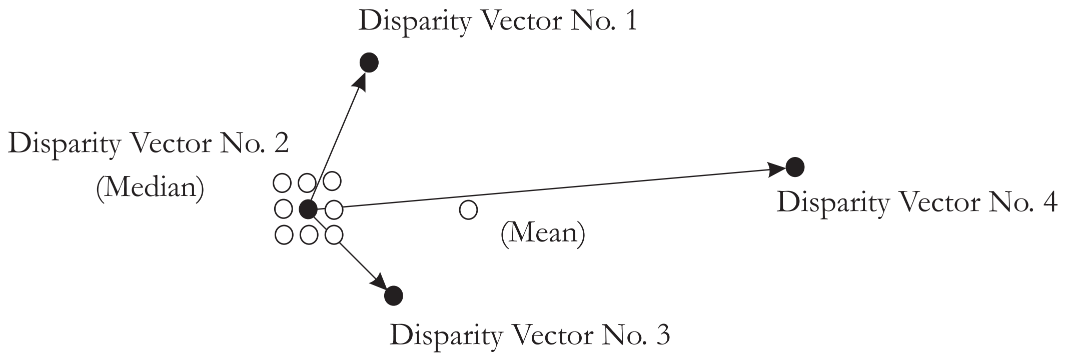


Figure 5.19: Four criterion for selecting the best disparity vector among four existing DV candidates

explained coordinate mappings properly, we will be able to extract disparity vectors between any two rendered views from the existing disparity vectors already calculated in the compression stage of the virtual navigation system, information which is highly valuable.

Since in practice we are working with blocks of pixels and in transformations, from 2D rendered current view coordinates to 3D cube coordinates, the centers of each block on the current rendered view will lie somewhere in the middle of four neighbor blocks on the corresponding current cube face, therefore for each block on the current rendered view we will obtain four different disparity vector candidates, i.e., four disparity vectors corresponding to the four nearest neighbor block centers in the reference cube frame. Therefore, for each block we will have four disparity vector candidates and a decision making criterion should be envisioned. This phenomenon is depicted visually in figure 5.19. A refinement stage can also be applied around the best choice in each case afterwards. Specifically, we use four different methods:

Median Let $DV = \{DV_1, DV_2, DV_3, DV_4\}$ represent the four adjacent disparity vectors. The distance between each vector and the rest is calculated as the sum of their Euclidean distances as follows:

$$d_i = \sum_{j=1, j\neq i}^{4} \|DV_i - DV_j\|. \tag{5.8}$$

The median vector is defined as one of these vectors that has the least distance from all, i.e.

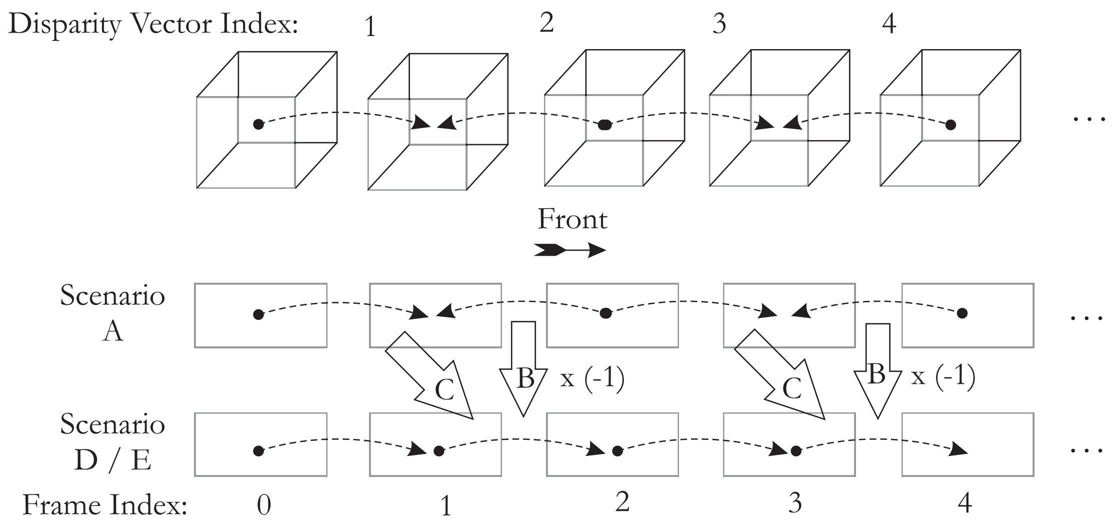


Figure 5.20: Five different scenarios used for converting the frame prediction structure

$$med(DV) = DV_k \in DV \text{ such that } \min d_i = d_k. \tag{5.9}$$

This method extracts the disparity vector situated in the middle of the rest of the disparity vectors.

Mean To calculate the average or mean of the input disparity vectors, given by:

$$DV = 1/4 \sum_{i=1}^{4} DV_i. \tag{5.10}$$

Best To pick the disparity vector with the minimum residual error at the storage stage.

Rand To randomly pick one among four disparity vector candidates.

Computational complexities with or without refinement for each case are summarized in table 5.2. Great computational savings are noticeable here. Input cubic-panorama image dataset is assumed to be stored in $IBIB$ format. Regarding the frame prediction structure conversion five different scenarios are considered as the following, see figure 5.20:

Scenario A Desired views by the user in assumed to be in $IBIB$ format, i.e., the same format as the cubic-panoramas are stored.

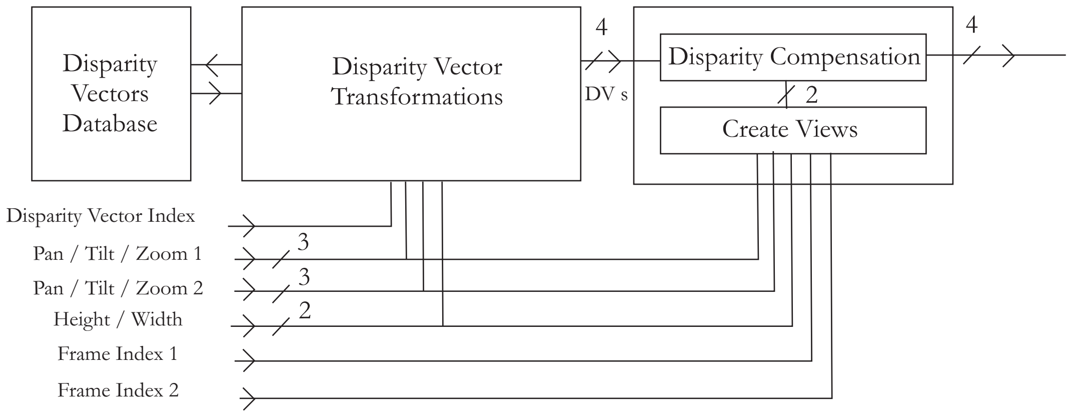


Figure 5.21: Disparity vector transformation algorithm in more detail

Scenario B It is assumed that the user is interested in $IPPP$ prediction structure. This is more realistic assumption since with B frames the user has to wait for a future frame to be decoded before getting the current frame and that is not desired in real time navigation applications. In this scenario and in disparity vector conversions, all backward predictions are multiplied by factor -1 and reused. At this stage only even disparity vector indexes are transformed to be able to have a fair comparison with results in scenario C below.

Scenario C Similar to scenario B, but this time instead of negating current backward predicted disparity vector, we use the immediate previous forward predicted disparity vector instead. In this scenario too, only even disparity vector indexes are transformed and evaluated for comparison purposes. Obtained outcomes will be compared in experimental results section.

Scenario D In this scenario a mixture of scenarios B and C are used. Here again only even disparity vector indexes are transformed and evaluated and in each case the best of negated current backward disparity vector and the immediate previous forward predicted disparity vector is used. This will add to the computational complexity but improved performance is expected.

Scenario E Eventually, results obtained in scenario D are used for even disparity vector indexes. For odd indexes, results of Scenario A are applied directly. This scenario realizes a $IBIB$ to $IPPP$ conversion as far as disparity vectors are concerned.

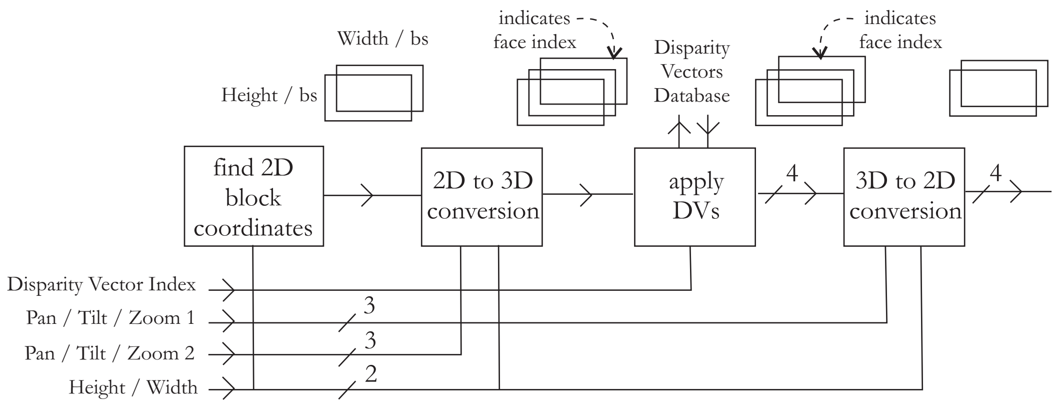
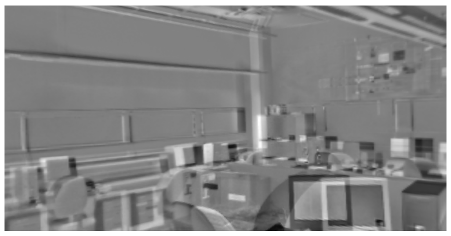
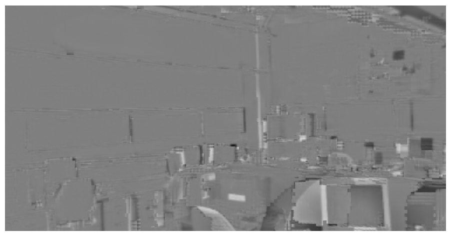


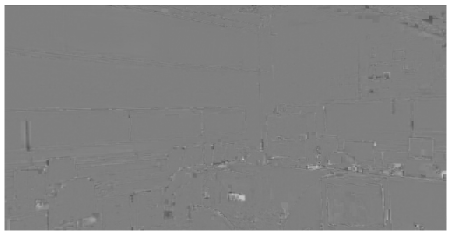
Figure 5.22: Disparity vector transformation unit details

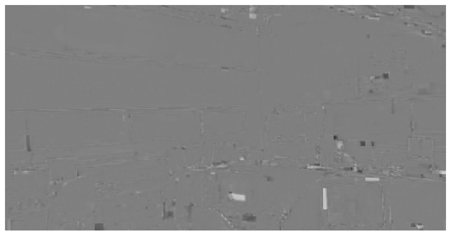
Details of disparity vector transformations that can be used in aforementioned scenarios are depicted in two figures. First, the desired rendered view has a certain Width and Height which does not change throughout the navigation process, see figure 5.21. Parameters that change include the frame index number 1 (reference frame), frame index number 2 (current frame), pan, tilt, and zoom factors associated with each frame index and the disparity vector index, i.e. the set of stored disparity vectors that are to be used as was already shown in figure 5.20. This information at each step is provided to the disparity vector transformation unit. This unit is interacting with the stored disparity vectors unit. At each step, four sets of disparity vectors are generated and fed into the disparity compensation unit. At the same time, two current and reference views are created using similar data. Based on four different sets of input disparity vectors at this stage, four different error images are obtained.

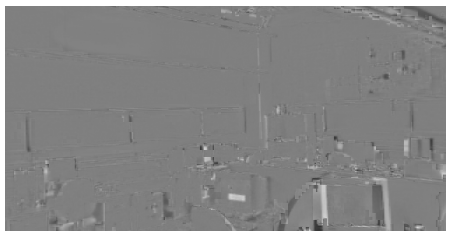
The disparity vector transformation unit is shown in more detail in figure 5.22. For each set of input information, 2D block coordinates are calculated first. These block coordinates are converted from rendered view format coordinates, into 3D coordinates on cubic panoramas. For each coordinate on a cube, four disparity vectors are extracted by interaction with the disparity vectors database. Therefore, four new 3D coordinates are found for each 8×8 block. These coordinates are mapped again back to 2D rendered view coordinates. Therefore, four sets of disparity vectors are generated for each block in rendered view coordinates.


(a) No Estimation (Intra)


(b) Three Step Search (TSS) algorithm


(c) Directional Search (Best method)


(d) Directional Search (Median method)


(e) Directional Search (Mean method)

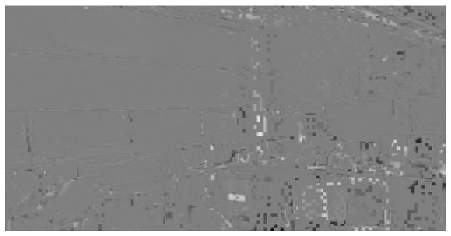

(f) Directional Search (Rand method)

Figure 5.23: Residual error image (indoor image sequence LAB, $\theta = -\pi/4$ frames No. 18 and 19)

5.3.2 Experimental Results

Disparity vectors that have been already estimated, encoded, and stored in the previous section, are accessed, decoded, and reused in this section based on navigation scenarios assumed in each experiment in order to obtain a new set of disparity vectors appropriate for the required rendered views format. Figure 5.23 shows the results, specifically resultant error images for indoor sequence LAB performing disparity estimation between cubic-panoramas number 18 and 19. Values of $FOV = \pi/3$, $Width = 512$, and $Height = 256$ are used and it is assumed that $\theta = -\pi/4$, that is when the user is looking 45 degrees to the right. Particularly, that corresponds to what is depicted in figure 5.17(b). The

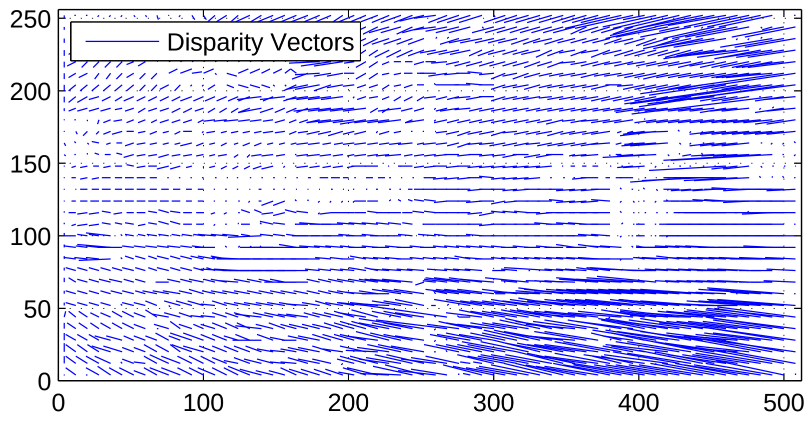


Figure 5.24: Estimated disparity vectors (indoor image sequence LAB, $\theta = -\pi/4$ frames No. 18 and 19)

difference values between rendered frames are shown in 5.23(a) while the result of TSS algorithm are given in 5.23(b). Performance of Best, Median, Mean, and Rand can be compared visually in 5.23(c) to (f) respectively. It is interesting to notice that the Mean operator gives poor results in comparison. The Rand operator also generates salt and pepper effects. The Median operator gives comparable subjective results as compared to the Best operator. Estimated disparity vectors for this test are also depicted in figure 5.24 in order to provide a better visual understanding.

Results considering five test scenarios will be shown here. Tests are run on 65 frames of CHPL and TBT1 sequences, indexed from 0 to 64, using $\theta = -\pi/4$ and $\theta = 0$ respectively. These assumptions will be equivalent with what was already shown in figure 5.17(b) and (a) respectively. Table 5.4 summarizes reference frame, current frame, and disparity vector indexes required in test scenario A. For example at the beginning, the reference frame index is 0 and the current frame is 1 and we use disparity vector

index 1	0	2	2	4	4	6	6	8	8	10	...
index 2	1	1	3	3	5	5	7	7	9	9	...
DV index	1	2	3	4	5	6	7	8	9	10	...

Table 5.4: Disparity estimation for transmission: Scenario A, index table

DE Method	CHPL	TBT1
TSS	24.6 dB	24.9 dB
4SS	27.5 dB	27.7 dB
Best	29.8 dB	31.2 dB
Median	26.7 dB	28.1 dB
Mean	23.5 dB	26.9 dB
Rand	23.9 dB	27.5 dB
Time		
TSS	476	490
4SS	615	635
Best	163	168
Median	112	114
Mean	108	111
Rand	109	112

Table 5.5: Disparity estimation for transmission: Scenario A, Average PSNR and Time (in seconds)

index 1	1	3	5	7	9	11	13	15	17	19	...
index 2	2	4	6	8	10	12	14	16	18	20	...
DV index	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	...

Table 5.6: Disparity estimation for transmission: Scenario B, index table

index 1 as shown in figure 5.20. Immediately after, the reference frame index will become equal to 2 and the current frame will become equal to 1 and we use disparity vector index 2. The indexing scheme is such because we are converting from $IBIB$ cubic format to $IBIB$ planar format. Disparity compensation PSNR values using four proposed methods and corresponding required time are summarized in table 5.5. Performance of TSS and 4SS algorithms are shown for comparison. These two algorithms are applied on rendered views without any reference to the already calculated and stored disparity vectors at the database. As expected, great improvement is achieved in terms of computational complexity, however, the output format is yet $IBIB$ and therefore more test scenarios should be run to get to the desirable state.

DE Method	CHPL	TBT1
TSS	24.6 dB	253 dB
4SS	27.8 dB	326 dB
Best	26.8 dB	87 dB
Median	24.4 dB	59 dB
Mean	23.1 dB	57 dB
Rand	22.7 dB	57 dB
Time		
TSS	237	25.4
4SS	306	28.7
Best	81	28.3
Median	55	25.8
Mean	53	25.3
Rand	54	25.4

Table 5.7: Disparity estimation for transmission: Scenario B, Average PSNR and Time (in seconds)

Table 5.6 summarizes reference frame, current frame, and disparity vector indexes required in test scenario B. In this scenario we use even disparity vector indexes and also negate disparity vectors, see figure 5.20. For example at the beginning the reference frame index is 1 and the current frame is 2 and we use disparity vector index 2 as shown in figure 5.20. Immediately after, the reference frame index will become equal to 3 and the current frame will become equal to 4 and we use disparity vector index 4. The indexing scheme is such because we are testing only even disparity vector indexes here. Disparity compensation PSNR values using four proposed methods and corresponding required time are summarized in table 5.7 along with the performance of TSS and 4SS algorithms. In this experiment we witness a decrease in quality due to using backward disparity estimation in cube space for forward disparity estimation of required views. Now we move to scenario C to test an alternative scenario.

Table 5.8 summarizes reference frame, current frame, and disparity vector indexes required in test scenario C. In this scenario we use odd disparity vector indexes instead, see figure 5.20. For example at the beginning the reference frame index is 1 and the current frame is 2 and we use disparity vector index 1 as shown in figure 5.20. Immedi-

index 1	1	3	5	7	9	11	13	15	17	19	...
index 2	2	4	6	8	10	12	14	16	18	20	...
DV index	1	3	5	7	9	11	13	15	17	19	...

Table 5.8: Disparity estimation for transmission: Scenario C, index table

DE Method	CHPL	TBT1
TSS	24.6 dB	25.4 dB
4SS	27.8 dB	28.7 dB
Best	24.3 dB	26.6 dB
Median	22.7 dB	24.5 dB
Mean	22.5 dB	24.5 dB
Rand	21.8 dB	24.3 dB
Time		
TSS	243	254
4SS	313	327
Best	83	87
Median	57	60
Mean	55	57
Rand	55	58

Table 5.9: Disparity estimation for transmission: Scenario C, Average PSNR and Time (in seconds)

ately after, the reference frame index will become equal to 3 and the current frame will become equal to 4 and we use disparity vector index 3. The indexing scheme is such because we are testing only odd disparity vector indexes. Disparity compensation PSNR values using four proposed methods and corresponding required time are summarized in table 5.9 along with the performance of TSS and 4SS algorithms. In this experiment we witness a relatively better performance, but results are not satisfying yet. That is mainly due to the fact that disparity vectors between immediate past neighbor pair of cubes are used to predict the current pair. In scenario D we use a combination of test scenarios B and C.

Table 5.10 summarizes reference frame, current frame, and disparity vector indexes

index 1	1	3	5	7	9	11	13	15	17	19	...
index 2	2	4	6	8	10	12	14	16	18	20	...
DV index 1	-2	-4	-6	-8	-10	-12	-14	-16	-18	-20	...
DV index 2	1	3	5	7	9	11	13	15	17	19	...

Table 5.10: Disparity estimation for transmission: Scenario D, index table

required in test scenario D. In this scenario we use the best choice in Scenario B and C for each block while concentrating on every second pair of frames, see figure 5.20. For example at the beginning the reference frame index is 1 and the current frame is 2 and we use disparity vector index 2 and 1 as shown in figure 5.20. Immediately after, the reference frame index will become equal to 3 and the current frame will become equal to 4 and we use disparity vector index 4 and 3. The indexing scheme is such because we are testing both odd and even disparity vector indexes. Disparity compensation PSNR values using four proposed methods and corresponding required time are summarized in table 5.11 along with the performance of TSS and 4SS algorithms. As we could expect, this scenario outperforms scenarios B and C while computational complexity and therefore timing is still very reasonable.

Finally, table 5.12 summarizes reference frame, current frame, and disparity vector indexes required in test scenario E. In this scenario we use the results of Scenario A and D for originally forward and backward prediction structures respectively, see figure 5.20. For example at the beginning the reference frame index is 0 and the current frame is 1 and we use disparity vector index 1 (scenario A) as shown in figure 5.20. Immediately after, the reference frame index will become equal to 1 and the current frame will become equal to 2 and we use disparity vector index 2 and 1 (scenario D). Disparity compensation PSNR values using four proposed methods and corresponding required time are summarized in table 5.13 along with the performance of TSS and 4SS algorithms.

DE Method	CHPL	TBT1
TSS	24.6 dB	25.4 dB
4SS	27.8 dB	28.7 dB
Best	28.3 dB	31.1 dB
Median	23.9 dB	25.7 dB
Mean	22.9 dB	25.4 dB
Rand	22.2 dB	24.8 dB
Time		
TSS	250	248
4SS	324	322
Best	144	141
Median	82	80
Mean	77	75
Rand	77	76

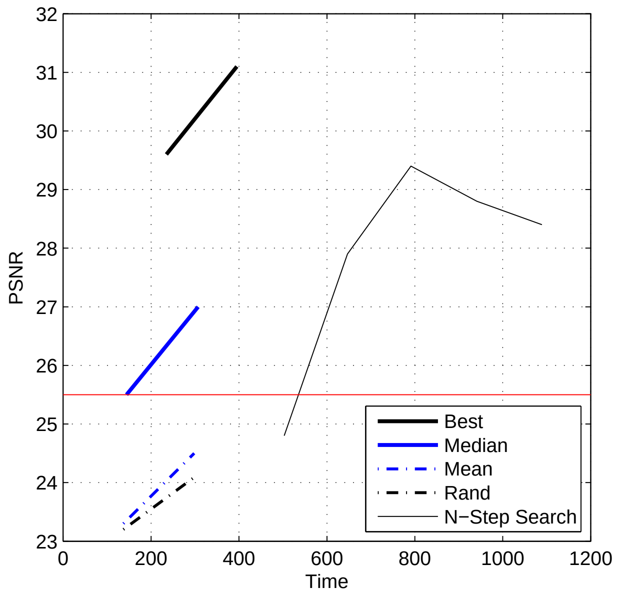
Table 5.11: Disparity estimation for transmission: Scenario D, Average PSNR and Time (in seconds)

index 1	0	1	2	3	4	5	6	7	8	9	...
index 2	1	2	3	4	5	6	7	8	9	10	...
DV index 1	1	-2	3	-4	5	-6	7	-8	9	-10	...
DV index 2	-	1	-	3	-	5	-	7	-	9	...

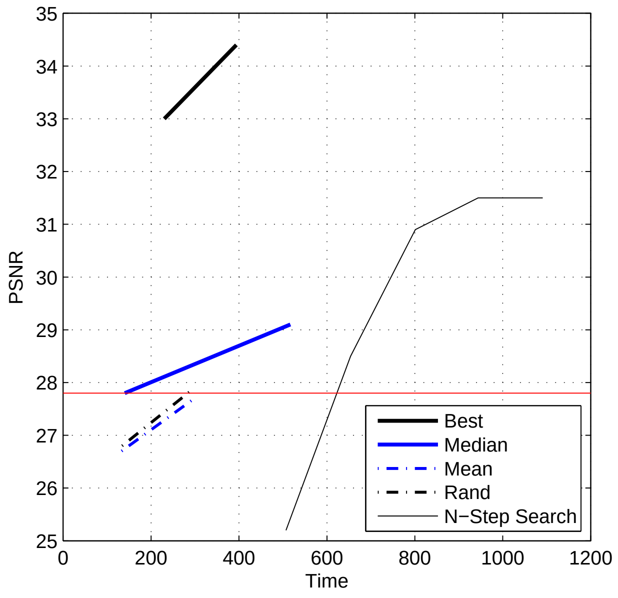
Table 5.12: Disparity estimation for transmission: Scenario E, index table

DE Method	CHPL	TBT1
TSS	24.8 dB	25.2 dB
4SS	27.9 dB	28.5 dB
5SS	29.4 dB	30.9 dB
6SS	28.8 dB	31.5 dB
7SS	28.4 dB	31.5 dB
Best	29.6 dB	33.0 dB
Median	25.5 dB	27.8 dB
Mean	23.3 dB	26.7 dB
Rand	23.2 dB	26.8 dB
Best + refinement	31.1 dB	34.4 dB
Median + refinement	27.0 dB	29.1 dB
Mean + refinement	24.5 dB	27.7 dB
Rand + refinement	24.1 dB	27.9 dB
Time		
TSS	503	507
4SS	647	654
5SS	791	801
6SS	941	944
7SS	1089	1091
Best	235	230
Median	144	140
Mean	137	133
Rand	137	134
Best + refinement	395	394
Median + refinement	307	517
Mean + refinement	298	299
Rand + refinement	300	299

Table 5.13: Disparity estimation for transmission: Scenario E, Average PSNR and Time (in seconds)



(a) CHPL image sequence, $\theta = -\pi/4$



(b) TBT1 image sequence, $\theta = 0$

Figure 5.25: Disparity estimation for transmission: Residual error PSNR (dB) versus Time (in seconds), CHPL and TBT1 image sequences

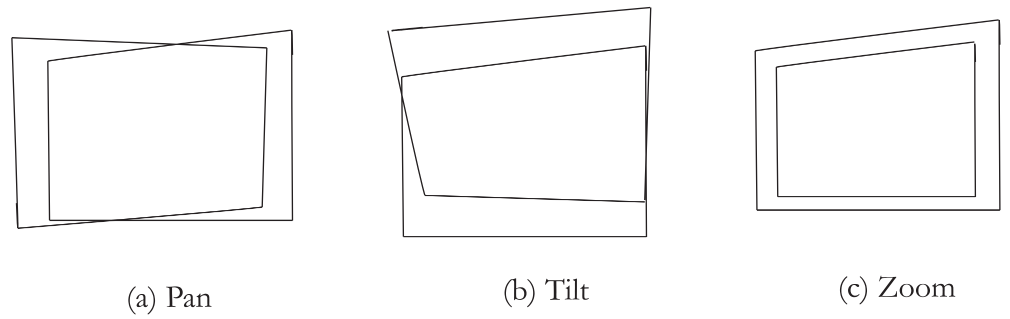


Figure 5.26: Visual demonstration of Pan, Tilt, and Zoom

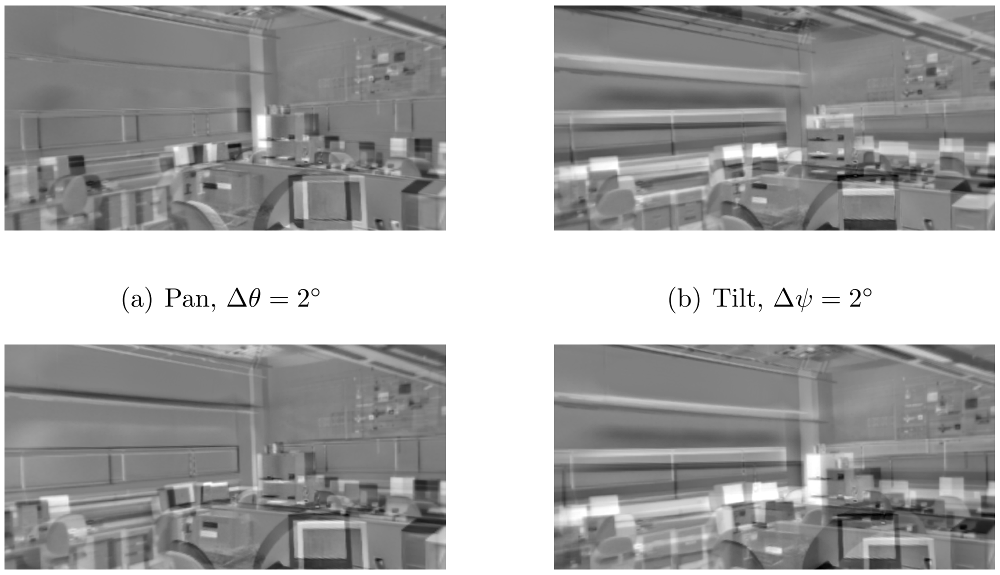
Adopting this scenario, more results are depicted as disparity compensation error PSNR versus spent time in figure 5.25. One interesting observation is that using step search at higher search ranges in CHPL sequence disparity compensation error increases after a certain point. That means although the overall search range is increasing, the accuracy of vector estimations are decreasing mainly due to large step sizes. After Best method, Median achieves highest performance and this is also in line with out visual observations of figure 5.23. It can be observed in this figure that for a given similar disparity compensation quality, required disparity calculation time can be decreased to a great extent using our proposed disparity conversion algorithm. For example consider horizontal bars in figures 5.25(a) and (b). To maintain similar disparity compensation error, by using our proposed algorithm and Median method around 73 to 78 percent of time will be saved respectively, resulting in similarly accurate disparity vectors.

5.4 Dealing with Pan, Tilt, and Zoom

Figure 5.26 shows Pan, Tilt, and Zoom when the user is not changing the view point but the view angles or field of view. This situation is to a great extent similar to the view point change which was already explained in the previous section except the fact that the second step in the disparity vector search is not required here and therefore the search algorithm will consist of only two steps as explained below, see also figure 5.2:

1. First find the coordinates of the point on the cube face in the 3D space (x_c, y_c) by applying an inverse mapping transformation.

2. Then, by using a forward projection transformation, desired 2D image coordinates



(a) Pan, $\Delta\theta = 2°$ (b) Tilt, $\Delta\psi = 2°$

(c) Zoom, $factor = 0.9$ (d) Pan, Tilt, and Zoom

Figure 5.27: Pan, Tilt, and Zoom, Intra (indoor image sequence LAB, frame No. 18)

(X_r, Y_r) can be calculated. Overall Disparity Vector (DV) can be obtained as in equation 5.7.

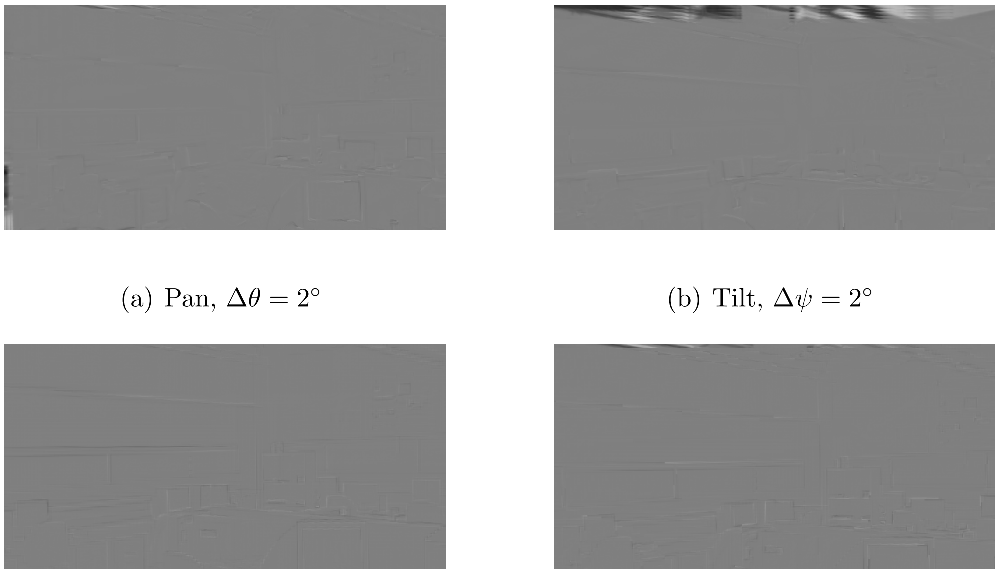
As before, in order to find all the required disparity vectors, this procedure should be repeated for all the smallest coding units, e.g. 4×4 or 8×8 blocks, at each frame.

Since disparity vectors will be calculated using the pan and tilt angles and the zooming factor issued by the user, disparity vectors need not be transmitted through the channel and the decoder will be able to calculate the disparity components only based on the three parameters of pan and tilt angles and the zooming factor.

5.4.1 Experimental Results

Figure 5.27 shows difference images for LAB sequence frame number 18, that includes, 2 degrees pan, 2 degrees tilt, zooming factor 0.9 and eventually combination of the three possibilities. One noticeable observation is that, unlike CHPL image sequence, since this image sequence consists of more horizontal lines, higher error is produced in tilt scenario as compared to pan case.

Results for disparity compensation error using the above mentioned method are given in figure 5.28. The residual error in this case is reduced to almost zero everywhere, except



(a) Pan, $\Delta\theta = 2°$ (b) Tilt, $\Delta\psi = 2°$

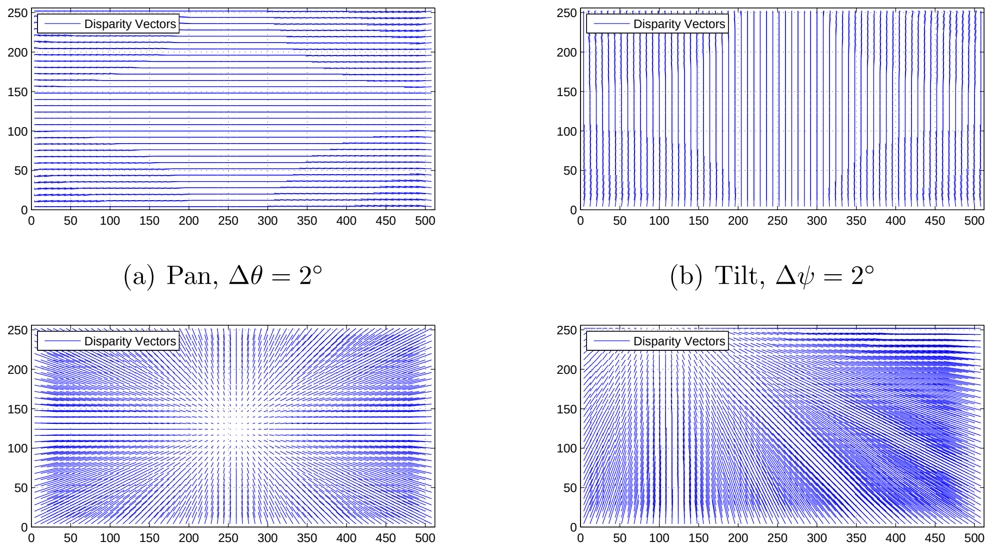
(c) Zoom, $factor = 0.9$ (d) Pan, Tilt, and Zoom

Figure 5.28: Pan, Tilt, and Zoom, PTZ (indoor image sequence LAB, frame No. 18)

at the left border in pan and the top border in tilt. Estimated disparity vectors for these pan, tilt, and zoom values are also depicted in figure 5.29. Finally, simulation results for 90 degrees pan with 2 degree step size performed on CHPL and TBT1 sequence are summarized in table 5.14 and are compared with the TSS algorithm.

5.5 Summary

In this chapter we pursued three major goals. First, to obtain disparity vectors for storage of cubic-panorama image dataset using results obtained in previous chapters. In chapter 3 it was explained how to analyze the pair of panoramas and the concept of essential matrix and epipolar geometry was introduced. In chapter 4 different prediction structures as well as GOP sizes were examined. It was shown that $IBIB$ prediction structure is useful in storing large number of image datasets due to properties of B frames. Combining the two, in this chapter we applied the epipolar geometry constraints to sequences in $IBIB$ format to achieve a fast disparity estimation algorithm which was named Dir. It was shown that to maintain similar disparity compensation error, by using our proposed algorithm around 34 to 40 percent of time will be saved, resulting in no less accurate disparity vectors.



(a) Pan, $\Delta\theta = 2°$

(b) Tilt, $\Delta\psi = 2°$

(c) Zoom, $factor = 0.9$

(d) Pan, Tilt, and Zoom

Figure 5.29: Pan, Tilt, and Zoom, Estimated disparity vectors

In the second step, an algorithm was explained in detail to extract the already calculated and stored disparity vectors at the time of virtual navigation. It includes disparity vector extraction, disparity vector transformations, and also prediction structure conversion followed by an optional refinement stage. It was shown that by appropriate disparity conversion as well as prediction format conversion, an efficient outcome can be achieved with much less computational complexity. Differences were shown through disparity compensation error images, disparity vector plots as well as quality versus computational complexity plots. It was shown that to maintain similar disparity compensation error, by using our proposed algorithm and Median method around 73 to 78 percent of time will be saved, resulting in no less accurate disparity vectors.

In the third and last part, it was explained how to find disparity vectors when the user turns around or decides to zoom in or zoom out. A real navigation includes a combination of the two above mentioned types of navigation. Altogether, our approach facilitates free navigation. That means the remote user can enter the sequence at any I frame, that means every second frame. He or she may decide to move to navigation in the right or the left direction. Since our disparity estimation method considers the $IBIB$ to $IPPP$ conversion, moving in either direction will not be a problem. Also since this method takes into account pan and tilt, returning back from the already walked navigation path at any point of interest is also facilitated. In short, disparity vectors are estimated accurately

DE Method	CHPL	TBT1
TSS	25.2 dB	27.7 dB
PTZ	34.5 dB	36.2 dB
Time		
TSS	176	177
PTZ	15	15

Table 5.14: Pan, Tilt, and Zoom, Average PSNR and Time (in seconds) for 90° Pan, $\Delta\theta = 2°$

first hand at the storage time and reused efficiently at the transmission time [101].

At the end, notice that different users' spatial capabilities requirement, explained among the requirements of the system in the first chapter, is automatically addressed by using our approach. Since recording and storage of data is performed separate from the transcoding stage, using different rendered view sizes throughout the navigation for each specific user will help users with different capabilities to access and use the stored data effectively.

Chapter 6

Conclusion: Contributions and Future Work

In this thesis we worked on a virtual navigation system and particularly considered the problem of analysis of cubic-panorama image datasets for storage and transmission of data. We started by introducing an image-based virtual environment navigation system with its main components including cubic-panorama image dataset compression and image sequence transmission. We discussed the interesting features of such image datasets as well as requirements for cubic-panorama image dataset compression and image sequence transmission. A chapter was dedicated to literature review, including image based rendering techniques, video compression schemes and standards, image matching techniques and existing classic video transcoding schemes. In order to be able to evaluate any proposed idea we needed to obtain a number of cubic-panorama image datasets first. All the images shown and used in this work have been acquired by our image acquisition system in the lab:

Image dataset acquisition Twelve cubic-panorama image datasets have been captured. Various indoor and outdoor locations were chosen for image acquisition in the city of Ottawa in the Fall of 2009, including the National Gallery of Canada, Tabaret Hall, and Colonel By building on campus at the University of Ottawa. Captured cubic-panorama image datasets can be used for testing and comparison purposes in various applications including the virtual navigation system.

In this work we have proposed various ideas for omnidirectional cubic-panorama image dataset preprocessing, compression, and transmission stages of a virtual navigation system, with special attention on disparity estimation, and have presented experimental results and a number of contributions, including the following.

6.1 Contributions

This section describes the contributions of this thesis to knowledge. Supporting contributions include the following:

Image dataset analysis and preprocessing A novel method which aims to achieve alignment of a large number of cubic image panoramas was presented. Also a measure was introduced for assessment of the overall alignment level among the basis images in a given cubic-panorama image dataset. Subsequently, the proposed method was tested using the introduced measure and the simulation results confirm the desired improvements. These results were presented in *ICASSP* conference in year 2011 [93].

Standardized versus wavelet-based methods The two major existing approaches in the literature, i.e., standards-based scheme and wavelet-based scheme, have been compared using two different quality measures, namely PSNR and SSIM for both indoor and outdoor cubic-panorama image datasets. The standardized based scheme has shown better rate distortion performance for captured cubic-panorama image datasets over various range of bitrates.

Efficient prediction structure A state-of-the-art standardized video codec, H.264/AVC, was adopted and used for compressing the acquired cubic-panorama image datasets. In search for an efficient prediction structure, various GOP sizes as well as various frame types and structures were tested and discussed. Presenting an efficient prediction structure for cubic-panorama image datasets, as contrasted with the case of conventional video sequences, constitutes another contribution of our work. The case of 4D cubic-panorama image datasets has been addressed similarly and a practical prediction structure was presented as an extended application of the already explained 3D cubic-panorama image datasets. A bitstream syntax, at the application layer of the network, was presented in order to facilitate organizing and randomly accessing the required data in such huge size captured cubic-panorama image datasets. These two set of results were presented in *SPIE* Electronic Imaging symposium in year 2012 [98].

Main contributions include the following:

Using the essential matrix for displacement estimation We proposed using the properties of the essential matrix and the epipolar constraint along with the cubic format of existing panoramas, mainly in order to reduce the computational complexity in the encoding process of the captured cubic-panorama image datasets.

In conventional video coding approaches, usually a block matching algorithm is used in motion vector search. This task is performed over a search window around the current reference point, restricted by a parameter called the maximum search range in both horizontal and vertical direction. No information about the nature of the video is usually exploited and the previously estimated motion vector is used as the initial guess for the current motion vector, the assumption that implies a uniform motion across the video frame.

First, as has been explained earlier, in video coding, as well as in cubic-panorama image dataset compression, motion or disparity estimation compromises 60-70% of the encoding complexity. By using a wiser approach, e.g., exploiting the properties of the essential matrix, and the specific cubic format of the basis images, we were able to reduce the computational complexity to a great extent, by restricting our search with a directional search algorithm.

More accurate disparity vectors were achieved, because we are basically searching in the right place for the disparity vector when using a directional search algorithm. We also did not need to assume that the disparity vectors possess smooth changes across the basis image; instead we were able to avoid using a predictive structure in the search for disparity vectors. This is a more realistic assumption since objects are usually located at different depths. As an example, objects located farther from the camera possess less disparities compared to the ones which are located closer to the camera. In addition, we realized higher search ranges along the epipolar line due to the minimal 1D nature of the search algorithm.

A new video transcoding scheme A new type of video transcoding was proposed which converts the information extracted from the storage unit from the cube format into a planar video sequence ready for transmission through the communication channel. In practice, the residual error should also be decoded and re-encoded. Our concern in video transcoding was mainly to extract the precalculated disparity vectors and reuse them in an efficient manner to be able to reduce the computational complexity of the transcoding stage to a great extent. Also, since disparity vectors are usually calculated for each image block (usually of size 4×4 or 8×8), when going back and forth from the panoramic images in cube format to the planar image format, we considered a few candidate vectors in disparity vector estimation inspired from the existing works in standards transcoding applications.

The case of pan, tilt, and zoom was also explained and a method for finding disparity vectors were explained. As can be seen by comparing the two figures of 5.1 and 5.2, the idea which was proposed for dealing with the cases of pan, tilt, and zoom, can be considered as a simplified case of the previously explained video transcoding step. In other words, our goals at this step were achieved by taking the transcoding algorithm and simply discarding its second step (among the three steps introduced for video transcoding). The aforementioned set of contributions were presented in *SPIE* Electronic Imaging symposium in year 2013 [101].

6.2 Future Directions

Eventually, a number of future directions can also be envisioned as below:

Real world application We have already made several simulations using different assumptions and test scenarios without directly involving the existing implementation of the source codes. However, our method can be embedded in any existing standardized video codec. Specifically, the motion/disparity estimation module needs to be modified as explained in chapter five such that it can be used in different ways for comparing and testing various approaches. The existing reference software for H.264/AVC, along with the software manual[1], the ITU-T recommendation for H.264[2], as well as the email listing of the JVT experts team will be a great help for anyone who wishes to extend this work.

Therefore, by using our method we believe that it is possible not only to reduce the computational complexity, but also we look forward to increasing the compression efficiency, due to a more realistic approach towards estimating the disparity vectors.

Note that the task of video transcoding can be considered as a combination of an encoding and a decoding module. Therefore, if enough experience in using the

[1]Available here: http://iphome.hhi.de/suehring/tml/

[2]Available here: http://www.itu.int/rec/T-REC-H.264

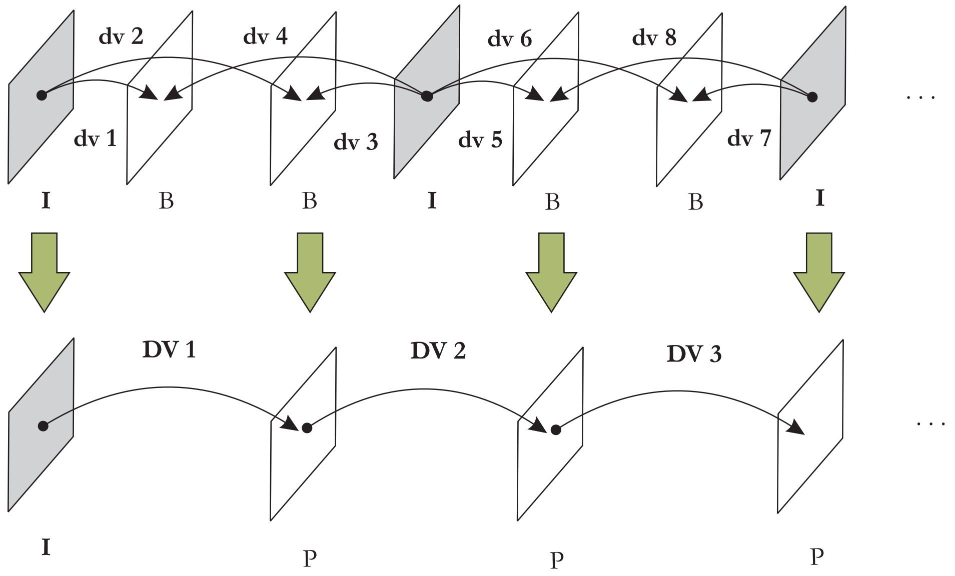


Figure 6.1: An example of view scalability requirement in image sequence transmission

encoding/decoding stages is achieved, we expect to achieve similar improvements in the video transcoding step in a relatively shorter time period as explained in chapter five. Existing standardized video codec can be modified to be able to test the proposed disparity estimation algorithm. Once this task is achieved, we will be able to extract the disparity vectors out of the resultant bitstream in an inverse manner, and subsequently we will be able to combine the two tasks in order to reaffirm improvements obtained in the video transcoding step as well as the last step of dealing with pan, tilt, and zoom. Notice that negligible residual error in pan, tilt, and zoom scenario is equivalent with lower bit rate and therefore higher encoding efficiency. Also since the user is aware of the view angles he is submitting to the transmitter, in practice, no disparity vector need to be transmitted to the receiver side so far as pan, tilt, or zoom scenario is considered.

View scalability As can be seen in Figure 6.1, by using this method in the image dataset transcoding stage, the desired view scalability feature can also be achieved for more complex scenarios, for example IBB prediction structure, given required essential matrixes are already calculated and stored. If normal speed is sought, between any pair of frames in this setting, two candidate disparity vectors exist for example +dv1 and half of dv2 (briefly +0.5dv2) between the first pair can be used and so on for the rest. However, for fast navigation, i.e. twice the normal speed, a typical conversion scenario is briefly shown. disparity vectors (DVs) are to be estimated from already calculated and stored ones noted by dv in the figure. Below is a candidate decision scenario for each possibility:

possible candidates for DV1: +dv2, +2dv1, and -dv4.

possible candidates for DV2: +2dv5, -2dv3, +dv6, and -dv4.

possible candidates for DV3: -dv8, -2dv7, and +dv6.

The rest will be similar. If this scenario is developed carefully in a similar manner as it was done in chapter five, the user will be able to navigate twice faster and the goal of view scalability explained earlier in the first chapter can be fulfilled.

View synthesis One interesting aspect of any panoramic image dataset is view synthesis. We elaborated enough on free virtual walk-through, yet the view locations are limited to the center of cubic-panoramas in our work. However, analysis of cubic-panoramas give us tools to work on view synthesis. As shown in [27, 28], by exploiting properties of the geometry of the scene, the triangular re-projection method can be extended to virtual cubic-panorama synthesis. In their work, the desired panorama is synthesized from three neighboring reference cubic panoramas in two steps. The first step is to synthesize the central part in each cube face, and the second step is to synthesize the region near the cube edges, with emphasis on solving the difficulties due to the discontinuity at the edges. Also If you would like to see an interesting work on spherical-panorama stereo with wavelet filtering for novel view synthesis, visit chapter six of Brunton's thesis [25].

Therefore in addition to the aforementioned contributions, our work can be used as a starting point in a number of future directions in the area of cubic-panorama image dataset processing.

Appendix A

Glossary of Terms

4SS Four Step Search

5SS Five Step Search

6SS Six Step Search

7SS Seven Step Search

AVC Advanced Video Coding

CBY Colonel By

CCD Charge-Coupled Device

CFA Color Filter Array

CHPL Chapel

CM Concentric Mosaic

CR Compression Ratio

DCP Disparity Compensation/Prediction

DCT Discrete Cosine Transform

DE Disparity Estimation

DV Disparity Vector

DWT Discrete Wavelet Transform

EBCOT Embedded Block Coding with Optimal Truncation

EZW Embedded Zero-tree Wavelet

FOV Field Of View

FSA Full Search Algorithm

FTV Free viewpoint TV

GOP Group Of Pictures

HDTV High Definition Television

HEVC High Efficiency Video Coding

HVS Human Visual System

IBR Image-Based Rendering

IEC International Electrotechnical Commission

IEEE Institute of Electrical and Electronics Engineers

ISO International Standards Organization

ITU International Telecommunication Union

JPEG Joint Photographic Experts Group

JVT Joint Video Team

LAB Laboratory

LBY Lobby

MCTF Motion Compensated Temporal Filtering

MIC Multiview Image Coding

MPEG Moving Picture Experts Group

MSE Mean Square Error

MV Motion Vector

MVC Multiview Video Coding

NAVIRE Virtual Navigation in Image-based Representations of Real World Environments

NCC Normalized Cross Correlation

NGC National Gallery of Canada

NSERC Natural Sciences and Engineering Research Council

PSNR Peak Signal to Noise Ratio

PTZ Pan, Tilt, and Zoom

RGB Red, Green, Blue

RMSE Root Mean Square Error

SAD Sum of Absolute Differences

SIFT Scale-Invariant Feature Transform

SPIHT Set Partitioning in Hierarchical Trees

SSD Sum of Squared Differences

SSIM Structural Similarity

SVD Singular Value Decomposition

TBT Tabaret

TSS Three Step Search

VCR Video Cassette Recorder

VLC Variable Length Coding

VQ Vector Quantization

Appendix B

List of Symbols

a, b, c	Epipolar plane
a_i, b_i, c_i	Epipolar line
c_1, c_2	Variables to stabilize the division with weak denominator
d, u, l, r, b, f	Down, up, left, right, back and front side images
d_1, d_2	Vertical and horizontal displacement
E or $(e_{ij})_{33}$	Essential matrix
$\mathcal{E}_{\mathbf{MS}}$	Mean square error
$\mathcal{E}_{\mathbf{RMS}}$	Root mean square error
F	Focal length of the camera
I_{33}	Identity matrix (3 x 3)
$\mathcal{I}_{\mathcal{B},k}$	k^{th} basis image in an image dataset
$\mathcal{I}_{\mathcal{B},k,j}$	j^{th} image segment in $\mathcal{I}_{\mathcal{B},k}$
$\mathcal{I}_{\mathcal{B},c}$, $\mathcal{I}_{\mathcal{B},r}$	Current and reference basis images
$\hat{\mathcal{I}}_{\mathcal{B},k}$	Reconstructed $\mathcal{I}_{\mathcal{B},k}$
J_0	Total number of image datasets available in the image database

K Size of the image dataset

K_0 Number of image dataset segments in each image dataset

L Size of a cube

L_0 Number of basis images in each image dataset segment

$\mathcal{L}_P$ Plenoptic function of the light ray intensity distribution

$\mathcal{L}_{Pm}$ m-dimensional plenoptic function

M Number of image correspondences

M_0 Number of group of pictures

N Number of search operations in the first step

N_0 Number of frames in each group of picture

N_1, N_2 Width and Height of each side image in pixels

$\mathbb{P}_M$ Model parameter set

R or $(r_{ij})_{33}$ Rotation matrix

$\mathbf{R}_k$ Rotation matrix between $\mathcal{I}_{\mathcal{B},k}$ and $\mathcal{I}_{\mathcal{B},k+1}$

$\mathbf{R}_\mathbf{T}$ Rotation matrix (resulted from the translation vector $\mathbf{T}$)

$\mathcal{R}_C$ Compression ratio

S_E Sum of squared errors

$\mathbf{T}$ Translation vector

T_x, T_y, T_z Translation vector components

$\mathbf{T}_d$, $\mathbf{T}_u$, $\mathbf{T}_l$, $\mathbf{T}_r$, $\mathbf{T}_b$, $\mathbf{T}_f$ Transformation matrices of down, up, left, right, back and front side images

w_i Image window

Δx, Δy, Δz Scaled translation vector components

x, y, z Coordinates of a scene point

X, Y Coordinate of the scene point projected on the image plane of the camera

μ_{w_i} The average of w_i

$\sigma^2_{w_i}$ The variance of w_i

$\sigma_{w_1 w_2}$ The covariance of w_1 and w_2

θ, ψ, φ Rotation angles

θ', ψ' Rotation angles (resulted from $\mathbf{T}$)

Bibliography

[1] D. Bradley, A. Brunton, M. Fiala and G. Roth, "Image-based Navigation in Real Environments Using Panoramas," *Proc. IEEE Int. Workshop on Haptic Audio Visual Environments and their Applications (HAVE'05)*, Ottawa, ON, Canada, Oct. 2005, pp. 57-59.

[2] S. E. Chen, "QuickTime VR - An Image-Based Approach to Virtual Environment Navigation," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'95),* Aug. 1995, pp. 29-38.

[3] M. Uyttendaele, A. Criminisi, S. B. Kang, S. Winder, R. Szeliski and R. Hartley, "Image-Based Interactive Exploration of Real-World Environments," *IEEE Trans. Computer Graphics and Applications*, vol. 24, no. 3, pp. 52-63, May 2004.

[4] C. Zhang and J. Li, "Interactive Browsing of 3D Environment over the Internet," *Proc. SPIE Visual Communications and Image Processing Symp.,* San Jose, CA, USA, Jan. 2001, vol. 4310, pp. 509-520.

[5] M. Fiala and G. Roth, "Automatic Alignment and Graph Map Building of Panoramas," *Proc. IEEE Int. Workshop on Haptic Audio Visual Environments and their Applications (HAVE'05)*, Ottawa, ON, Canada, Oct. 2005, pp. 104-109.

[6] X. Huang and E. Dubois, "3D Reconstruction Based on a Hybrid Disparity Estimation Algorithm," *IEEE Int. Conf. on Image Processing,* pp. 1025-1028, Oct. 2006.

[7] F. Kangni and R. Laganière, "Epipolar Geometry for the Rectification of Cubic Panoramas," *Proc. of the 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*, June 2006, pp. 70-77.

[8] X. Sun and E. Dubois, "A Matching-Based View Interpolation Scheme," *Proc. IEEE Int. Conf. Acoustics Speech Signal Processing*, pp. II 877-880, March 2005.

[9] L. Zhang, D. Wang and A. Vincent, "Adaptive Reconstruction of Intermediate Views from Stereoscopic Images," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 16, pp. 102-113, Jan. 2006.

[10] Kehua Jiang, "Cubic-Panorama Image Dataset Compression for Image-Based Virtual Environment Navigation," *PhD Thesis, University of Ottawa*, Feb. 2009.

[11] NAVIRE: http://www.site.uottawa.ca/research/viva/projects/ibr.

[12] S.-C. Chan and H.-Y. Shum, "A Spectral Analysis for Light Field Rendering," *Proc. IEEE Int. Conf. Image Processing (ICIP'00)*, Vancouver, BC, Canada, Sept. 2000, vol. 2, pp. 25-28.

[13] C. Zhang and T. Chen, "Spectral Analysis for Sampling Image-Based Rendering Data," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 13, no. 11, pp. 1038-1050, Nov. 2003.

[14] J.-X. Chai, X. Tong, S.-C. Chan and H.-Y. Shum, "Plenoptic Sampling," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'00)*, New Orleans, LA, USA, 2000, pp. 307-318.

[15] Z. C. Lin and H.-Y. Shum, "On the Number of Samples Needed in Light Field Rendering with Constant-Depth Assumption," *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR'00)*, Hilton Head Island, SC, USA, June 2000, vol. 1, pp. 588-595.

[16] H.-Y. Shum and L.-W. He, "Rendering with Concentric Mosaics," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'99)*, Los Angeles, CA, USA, Aug. 1999, pp. 299-306.

[17] S. Ikeda, T. Sato and N. Yokoya, "High-Resolution Panoramic Movie Generation from Video Streams Acquired by an Omnidirectional Multi-Camera System," *Proc. IEEE Int. Conf. Multisensor Fusion and Integration for Intelligent Systems (MFI'03)*, Aug. 2003, pp. 155-160.

[18] M. Beermann and E. Dubois, "Capture and Generation of Cubes," *NAVIRE Internal Technical Report*, Nov. 2006.

[19] M. Beermann and E. Dubois, "Acquisition Processing Chain for Dynamic Panoramic Image Sequences," *Proc. IEEE Int. Conf. Image Processing (ICIP'07)*, Sept. 2007, vol. V, pp. 217-220.

[20] L. E. Gurrieri and E. Dubois, "Optimum alignment of panoramic images for stereoscopic navigation in image-based telepresence systems," *proc. of the 11th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras OMNIVIS2011*, Barcelona, Spain, Nov. 2011.

[21] L. E. Gurrieri and E. Dubois, "Efficient panoramic sampling of real-world environments for image-based stereoscopic telepresence," *Proc. of the IS&T/SPIE Annual Symposium on Electronic Imaging Science and Technology, Stereoscopic Display and Applications*, San Francisco, USA, Jan. 2012.

[22] R. Szeliski and H. Shum, "Creating Full View Panoramic Image Mosaics and Environment Maps," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'97),* 1997, pp. 251-258.

[23] Alan Brunton, Jochen Lang, and Eric Dubois, "Spherical Harmonic Transforms and Convolutions on the GPU," *Journal of Graphics, GPU, and Game Tools*, 15(1):13-27, 2010.

[24] A. Brunton, J. Lang, and E. Dubois, "Efficient Multi-Scale Stereo of High-Resolution Planar and Spherical Images," *International Conference on 3D Imaging Modeling Processing Visualization and Transmission (3DIMPVT)*, 2012.

[25] Alan Brunton, "Multi-Scale Methods for Omnidirectional Stereo with Application to Real-Time Virtual Walkthroughs," *Ph.D. Thesis, University of Ottawa*, Nov. 2012.

[26] N. Greene, "Environment Mapping and Other Applications of World Projections," *IEEE Computer Graphics and Applications,* vol. 6, no. 11, pp. 21-29, Nov. 1986.

[27] C. Zhang, E. Dubois, and Y. Zhao, "Intermediate cubic-panorama synthesis based on triangular re-projection," *17th IEEE International Conference on Image Processing (ICIP'10)*, September 2010.

[28] C. Zhang, Y. Zhao, and F. Wu, "Triangulation of cubic panorama for view synthesis," *Applied Optics*, vol. 50, no. 22, pp. 4286-4294, August 2011.

[29] X. Sun and E. Dubois, "View Morphing and Interpolation through Triangulation," *Proc. SPIE Electronic Imaging Symp., Conf. on Image and Video Communications and Processing,* vol. 5685, pp. 513-521, Jan. 2005.

[30] E. H. Adelson and J. Bergen, "The Plenoptic Function and the Elements of Early Vision," *Computer Models of Visual Processing,* pp. 3-20, MIT Press, Cambridge, MA, USA, 1991.

[31] D. Bradley, "Navire Cube Viewer," Navire Internal Technical Report, NAV-TR-2005-01, Aug. 2005.

[32] C. Zhang and T. Chen, "A Survey on Image-Based Rendering – Representation, Sampling and Compression," *Signal Processing: Image Communication,* vol. 19, pp. 1-28, Jan. 2004.

[33] Dirac video codec: http://www.diracvideo.org.

[34] G. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE Signal Processing Magazine*, vol.15, no.6, pp.74-90, Nov. 1998.

[35] G. Sullivan and T. Wiegand, "Video Compression - From Concepts to the H.264/AVC Standard," *Proceedings of the IEEE*, vol.93, no.1, pp.18-31, Jan. 2005.

[36] M. Levoy and P. Hanrahan, "Light Field Rendering," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'96),* New Orleans, LA, USA, Aug. 1996, pp. 31-42.

[37] D. G. Aliaga and I. Carlbom, "Plenoptic Stitching: A Scalable Method for Reconstructing 3D Interactive Walkthroughs," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'01),* Los Angeles, CA, USA, Aug. 2001, pp. 443-450.

[38] S. Gortler, R. Grzeszcczuk, R. Szeliski and M. F. Cohen, "The Lumigraph," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'96),* New Orleans, LA, USA, Aug. 1996, pp. 43-52.

[39] H. Y. Shum, S. B. Kang and S. C. Chan, "Survey of Imaged-Based Representations and Compression Techniques," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 13, no. 11, pp. 1020-1037, Nov. 2003.

[40] S. M. Seitz and C. R. Dyer, "View Morphing," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'96),* New Orleans, LA, USA, Aug. 1996, pp. 21-30.

[41] J. Shade, S. Gortler, L. W. He and R. Szeliske, "Layered Depth Images," *Proc. ACM Annu. Conf. Computer Graphics and Interactive Techniques (SIGGRAPH'98),* Orlando, FL, USA, July 1998, pp. 231-242.

[42] X. Tong and R. M. Gray, "Interactive Rendering from Compressed Light Fields," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 13, no. 11, pp. 1080-1091, Nov. 2003.

[43] K. Jiang and E. Dubois, "Motion-Oriented Coding Scheme for Compression of Concentric Mosaic Scene Representations," *Proc. IEEE Int. Workshop on Haptic Audio Visual Environments and their Applications*, Ottawa, ON, Canada, Oct. 2004, pp. 37-42.

[44] H. Y. Shum, K. T. Ng and S. C. Chan, "Virtual Reality Using the Concentric Mosaic: Construction, Rendering and Data Compression," *Proc. IEEE Int. Conf. Image Processing (ICIP'00),* Sept. 2000, vol. 3, pp. 644-647.

[45] G. Sullivan, "Efficient Scalar Quantization of Exponential and Laplacian Variables," IEEE Trans. Information Theory, vol. 42, no. 5, pp. 1365-1374, Sept. 1996.

[46] B. Girod, C. L. Chang, P. Ramanathan and X. Zhu, "Light Field Compression Using Disparity-Compensated Lifting," *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP'03),* Hong Kong, China, Apr. 2003, pp. IV760-763.

[47] L. Luo, Y. Wu, J. Li and Y. Q. Zhang, "Compression of Concentric Mosaic Scenery with Alignment and 3D Wavelet Transform," *Proc. SPIE Image and Video Communications and Processing Symp.,* San Jose, CA, USA, Jan. 2000, vol. 3974, pp. 89-100.

[48] X. Tong and R. M. Gray , "Coding of Multi-View Images for Immersive Viewing," *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP'00),* Istanbul, Turkey, June 2000, vol. 4, pp. 1879-1882.

[49] M. Magnor and B. Girod, ”Hierarchical Coding of Light Fields with Disparity Maps,” *Proc. IEEE Int. Conf. Image Processing (ICIP’99),* Kobe, Japan, Oct. 1999, vol. 3, pp. 334-338.

[50] M. Magnor and B. Girod, ”Data Compression for Light-Field Rendering,” *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 10, no. 3, pp. 338-343, Apr. 2000.

[51] C. Zhang and J. Li, ”Compression of Lumigraph with Multiple Reference Frame (MRF) Prediction and Just-in-Time Rendering,” *IEEE Data Compression Conf.,* Snowbird, UT, USA, Mar. 2000, pp. 254-263.

[52] C. Zhang and J. Li, ”Compression and Rendering of Concentric Mosaics with Reference Block Codec (RBC),” *Proc. SPIE Visual Communications and Image Processing Symp.,* Perth, Australia, June 2000, vol. 4067, pp. 43-54.

[53] Y. Wu, L. Luo, J. Li and Y. Q. Zhang, ”Rendering of 3D-Wavelet Compressed Concentric Mosaic Scenery with Progressive Inverse Wavelet Synthesis (PIWS),” *Proc. SPIE Visual Communications and Image Processing Symp.,* Perth, Australia, June 2000, vol. 4067, pp. 31-43.

[54] Y. Wu, C. Zhang and J. Li, ”Smart Rebinning for the Compression of Concentric Mosaic,” *IEEE Trans. on Multimedia,* vol. 4, no. 3, pp. 332-342, Sept. 2002.

[55] M. Ghanbari, ”Standard Codecs: Image Compression to Advanced Video Coding,” *The Institution of Electrical Engineers, London,* 2003.

[56] ISO/IEC JTC1/SC29/WG1, ”Information Technology — Digital Compression and Coding of Continuous-tone Still Images: Requirements and Guidelines,” ISO/IEC 10918-1 and ITU-T Rec. T.81 (JPEG), 1992.

[57] ISO/IEC JTC1/SC29/WG1, ”Information Technology — JPEG 2000 Image Coding System, Part 1: Core Coding System,” ISO/IEC 15444-1 and ITU-T Rec. T.800 (JPEG2000), Aug. 2000.

[58] M. D. Adams and F. Kossentini, ”JasPer: A Software-Based JPEG 2000 Codec Implementation,” *Proc. IEEE Int. Conf. Image Processing (ICIP’00),* Vancouver, BC, Canada, Sept. 2000, vol. 2, pp. 53-56.

[59] D. T. Lee, ”JPEG 2000: Retrospective and New Developments,” *Proceedings of the IEEE,* vol. 93, no. 1, pp. 32-41, Jan. 2005.

[60] J. M. Shapiro, "Embedded Image Coding Using Zerotrees of Wavelet Coefficients," *IEEE Trans. Signal Processing,* vol. 41, no. 12, pp. 3445-3462, Dec. 1993.

[61] A. Said and W. A. Pearlman, "A New Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 6, pp. 243-250, June 1996.

[62] D. Taubman, "High Performance Scalable Image Compression with EBCOT," *IEEE Trans. Image Processing,* vol. 9, no. 7, pp. 1158-1170, July 2000.

[63] ITU-T, "Video Codec for Audiovisual Services at p x 64 kbit/s," ITU-T Rec. H.261, Mar. 1993.

[64] ISO/IEC JTC1/SC29/WG11, "Information Technology — Coding of Moving Pictures and Associated Audio for Digital Storage Media at Up To About 1.5 Mbit/s, Part 2: Video," ISO/IEC 11172-2 (MPEG-1), Nov. 1992.

[65] ISO/IEC JTC1/SC29/WG11, "Information Technology — Generic Coding of Moving Pictures and Associated Audio Information, Part 2: Video," ISO/IEC 13818-2 and ITU-T Rec. H.262 (MPEG-2), Nov. 1994.

[66] ITU-T, "Video Coding for Low Bit Rate Communication," ITU-T Rec. H.263, Version 2, Jan. 1998.

[67] S. C. Tai, Y. Y. Chen and S. F. Sheu, "Deblocking filter for Low Bit Rate MPEG-4 Video," *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 15, no. 6, pp. 733-741, June 2005.

[68] ITU-T and ISO/IEC, "Advanced Video Coding for Generic Audiovisual Services," ITU-T Rec. H.264 and ISO/IEC 14496-10 AVC (H.264/AVC), 2003.

[69] ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6, "H.264 /MPEG-4 AVC Reference Software," ISO/IEC and ITU JVT-X072, July 2007.

[70] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer and T. Wedi, "Video Coding with H.264/AVC: Tools, Performance, and Complexity," *IEEE Circuits and Syst. Magazine,* vol. 4, no. 1, pp. 7-28, First Quarter 2004.

[71] A. Puri, X. Chen and A. Luthra, ”Video Coding Using the H.264/MPEG-4 AVC Compression Standard,” *Signal Processing: Image Communication,* vol. 19, pp. 793-849, 2004.

[72] T. Wiegand, G. J. Sullivan, G. Bjontegaard and A. Luthra, ”Overview of the H.264/AVC Video Coding Standard,” *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 13, no. 7, pp. 560-576, July 2003.

[73] G. J. Sullivan, P. N. Topiwala, and A. Luthra, ”The h.264/avc advanced video coding standard: overview and introduction to the fidelity range extensions,” *Applications of Digital Image Processing XXVII, SPIE 2004*, vol. 5558, no. 1, pp. 454-474, August 2004.

[74] M. Karczewicz and R. Kurceren, ”The SP- and SI-Frames Design for H.264/AVC,” *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 13, no. 7, pp. 637-644, Jul. 2003.

[75] X. Sun, S. Li, F. Wu, J. Shen, and W. Gao, ”The improved sp frame coding technique for the JVT standard,” *Proc. IEEE Int. Conf. Image Processing (ICIP'03)*, vol. 3, pp. III 297-300 vol. 2, 14-17 Sept. 2003.

[76] H. Schwarz, D. Marpe and T. Wiegand, ”Analysis of Hierarchical B Pictures and MCTF,” *IEEE International Conference on Multimedia and Expo*, pp.1929-1932, 2006.

[77] ”ITU-T Recommendation H.264 : Advanced video coding for generic audiovisual services,” *International Telecommunications Union*, Nov. 2007.

[78] A. M. Tourapis, A. Leontaris, K. Suehring, and G. Sullivan, ”H.264/mpeg-4 AVC Reference Software Manual, JVT-AD010,” *Tech. Rep.,* January 2009.

[79] K. P. Lim, G. Sullivan, and T. Wiegand, ”Text Description of Joint Model Reference Encoding Methods and Decoding Concealment Methods, JVT-O079,” *Tech. Rep.,* April 2005.

[80] Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand, ”Overview of the High Efficiency Video Coding (HEVC) Standard,” *IEEE Trans. Circuits and Syst. for Video Technol.*, vol. 22, no. 12, December 2012.

[81] M. Z. Brown, D. Burschka and G. D. Hager, "Advances in Computational Stereo," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 25, no. 8, pp. 993-1008, August 2003.

[82] F. Remondino, S. F. El-Hakim, A. Gruen, and L. Zhang, "Turning images into 3-D models," *IEEE Signal Processing Magazine,* vol. 25, no. 4, pp. 55-65, July 2008.

[83] J. Xin, C. Lin and M. Sun, "Digital Video Transcoding," *Proceedings of the IEEE,* vol. 93, no. 1, pp. 84-97, January 2005.

[84] J. H. Hur, S. Cho and Y. L. Lee, "Adaptive Local Illumination Change Compensation Method for H.264/AVC-Based Multiview Video Coding," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 17, no. 11, pp. 1496-1505, November 2007.

[85] P. Merkle, A. Smolic, K. Muller and T. Wiegand, "Efficient Prediction Structures for Multiview Video Coding," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 17, no. 11, pp. 1461-1473, November 2007.

[86] X. San, H. Cai, J. G. Lou and J. Li, "Multiview Image Coding Based on Geometric Prediction," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 17, no. 11, pp. 1536-1548, November 2007.

[87] E. Kurutepe, M. R. Civanlar and A. M. Teklap, "Client-Driven Selective Streaming of Multiview Video for Interactive 3DTV," *IEEE Trans. Circuits and Syst. for Video Technol.,* vol. 17, no. 11, pp. 1558-1565, November 2007.

[88] R. Hartley, "In defense of the 8-point algorithm," *IEEE trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 6, pp. 580-593, June 1997.

[89] A. Zakhor and F. Lari, "Edge-Based 3-D Camera Motion Estimation with Application to Video Coding," *IEEE Trans. Image Processing*, vol. 2, no. 4, pp. 481-498, Oct. 1993.

[90] T. S. Huang and A. N. Netravali, "Motion and Structure from Feature Correspondences: A Review," *Proc. IEEE*, vol. 82, no. 2, pp. 252-268, Feb. 1994.

[91] J. J. Weng and T. S. Huang, "3-D Motion Analysis from Image Sequences Using Point Correspondences," *Handbook of Pattern Recognition and Computer Vision*, pp. 395-441, World Scientific Publishing Co., 1993.

[92] J. Weng, T. S. Huang and N. Ahuja, "Motion and Structure from Two Perspective Views: Algorithms, Error Analysis, and Error Estimation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 11, no. 5, pp. 451-476, May 1989.

[93] S. Salehi and E. Dubois "Alignment of Cubic-Panorama Image Datasets Using Epipolar Geometry," *Proceedings, Intl. Conf. on Acoustics, Speech and Signal Processing*, 22-27 May 2011, Prague, pp. 1545-1548.

[94] J. R. Ohm, "Advances in Scalable Video Coding," *Proceedings of the IEEE,* vol. 93, no. 1, pp. 42-56, Jan. 2005.

[95] T. Halbach, "Comparison of Open and Free Video Compression Systems," *Proc. International Conference on Imaging Theory and Applications (IMAGAPP)*, Lisboa (Portugal), Feb. 2009.

[96] A. Ravi, "Performance Analysis and Comparison of Dirac Video Codec With H.264 / MPEG-4 Part 10 AVC," *MSc Thesis, University of Texas at Arlington*, Aug. 2009.

[97] Z. Wang, A. C. Bovik, H. R. Sheikh and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing,* vol. 13, no. 4, pp. 600-612, Apr. 2004.

[98] S. Salehi and E. Dubois "Cubic-Panorama Image Dataset Compression," *Proc. SPIE Electronic Imaging Symp., Conf. on Visual Information Processing and Communication III*, 22-26 January 2012, San Francisco, vol. 8305.

[99] T. Shanableh and M. Ghanbari, "Heterogeneous Video Transcoding to Lower Spatio-Temporal Resolutions and Different Encoding Formats," *IEEE Trans. Multimedia*, vol. 2, no. 2, pp.101-110, 2000.

[100] F. Dufaux and J. Konrad, "Efficient, Robust, and Fast Global Motion Estimation for Video Coding," *IEEE Trans. Image Processing,* vol. 9, no. 3, pp. 497-501, Mar. 2000.

[101] S. Salehi and E. Dubois "Cubic-Panorama Image Dataset Analysis for Storage and Transmission," *Proc. SPIE Electronic Imaging Symp., Conf. on Visual Information Processing and Communication IV*, 3-7 Februrary 2013, San Francisco. vol. 8666.