Compression and Transmission of Multiview Video Data

Ricardo da Silva, Rong Zhang, *Member, IEEE*, and Lajos Hanzo, *Fellow, IEEE*

***Abstract*—As demand increases for visually detailed and immersive media, efforts to achieve better transmission schemes also increases. 3D displays are a relatively novel technology, and there are many different ways of creating a 3D viewing experience. This report investigates compression and transmission schemes for multi-view and stereoscopic video, and presents implementation results of simple motion compensation and DPCM codecs. A block based motion estimation codec is designed, which is shown to work better than frame differencing, especially for video sequences with high amounts of motion. A tree based motion search algorithm was shown to be far less complex, with only a small reduction in quality, when compared with an exhaustive search. Furthermore, a differential pulse code modulation codec is implemented and paired with motion compensation, which reduced the energy of signals to be transmitted. Finally, the motion compensation codec is used to predict right from left frames in stereoscopic video, but it is found to be unsuited to this task. For these simple algorithms, it is more efficient to simply encode the streams separately.**

***Index Terms*—multiview coding, stereoscopic, holographic display, video codec, motion compensation, dpcm, FEC, UEP, video transmission**

I. Introduction

STATE OF THE ART three dimensional display systems, such as the Holografika Holovizio system, are capable of displaying 3D video as a light field, such that each spatial point in the object is projected individually. This is in contrast with current consumer grade 3D displays that require goggles, or use some form of viewer tracking to correctly project the scene. As 3D visualisation can be considered to be the ultimate goal of non-interactive display technology, a related goal is how to achieve error resilient transmission of 3D video. Currently, most research in this area revolves around commonly used 3D systems, including stereoscopic and multiview autostereoscopic displays.

In general there are several categories of 3D video types that may considered here. Stereoscopic video, where there are two views for every frame, is the simplest. More views may be added, however, to create MultiView video. The simplest, least efficient way of compressing stereoscopic video is simply to treat each view as independent and use an industry standard such as H.264 to compress each stream. However, this neglects the opportunity for optimisation based on inter-view correlations. Thus one question of interest is how multiview video may be efficiently coded in order to minimise bitrate, while maintaining error resilience. In addition, the Holovizio system uses its own specialised video transport mechanism which must be taken into account when designing the coding scheme [1].

Error resilience is a desirable property of any transmission scheme, and there is substancial research into source and channel coding schemes that improve this. Many techniques that are applicable to multi-layer or multi-view video are equally applicable to more complex forms of 3D video. Once source coding is applied, the data may be protected by one of the common error correcting codes, such as Turbo or Low Density Parity Check (LDPC) codes. In addition, there are options to use an equal error protection (EEP) or unequal error protection (UEP) scheme, each of which have trade-offs [2], [3]. Error concealment techniques can also be beneficial, and can be used to correct errors in a video up to a visually acceptable level [4].

This report is divided into two main sections. First, currently used 3D data formats and coding systems are reviewed, giving an overview of modern high complexity source and channel coding schemes. Secondly, a more detailed treatment of simple error correction and motion compensation techniques is given, including implementation of a DPCM codec combined with block based inter-frame motion compensation. The application of these techniques to stereoscopic video is also investigated.

II. Modern Techniques

A. Current 3D Video Systems

By far the most common form of 3D display is those that require glasses and provide a stereoscopic view. They are limited in that they only create a single fixed 3D view which is the same irrespective of viewing position [5].

Autostereoscopic displays improve on this by projecting a 3D image that is viewable without glasses or other special hardware. This is accomplished with either viewer (head, or eye position) tracking, or through some form of multi-view display. The second method includes holographic, volumetric, parallax barrier or lenticular lens displays, and finally, light field displays.

B. The Holovisio Display System

Unlike the majority of autostereoscopic 3D displays currently in use, which rely on some form of view partioning, the Holovizio system uses light field technology to create a seamless three dimensional view. This results in a display that requires no glasses to view it, and is free of glitches when the viewing position changes [1]. Currently the Holovizio display is more useful for displaying 3D model data, as a library was created to allow direct interception of OpenGL rendering calls.

For simple use of the display, a custom video renderer and player is provided by Holografika, which allow arbitrary 3D

scenes to be converted into the form required to build a light field, and then sent to the display. However, as mentioned by Balogh [1], applying Multiview Coding (MVC) to the light field data for transmission is extremely desirable in order to reduce bitrate.

In addition, if joint source channel coding (JSCC) is desired, several works exist where MVC is paired with channel coding such as Turbo codes (in [6]) in order to realise this.

C. 3D Representations

3D video data may be represented in several different ways, each of which have drawbacks and advantages. In general, these can be grouped into three types – polygon based representation, single video plus depth information, and parallel video [2].

Polygon based representations are mainly associated with 3D graphics software, or Computer Aided Design (CAD), where a 3D scene is represented by groups of polygons. This is ideal for synthetic scenes, but is not useful for representing real world scenes, as complex algorithms would be required to construct polygons from a set of views, and the level of detail required is very high.

Video plus depth (V+D) coding has the bonus of being backward compatible with monoscopic coding, and is not fixed to a certain display size. Each frame in the video is comprised of an image containing colour, and a secondary image containing (grayscale) depth information, allowing the 3D scene to be rendered by the receiver. However, generating the depth information is not simple, and requires either computer vision analysis of the images, or through special hardware. Once acquired however, relatively simple calculations may be used to reconstruct the 3D scene [5].

Parallel video streams (one for each view of the scene) may be compressed jointly or independently, referred to as multiview coding and simulcast, respectively. Simulcast is equivalent to simply compressing multiple video streams, and thus the least complex, but it is also the least bandwidth efficient. Multiview coding (MVC) targets inter-view as well as temporal redundency in the video in order to aid compression [7]. MVC is also backward compatible, in that displays can choose to ignore the extra 3D data transmitted alongside the 2D video. In the stereoscopic case, this would simply be a case of ignoring one view, for example.

D. Advanced Video Formats

As the demand for media increases, so does the complexity and efficacy of video compression formats.

High Efficiency Video Coding (HEVC) is being prepared as the next generation video coding standard (of the ISO/IEC MPEG and the ITU-T Video Coding Experts Group) [8]. This aims to improve on the current H.264/MPEG-4 Advanced Video Coding (AVC) standard. The main goals of HEVC are to improve compression performance and data loss robustness, while maintaining perceived image quality, in order to allow transmission and storage of very high bandwidth applications. These include HD and 'beyond HD' formats (4k and 8k resolutions), and high resolution stereo or multiview media.

In addition, Multiview Video Coding (MVC, ISO/IEC 14496-10:2008 Amendment 1 ITU-T H.264) is an extension of AVC which is aimed compressing at multiple camera video data [7]. MVC is therefore a good candidate for use with the Holovizio system, as it can exploit the multi-view format that Holovizio uses.

Mixed resolution stereo coding (MRSC) is a parallel coding technique in which one of the views (in a stereoscopic system, for example) is kept intact, while the other is spcially decimated, up to a point where depth is still perceived. As this technique is based on a perceptual quality assessment (perceived signal to noise ratio, PSNR), it is hard to determine its effectiveness. Brust et al. [9] performed a series of tests comparing the perceived quality of mixed resolution (MR) and full resolution (FR) stereo sequences, while varying the mixed resolution coding parameters. It was found that at lower bitrates, while appropriate video coding was used, the MR sequences were perceived to be higher quality. However, the optimum bit rate was sequence dependent.

III. 3D Video Transmission

A good overview of current feasible transmission schemes is given in [2]. A variety of different coding methods are tested and evaluated, and results showed that the H.264/AVC MVC extension (with simplified referencing) performed relatively well. Rate-distortion (RD) analysis showed that MVC performed better than simulcast, and that video plus depth has better RD performance than stereo video with simulcast. In addition, several different coding methods (MRSC, MVC, Simulcast, Video+Depth) were used to code a set of test videos at fixed bitrates, which were then tested for viewer satistfaction. However, MVC and Video+Depth consistently scored higher satisfaction levels in the tests.

A complete transmission simulation was also built by [2], and evaluated the use of DVB-H as a transmission channel. DVB-H extends DVB-T with various features targetted at improving mobile transmission performance. Variable-length Reed-Salomon (RS) coding is used to protect the data, and time slicing (burst-style transmission) is used to improve power performance at the receiver. MVC and V+D coding methods were tested, and various EEP and UEP schemes were used. Overall, it was found that MVC performed better with EEP, and required a less complex receiver.

Ramzan et al. [6] present an approach for a Joint Source and Channel Coding (JSCC) scheme, based on multiview coding and double binary Turbo code (DBTC) forward error correction. The MVC extension to H.264/AVC is used, and the JSCC scheme is designed in order to achieve the required bitrate. Their system employs stereoscopic MVC predictive coding, such that there are three levels of frame importance, depending on the view and type of frame, allowing unequal error protection (UEP) to be used. An iterative algorithm is proposed which converges on the optimal allocation of code rate across the different levels. This so-called "Optimal Protection Scheme" is compared with equal error protection schemes, and other UEP methods, and found to outperform them at low bitrates.

IV. Low Complexity Techniques

The high complexity codecs mentioned in the previous section are all products of years of research, but at their cores are fairly simple concepts. Video contains both temporal and spatial redundancy, and a primary goal is to reduce this redundancy in order to reduce the bitrate, while maintaining the perceived quality. Motion compensation, which is a component of practically all modern video codecs, is intended to reduce temporal redundancy between sequential video frames. This can be done in several ways, such as simple block based movement tracking, or by performing the motion estimation in a transform domain such as the discrete cosine transform (DCT). Typically, in motion compensation algorithms, motion vectors are computed, and then a (hopefully) low energy error signal is calculated, which is used for reconstruction. In theory, the video can be losslessly compressed using motion compensation, although in practical usage some information is lost during quantisation of the error signal for transmission. This loss can be reduced by encoding the signal. Differential Pulse Code Modulation (DPCM) encodes a stream of samples using a predictive function, such that the total energy of the signal is reduced, and a lower bitrate can be achieved. This is beneficial as it reduces the range the quantiser must cover, and hence improves the quality of the reconstructed data. Block truncation coding (BTC) is another relatively simple coding technique, which has the advantage of not inflicting any spatial frequency limitations, unlike other transform based methods [10].

The goal of the following sections is to investigate simple motion compensation and coding methods, with a specific focus on block based motion compensation, and DPCM coding for transmission. To test the implementations, commonly available video sequences are used, including 'Suzie', 'Carphone', 'Football', and 'Miss America', in YUV (QCIF) format.

A. Qualitative Measures

Before we can evaluate the performance of a compression scheme or codec, it is necessary to choose a criterion that gives an approximation of the quality of a reproduced image. The Peak Signal to Noise Ratio (PSNR) will be used for this purposed, and is defined as:

$$PSNR = 10\log_{10}\frac{\sum_{i=1}^{p}\sum_{j=1}^{q}\max^2}{\sum_{i=1}^{p}\sum_{j=1}^{q}\left(f_n(i,j)-\tilde{f}_n(i,j)\right)^2} \quad (1)$$

where $\tilde{f}_n$ is the reconstructed $p \times q$ sized frame as a function of horizontal and vertical pixel locations i, j, and $\max^2$ is the maximum possible signal energy (255^2 in the case of an 8-bit image). This is chosen over the more common Signal to Noise Ratio (SNR), as it is believed to correlate better with human perception of quality.

B. Motion Compensation

Image sequences contain temporal redundancy between frames if there has been no drastic change of scene, and objects in the view have simply moved around. In other words, consecutive frames are often fairly correlated, and hence could be represented in terms of each other. A simple way to reduce redundancy is to compute the pixel-wise difference between two frames, and use this as the representation of the second frame. The result is referred to as the Motion Compensated Error Residual (MCER), and is given by $e_n = f_n - f_{n-1}$, where f_n is the n'th frame of the sequence and f_{n-1} is the previous frame. e_n can then be encoded (using DPCM, for example) for transmission, and will require fewer bits to represent than f_n. The receiver can reconstruct the current frame by adding the MCER to the previously received frame. This scheme is shown in Fig. 1.

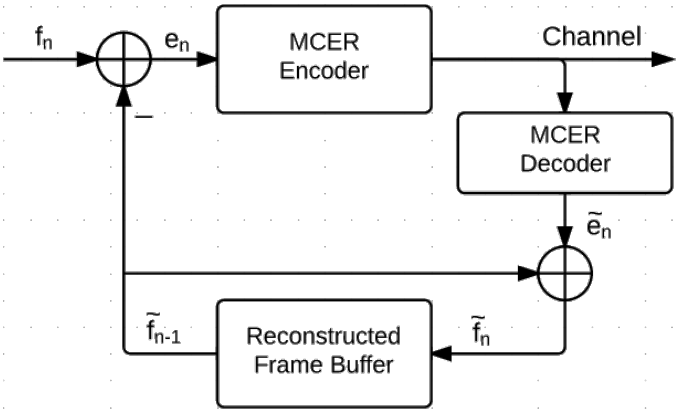

Fig. 1. Frame-differencing video codec block diagram

Note that the receiver only has access to a quantised version of e_n, denoted by $\tilde{e}_n$, and hence the reconstructed frame will be distorted. In order to synchronise transmitter and receiver states as much as possible, the transmitter makes use of a "local decoder", which performs the same operations as the receiver. Thus, the frame used for motion prediction is the locally decoded previous frame, $\tilde{f}_{n-1}$, which will contain the same quantisation noise that is present at the receiver. As the receiver does not have access to the original previous frame, it cannot be used to perform the motion compensation, or there would be errors in the reconstruction due to the mismatches between the original and quantised versions. Of course, channel effects may cause further errors which will accumulate in the receiver, unless error correction measures are used.

1) Block Based Motion Compensation: The simple frame differencing scheme is only useful in sequences with very little movement, and does not perform well with more dynamic video. True motion estimation would represent every pixel of a frame in terms of the motion of a pixel in the previous frame. However, this is overly complex, and good performance can be achieved by performing motion estimation on blocks of pixels instead. In principle, this is performed by dividing each frame into a number of perfectly tiling blocks, and finding a block in the previous frame that best matches each block in the current frame. The "motion vectors" are then the vectors that describe the displacement of each block from the best matching previous block. To phrase this more explicitly, the $x \times y$ image is divided into a number of $b \times b$ pixel sized tiles, where b divides x and y. For each block, a search area is

defined that surrounds the corresponding block in the previous frame, which is explored to find the block that best matches the current block. Once this is identified, the two blocks are subtracted from each other, and the result is taken as the corresponding block of the MCER for that frame. Thus the MCER should have very low energy if the scene change is primarily made up of small movements.

The Mean Square Error criterion is used to choose the "best matching" block referred to previously, which is defined as:

$$M_{mse} = \arg\min_{m_x,m_y} \sqrt{\sum_{i=1}^{b}\sum_{j=1}^{b}\left(\begin{array}{l} f_n(x+i,y+j) \\ -f_{n-1}(x+i-m_x,y+j-m_y)\end{array}\right)^2}$$

where m_x, m_y are the components of the displacement vector (measured in pixels). Essentially, this equation finds a displacement vector which minimises the pixel-wise difference between the current block and another block in the search area.

The motion compensation block diagram is shown in Fig. 2, including the local decoder for the same reasons as with simple frame differencing.

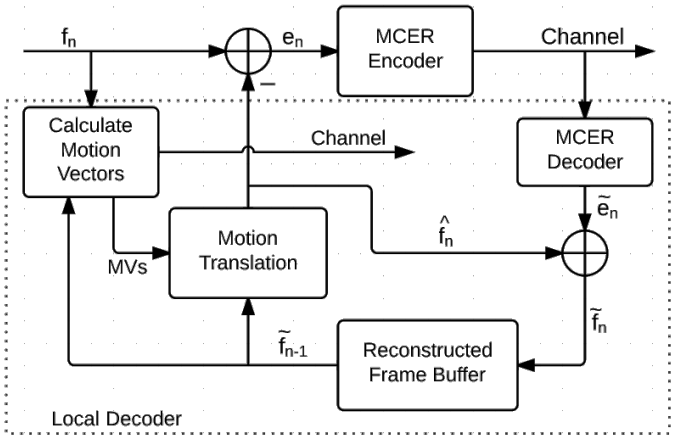

Fig. 2. Block-based motion compensation schematic

There are many ways this basic algorithm may be improved. For example, the naive approach of transmitting all the motion vectors results in a large bitrate overhead, and in many cases may be unnecessary. In the areas of the frame where little movement occurs, simple frame differencing is far more efficient. Hanzo et al. [10] noted that only retaining the most significant 25% of motion vectors of a given frame still resulted in relatively good performance, at a far lower bit rate. Another issue is the size of the search space relative to the block size, and hence number of bits to be assigned to the motion vectors.

A series of experiments was performed with this algorithm, and results are shown here. Firstly, the ability of motion estimation to reduce the energy of the transmitted signal is evident when comparing the PDF of the original signal, shown in Fig. 3, with the PDF of the MCER, shown in Fig. 4. In this case the Suzie sequence was used, which has fairly low motion, and if a video with higher motion (such as the Football sequence) was used, the PDF would be more widely spread. However, these results show how the motion compensation reduces the signal energy substantially.

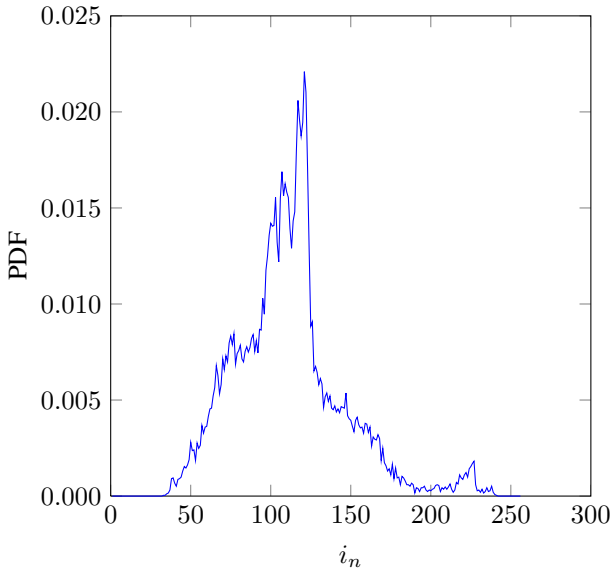

Fig. 3. PDF of a typical frame (Suzie)

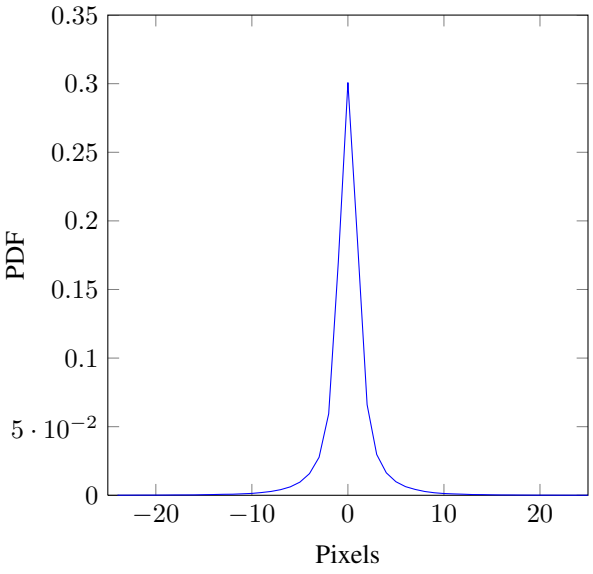

Fig. 4. PDF of a typical MCER frame (Suzie)

Another interesting aspect is the performance of the algorithm as the number of bits allocated to the MCER is altered. Fig. 5 shows this relationship. However, it is also interesting to compare the performance of block based motion compensation with simple frame differencing. The algorithms were tested with three sequences, and the PSNR vs bitrate performance is shown in Fig. 6. The y-axis shows the difference in PSNR between the two methods for a given bitrate, averaged over several frames. As expected, for sequences with more motion, the frame differencing method performs worse, but with low motion videos the two methods are similar.

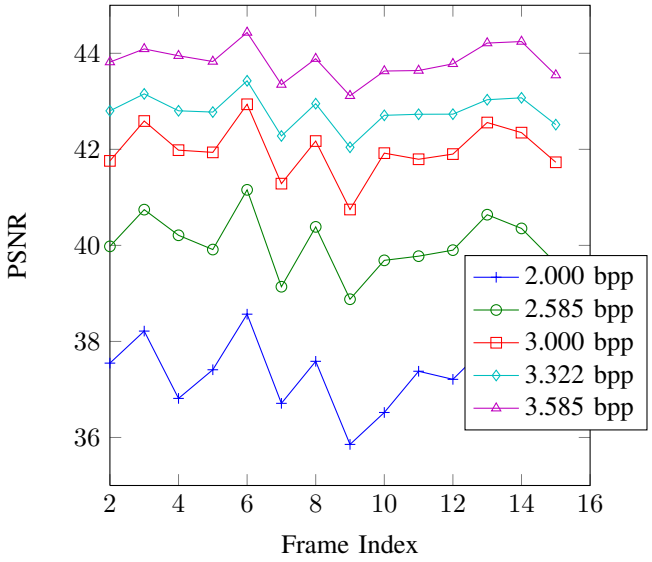

Fig. 5. PSNR vs bitrate (in bits/pixel, bpp) for Motion Compensation (Carphone sequence)

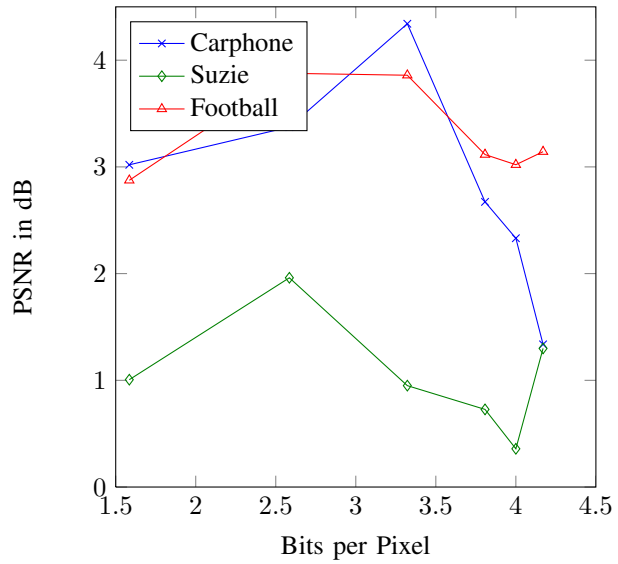

Fig. 6. Gain in PSNR of block based MC over frame differencing

Note that in these tests, a block size of 8×8 pixels was used, with a search window of 16×16. This resulted in motion vectors in the range of $[-7, 7]$, and hence each motion vector incurs $8/64 = 0.125$ bits/pixel overhead. Interestingly, for the Miss America sequence, it was found that simple frame differencing sometimes outperformed the block based motion compensation, and resulted in higher PSNR scores for a given bitrate. The movements occurring in this sequence are all small, and hence using 8×8 blocks essentially introduces more movement than there actually was, resulting in worse PSRNs.

In addition, Max-Lloyd quantisers were trained for each sequence in order to perform the required quantisation for each experiment. This was chosen due to its simplicity and relatively good performance.

2) Motion Search Algorithms: The search algorithm is another consideration of this type of codec, as it influences the computational complexity and feasibility.

a) Full Search: The exhaustive or full search simply applies the matching criterion to every possible previous block in the search window and finds the minimum. Thus it finds the optimal motion vector, but at a high computational cost. For example, with a window size of 16×16 and block size of 8×8 pixels, a full search requires 256 block comparisons.

b) Tree Search: It is possible to reduce the number of required comparisons, while maintaining high performance, using an iterative tree based search. One such method is known as a tree search, or the three step search. It assumes that the image sequence satisfies the so-called gradient constraint equation, discussed in [10]. There are several variations of this algorithm, but at the core is the idea of repeatedly partitioning the search space until a minimum is found. The variations mainly come from how to optimally partition the search space.

For this simple implementation, the algorithm converges in 3 iterations, using step sizes of 4, 2, and 1. Thus only 27 block comparisons have to be performed. In the first iteration, 8 points spread around the $(0, 0)$ point are visited, and the MCER value is computed for each of them. The lowest scoring point is chosen to be the next centre point, around which 8 more points are visited in a smaller search space. In the final iteration, the search space is reduced to a 3×3 grid, and the lowest scoring point is taken to be the optimal vector for the whole block. Figs. 7, 8 illustrates how even though this is a huge reduction in complexity, the PSNR is not affected greatly, especially in low motion video.

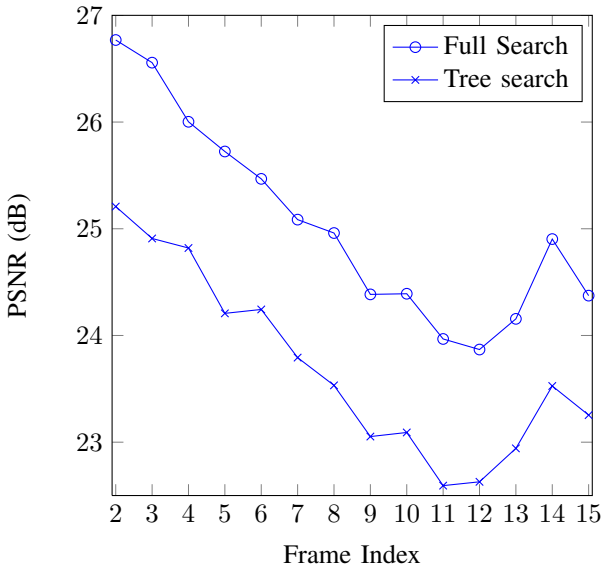

Fig. 7. High motion video (Football)

C. Differential Pulse Code Modulation (DPCM)

DPCM is based on the principle that adjacent pixels in an image are likely to be similar, and hence redundancy can be removed by representing each pixel in terms of an adjacent (usually previous) pixel. This is useful for encoding the MCER

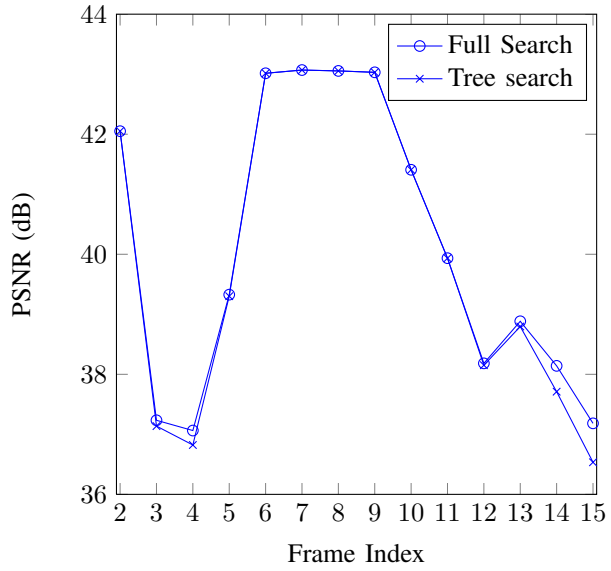

Fig. 8. Low motion video (Suzie)

signals derived in motion estimation algorithms, which often exhibit spatial correlation over adjacent pixels. A simple form of DPCM considers the current and previous pixel, i_n and i_{n-1} respectively, and defines the predicted value to be $e_n = i_n - a \cdot i_{n-1}$, where a is a constant between 0 and 1. It can be shown that the optimal value for a is the one-step correlation of the signal [10], which also makes sense intuitively. However, for this implementation the value was simply set at 0.95 for all sequences.

When this prediction function is applied to a video sequence, Suzie, for example, the resulting MCER signal is far more compressed than the original video, as shown in Figs. 3, 9.

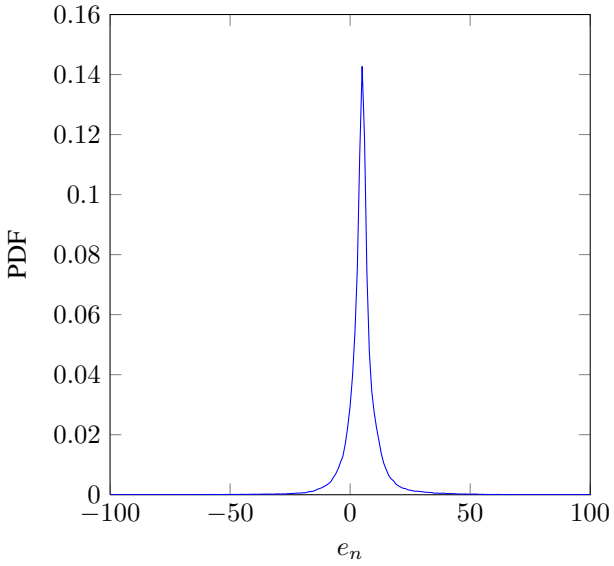

Fig. 9. PDF of a DPCM encoded stream from Suzie sequence

In this implementation, a "feedback" scheme is used, such that a local decoder is present at the encoder, similar to the motion compensation system. The transmitter reconstructs each sample locally, using a quantised version of e_n, so that the transmitter and receiver are in sync as much as possible.

D. DPCM with Motion Compensation

Now that a simple motion compensation scheme and an encoding method have been introduced, it is possible to combine the two, such that the compression is further improved.

The MCER output of a motion compensation algorithm, for example frame differencing, or the more complex block based method, must be transmitted over a channel (along with motion vectors if required). It has already been shown that the PDF of the block based MCER is already fairly compressed. However, it can be compressed further by using DPCM to encode the MCER, and transmitting the DPCM error residual. A set of experiments were performed, and the results are shown in Fig. 10. This shows the performance of the combined MC and DPCM codec, for different sequences and over a range of quantisation levels.

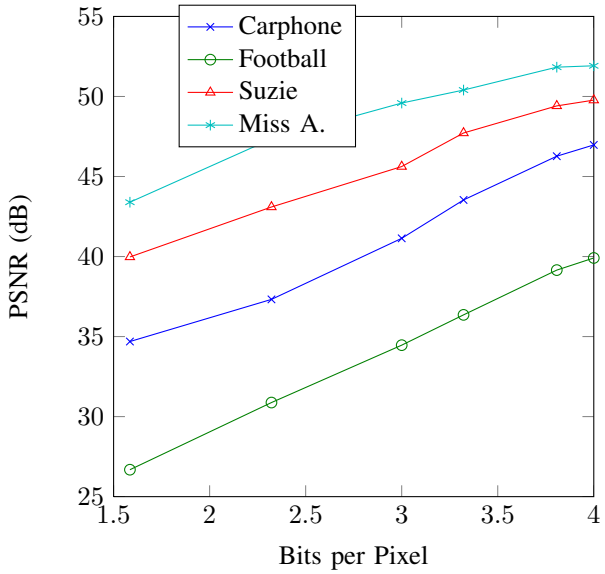

Fig. 10. PSNR vs bitrate for four sequences, combined MC and DPCM

V. Application to Stereoscopic Video

The techniques explored in the previous section were applied to monoscopic test video sequences in order to assess their performance. However, as this report began with a focus on multiview data, it is interesting to investigate the performance of such techniques when applied to stereoscopic video data. The "Book arrival" test sequence (in 432×240 resolution) was used for this purpose. This 3D sequence is represented by two separate video streams for left and right views (rather than Video+Depth or other possible 3D representations).

There are several ways stereoscopic video can be compressed, as discussed in Section II. The simplest and least bandwidth efficient method is to compress the two streams individually (simulcast), which is easily accomplished by adjusting the system to interleave the frames of each stream, and

use separate encoders for each. A more interesting approach is to remove redundancy that exists between left and right views by representing one in terms of the other. MVC coding, as used in [6], employs a predictive structure where the right view is dependent on the left view and the previous right view, which removes substantial redundancy.

It was decided to investigate simple approaches to the problem of removing redundancy between left/right frames. One option is to compute the difference between right and left frames, such that $e_n = f_{Ln} - f_{Rn}$, where f_{Ln} and f_{Rn} are the n'th left and right frames respectively. Then e_n can be interleaved with f_{Ln} and encoded as before (simulcast). However, it was hypothesised that the right view would be better represented by a horizontal shift of the left, rather than simple differencing, due to the parallax effect. To test this, the block based motion compensation algorithm was used to create a representation of the right view in terms of movement in the left view. The resulting signal has lower energy than the original view, with a PDF shown in Fig. 11.

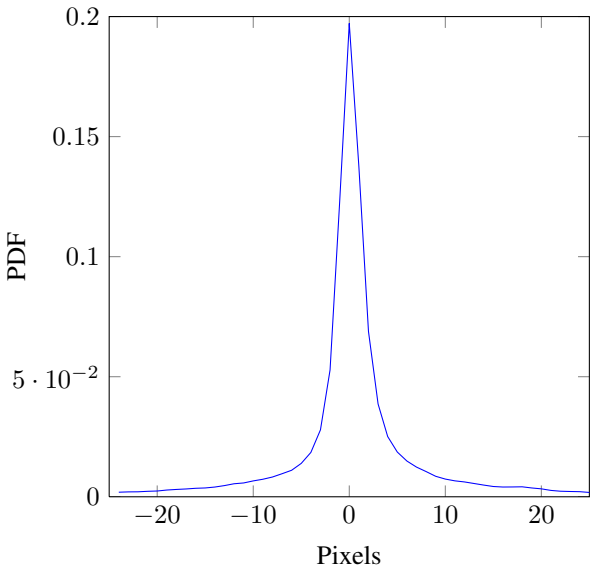

Fig. 11. PDF of the MCER for 5 frames of L/R motion estimation

However, it was found that this (and simpler view differencing) introduced substantial noise into the sequence, meaning that further compression became less effective. Encoding the motion estimation compressed right stream instead of the original stream ultimately resulted in a lower PSNR than if the original streams had simply been compressed separately (at the same bitrate), as shown in Fig. 12. It was concluded that simple motion prediction algorithms are inadequate for predicting the right view from the left in stereoscopic video.

VI. Conclusion

State of the art three dimensional viewing systems were introduced, along with the relevant video representations and compression techniques. In order to build understanding of the fundamentals of these high complexity codecs, a simple motion compensation predictive codec was implemented, as well as a differential pulse code modulation codec. The

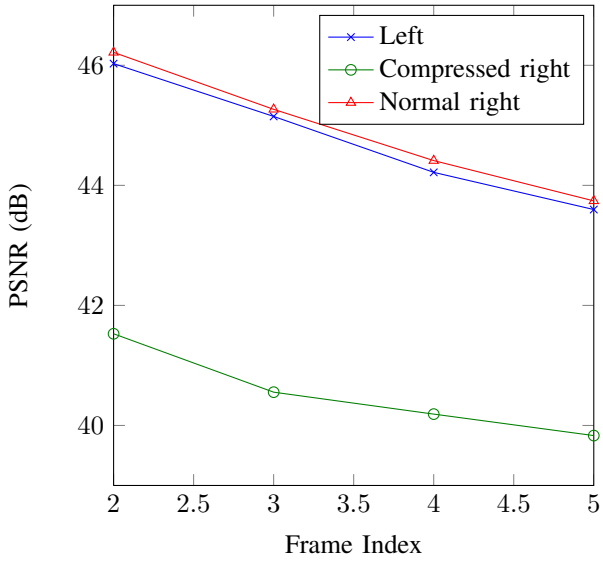

Fig. 12. PSNR values for left and right views with MC compression

performance of these two systems was investigated, and results were presented that showed how they improve the PSNR for a given bitrate. Stereoscopic video was also tested with the same algorithms. It was found that the simple motion compensation algorithm is not suited to predicting right from left frames in stereoscopic video, as the resulting prediction is too noisy.

References

[1] T. Balogh, "The holovizio system," in *Electronic Imaging 2006*. International Society for Optics and Photonics, 2006, pp. 60 550U–60 550U.

[2] A. Gotchev, G. Akar, T. Capin, D. Strohmeier, and A. Boev, "Three-dimensional media for mobile devices," *Proceedings of the IEEE*, vol. 99, no. 4, pp. 708–741, April 2011.

[3] Y. Huo, M. El-Hajjar, R. Maunder, and L. Hanzo, "Layered wireless video relying on minimum-distortion inter-layer fec coding," *Multimedia, IEEE Transactions on*, vol. 16, no. 3, pp. 697–710, April 2014.

[4] O. Stankiewicz, K. Wegner, and M. Domanski, "Error concealment for mvc and 3d video coding," in *Picture Coding Symposium (PCS), 2010*, Dec 2010, pp. 498–501.

[5] Z. Megyesi, A. Barsi, and T. Balogh, "3d video visualization on the holovizio system," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2008*, May 2008, pp. 269–272.

[6] N. Ramzan, A. Amira, and C. Grecos, "Efficient transmission of multiview video over unreliable channels," in *Image Processing (ICIP), 2013 20th IEEE International Conference on*, Sept 2013, pp. 1885–1889.

[7] A. Vetro, T. Wiegand, and G. J. Sullivan, "Overview of the stereo and multiview video coding extensions of the h. 264/mpeg-4 avc standard," *Proceedings of the IEEE*, vol. 99, no. 4, pp. 626–642, 2011.

[8] G. J. Sullivan, J. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (hevc) standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 12, pp. 1649–1668, 2012.

[9] H. Brust, A. Smolic, K. Mueller, G. Tech, and T. Wiegand, "Mixed resolution coding of stereoscopic video for mobile devices," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2009*, May 2009, pp. 1–4.

[10] L. L. Hanzo, P. Cherriman, and J. Streit, *Video compression and communications: from basics to H. 261, H. 263, H. 264, MPEG4 for DVB and HSDPA-style adaptive turbo-transceivers*. John Wiley & Sons, 2007.