Proxy-Based Compression of $2\frac{1}{2}$D Structure of Dynamic Events for Tele-immersive Systems

Pooja Verlani, P. J. Narayanan
Centre for Visual Information Technology
IIIT, Hyderabad 500032 INDIA

Abstract

Capture of dynamic events is an active research area today. Capturing the $2\frac{1}{2}$D geometric structure and photometric appearance of dynamic scenes finds applications in 3D tele-conferencing systems, 3DTV etc. The captured "depth movies" contain aligned sequences of depth maps and textures and are often streamed to a distant location for immersive viewing. The depth maps are heavy and need efficient compression schemes. In this paper, we present a scheme to compress depth movies of human actors using a parametric proxy model for the underlying action. We use a generic articulated human model as the proxy and the various joint angles as its parameters for each time instant to represent a common prediction of the scene structure. The difference or residue between the captured depth and the depth of the proxy represents the scene to exploit spatial coherence. Differences in residues across time are used to exploit temporal coherence. Intra-frame coded frames and difference-coded frames provide random access and high compression. We show results on several synthetic and real actions to demonstrate the compression ratio and resulting quality using a depth-based rendering of the decoded scene.

1 Introduction

Digital capture of dynamic events has assumed great importance recently [5]. Sensors for 3D data including multi-camera systems, laser range scanners, etc are common today. Some of them are suitable for the real-time capture of the shape and appearance of dynamic events, like in the other previously existing multiview capture teleimmersive systems. The $2\frac{1}{2}$D model of aligned depth map and image, called a Depth Image, has been popular for Image Based Modeling and Rendering (IBMR). Time varying depth and image sequences, called *Depth Movies*, can extend IBMR to the dynamic events. The applications of such systems include virtual-space tele-conferencing, remote 3D immersion, 3D entertainment, etc. Tele-immersive environments are now emerging as the next generation of communication medium to allow remote users more effective interaction and collaboration in joint activities at various research, academic and social levels. Tele-immersion creates an illusion that the user is in the same physical space like the other remote participants, though they are miles apart.

Depth Movies are time-varying sequences of depth maps and texture, as shown in figure 1. A depth movie consists of three channels of data: (a) The *I Channel* contains the sequence of image with $I_k[i,j]$ giving the colour at pixel (i,j) at time instant k, (b) the *D Channel* contains the sequence of depth maps with $D_k[i,j]$ giving the depth or distance at pixel (i,j) at time instant k, and (c) the *C Channel* gives the sequence of time varying calibration parameters that help map the depth into the 3D coordinates in a common reference frame, with C_k giving the 3×4 calibration matrix. Continuous, hole-free capture requires multiple depth movies to cover the event from all directions. In this paper, we study dynamic events involving a single human actor, a common case for entertainment and tele-conferencing. We assume that the event is captured using m depth movies of length n. That is, the $2\frac{1}{2}$D structure of the event is available from m different positions or views for n uniformly sampled time-instants or "frames".

Depth movies are bulky and call for effective representations and compression methods. Compression is essential as the captured event is often transmitted to a remote location for 3D playback. For such a setting, the server at the capture site is linked over a network to a client at the rendering site. Video compression can be used effectively on the I Channel. Compression of static and dynamic light-fields have been studied before [8, 9, 14]. The C Channel contains light and slow-varying data and a standard data compression scheme can compress it effectively. The D Channel, however, contains time-varying depth maps from multiple sources. Effective compression of such data is important and is the focus of this paper. The D channel is a video of depth values. It may appear that video compression schemes like MPEG can work well on it. Critical differences between images and depth maps make this tricky

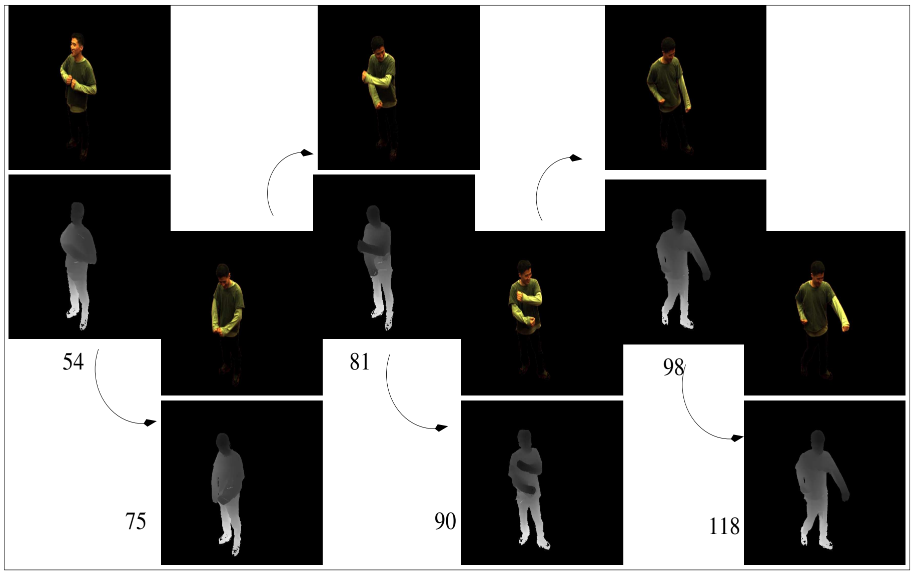


Figure 1. Doo-Young Dataset Depth Movie from ETH-Z.

and undesirable. Video compression is psycho-visually motivated and give less emphasis to the high frequency components. However, the high frequency regions represent occlusion boundaries; they are critical for depth maps [6], especially if used rendering.

Depths from different viewpoints contains redundant information derived from the common geometric structure of the scene. We exploit this spatial redundancy in addition to the temporal redundancy for effective compression of depth movies. The geometric structure is approximated using a light-weight, parametric *proxy* model for each time instant. We use an articulated human model as the parametric proxy with the joint angles serving as its parameters. The time-varying parameters approximate the underlying geometric structure of the action in a viewpoint independent manner. The depths from each viewpoint is replaced with the residue or the difference from the depth due to the proxy model. The residues are small in range and are correlated spatially and temporally. We show different ways to encode them for different quality/size trade-offs.

We present related work in section 2. Section 3 presents our work on proxy-based compression of depth movies, followed by results in section 4.

2 Related Work

Geometric structure of real-life scenes can be captured using multicamera setups, range scanners, etc. Several teleimmersion systems have been built for this purpose over the past decade or so [11, 15, 10, 2, 16, 4, 17]. They attempt to capture dense or sparse 3D structure of the scene using cameras as time-varying depth and texture maps or depth movies.

Disparity compensated compression of multiview images exploit the spatial redundancy in a lightfield. Levkovich-Maslyuk et al. compute the disparity maps and use them to predict the appearance in other views to exploit spatial redundancy of the lightfield [8]. Magnor et al. use block-based disparity compensation to predict other views of a multiview set [9]. They computed the disparity or depth maps for encoding but never stored them. Wu et al. extended disparity compensation to dynamic lightfields to compress video objects [14]. They used an MPEG-like algorithm to compress multiple video data using temporal correlation within one video stream and spatial correlation using depth values across views. They, however, treated depth maps as incompressible and encoded it in a separate layer without any loss.

The compression of depth maps hasn't attracted much attention before. Standard image compression methods like JPEG emphasize perceived visual quality and do not suit depth image compression [6]. They advocate the use of region of interest (RoI) and reshaping of the depths to preserve depth edges or occlusion boundaries when dealing with depth maps. Kum and Mayer-Patel compress multiple depth streams of a scene using motion vectors derived from color and depths [7]. They concluded that separate motion vectors to encode color and depth perform better than a common one. Penta and Narayanan introduced proxy-based compression of depth images of static scenes [12]. The geometric proxy model was used as the common structure of the scene and each depth map was represented as difference with the proxy. Simple triangulated proxies and paramet-

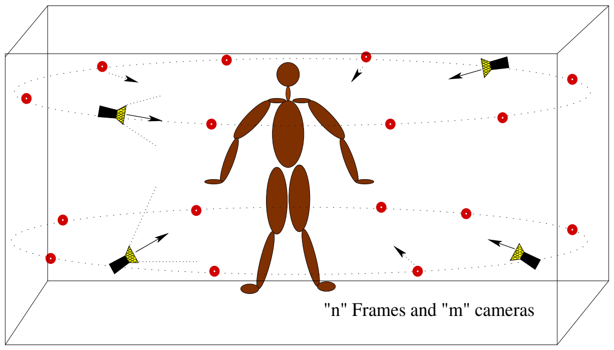


Figure 2. Setting of 20 cameras around the scene.

ric proxies like ellipsoids provided good compression and quality on static 3D scenes by them.

We extend the proxy-based compression to depth movies of human actors in this paper. The input to the system is multiple depth movies of a scene captured from a setup of cameras all around the scene as showing in figure 2.

We first fit an articulated proxy model to the point cloud for each frame. The output of this process is the joint angles of the model, which are the parameters of the proxy. Parametrization of the proxy reduces its memory requirements. Several methods try to fit an articulated model to real human data [13, 3, 1]. The fitted proxy for each frame is projected to each view to give predictions of the input depth maps. The residue or prediction error is sufficient to recover the input depth maps losslessly. The temporal correlation is exploited by encoding the residue differences between successive time instants.

3 Proxy-Based Compression

We consider depth movies involving a human performer in this paper. We use 16-bit integer depth values for the depth maps. This gives a compact and exact representation than using floating point without compromising on the range of values in practice.

Parametrized Proxy Model: An articulated, parametric human model is used as the proxy model. The joint angles and bone lengths are the free variables of the model and serve as the parameters using which any position of the actor can be represented. The parameters change over time, but are same for all views at any time instant. We use an articulated model with 18 joints. Each joint has 3 angles for heading, pitch, and roll. We currently use a single size parameter to scale the base model uniformly. Thus, the model is defined completely using only 55 parameters in each frame. The basic model is skeletal. A triangulated

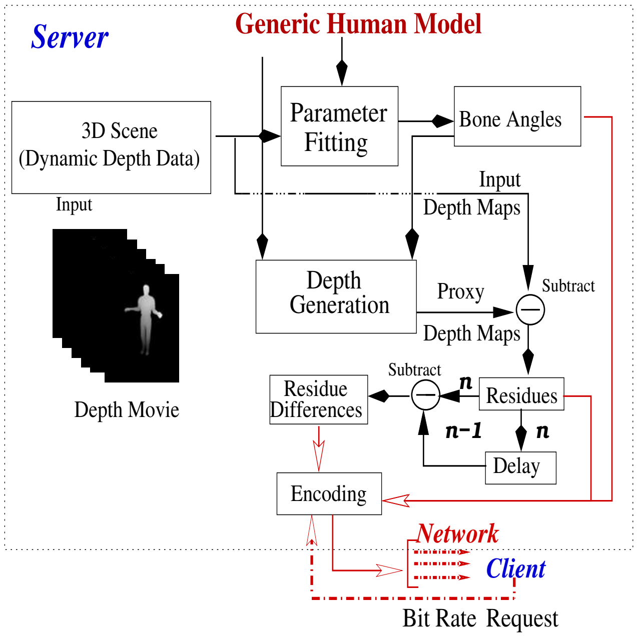


Figure 3. Encoding and transmission of compressed 3D scene at the Server

skin-model is attached to it and provides a realistic human model. The basic skeletal and skinned models are available to both the encoder and the decoder. Thus, the 55 parameters provide a decent prediction for the common geometric structure of the actor for each frame.

We compute the parameters of the proxy by first unprojecting each depth map from each view to get a point cloud in the world. The problem of fitting an articulated model from images and from depths has been studied in the past [13, 3, 1]. The fitting can be performed by optimizing the error between the skinned model and the point cloud. Fitting subsequent frames is easier as the parameters change slowly. Since the model fitting is not our main focus, we fit the model interactively using a semi-automatic tool built for the purpose to get a basic proxy with minimal efforts. In the end, the parameters of the articulated model for each frame represent the scene parametrically as a proxy.

Residues: Our basic idea is to replace each depth map as the deviation from the common proxy model, thus needing fewer bits to encode the information. We first compute the residue depth maps by projecting the fitted proxy model to each view for each frame. The *residue map* for each view is the pixel-by-pixel difference between its input depth map and the proxy depth map. This is the prediction error when the proxy model is used as a common prediction for all views. After this process, each of the mn depth maps is replaced by a residue map losslessly. The residues have a

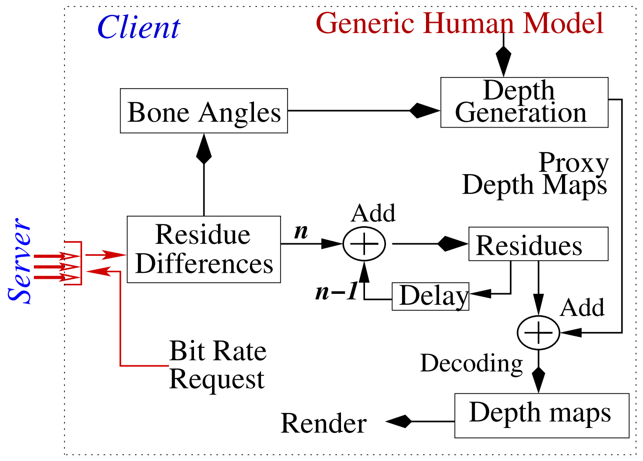
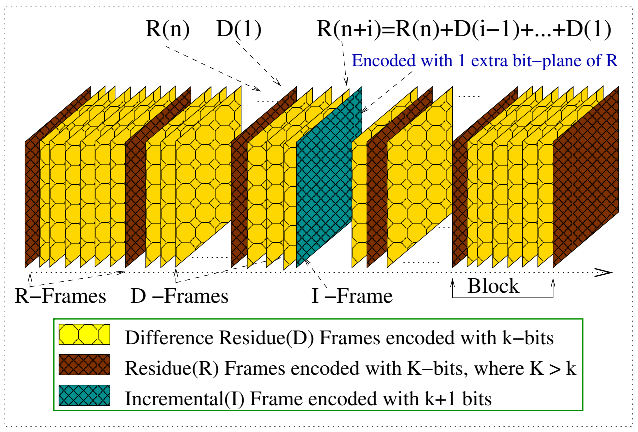


Figure 4. Decoding of residues at the Client



R(n)
D(1)
R(n+i)=R(n)+D(i−1)+...+D(1)
Encoded with 1 extra bit−plane of R
R−Frames
D −Frames
I −Frame
Block
Difference Residue(D) Frames encoded with k−bits
Residue(R) Frames encoded with K−bits, where K > k
Incremental(I) Frame encoded with k+1 bits

Figure 5. The structure of a block with R frames, D frames and optional I frames.

smaller range of values and are less bulky to represent. They are also more correlated spatially and temporally within a view, being the differences with a good approximation of the underlying structure.

Residue Encoding: For each residue map, we also compute a foreground mask bit to identify pixels which belong to the actor in each view. The mask is obtained by thresholding the input depth values. The residue values are represented using a sign-magnitude format to facilitate bit-plane encoding of residues. We explore two compression schemes for the residues.

Since residues are correlated spatially and temporally and are otherwise like videos for each view, we can encode them using a standard video coding scheme like MPEG. This gives m residue movies and has the random access properties and bit-rate control as the underlying video coding scheme. The video scheme may incur high reconstruction costs.

The other method uses bit-plane encoding of the residues, using as many bits as client demands. The scheme (Figure 3), inspired by video coding, is as follows.

1. Exploit temporal correlation by computing *residue differences* as $D_i = R_i - R_{i-1}$, where R_i is the residue for frame i.
2. Encode the residue map as *blocks* that contain one **R** frame of residue values and several **D** frames of residue differences. A block is a random-access unit and its length is determined by the requirements for random access.
3. Encode **R** frames with K most-significant bits of the residues and the **D** frames with k most-significant bits of the residue difference.
4. Encode mask bits using the JBIG algorithm.

Different quality points can be obtained by varying K and k, which can be varied online in a real-time client-server setup. We also use an incremental representation to exploit any additional available bandwidth. We send the next (i.e., $K+1$th) bit-plane of the residue as an **I** frame when this happens. The value received is added to the current **R** frame, thus improving the quality of all frames till the end of the block. As shown in Figure 5, an extra bit at **I** frame provides the increment in the bit representation for the following frames, thus there on increasing the quality of the movie.

$$R_i = R_0 + D_1 + D_2 + ... + D_{i-1} \quad (1)$$

$$R_i^{'} = R_0 + D_1 + \cdots + D_j^{'} + D_{j+1}^{'} \cdots + D_{i-1}^{'} \quad (2)$$

Equation 1 shows R_0 encoded with K bits and subsequent D frames encoded with k bits. When 1 extra bit is added to the j^{th} D frame as shown in equation 2, $D \rightarrow D^{'}$, and the subsequent D frames also get better representation. Thus, the quality of $R_i^{'}$ gets more than R_i.

Compressed Representation: The data to be sent to the rendering client includes the following. (a) The joint-angle parameters for each frame, (b) the mask bits for each view and each frame, (c) m MPEG streams for the residue values when using MPEG or the sign bit and **R** or**D** values for the each frame. We use the following compression schemes for each. Please note that the articulated model and the skinning triangles are available at both ends and need not be transmitted at all.

1. The model parameters (joint-angles) take only few bytes per frame. Only entropy encoding using zip is used to reduce its size.

2. The mask planes are bit-maps. They are compressed using a lossless JBIG format to get maximum compression. Run-length encoding and other schemes were also tried, but JBIG gave the best performance in all cases.
3. When using MPEG, the data is encoded using the `ffmpeg` library with the given bit-rate.
4. When using bit-plane encoding, the sign bits are compressed using the JBIG scheme. The **R** and **D** frames are compressed into a long sequence of bits and then entropy coded using zip.

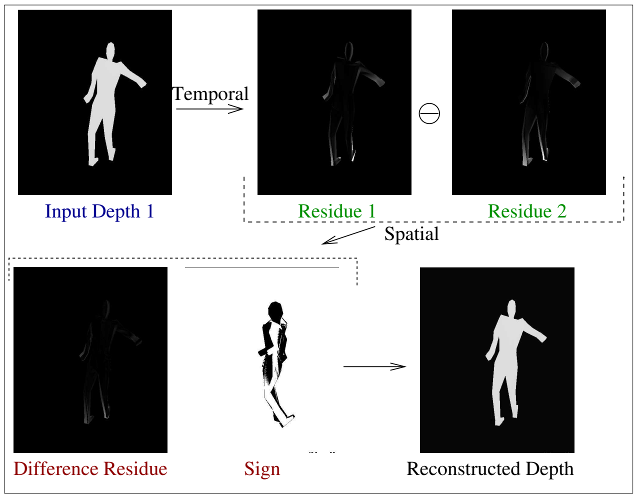


Figure 6. Results showing input depth, residues, residue differences, sign and reconstructed depth.

Each of the m views are treated independently for this process. The frames of each view are also independent, with some using residue values and others using residue differences. All data is zipped together at the end as a simple entropy-encoding method. This information is sent to the rendering client, which decodes the depths back. The final representation contains one parameter file per frame, one mask, one sign bit, and an **R** or **D** plane per frame per view. The data for a whole block (between two **R** frames for bit-plane coding or between two **I** frames for MPEG) needs to be together logically and can be treated like a package to be sent to the client.

Decoding: The decoding process is shown Figure 4. Each block of data is treated independently by the client. The joint-angle parameters are applied to the articulated model. The resulting model projected to the camera of each depth stream, giving the proxy depth maps. The residue maps are recovered from the **R** frames of the packet. The residue map is added to the proxy depth map to get the decoded depth map for that view and frame. For **D** frames, the residue differences recovered from the packet are added to the current residue map R_i to get the next residue map R_{i+1}, which is added to the proxy depth map for that frame to get the decoded depth values as given in equation 2. The foreground mask is needed to keep track of the changing silhouette of the actor. The depth map of frame $i+1$ may include pixels not in frame i. If frame $i+1$ is a **D** frame, the current average residue value is used as the reference for the residue difference. If the incremental frame arrives, the bit-plane for it is assembled and added to the current running residue R'_i. The improved quality results till the end of the current block.

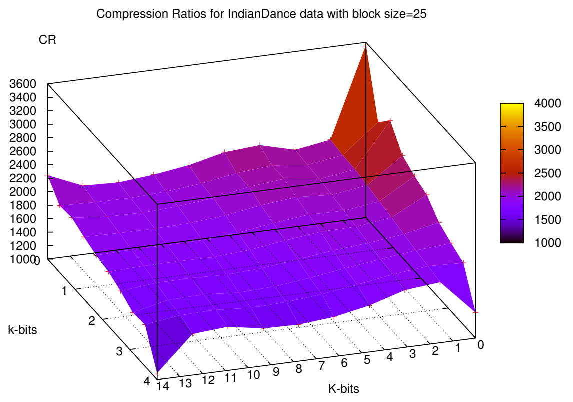
4 Experiments and Results

We present the results of our scheme now. We have experimented on real depth datasets and MOCAP datasets for dynamic scenes. Real dataset we have used is from ETH-Z, a Doo-Young karate sequence as shown in figure 1. As such real data is very rare, we have also experimented on real MOCAP (Motion Capture) data available freely with CMU. A standard POSER human model was animated using the MOCAP parameters and 16-bit depth maps were captured from 20 viewpoints. 16-bit depthmaps help in capturing depth to 65535 meters. The joint-angle parameters for compression are very similar to the MOCAP files. We, however, added noise to the joint angles to simulate bad fitting of the model to real data. We also added slow-varying noise to the depth values to simulate errors of the depth-recovery process. The depth noise has a small random component at each pixel which rides on top of a larger component that varies slowly over the whole depth map.

Fitting Procedure is simple with minimal human involvement. We can fit the first frame of a sequence in less than 60 seconds and the subsequent frames in less than 15 seconds, on an average.

The proxy model is articulated using the noisy angles to generate the depth maps and residues are generated by subtracting the fitted proxy depth maps from the input depth maps. Residue differences are obtained by subtracting R_i from R_{i+1}. Mask bit is obtained by thresholding the depth map. Encoding is performed as described in the previous section. Decoding is performed as the reverse process at the receivers end.

Results on a few synthetic datasets are shown in graphs of figure 8. We experimented on three MOCAP sequences, Indian dance, Ballet and Exercise, each with around 300-400 frames each. The compression ratio is with respect to the original, uncompressed depth maps. The PSNR is calculated by comparing the reconstructed depth maps with the input depth maps. The residue compression exploits the



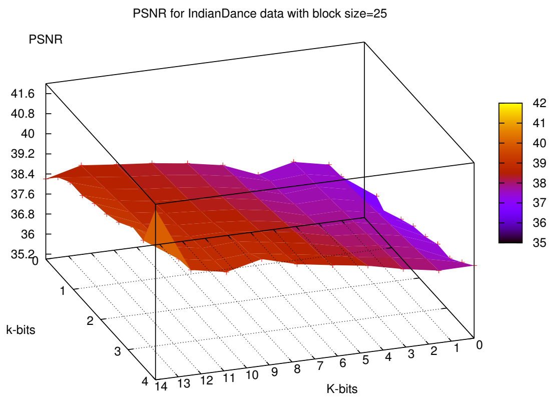


Figure 7. Results for IndianDance Dataset with block size=25, joint angle noise=5, and K=0,5-14, k=0-4

spatial redundancy but we do residue difference encoding scheme that exploits both the temporal and spatial aspects of Depth movies. The bit-plane scheme provides high compression and moderate quality at low number of bits and good compression and excellent quality at higher number of bits. As we increase the number of bits in encoding, the compression ratio, as expected, decreases with increase in the quality factor. Also, it provides totally random access of the depths of individual frames. Most interestingly, the option of using 0 bits of residue provides a very low bit-rate approximation of the input scene.

Other than joint angle noise and depth noise, we varied the block sizes for coding the depth movies to get nice compression figures with good quality. We observed that as the block size increases, the average compression ratio increases and the PSNR decreases. Thus, higher blocks are preferable for coding as encoding with K bits is lesser than k bits for D-Frames, where $K > k$. Keeping the block size constant, with increasing the joint noise the compression ratio reduces as higher bits are needed to fully represent the residues. Figure 7 plots the PSNR and the compression ratio against the number of bits used to encode the residues, K and number of bits used to encode residue differences, k, for one dataset. It can be seen that the PSNR varies slowly with the number of bits, but the compression ratio of bit-plane encoding is good.

We compared our method with the present state of Art, MPEG. The MPEG compression of depth and residues (MPEG-R and MPEG-D in graphs of figure 8) provides decent compression and quality, but the bit-plane encoding scheme provides more size to quality trade-offs to suit any situation.

With real dataset we carried out the same experiments. Doo-Young dataset consists of 8-bit images. The results for compression ratios and quality are as shown in figure 9. The point cloud, is fitted in the same manner as in the MOCAP dataset. Here, we do not have any noise levels since no simulation of noise is required as it being a real dataset. We observed that the trade-offs are much similar to the MOCAP simulated real dataset.

We observed from graphs in figures 8, 9, if the remote client asks for a particular range of compression ratios and quality, he has a set of choices among various combinations of K-bits, k-bits and block sizes. This makes the system effective for a remote-server-client teleimmersion environment with user compatible service options.

More results and videos for sequences with varying bits for encoding, levels of joint noise, depth noise and block sizes are provided in the supplementary material.

5 Conclusion

We presented a proxy-based compression scheme for multiple depth movies of a scene involving dynamic human action in this paper. We used a simple articulated model as the proxy and joint angles as the parameters that approximate the model for a particular frame. Spatial coherence of the structure between views is exploited using the common proxy model and replacing depths with differences with it. Temporal coherence is exploited using incremental encoding of these differences across time. The scheme provides good compression ratios at acceptable quality levels. The proxy-based scheme provides several controls on the amount of data to be sent. This makes it ideal for sending the captured data for applications like 3D teleconferencing.

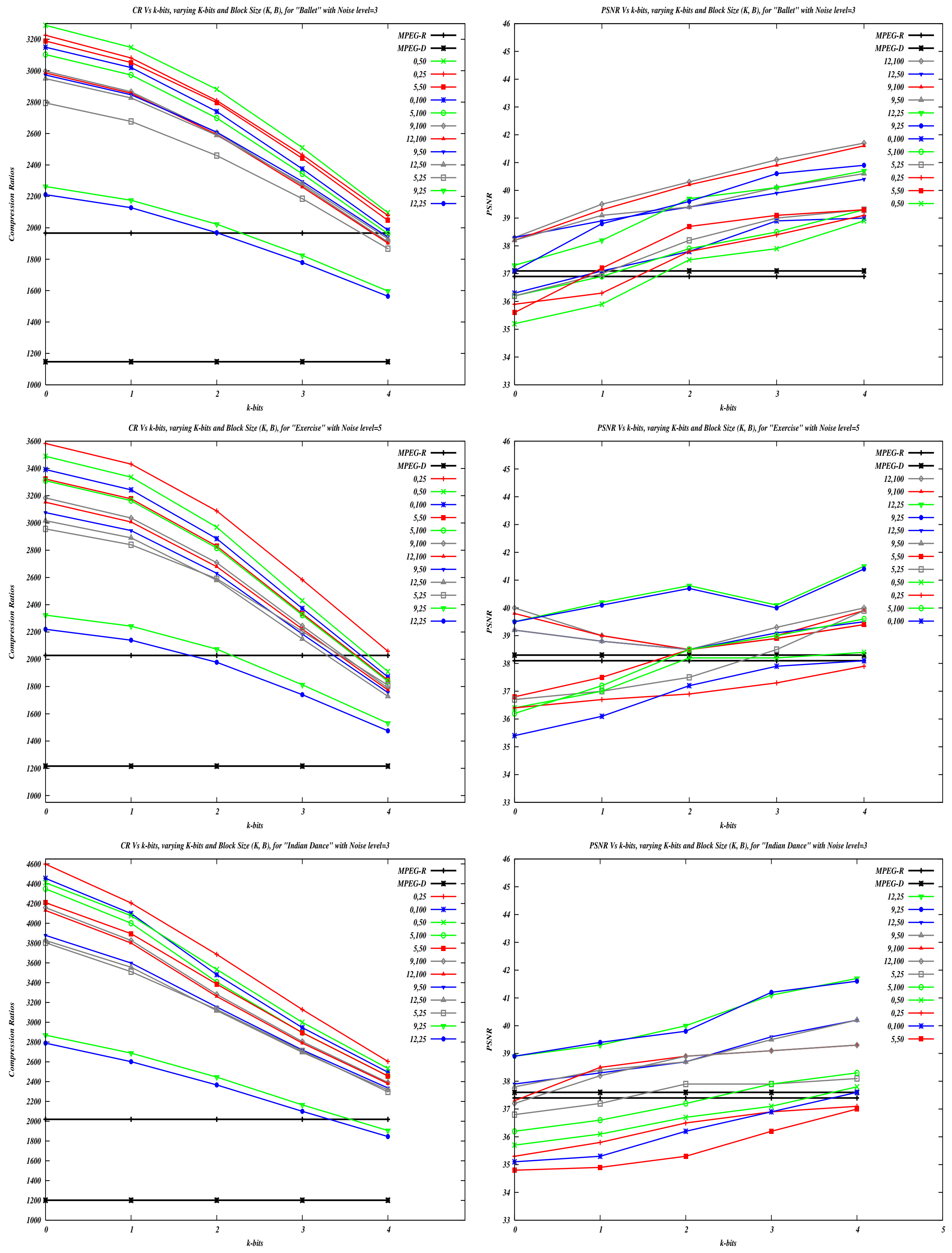
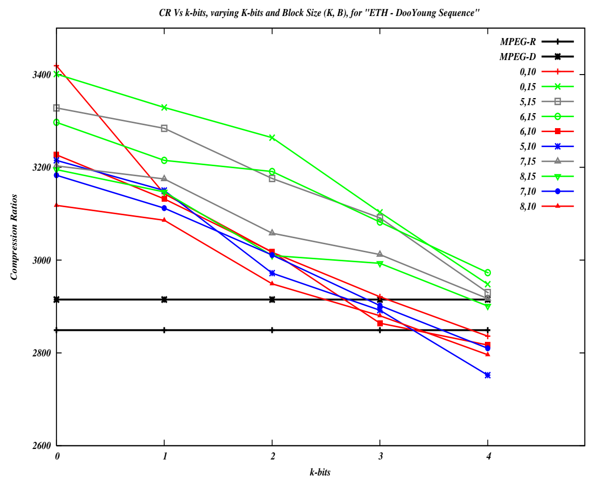


Figure 8. Compression Ratio and PSNR results for Ballet, Exercise and IndianDance Dataset with block size=25/50/100 and varying joint angle noise levels, K=0,5,9,12, plotted against k



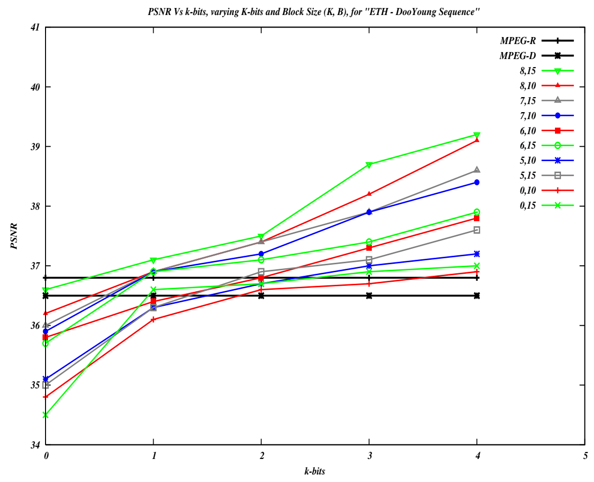


Figure 9. Compression ratio and PSNR for Doo-Young real dynamic depth movie dataset.

References

[1] A. Agarwal and B. Triggs. Recovering 3d human pose from monocular images. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(1):44–58, 2006.

[2] J. Carranza, C. Theobalt, M. A. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. *ACM Trans. Graph.*, 22(3):569–577, 2003.

[3] D. Gavrila and L. Davis. Tracking of humans in action: A 3d model-based approach, 1996.

[4] M. H. Gross, S. Würmlin, M. Näf, E. Lamboray, C. P. Spagno, A. M. Kunz, E. Koller-Meier, T. Svoboda, L. J. V. Gool, S. Lang, K. Strehlke, A. V. Moere, and O. G. Staadt. blue-c: a spatially immersive display and 3d video portal for telepresence. *ACM Trans. Graph.*, 22(3):819–827, 2003.

[5] T. Kanade and P. J. Narayanan. Virtualized reality: Perspectives on 4d digitization of dynamic events. *IEEE Computer Graphics and Applications*, pages 32 – 40, May 2007.

[6] R. Krishnamurthy, B. Chai, H. Tao, and S. Sethuraman. Compression and transmission of depth maps for image-based rendering. In *ICIP01*, pages III: 828–831, 2001.

[7] S. Kum and K. MayerPatel. Intra-stream encoding for multiple depth streams. *NOSSDAV*, pages 62–67, 2006.

[8] L. Levkovich-Maslyuk, A. Ignatenko, A. Zhirkov, A. Konushin, I. K. Park, M. Han, and Y. Bayakovski. Depth image-based representation and compression for static and animated 3-d objects. *IEEE Trans. Circuits Syst. Video Techn.*, 14(7):1032–1045, 2004.

[9] M. A. Magnor, P. Ramanathan, and B. Girod. Multi-view coding for image-based rendering using 3-d scene geometry. *IEEE Trans. Circuits Syst. Video Techn.*, 13(11):1092–1106, 2003.

[10] W. Matusik and H. Pfister. 3d tv: a scalable system for real-time acquisition, transmission, and autostereoscopic display of dynamic scenes. *ACM Transactions on Graphics*, 23(3):814–824, Aug. 2004.

[11] P. J. Narayanan, P. Rander, and T. Kanade. Constructing virtual worlds using dense stereo. In *ICCV*, pages 3–10, 1998.

[12] S. K. Penta and P. J. Narayanan. Compression of multiple depth maps for ibr. *The Visual Computer*, 21(8-10):611–618, 2005.

[13] C. Theobalt, E. de Aguiar, M. A. Magnor, H. Theisel, and H.-P. Seidel. Marker-free kinematic skeleton estimation from sequences of volume data. In *VRST*, pages 57–64, 2004.

[14] Q. Wu, K. T. Ng, S.-C. Chan, and H.-Y. Shum. On object-based compression for a class of dynamic image-based representations. In *ICIP (3)*, pages 405–408, 2005.

[15] S. Würmlin, E. Lamboray, O. G. Staadt, and M. H. Gross. 3d video recorder: a system for recording and playing free-viewpoint video. *Comput. Graph. Forum*, 22(2):181–194, 2003.

[16] Z. Yang, K. Nahrstedt, Y. Cui, B. Yu, J. Liang, S.-H. Jung, and R. Bajcsy. Teeve: The next generation architecture for tele-immersive environment. In *ISM*, pages 112–119, 2005.

[17] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. A. J. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23(3):600–608, 2004.