Model-based Multi-view Video Compression Using Distributed Source Coding Principles

Jayanth Nayak, Bi Song, Ertem Tuncel, Amit K. Roy-Chowdhury

1 Introduction

Transmission of video data from multiple sensors over a wireless network requires enormous amount of bandwidth, and could easily overwhelm the system. However, by exploiting the redundancy *between* the video data collected by different cameras, in addition to the inherent temporal and spatial redundancy *within* each video sequence, the required bandwidth can be significantly reduced. Well-established video compression standards, such as MPEG-1, MPEG-2, MPEG-4, H.261, H.263, and H.264, all rely on efficient transform coding of motion-compensated frames, using the discrete cosine transform (DCT) or computationally efficient approximations to it. However, they can only be used in a protocol that encodes the data of each sensor independently. Such methods would exploit spatial and temporal redundancy within each video sequence, but would completely ignore the redundancy between the sequences.

In this chapter, we develop novel multi-terminal, model-based video coding algorithms combining DSC and computer vision techniques. In broad terms, our schemes rely on model-based tracking of individual video sequences captured by cameras (which could be located arbitrarily in space), leading to removal of spatial, temporal, and *inter-camera* redundancies. The presence of the 3D model provides correspondence between overlapping feature points in the different views, provided that the tracking of the individual sequences is accurate. The performance of our algorithm depends, most crucially, on the quality of tracking and the coding efficiency of the distributed quantization scheme. The tracking must result in correspondences between pixels that are maximally correlated, and the distributed coding must optimally exploit this correlation.

Although distributed source coding (DSC) has been introduced more than three decades ago by Slepian and Wolf (1973), and underlying ideas as to how it should be practically implemented had been outlined by Wyner (1974), the papers (Zamir and Shamai 1998) and (Pradhan and Ramchandran 1999) arguably demonstrated the feasibility of implementing distributed source codes for the first time. Following the publication of these papers, considerable effort has been devoted to adapting DSC ideas to application scenarios such as distributed image and video coding (e.g., (Wagner et al. 2003, Zhu et al. 2003, Girod et al. 2005, Gehrig and Dragotti 2005, Puri and Ramchandran 2007, Yang et al. 2007)), but the results have been mixed. The reason is that while there are many scenarios where distributed processing appears to be a natural choice, the conditions necessary for known DSC schemes to perform well are rarely satisfied and the more mature non-DSC techniques either easily outperform the DSC schemes or the gains of DSC schemes are relatively small. We make similar observations in this work. More specifically, the gains we achieve over separate coding are diminished because it seems that the temporal redundancies are much larger than the inter-camera ones. The exception is at very low rates where even small gains become significant. We therefore focus particularly on very low bit rates in this chapter.

Although early work on distributed video coding focused on the single view case, there has been some attention to applying distributed coding for multi view video as well (Artigas et al. 2006, Guo et al. 2006, Ouaret et al, 2006, Song, Bursalioglu, Roy-Chowdhury and Tuncel 2006). Most of this work has been on block based coding and, as in our work, a key issue is construction of side information to optimally exploit intra sequence memory and inter sequence correlation.

The rest of the chapter is organized as follows. Section 2 outlines the model-based tracking algorithm. Section 3 presents an overview of our approach to distributed video coding. In Section 4, some experimental results are presented. Finally, Section 5 gives the conclusion and discusses some avenues for future research.

2 Model Tracking

Numerous methods exist for estimating motion and shape of an object from video sequences. Many of them can handle significant changes in the illumination conditions by *compensating* for the variations (Hager and Belhumeur 1998, Freedman and Turek 2005, Jin et al. 2001). However, there do not exist many methods that can *recover* the 3D motion *and* time-varying global illumination conditions from video sequences of moving objects. We achieve this goal by building upon a recently proposed framework for combining the effects of motion, illumination, 3D shape, and camera parameters in a sequence of images obtained by a perspective camera (Xu and Roy-Chowdhury 2007, Xu and Roy-Chowdhury 2008). This theory allows us to develop a simple method for estimating the rigid 3D motion, as presented in (Xu and Roy-Chowdhury 2007). However, this algorithm involves the computation of a bilinear basis (see Section 2.1) in each iteration, which is a huge computational burden. In this chapter, we show that it is possible to efficiently and accurately reconstruct the 3D motion and global lighting parameters of a rigid and non-rigid object from a video sequence within the framework of the inverse compositional (IC) algorithm (Baker and Matthews 2004). Details of this approach are available in Xu and Roy-Chowdhury (2008).

A well-known approach for 2D motion estimation and registration in monocular sequences is Lucas-Kanade tracking, which attempts to match a target image with a reference image or template. Building upon this framework, a very efficient tracking algorithm was proposed in (Hager and Belhumeur 1998) by inverting the role of the target image and the template. However, their algorithm can only be applied to restricted class of warps between the target and template (for details, see Baker and Matthews 2004). A forward compositional algorithm was proposed in (Shum and Szeliski 2000) by estimating an incremental warp for image alignment. Baker and Matthews (2004) proposed an IC algorithm for efficient implementation of the Lucas-Kanade algorithm to save computational cost in re-evaluation of the derivatives in each iteration. The IC algorithm was then used for efficiently fitting active appearance models (Matthews and Baker 2004) and the well-known 3D morphable model (3DMM) (Romdhani and Vetter 2003) to face images under large pose variations. However, none of these schemes estimate the lighting conditions in the images. A version of 3DMM fitting (Blanz and Vetter 2003) used a Phong illumination model, estimation of whose parameters in the presence of extended light sources can be difficult. Also, in contrast to our work, Blanz and Vetter (2003) did not use an IC approach for motion and lighting estimation.

Our lighting estimation can account for extended lighting sources and attached shadows. Further, our goal is to estimate 3D motion, unlike in (Hager and Belhumeur 1998, Freedman and Turek 2005, Shum and Szeliski 2000), which perform 2D motion estimation. The warping function in this work is different from (Baker and Matthews 2004, Romdhani and Vetter 2003) as we explain in Section 2.2. Since our IC approach estimates 3D motion, it allows us to perform the expensive computations only once every few frames (unlike once for every frame as in the image alignment approaches of (Baker and Matthews 2004)). Specifically, these computations are done only when there is a significant change of pose.

2.1 Image Appearance Model of a Rigid Object

Our goal is to describe the appearance of an image in terms of the 3D rigid motion and the overall lighting in the scene. For this purpose, we derive a generative image appearance model that is a function of these parameters. Given a sequence of images, we can estimate the parameters which lead to the best fit with this model. Details of the generative model, as well as the estimation framework, are available in (Xu and Roy-Chowdhury 2007) and (Xu and Roy-Chowdhury 2008). A brief overview is provided here.

In (Xu and Roy-Chowdhury 2007), it was proved that if the motion of the object (defined as the translation of the object centroid $\Delta\mathbf{T} \in \mathbb{R}^{3\times 1}$ and the rotation vector $\Delta\boldsymbol{\Omega} \in \mathbb{R}^{3\times 1}$ about the centroid in the camera frame) from time t_1 to new time instance $t_2 = t_1 + \delta t$ is small, then up to a first order approximation, the reflectance image $I(x, y)$ at t_2 can be expressed as

$$I_{t_2}(\mathbf{v}) = \sum_{i=1}^{9} l_i^{t_2} b_i^{t_2}(\mathbf{v}) \tag{1}$$

where

$$b_i^{t_2}(\mathbf{v}) = b_i^{t_1}(\mathbf{v}) + \mathbf{A}(\mathbf{v},\mathbf{n})\Delta\mathbf{T} + \mathbf{B}(\mathbf{v},\mathbf{n})\Delta\boldsymbol{\Omega}.$$

In the above equations, $\mathbf{v}$ represents the image point projected from the 3D surface with surface normal $\mathbf{n}$, and $b_i^{t_1}(\mathbf{v})$ are the original basis images before motion (precise format of $b_i^{t_1}(\mathbf{v})$ is defined in (Xu and Roy-Chowdhury 2007)). $\mathbf{l}_t = \begin{bmatrix} l_1^t & \dots & l_{N_l}^t \end{bmatrix}^T$ is the vector of illumination parameters. $\mathbf{A}$ and $\mathbf{B}$ contain the structure and camera intrinsic parameters, and are functions of $\mathbf{v}$ and the 3D surface normal $\mathbf{n}$. For each pixel $\mathbf{v}$, both $\mathbf{A}$ and $\mathbf{B}$ are $N_l \times 3$ matrices, where $N_l \approx 9$ for Lambertian objects with attached shadows. A derivation of (1) and explicit expressions for $\mathbf{A}$ and $\mathbf{B}$ are presented in (Xu and Roy-Chowdhury 2007). For the purposes of this chapter, we only need to know the form of the equations.

The left side of equation (1) is the image at time t_2, which is expressed in terms of the basis images and the lighting coefficients on the right hand side. The basis images, in turn, depend upon the motion of the object between two consecutive time instants. Thus, equation (1) expresses the image appearance in terms of the object's 3D pose and the scene lighting.

We can express the result in (1) succinctly using tensor notation as

$$\mathcal{I}_{t_2} = \left(\mathcal{B}_{t_1} + \mathcal{C}_{t_1} \times_2 \begin{bmatrix} \Delta\mathbf{T} \\ \Delta\mathbf{\Omega} \end{bmatrix}\right) \times_1 \mathbf{l}_{t_2}, \tag{2}$$

where $\times_n$ is called the *mode-n product* (Lathauwer et al. 2000) and $\mathbf{l} \in \mathbb{R}^{N_l}$ is the N_l-dimensional vector of l_i components. More specifically, the mode-n product of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_n \times \ldots \times I_N}$ by a vector $\mathbf{V} \in \mathbb{R}^{I_n \times 1}$, denoted by $\mathcal{A} \times_n \mathbf{V}$, is the $I_1 \times I_2 \times \ldots \times 1 \times \ldots \times I_N$ tensor

$$(\mathcal{A} \times_n \mathbf{V})_{i_1 \ldots i_{n-1} 1 i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_{n-1} i_n i_{n+1} \ldots i_N} v_{i_n}.$$

Thus, the image at t_2 can be represented using the parameters computed at t_1. For each pixel (p, q) in the image, $\mathcal{C}_{pq} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \end{bmatrix}$ of size $N_l \times 6$. Thus for an image of size $M \times N$, $\mathcal{C}$ is $N_l \times 6 \times M \times N$, $\mathcal{B}_{t_1}$ is a sub-tensor of dimension $N_l \times 1 \times M \times N$, comprising the basis images $b_i^{t_1}(\mathbf{u})$, and $\mathcal{I}_{t_2}$ is a sub-tensor of dimension $1 \times 1 \times M \times N$, representing the image.

2.2 Inverse Compositional Estimation of 3D Motion and Illumination

We now present the estimation algorithm for the bilinear model in (2). The detailed proof of convergence and extension to video for the rigid motion case can be found in (Xu and Roy-Chowdhury 2008).

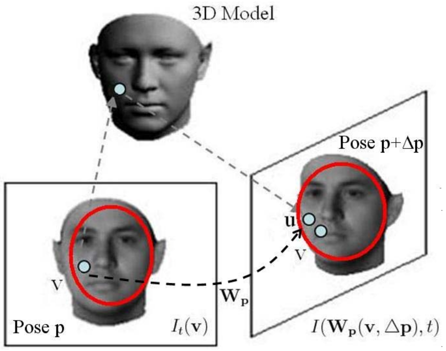


Figure 1: Illustration of the warping function $\mathbf{W}$. A point $\mathbf{v}$ in image plane is projected onto the surface of the 3D object model. After the pose transformation with $\triangle\mathbf{p}$, the point on the surface is back projected onto the image plane at a new point $\mathbf{u}$. The warping function maps from $\mathbf{v} \in \mathbb{R}^{\mathbf{2}}$ to $\mathbf{u} \in \mathbb{R}^{\mathbf{2}}$. The red ellipses show the common part in both frames that the warping function $\mathbf{W}$ is defined upon.

We begin by estimating the 3D motion assuming that illumination is constant across two consecutive frames. We will then estimate variations in illumination. Let $\mathbf{p} \in \mathbb{R}^{6 \times 1}$ denote the pose of the object. Then the image synthesis process can be considered as a rendering function of the object at pose $\mathbf{p}$ in the camera frame to the pixel coordinates $\mathbf{v}$ in the image plane as $f(\mathbf{v}, \mathbf{p})$. Using the bilinear model described above, it can be implemented with (2). Given an input image $I(\mathbf{v})$, we want to align the synthesized image with it so as to obtain

$$\hat{\mathbf{p}} = \arg\min_{\mathbf{p}} \sum_{\mathbf{v}} \left(f(\mathbf{v}, \mathbf{p}) - I(\mathbf{v})\right)^2, \tag{3}$$

where $\hat{\mathbf{p}}$ denotes the estimated pose for this input image $I(\mathbf{v})$. This is the cost function of Lucas-Kanade tracking in (Baker and Matthews 2004) modified for 3D motion estimation.

We will consider the problem of estimating the pose change, $\mathbf{m}_t \triangleq \triangle\mathbf{p}_t$, between two consecutive frames, $I_t(\mathbf{v})$ and $I_{t-1}(\mathbf{v})$. Let us introduce a warp operator $\mathbf{W} : \mathbb{R}^\mathbf{2} \rightarrow \mathbb{R}^\mathbf{2}$ such that, if we denote the pose of $I_t(\mathbf{v})$ as $\mathbf{p}$, the pose of $I_t(\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p}))$ is $\mathbf{p} + \triangle\mathbf{p}$. Specifically, a 2D point on the image plane is projected onto the 3D object surface. Then we transform the pose of the object surface by $\triangle\mathbf{p}$ and back project the point from the 3D surface onto the image plane. Thus, $\mathbf{W}_\mathbf{p}$ represents the displacement in the image plane due to a pose transformation of the 3D model. Note that this warping involves a 3D pose transformation (unlike (Baker and Matthews 2004)). In (Romdhani and Vetter 2003), the warping was from a point on the 3D surface to the image plane, and was used for fitting a 3D model to an image. Our new warping function can be used for the IC estimation of 3D rigid motion and illumination in video sequence, which (Baker and Matthews 2004) and (Romdhani and Vetter 2003) do not address. A key property of $\{\mathbf{W}_\mathbf{p}\}$ is that these warps form a group with respect to function composition (See (Xu and Roy-Chowdhury 2008) for a detailed proof of this and other properties of the set of warps), which is necessary for applying the IC algorithm. We shall here only show how the inverse of a warp is another warp. The inverse of the warp $\mathbf{W}$ is defined to be the $\mathbb{R}^2 \rightarrow \mathbb{R}^2$ mapping such that if we denote the pose of $I_t(\mathbf{v})$ as $\mathbf{p}$, the pose of $I_t(\mathbf{W}_\mathbf{p}(\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p}), \triangle\mathbf{p})^{-1})$ is $\mathbf{p}$ itself. As the warp $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})$ transforms the pose from $\mathbf{p}$ to $\mathbf{p} + \triangle\mathbf{p}$, the inverse $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})^{-1}$ should transform the pose from $\mathbf{p} + \triangle\mathbf{p}$ to $\mathbf{p}$, i.e. $\mathbf{W}_\mathbf{p}(\mathbf{v}, \triangle\mathbf{p})^{-1} = \mathbf{W}_{\mathbf{p}+\triangle\mathbf{p}}(\mathbf{v}, -\triangle\mathbf{p})$.

Using this warp operator, for any frame $I_t(\mathbf{v})$, the cost function can be written as

$$\hat{\mathbf{m}}_t = \arg\min_{\mathbf{m}} \sum_{\mathbf{v}} \left(f(\mathbf{v}, \hat{\mathbf{p}}_{t-1}) - I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v}, -\mathbf{m}))\right)^2. \tag{4}$$

Rewriting the cost function (4) in the IC framework (Baker and Matthews 2004), we consider minimizing

$$\arg\min_{\triangle\mathbf{m}} \sum_{\mathbf{v}} \left(f(\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{v}, \triangle\mathbf{m}), \hat{\mathbf{p}}_{t-1}) - I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v}, -\mathbf{m}))\right)^2 \tag{5}$$

with the update rule

$$\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m}) \leftarrow \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m}) \circ \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},\triangle\mathbf{m})^{-1}. \tag{6}$$

The compositional operator $\circ$ in (6) means the second warp is composed into the first warp, i.e., $\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m}) \equiv \mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},\triangle\mathbf{m})^{-1},-\mathbf{m})$. According to the definition of the warp $\mathbf{W}$, we can replace $f(\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{v},\triangle\mathbf{m}),\hat{\mathbf{p}}_{t-1})$ in (5) with $f(\mathbf{v},\hat{\mathbf{p}}_{t-1}+\triangle\mathbf{m})$. This is because $f(\mathbf{v},\hat{\mathbf{p}}_{t-1}+\triangle\mathbf{m})$ is the image synthesized at $\hat{\mathbf{p}}_{t-1}+\triangle\mathbf{m}$, while $f(\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{v},\triangle\mathbf{m}),\hat{\mathbf{p}}_{t-1})$ is the image synthesized at $\hat{\mathbf{p}}_{t-1}$ followed with the warp of the pose increments $\triangle\mathbf{m}$. Applying the first order Taylor expansion on it, we have

$$\sum_{\mathbf{v}}\left(f(\mathbf{v},\hat{\mathbf{p}}_{t-1}) + \frac{\partial f(\mathbf{v},\mathbf{p})}{\partial \mathbf{p}}|_{\mathbf{p}=\hat{\mathbf{p}}_{t-1}}\triangle\mathbf{m} - I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m}))\right)^2.$$

Taking the derivative of the above expression with respect to $\triangle\mathbf{m}$ and setting it to be zero, we have

$$\sum_{\mathbf{v}}\left(f(\mathbf{v},\hat{\mathbf{p}}_{t-1}) + \mathcal{G}^{\mathbf{T}}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}}\triangle\mathbf{m} - I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m}))\right)\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}} = 0,$$

where $\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{t-1}}$ is the derivative $\frac{\partial f(\mathbf{v},\mathbf{p})}{\partial \mathbf{p}}|_{\mathbf{p}=\hat{\mathbf{p}}_{t-1}}$. Solving for $\triangle\mathbf{m}$, we get:

$$\triangle\mathbf{m} = \mathbf{H_{IC}}\sum_{\mathbf{v}}\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}}\left(I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v},-\mathbf{m})) - f(\mathbf{v},\hat{\mathbf{p}}_{t-1})\right) \tag{7}$$

where

$$\mathbf{H_{IC}} = \left[\sum_{\mathbf{v}}\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}}\mathcal{G}^{\mathbf{T}}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}}\right]^{-1}.$$

Note that the derivative $\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}}$ and Hessian $\mathbf{H_{IC}}$ in (7) do not depend upon the updating variable $\mathbf{m}$, which is moved into the warp operator $\mathbf{W}$. The computational complexity of $\mathbf{W}_{\hat{\mathbf{p}}_{\mathbf{t-1}}}(\mathbf{v},-\mathbf{m})$ will be significantly lower than that of recomputing $\mathcal{G}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}+\mathbf{m}}$ and the corresponding Hessian $\mathbf{H}$ in every iteration.

Reintroducing the illumination variation, the lighting parameter $\mathbf{l}$ can be estimated using

$$\hat{\mathbf{l}} = (\mathcal{B}_{l(l)}\mathcal{B}^{\mathbf{T}}_{l(l)})^{-1}\mathcal{B}_{l(l)}\mathcal{I}^{\mathbf{T}}_{(l)},$$

where the subscripts $_{(l)}$ indicates the unfolding operation (Lathauwer et al. 2000) along the illumination dimension. That is, assuming an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times\ldots\times I_N}$, the matrix unfolding $\mathbf{A}_{(n)} \in \mathbb{R}^{I_n\times(I_{n+1}I_{n+2}\ldots I_N I_1 I_2\ldots I_{n-1})}$ contains the element $a_{i_1 i_2\ldots i_N}$ at the position with row number i_n and column number equal to $(i_{n+1}-1)I_{n+2}I_{n+3}\ldots I_N I_1 I_2\ldots I_{n-1} + (i_{n+2}-1)I_{n+3}I_{n+4}\ldots I_N I_1 I_2\ldots I_{n-1} + \cdots + (i_N - 1)I_1 I_2\ldots I_{n-1} + (i_1-1)I_2 I_3\ldots I_{n-1} + \cdots + i_{n-1}$.

Following the same derivation as (7), we have

$$\triangle \mathbf{m} = \mathbf{H_{IC}} \sum_{\mathbf{v}} (\mathcal{C}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}} \times_1 \hat{\mathbf{l}}) \left(I_t(\mathbf{W}_{\hat{\mathbf{p}}_{t-1}}(\mathbf{v}, -\mathbf{m})) - \mathcal{B}_{\mathbf{v}|\hat{\mathbf{p}}_{t-1}} \times_1 \hat{\mathbf{l}} \right) \tag{8}$$

where

$$\mathbf{H_{IC}} = \left[\sum_{\mathbf{v}} (\mathcal{C}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}} \times_1 \hat{\mathbf{l}})(\mathcal{C}_{\mathbf{v}|\hat{\mathbf{p}}_{\mathbf{t-1}}} \times_1 \hat{\mathbf{l}})^{\mathbf{T}} \right]^{-1}.$$

3 Distributed Compression Schemes

The distributed compression schemes that we shall discuss in this section can all be represented by the block diagram of Figure 2. The choices we make for the following blocks result in the various schemes as described later in this section.

- Feature extraction
- Mode decision
- Side information extraction

3.1 Feature Extraction and Coding

In order to extract the image features, we need to detect them in the first frame and track them in subsequent frames. Since we estimate 3D pose, a 3D mesh model is registered to the first frame and then tracked using the method in Section 2. The mesh model consists of a set of mesh points $\mathcal{V} = \{V_i = (x_i, y_i, z_i), i = 1, \dots, N\}$ and a set of triangles $\mathcal{T} = \{T_{abc} = (V_a, V_b, V_b)\} \subset \mathcal{V}^3$ formed by mesh points. Figure 3 depicts the 2D projection of a triangular mesh model on a pair of frames. The model pose parameters, namely the translation and rotation parameters, are computed independently at each view and transmitted to the decoder at high fidelity. Each encoder also receives the initial pose parameters of the other encoder.

The visibility of a mesh point V in both views is computed at each encoder by computing its position relative to each triangle $T = (V_1, V_2, V_3)$ of which it is not a member. Given the estimated pose of a certain view, let (x, y, z) and (x_i, y_i, z_i) be the coordinates of V and V_i respectively. The triangle, which is the convex hull of the three points occludes V if the segment connecting the origin to V passes through the

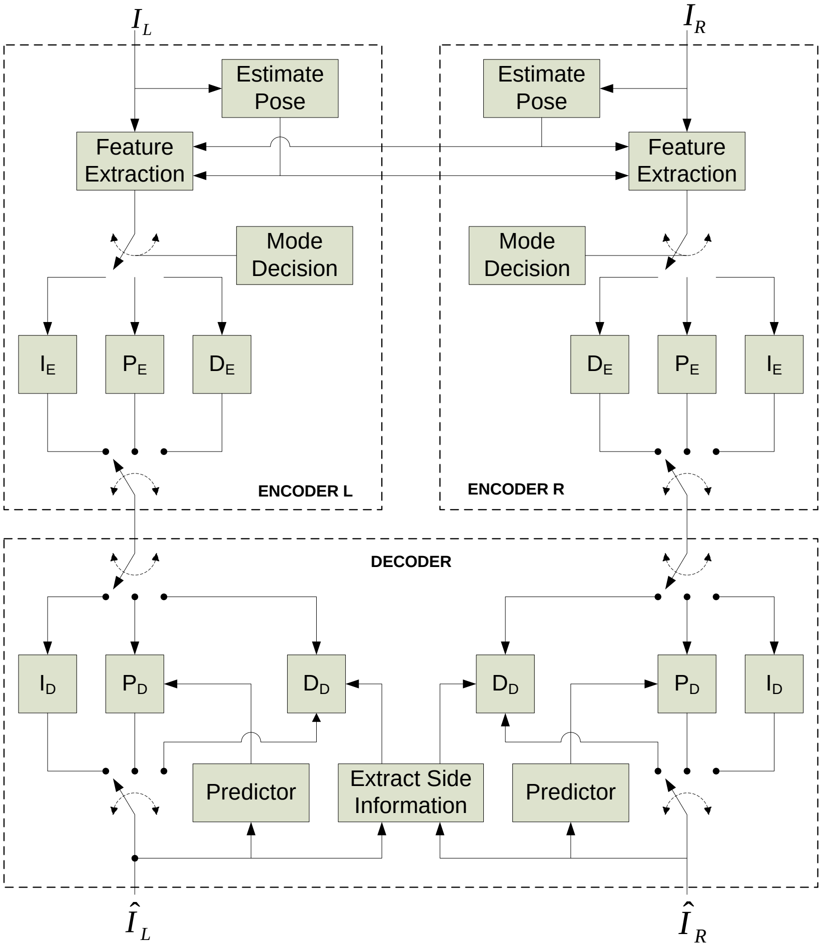


Figure 2: Block diagram of coding schemes

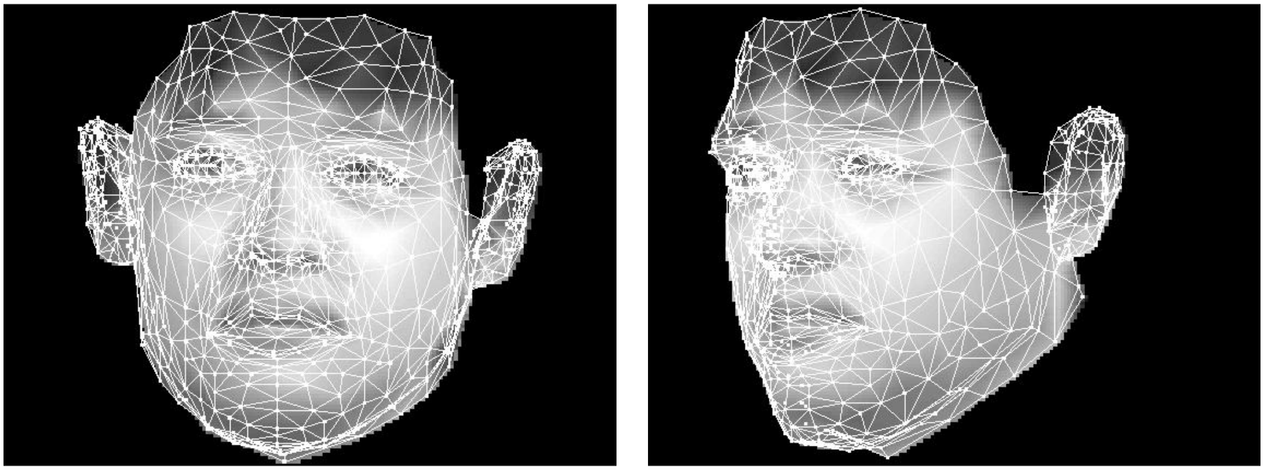


Figure 3: Mesh points $\mathcal{V}$ and triangles $\mathcal{T}$ are overlaid on the two views.

triangle. To test occlusion, we project the point onto the basis formed by the vertices of T:

$$\begin{bmatrix} c_x \\ c_y \\ c_z \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ z_1 & z_2 & z_3 \end{bmatrix}^{-1} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{9}$$

The triangle occludes the point if and only if c_x, c_y and c_z are all positive and $c_x + c_y + c_z > 1$. Observe that if all coefficients are positive and sum to 1, the point lies in the convex hull of the three points forming T i.e., it lies on the triangle. If the mesh point is visible with respect to every triangle, we declare the point visible.

We consider three feature extraction techniques.

- Point Sampled Intensity (PSI): The feature set for an image is the set of intensity values at the visible mesh points projected onto the image plane.
- Triangles with Constant Intensity (TCI): For all triangles containing at least one visible vertex, this feature is the average of the intensity values over the visible pixels in the triangle.
- Triangles with Linearly varying Intensity (TLI): For triangles with all vertices visible, we compute a linear least squares fit to the intensity profile. The intensity values at the vertices according to the estimated planar intensity profile form a part of the feature set. For those triangles where not all vertices are visible, we use the average intensity feature as in TCI.

For distributed encoding of the extracted features, we utilized a very simple binning technique whereby each feature is independently encoded using

$$\mathcal{C}(i) = i \bmod W$$

where $\mathcal{C}(i)$ is the codeword corresponding to the quantization index i of the coded feature and W is a parameter controlling the rate of the code. Even though the above binning scheme can be improved by more sophisticated DSC techniques (e.g., LDPC- or turbo-based codes) in general, the improvement would be marginal at very low bit rates, which we particularly focus on.

3.2 Types of frames

In the compression schemes that we consider, each frame of each view can belong to one of three classes:

- Intra frame or I-frame: Each feature is encoded and decoded independently of both past frames in the same view and all frames in the other view.
- Predictive frame or P-frame: If any feature is common to the current frame and the previous frame in the same view, we only encode the difference between the two features. Features which are unique to the current frame are coded as in an I-frame.
- Distributed frame or D-frame: The features for which side information is available at the decoder are distributively coded, while the rest of the features are coded as in an I-frame.

We considered two schemes for coding sequences:

- Scheme 1: We begin with an I-frame in both views. For the subsequent `DRefreshPeriod`-1 sampling instants, both views are coded as P-frames. At the following instant, the left view is transmitted as an I-frame and the other view is transmitted as a D-frame. The next `DRefreshPeriod`-1 are transmitted as P-frames. For the next instant, the right view is an I-frame while the left is a D-frame. This process then repeats. Every `IRefreshPeriod` instants, both views are coded as I-frames.
- Scheme 2: As in Scheme 1, we begin with I-frames in both views. For the subsequent `DRefreshPeriod`-1 sampling instants, the left view is coded as a P-frame while the other view is coded as a D-frame. At the next instant, the right view is coded as an I-frame and the other view is coded as a D-frame. This process is repeated in every block of `DRefreshPeriod` frame pairs with the roles of the right and left views reversing from block to block. Again, as in Scheme 1, every `IRefreshPeriod` instants, both views are coded as I-frames.

Table 1 shows some representative code sequences.

3.3 Types of side information

The side information in a D-frame can be one of three types:

- Other view (OD): The corresponding features between the frame being encoded and the one that is observed at the current instant at the other view form the side information. We can only encode features that are common to both views.

Table 1: Frame types of right and left views in the two schemes and separate coding. (`DRefreshPeriod`= 3 and `IRefreshPeriod`= 9)

Frame Number		1	2	3	4	5	6	7	8	9	10
Scheme 1	Left	I	P	P	I	P	P	D	P	P	I
	Right	I	P	P	D	P	P	I	P	P	I
Scheme 2	Left	I	P	P	D	D	D	I	P	P	I
	Right	I	D	D	I	P	P	D	D	D	I
Separate	Left	I	P	P	I	P	P	I	P	P	I
	Right	I	P	P	I	P	P	I	P	P	I

- Previous frame (PD): We use the corresponding features in the previous frame from the same view as side information. We can only encode features that are present in the previous and current views.

- Previous frame and Other view (POD): The side information is the optimal linear estimate of the source given the previous frame in the same view and the current frame in the other view. Only features that are common to the three frames involved can be distributively encoded by this method.

  The estimation coefficients are to be computed at the decoder before reconstructing a given view. So, if for example, frame R_n is distributively coded, we use the correlations from decoder reconstructions of R_{n-1}, R_{n-2} and L_{n-1} to obtain approximations to the optimal estimation coefficients.

4 Experimental Results

In this section, we evaluate the performance of the various schemes in compressing a sequence of a face viewed from two cameras under varying illumination. The face mesh model is assumed to be known to both encoders as well as the decoder. Transmission of the initial model pose parameters forms the sole inter-encoder communication.

For both schemes and all feature and side information types, we tested the performance on a 15-frame sequence from each view. Five consecutive original frames are shown in Figure 4(a). We fixed `DRefreshPeriod`= 5 and `IRefreshPeriod`= 15.

In Figures 4(b), 5(a), and 5(b), examples of reconstructed sequences are shown respectively for separate coding, Scheme 1, and Scheme 2, all using the PSI technique for feature extraction, and the latter two using

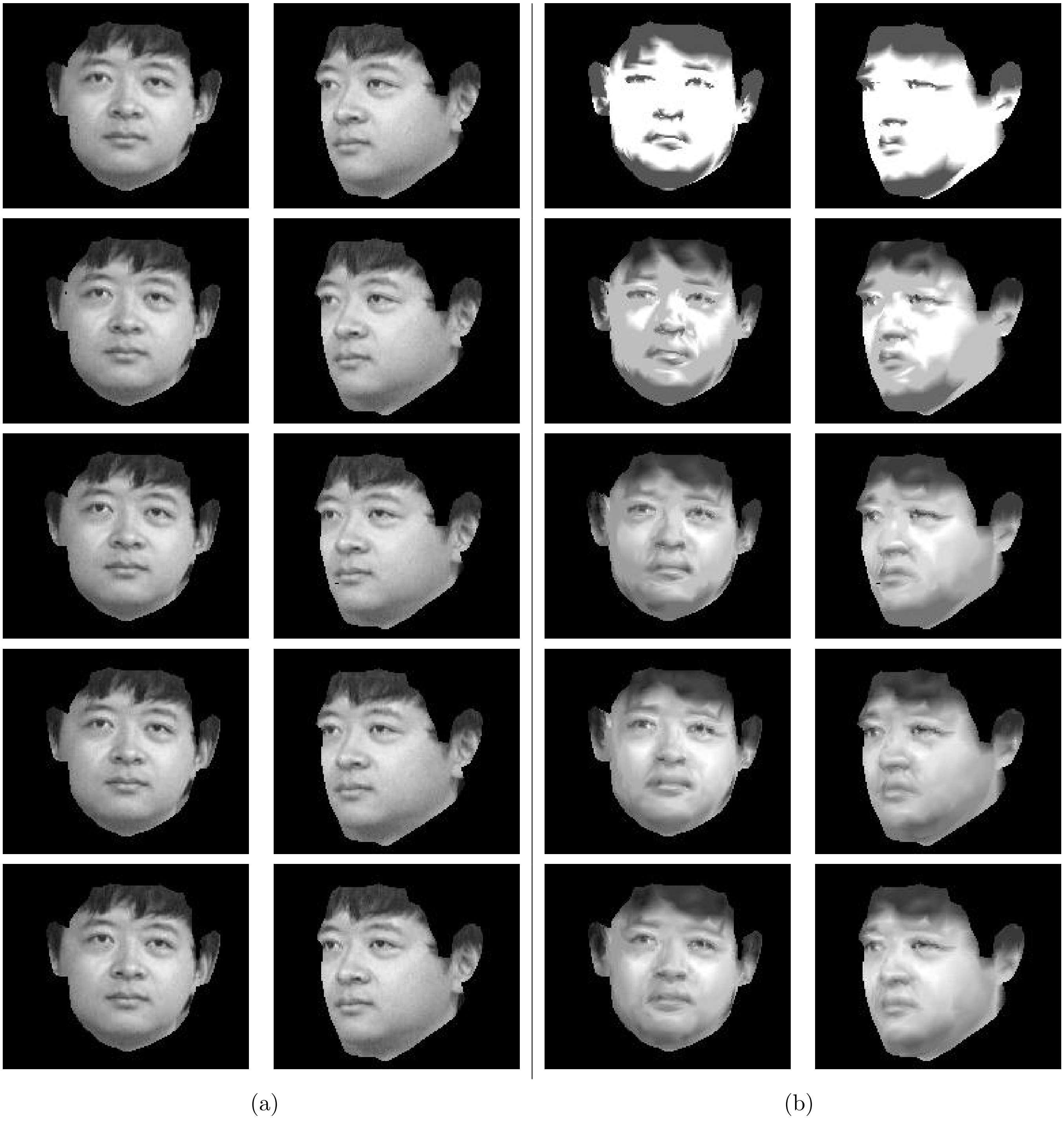


Figure 4: (a) Five consecutive frames from the original sequence. (b) Separate coding of the two views using the PSI technique and Scheme 1. The bit rate is ≈ 20kbps and the PSNR is ≈ 16.5dB.

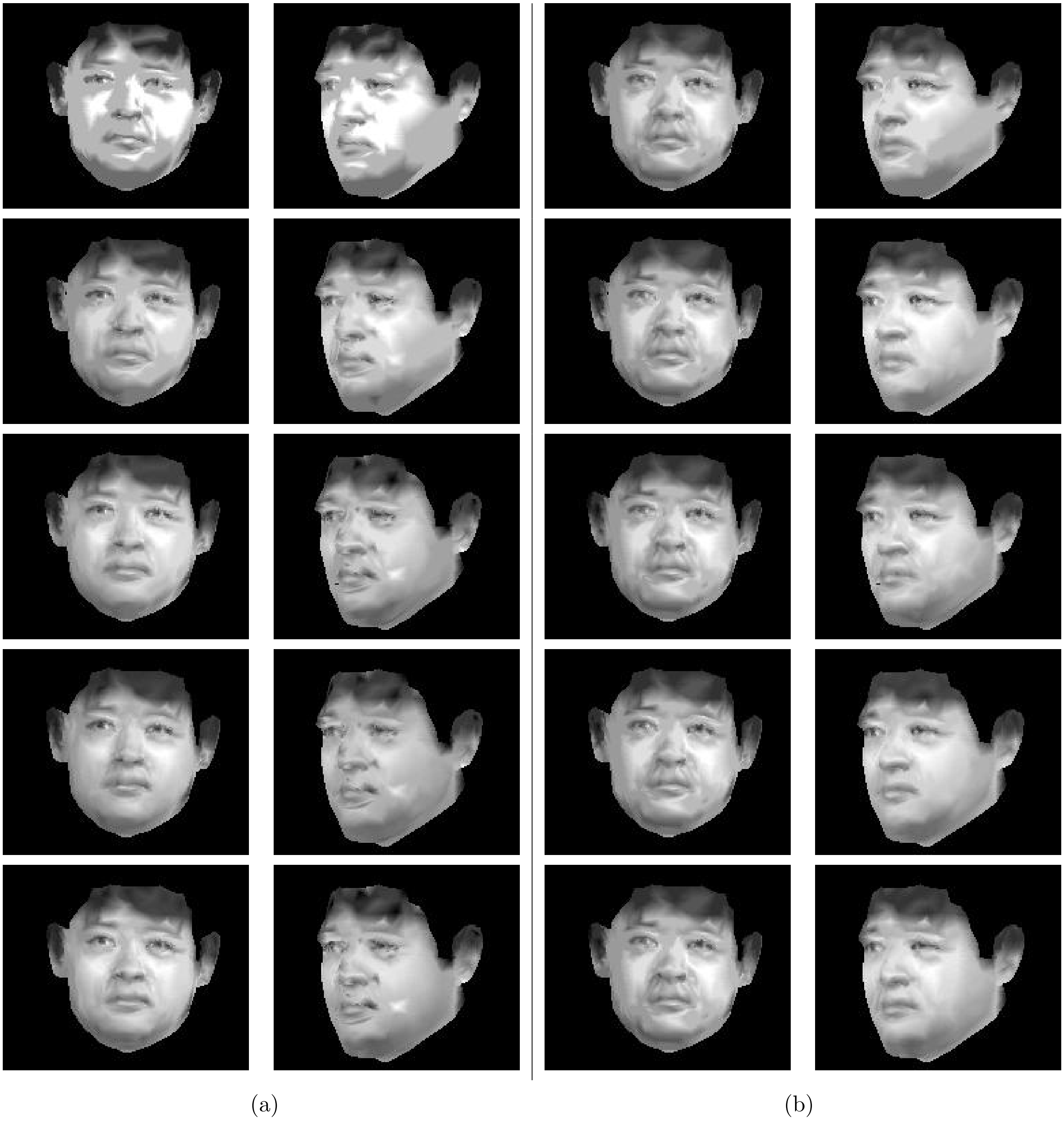


Figure 5: Distributed coding of the sequence using the PSI technique for feature extraction and POD-type side information. (a) Reconstruction using Scheme 1 with bit rate ≈ 22kbps and PSNR ≈ 21dB. (b) Reconstruction using Scheme 2 with bit rate ≈ 18kbps and PSNR ≈ 22.8dB.

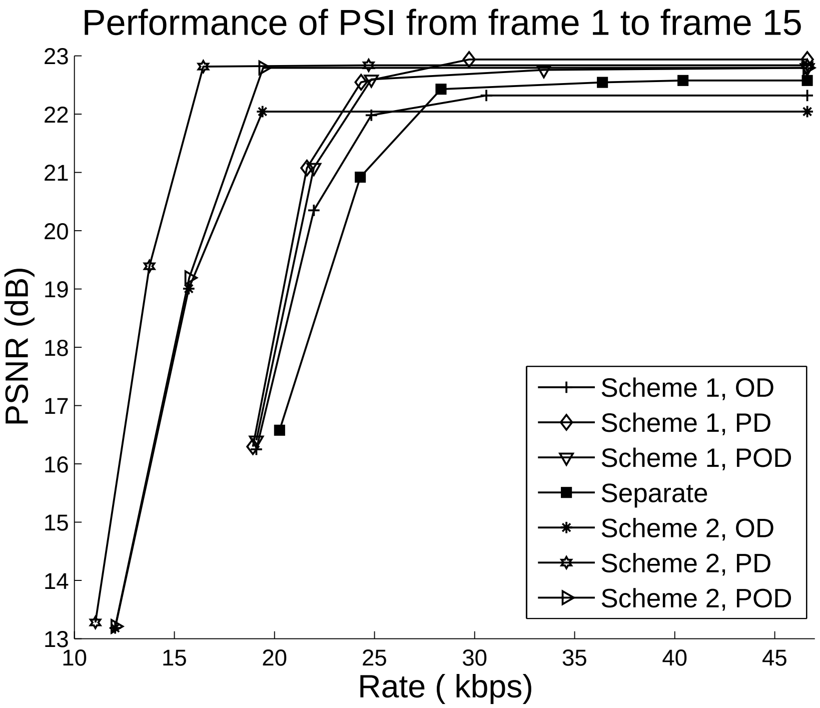
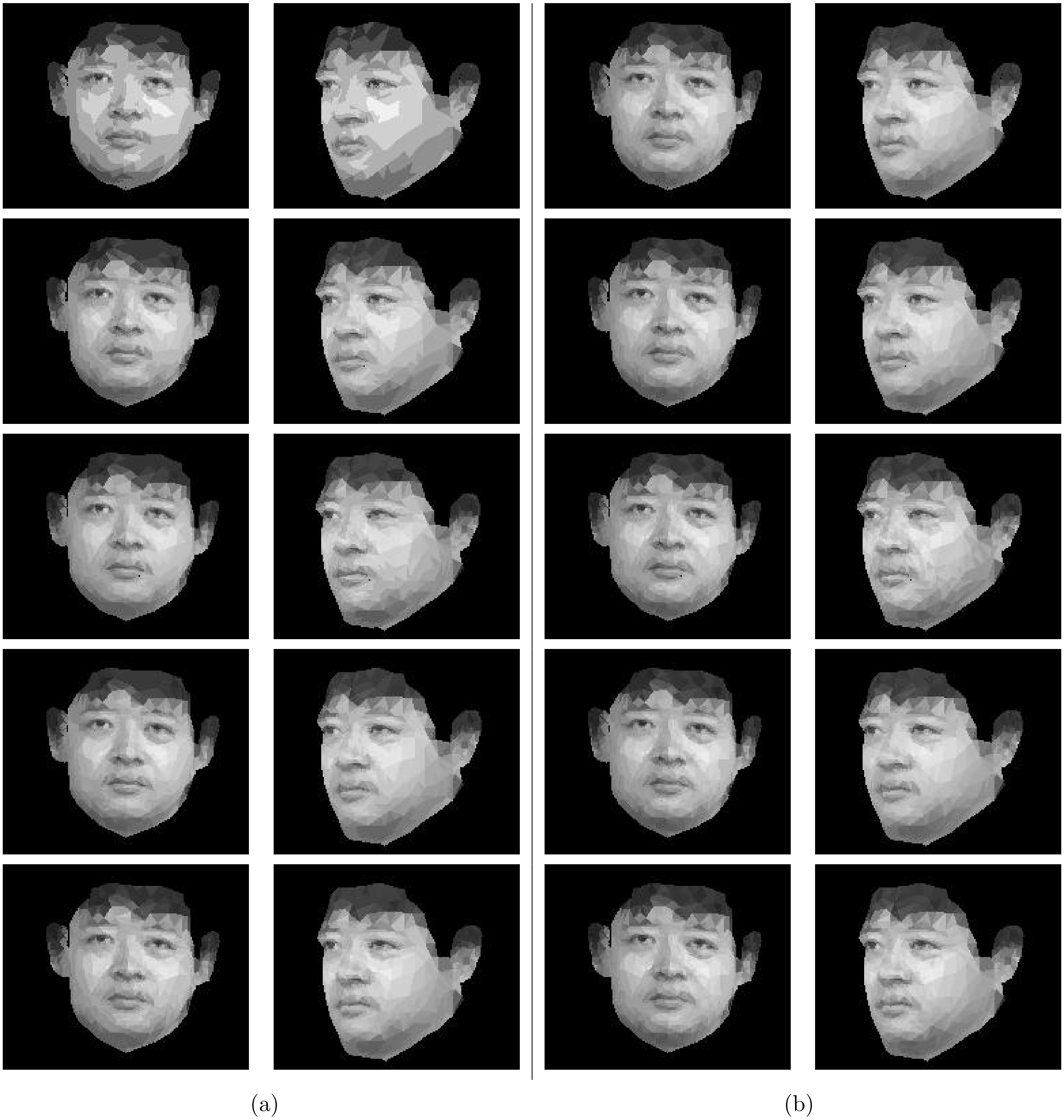


Figure 6: The rate-distortion tradeoff using the PSI feature extraction technique.

POD-type side information. The bit rates are fixed at around 20kbps. The complete rate-distortion tradeoff obtained by running all methods with various parameters is shown in Figure 6, where it is clearly seen that the gains are more significant at lower bit rates.

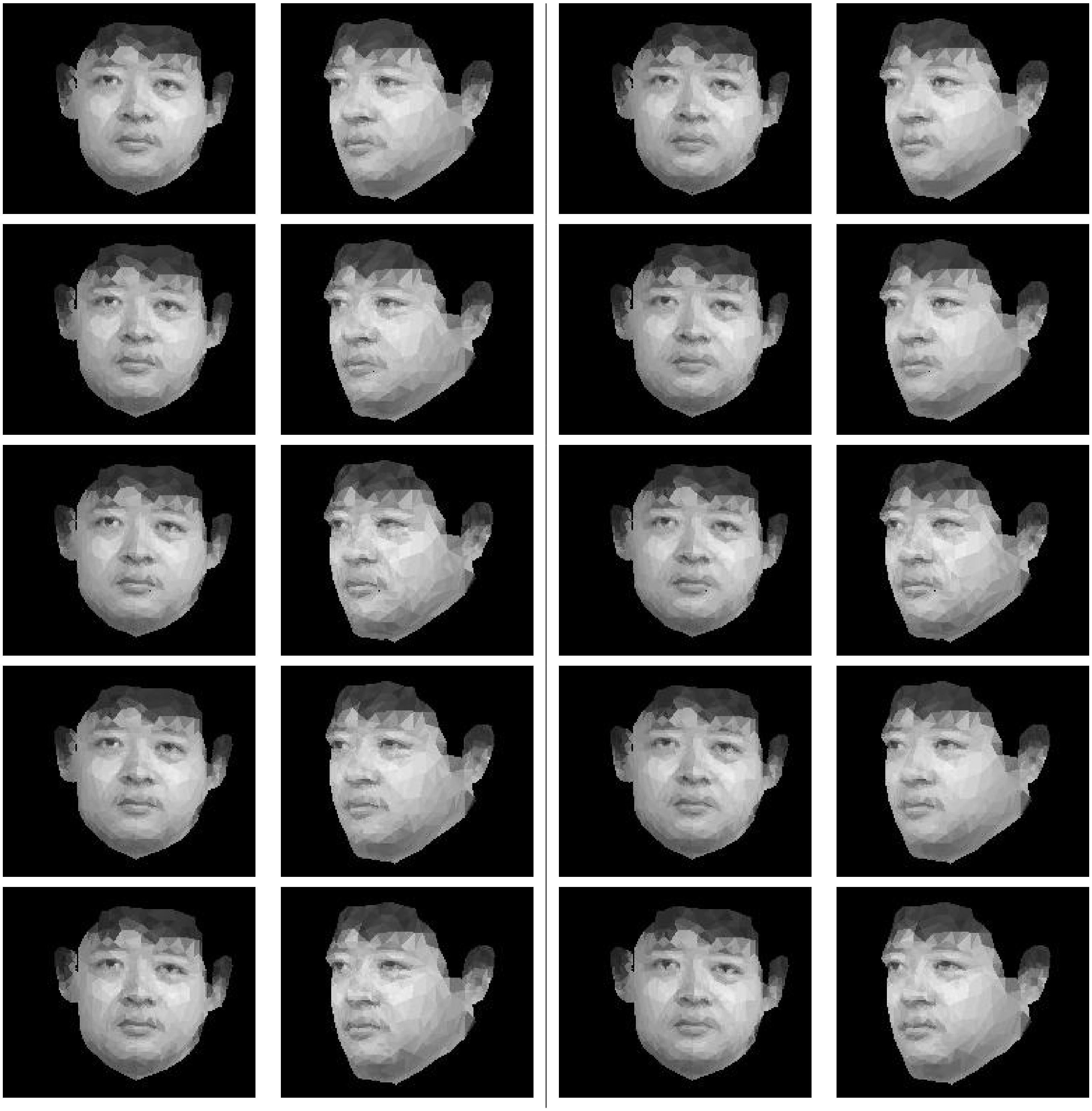
We then increase the target bit rate to around 41kbps by adopting the TCI technique for feature extraction. The reason for the rate increase even with the same quantization parameters is that the number of triangles is about twice as large as the number of mesh points. As apparently seen from the reconstructed sequences shown in Figures 7 and 8, this results in a quality increase. That is because the average of intensity values within a triangle is a much better representative of the triangle than a reconstruction based on only the three corner values. However, as can be seen from the complete rate-distortion tradeoff shown in Figure 9, the increase in the reconstruction quality comes at the expense of reduced distributed coding gains.

Using the TLI technique for feature extraction, we further increase the bit rate approximately by a factor of 3, since most triangles are now represented by three parameters. The reconstructed sequences are shown in Figures 10 and 11, and the complete rate-distortion tradeoff is depicted in Figure 12. The distributed coding gains are further diminished in this regime.



(a) (b)

Figure 7: Coding using the TCI technique for feature extraction. (a) Separate coding of the two views using Scheme 1 with bit rate ≈ 41kbps and PSNR ≈ 23.9dB. (b) Distributed coding using Scheme 2 and PD-type side information with the same bit rate PSNR ≈ 25.5dB.



(a) (b)

Figure 8: Coding using the TCI technique for feature extraction and POD-type side information. (a) Distributed coding using Scheme 1 with bit rate ≈ 40kbps and PSNR ≈ 24.4dB. (b) Distributed coding using Scheme 2 with bit rate ≈ 43kbps and the same PSNR.

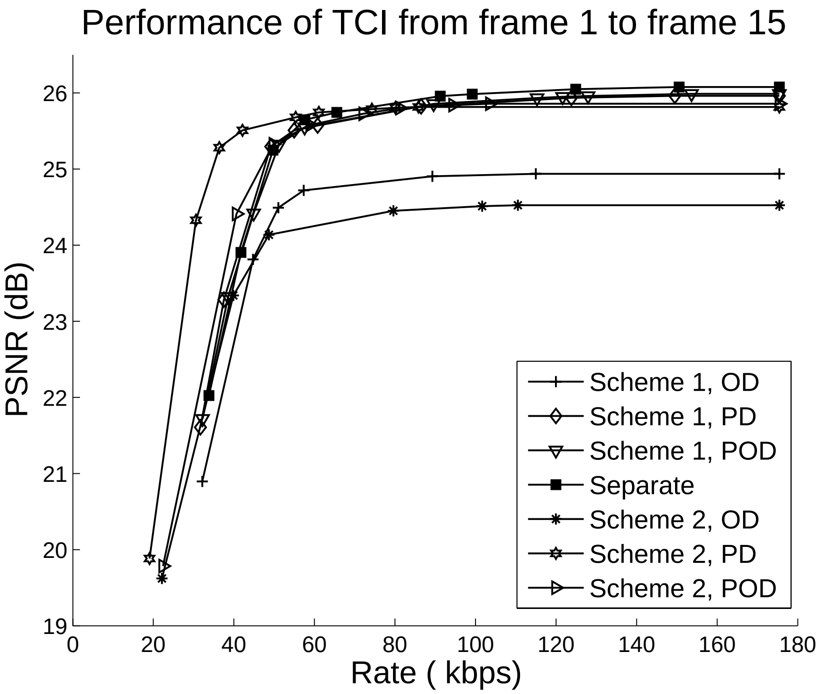
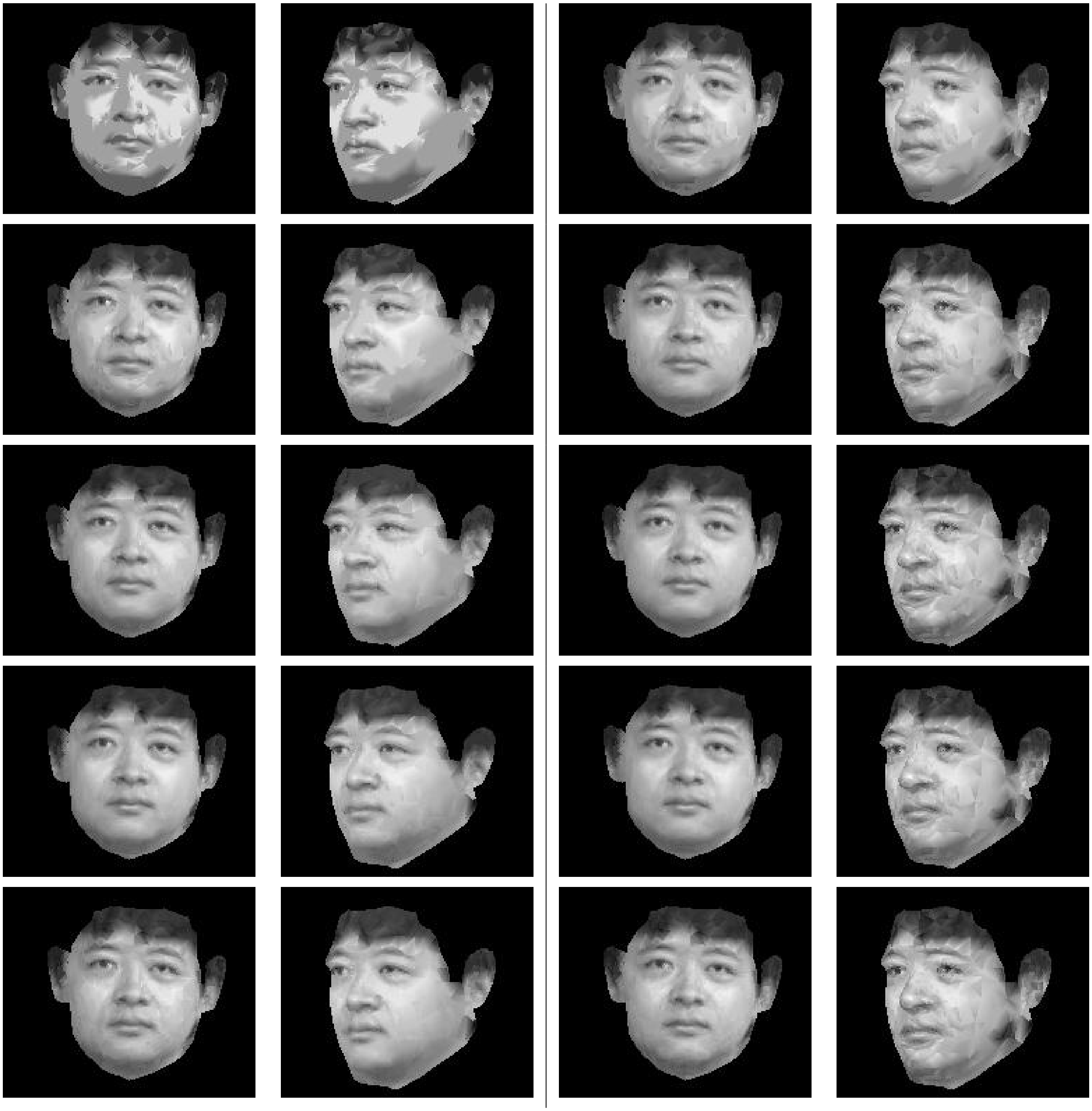


Figure 9: The rate-distortion tradeoff using the TCI feature extraction technique.

Even though PD-type side information yields the best performance at low bit rates as seen from Figures 6, 9, and 12, reconstruction based on POD-type side information is more beneficial in scenarios in which the channel carrying the D-frame is less reliable and can result in a very noisy reference for the future D-frames. On the other hand, since the temporal correlation is much higher than the correlation between the views, relying solely on the other view always yields worse performance.

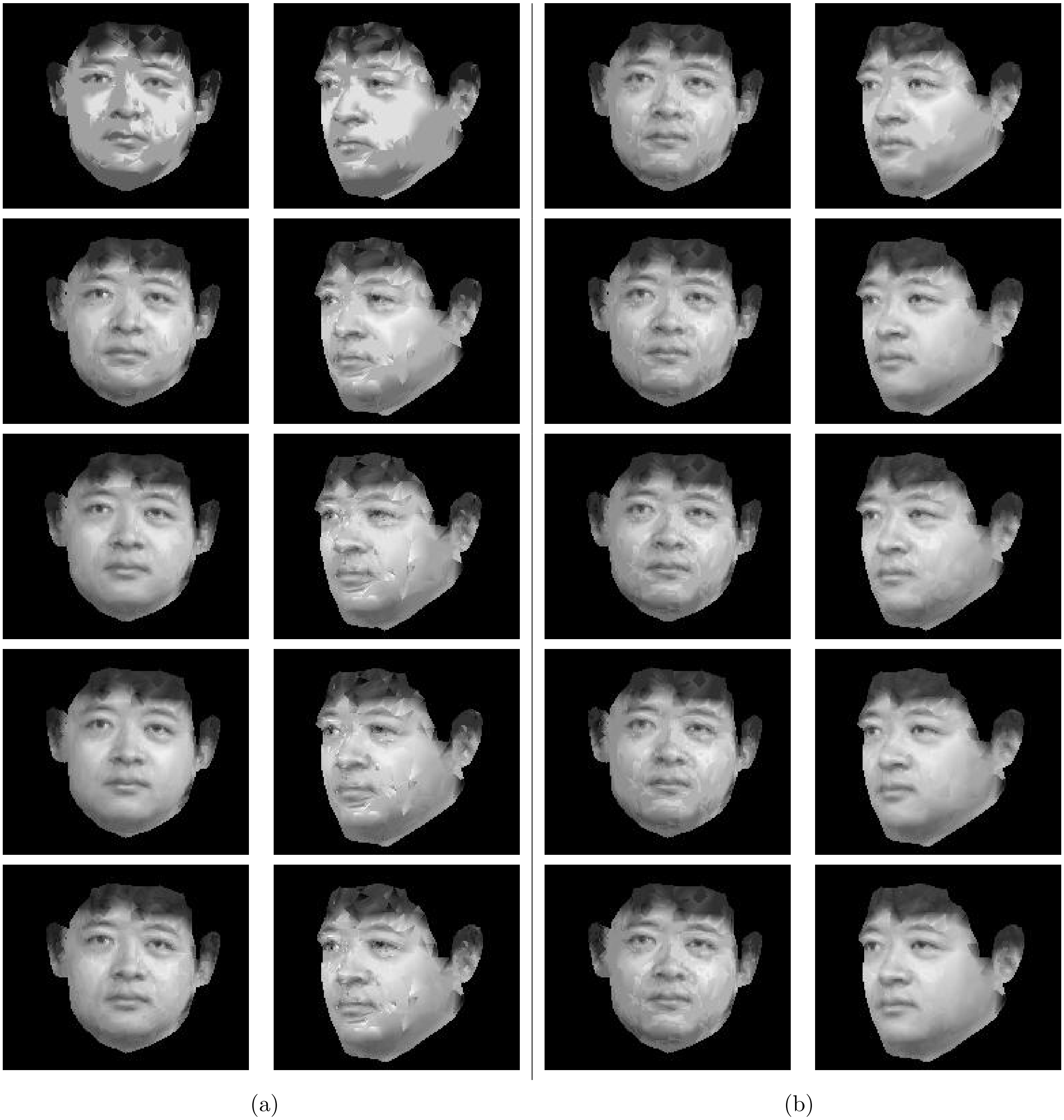
5 Conclusions

In this article, we have presented a method for distributed compression of two video sequences by using a combination of 3D model-based motion estimation and distributed source coding. For the motion estimation, we propose to use a newly developed inverse compositional estimation technique that is computationally efficient and robust. For the coding method, a binning scheme was used with each feature being coded independently. Different methods for rendering the decoded scene were considered. Detailed experimental results were shown. Analyzing the experiments, we found that the distributed video compression was more efficient than separate motion estimation based coding at very low bit rates. At higher bit rates, the ability of the motion estimation methods to remove most of the redundancy in each video sequence left very little



(a) (b)

Figure 10: Coding using the TLI technique for feature extraction. (a) Separate coding of the two views using Scheme 1 with bit rate ≈ 120kbps and PSNR ≈ 28dB. (b) Distributed coding using Scheme 1 and OD-type side information with bit rate ≈ 128kbps and PSNR ≈ 27dB.



(a) (b)

Figure 11: Coding using the TLI technique for feature extraction and POD-type side information. (a) Distributed coding using Scheme 1 with bit rate ≈ 128kbps and PSNR ≈ 28.7dB. (b) Distributed coding using Scheme 2 with bit rate ≈ 115kbps and PSNR ≈ 26.5dB.

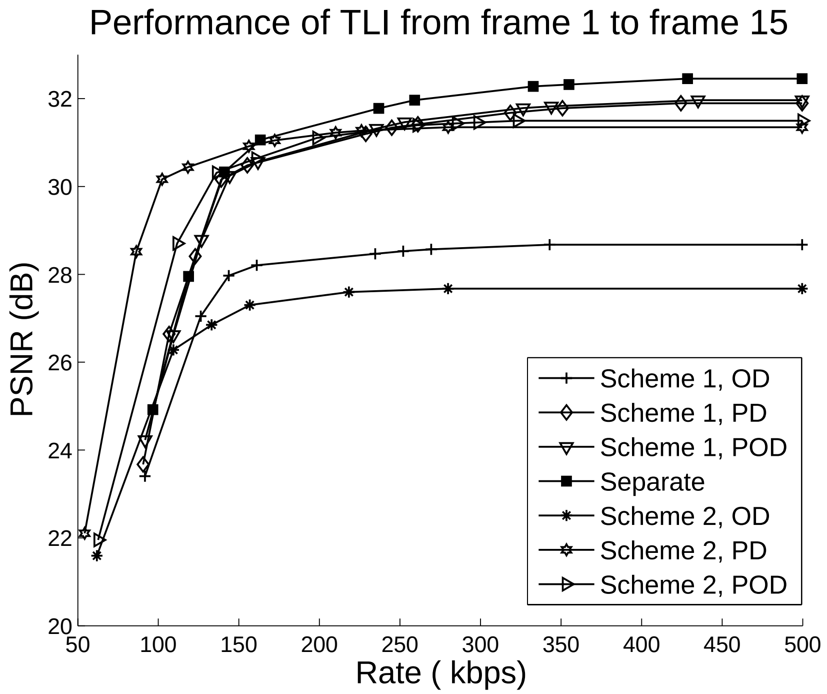


Figure 12: The rate-distortion tradeoff using the TLI feature extraction technique.

to be exploited by considering the overlap between the views. We believe that this should be taken into account in future while designing distributed coding schemes in video.

References

Artigas, X., Angeli, E. and Torres, L.: 2006, Side information generation for multiview distributed video coding using a fusion approach, *7th Nordic Signal Processing Symposium*, 250–253.

Baker, S. and Matthews, I.: 2004, Lucas-Kanade 20 years on: A unifying framework, *International Journal of Computer Vision* **56**(3), 221–255.

Blanz, V. and Vetter, T.: 2003, Face recognition based on fitting a 3D morphable model, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **25**(9), 1063–1074.

Freedman, D. and Turek, M.: 2005, Illumination-invariant tracking via graph cuts, *Proc. of IEEE Conference on Computer Vision and Pattern Recognition.*

Gehrig, N. and Dragotti, P. L.: 2005, DIFFERENT: Distributed and fully flexible image encoders for camera sensor networks, *International Conference on Image Processing.*

Girod, B., Margot, A., Rane, S. and Rebollo-Monedero, D.: 2005, Distributed video coding, *Proceedings of the IEEE* **93**(1), 71–83.

Guo, X., Lu, Y., Wu, F., Gao, W. and Li, S.: 2006, Distributed multi-view video coding, *Visual Communications and Image Processing 2006* **6077**(1), 60770T.1–60770T.8.

Hager, G. D. and Belhumeur, P.: 1998, Efficient region tracking with parametric models of geometry and illumination, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **20**(10), 1025–1039.

Jin, H., Favaro, P. and Soatto, S.: 2001, Real-time feature tracking and outlier rejection with changes in illumination, *IEEE Intl. Conf. on Computer Vision*.

Lathauwer, L. D., Moor, B. D. and Vandewalle, J.: 2000, A Multillinear Singular Value Decomposition, *SIAM J. Matrix Anal. Appl.* **21**(4), 1253–1278.

Matthews, I. and Baker, S.: 2004, Active appearance models revisited, *International Journal of Computer Vision* **60**(2), 135–164.

Ouaret, M., Dufaux, F. and Ebrahimi, T.: 2006, Fusion-based multiview distributed video coding, *4th ACM International Workshop on Video Surveillance and Sensor Networks*, 139–144.

Pradhan, S. and Ramchandran, K.: 1999, Distributed source coding using syndromes (DISCUS): design and construction, *Data Compression Conference*, 158–167

Puri, R. and Ramchandran, K.: 2007, PRISM: A video coding architecture based on distributed compression principles, *IEEE Transactions on Image Processing* **16**(10), 2436–2448.

Romdhani, S. and Vetter, T.: 2003, Efficient, robust and accurate fitting of a 3D morphable model, *IEEE International conference on Computer Vision 2003*.

Shum, H.-Y. and Szeliski, R.: 2000, Construction of panoramic image mosaics with global and local alignment, *International Journal of Computer Vision* **16**(1), 63–84.

Slepian, D. and Wolf, J. K.: 1973, Noiseless coding of correlated information sources, *IEEE Transactions on Information Theory* **19**(4), 471–480.

Song, B., Bursalioglu, O., Roy-Chowdhury, A. and Tuncel, E.: 2006, Towards a multi-terminal video compression algorithm using epipolar geometry, *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing*.

Wagner, R., Nowak, R. and Baranuik, R.: 2003, Distributed image compression for sensor networks using correspondence analysis and superresolution, *ICIP*.

Wyner, A.: 1974, Recent results in the Shannon theory, *IEEE Transactions on Information Theory* **20**(1), 2–10.

Xu, Y. and Roy-Chowdhury, A.: 2007, Integrating motion, illumination and structure in video sequences, with applications in illumination-invariant tracking, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **29**(5), 793–807.

Xu, Y. and Roy-Chowdhury, A.: 2008, A Theoretical analysis of linear and multi-linear models of image appearance, *IEEE International Conference on Computer Vision and Pattern Recognition*.

Xu, Y. and Roy-Chowdhury, A.: 2008, Inverse compositional estimation of 3d motion and lighting in dynamic scenes, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, to appear, July 2008.

Yang, Y., Stankovic, V., Zhao, W., and Xiong, Z.: 2007, Multiterminal video coding, *IEEE International Conference on Image Processing*, pp. III 28–28.

Zamir, R. and Shamai, S.: 1998, Nested linear/lattice codes for Wyner-Ziv encoding, *Information Theory Workshop*, pp. 92–93.

Zhu, X., Aaron, A. and Girod, B.: 2003, Distributed compression for large camera arrays, *IEEE Workshop on Statistical Signal Processing*.