In: Data Compression Conference 2011 proceedings

Lie Group Transformation Models for Predictive Video Coding

Ching Ming Wang, Jascha Sohl-Dickstein, Ivana Tošić, Bruno A. Olshausen
Redwood Center for Theoretical Neuroscience, UC Berkeley
{cmwang, jascha, ivana, baolshausen}@berkeley.edu

Abstract

We propose a new method for modeling the temporal correlation in videos, based on local transforms realized by Lie group operators. A large class of transforms can be theoretically described by these operators; however, we propose to learn from natural movies a subset of transforms that are statistically relevant for video representation. The proposed transformation modeling is further exploited to remove inter-view redundancy, i.e., as the prediction step of video encoding. Since the Lie group transformation coefficients are continuous, a quantization step is necessary for each transform. Therefore, we derive theoretical bounds on the distortion due to coefficient quantization. The experimental results demonstrate that the new prediction method with learned transforms leads to better rate-distortion performance at higher bit-rates, and competitive performance at lower bit-rates, compared to the standard prediction based on block-based motion estimation.

I. Introduction

Temporal correlations in dynamic scenes introduce an enormous amount of redundancy within natural movies, which plays a crucial role in video compression and video analysis. Even though the temporal correlation in 3D scenes arises from 3D rigid or non-rigid motions, which translate into affine, perspective or more complicated transforms on the image plane, video compression techniques remain limited to block-based motion estimation that exploits only correlation due to translational motion. More complicated transformation models such as those incorporating perspective or affine affine transformations stay limited to global predictive coding (e.g., background warping) [1], object and region-based coding [2], or motion refinement [3]. One of the main reasons for this is that the bit-rate overhead for encoding the transform parameters becomes prohibitive when the size of the blocks decreases. However, previous work has considered only hand-designed transforms, described by a fixed set of parameters. Such transforms are undoubtedly general enough to include many examples in data, but they might not all have equal statistical importance for video representation. Accordingly, if one could encode videos using only the spatial transforms that are statistically dominant, instead of using the full set of transforms, one could expect gains in rate-distortion performance in video compression.

C.M. Wang, J. Sohl-Dickstein and B.A. Olshausen are supported by Canadian Institute for Advanced Research - Neural Computation and Perception Program. I. Tošić is supported by the Swiss National Science Foundation under the fellowship no:PBELP2-127847.

The idea of learning a Lie, or continuous transformation, group representation of the dynamics which occur in the visual world was first introduced by Rao et al [4]. A large class of visual transformations, including all the affine transformations, intensity changes due to changes in lighting, contrast changes and spatially localized versions of all the preceding, can be described simply using Lie group operators. Although directly training such a model is of prohibitive computational complexity, one can efficiently learn Lie operators by re-parameterizing them in terms of their eigenvectors and eigenvalues, as we have previously shown in [5]. By additionally introducing transformation specific blurring operators, we can robustly infer transformations between frames of video in a multi-scale fashion using the learned operators.

We propose to learn Lie group transforms from natural movies and to exploit them for modeling inter-frame correlation within the prediction step of a hybrid video coding scheme. Each block in a prediction frame is approximated by a cascade of Lie operators applied to the corresponding region and its immediate surround in the reference frame. We derive bounds on distortion due to coefficient quantization. To the best of our knowledge, the proposed scheme represents the first predictive video coder based on transformations learned directly from visual data. Experimental results show that the prediction with the simplest transforms, that are translations, provide better rate-distortion performance by up to 1 dB at higher bit-rates compared to block-based motion estimation. Moreover, predictions with additional learned transforms, which represent more complex motion structures, offer even better quality of the predicted frame without requiring a large coding overhead.

The paper is structured as follows. In Section II, we briefly overview the theory of modeling the dynamics in visual scenes by Lie group operators, and describe an efficient method for learning those transforms from natural videos. In Section III we describe the proposed coding scheme and provide a distortion analysis for quantization of the coefficients for learned transformations. We give the experimental results in Section IV, and conclude the paper in Section V.

II. Learning Lie group transformations

Consider the class of continuous transformations described by the first order differential equation [4]

$$\frac{\partial \mathbf{x}(s)}{\partial s} = \mathbf{A}\ \mathbf{x}(s), \tag{1}$$

whose solution is

$$\mathbf{x}(s) = e^{\mathbf{A}s}\mathbf{x}(0) = \mathbf{T}(s)\,\mathbf{x}(0). \tag{2}$$

Here $\mathbf{A} \in \Re^{N\times N}$ is an infinitesimal transformation operator and the generator of the Lie group; $s \in \Re$ is a coefficient which modulates the amount of transformation; $\mathbf{T}(s) = e^{\mathbf{A}s}$ is a matrix exponential defined by its Taylor expansion and $\mathbf{x}(s) \in \Re^{N\times 1}$ is the signal $\mathbf{x}(0)$ transformed by $\mathbf{A}$ to a degree controlled by s.

The transformations of this form can be used to model the changes between adjacent frames $\mathbf{x}^{(t)}, \mathbf{x}^{(t+1)} \in \Re^{N\times 1}$ in video. We first overview the case of a single transform and then generalize to multiple ones. The goal of modeling the dynamics of video by Lie group operators is to find the model parameters $\mathbf{A}$ (adapted over an ensemble of video

image sequences) and coefficients $s^{(t-1)}$ (inferred for each pair of frames) that minimize the reconstruction error

$$E_r = \sum_t \left|\left|\mathbf{x}^{(t+1)} - \mathbf{T}\left(s^{(t)}\right)\mathbf{x}^{(t)}\right|\right|_2^2. \tag{3}$$

Estimating model parameters $\mathbf{A}$ is usually achieved by learning from a large batch of data samples. In order to derive a learning rule for $\mathbf{A}$, it is necessary to compute the gradient $\frac{\partial e^{\mathbf{A}s}}{\partial \mathbf{A}}$. This can be done efficiently if $\mathbf{A}$ is rewritten in terms of its eigen-decomposition $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$ and learning is instead performed directly in terms of $\mathbf{U}$ and $\mathbf{\Lambda}$.[1] $\mathbf{U} \in \mathbb{C}^{N\times N}$ is a complex matrix consisting of the eigenvectors of $\mathbf{A}$, and $\mathbf{\Lambda} \in \mathbb{C}^{N\times N}$ is a complex diagonal matrix holding the eigenvalues of $\mathbf{A}$. The matrices must be complex in order to facilitate periodic transformations, such as rotation. Recall also that the eigenvectors $\mathbf{U}$ of a matrix $\mathbf{A}$ need not be orthonormal. The benefit of this representation is that

$$e^{\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}s} = I + \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}s + \frac{1}{2}\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}\mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}s^2 + \ldots = \mathbf{U}e^{s\mathbf{\Lambda}}\mathbf{U}^{-1} \tag{4}$$

where the matrix exponential of a diagonal matrix is simply the element-wise exponential of its diagonal entries. This representation therefore replaces the full matrix exponential by two matrix multiplications and an element-wise exponential, and thus greatly facilitates learning.

Unfortunately, the reconstruction error described by Eq. 3 is typically highly non-convex in s and contains many local minima. To overcome this problem, we propose an alternative transformation, motivated by image matching algorithms [6]–[9], which adaptively smooths the error function in terms of the transformation coefficient. This is achieved by averaging over a range of transformations using a Gaussian distribution for the coefficient values

$$\mathbf{T}(\mu,\sigma) = \int_{-\infty}^{\infty} \mathbf{T}(s)\,\frac{1}{\sqrt{2\pi}\sigma} e^{\frac{||s-\mu||^2}{2\sigma^2}}\,\mathrm{d}s = \mathbf{U}e^{\mu\mathbf{\Lambda}}e^{\frac{1}{2}\mathbf{\Lambda}^2\sigma^2}\mathbf{U}^{-1} \tag{5}$$

and replacing $\mathbf{T}(s)$ with $\mathbf{T}(\mu,\sigma)$ in Eq. 3. The error is now minimized with respect to both μ and σ. Increasing σ blurs the signal along the transformation direction given by $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{-1}$, allowing local minima to be escaped. In the case of translation, for instance, this averaging over a range of transformations blurs the image along the direction of translation. During simultaneous inference of μ and σ, images are matched first at a coarse scale, and the match refines as the blurring of the image decreases.

The model presented above can be extended to multiple transformations by concatenating transformations in the following way:

$$\mathbf{T}_{\bar{n}}(\mu,\sigma) = \mathbf{T}_1(\mu_1,\sigma_1)\,\mathbf{T}_2(\mu_2,\sigma_2)\ldots = \prod_{k=1}^{n}\mathbf{T}_k(\mu_k,\sigma_k) \tag{6}$$

$$\mathbf{T}_k(\mu_k,\sigma_k) = \mathbf{U}_k e^{\mu_k\mathbf{\Lambda}_k}e^{\frac{1}{2}\mathbf{\Lambda}_k^2\sigma_k^2}\mathbf{U}_k^{-1} \tag{7}$$

where k indexes the transformation and $\bar{n}$ indicates a concatenation of transformations with indexes $1,..,n$. Note that the transformations $\mathbf{T}_k(\mu_k,\sigma_k)$ do not commute in general,

[1]This change of form enforces the restriction that $\mathbf{A}$ be diagonalizable. By construction, the A matrices learned with this algorithm are diagonalizable. A large variety of transformations, including all affine transforms, can be represented in this form.

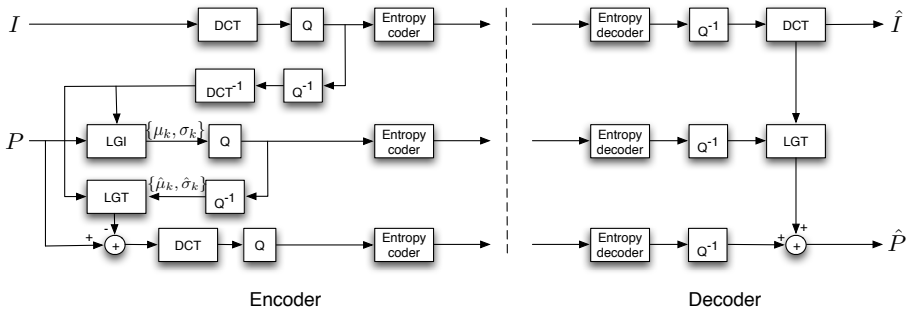

Fig. 1. Predictive hybrid video coder based on Lie Group transforms.

and thus the ordering of the terms in the product must be maintained. Also note that, again due to the lack of commutativity, transformation operators concatenated in this way may no longer form a Lie group, although they still allow a rich description of the transformations in natural scenes.

Finally, the full model's objective function is

$$E = \sum_t \left|\left|\mathbf{x}^{(t+1)} - \mathbf{T}_{\bar{n}}\left(\mu,\sigma\right)\mathbf{x}^{(t)}\right|\right|_2^2 + \eta_n \sum_{t,k} \mu_k \left|\left|\mathbf{A}_k e^{\frac{\mu_k}{2}\mathbf{A}_k}\mathbf{x}^{(t)}\right|\right|_2 + \eta_\sigma \sum_{t,k}(\sigma_k)^2, \quad (8)$$

where the final two regularization terms, described in more detail in [5], encourage the transformations to find the most direct path between initial and final frames, and speed convergence of the operators.

Learning of $\mathbf{U}$ and $\mathbf{\Lambda}$ can be performed via a variational Expectation-Maximization strategy, where the training data consists of a set of image patches from adjacent frames in natural video. A rescaling process is necessary to remove degeneracies between $\mathbf{U}_k$ and $\mathbf{U}_k^{-1}$. A more detailed description of the optimization scheme, along with derivatives of the energy function E, can be found in [5].

III. Learned Lie group transformations for predictive video coding

The model described in the previous section offers an efficient way to learn statistically important transformations that occur in natural videos. Such transforms can be beneficial in designing the prediction step in standard hybrid video coders. The block-based motion estimation and compensation, used in todays video coders, can be replaced by the estimation of Lie group transformation coefficients, as shown in Figure 1. For each block, the encoder finds parameters (μ_k, σ_k), $k = 1, ..., n$, which minimize the objective function E,where n is the total number of considered transforms. This is done within the block denoted as **LGI**, which stands for Lie Group Inference. **LGT** stands for the Lie Group Transform given by $\mathbf{x}^{(t+1)} = T_{\bar{n}}(\mu,\sigma)\mathbf{x}^{(t)}$. Since the Lie transformation coefficients (μ,σ) are continuous, we first have to quantize them and analyze the distortion introduced by the quantization.

A. Quantization of coefficients: distortion analysis

The total distortion between the original patch and the decoded patch is:

$$D_{tot} = \|\mathbf{x}^{(t+1)} - \hat{\mathbf{x}}^{(t+1)}\|^2 \leq \|\mathbf{x}^{(t+1)} - \tilde{\mathbf{x}}^{(t+1)}\|^2 + \|\tilde{\mathbf{x}}^{(t+1)} - \hat{\mathbf{x}}^{(t+1)}\|^2 = D_a + D_q, \quad (9)$$

where D_a is the distortion due to the approximation error in the inference process, D_q is the distortion due to quantization of the transform coefficients μ and σ; $\hat{\mathbf{x}}^{(t+1)} = \hat{\mathbf{T}}_{\bar{n}}\mathbf{x}^{(t)}$ and $\tilde{\mathbf{x}}^{(t+1)} = \mathbf{T}_{\bar{n}}\mathbf{x}^{(t)}$. For algebraic clarity, we will develop the distortion-quantization relationship for μ only. However, a relationship can be derived for σ^2 as well by following a nearly identical derivation. Let us denote the vector of coefficients μ_k, where $k = 1, ..., n$ represents the transformation index, as $\boldsymbol{\mu}$. Accordingly, its quantized version is denoted as $\hat{\boldsymbol{\mu}}$, and the vector of quantization errors is $\Delta\boldsymbol{\mu}$. $\hat{\mathbf{T}}_{\bar{n}}$ is thus the transform using the quantized coefficient $\hat{\boldsymbol{\mu}}$, $\hat{\mathbf{T}}_{\bar{n}} = \mathbf{T}_{\bar{n}}(\hat{\boldsymbol{\mu}}, \boldsymbol{\sigma})$. Substituting this back into Eq. 9, we get

$$D_q = \|\mathbf{T}_{\bar{n}}\mathbf{x}^{(t)} - \hat{\mathbf{T}}_{\bar{n}}\mathbf{x}^{(t)}\|^2 = \|(\mathbf{T}_{\bar{n}} - \hat{\mathbf{T}}_{\bar{n}})\mathbf{x}^{(t)}\|^2 = (\mathbf{x}^{(t)})^{\mathsf{T}}\Delta\mathbf{T}_{\bar{n}}{}^{\mathsf{T}}\Delta\mathbf{T}_{\bar{n}}\mathbf{x}^{(t)}. \tag{10}$$

Using the form of the transform given in Eq. 6, we obtain

$$\Delta\mathbf{T}_{\bar{n}} = \prod_{k=1}^{n}\mathbf{U}_k e^{\mu_k\boldsymbol{\Lambda}_k}e^{\frac{1}{2}\boldsymbol{\Lambda}_k^2\sigma_k^2}\mathbf{U}_k^{-1} - \prod_{k=1}^{n}\mathbf{U}_k e^{\hat{\mu}_k\boldsymbol{\Lambda}_k}e^{\frac{1}{2}\boldsymbol{\Lambda}_k^2\sigma_k^2}\mathbf{U}_k^{-1}. \tag{11}$$

Writing $e^{\hat{\mu}_k\boldsymbol{\Lambda}_k}$ as a Taylor expansion around $e^{\mu_k\boldsymbol{\Lambda}_k}$, we have

$$e^{\hat{\mu}_k\boldsymbol{\Lambda}_k} = e^{\mu_k\boldsymbol{\Lambda}_k} + \Delta\mu_k\boldsymbol{\Lambda}_k e^{\mu_k\boldsymbol{\Lambda}_k} + \dots\ . \tag{12}$$

Since the errors $\Delta\mu_k$, $k = 1, ..., n$, are typically small, especially in medium and high bit-rate regimes, we can approximate $\Delta\mathbf{T}_{\bar{n}}$ using the zeroth and first order terms. We substitute Eq. 12 back into Eq. 11, expand the product, drop the higher order terms in $\Delta\boldsymbol{\mu}$, and obtain

$$\Delta\mathbf{T}_{\bar{n}} = \prod_{k=1}^{n}\mathbf{U}_k e^{\mu_k\boldsymbol{\Lambda}_k}e^{\frac{1}{2}\boldsymbol{\Lambda}_k^2\sigma_k^2}\mathbf{U}_k^{-1} - \prod_{k=1}^{n}\mathbf{U}_k\left[e^{\mu_k\boldsymbol{\Lambda}_k} + \Delta\mu_k\boldsymbol{\Lambda}_k e^{\mu_k\boldsymbol{\Lambda}_k} + ...\right]e^{\frac{1}{2}\boldsymbol{\Lambda}_k^2\sigma_k^2}\mathbf{U}_k^{-1} \tag{13}$$

$$= \prod_{k=1}^{n}\mathbf{T}_k(\mu_k, \sigma_k) - \prod_{k=1}^{n}\mathbf{T}_k(\mu_k, \sigma_k) \tag{14}$$

$$- \sum_{l=1}^{n}\left(\prod_{k=1}^{l-1}\mathbf{T}_k(\mu_k, \sigma_k)\right)\mathbf{U}_l\Delta\mu_l\boldsymbol{\Lambda}_l e^{\mu_l\boldsymbol{\Lambda}_l}e^{\frac{1}{2}\boldsymbol{\Lambda}_l^2\sigma_l^2}\mathbf{U}_l^{-1}\left(\prod_{k=l+1}^{n}\mathbf{T}_k(\mu_k, \sigma_k)\right)$$

$$- \sum_{l=1}^{n}\sum_{m=1}^{n}\ \dots$$

$$\Delta\mathbf{T}_{\bar{n}} \approx \sum_{l=1}^{n}\Delta\mu_l\mathbf{M}^l \tag{15}$$

$$\mathbf{M}^l = -\left(\prod_{k=1}^{l-1}\mathbf{T}_k(\mu_k, \sigma_k)\right)\mathbf{U}_l\boldsymbol{\Lambda}_l e^{\mu_l\boldsymbol{\Lambda}_l}e^{\frac{1}{2}\boldsymbol{\Lambda}_l^2\sigma_l^2}\mathbf{U}_l^{-1}\left(\prod_{k=l+1}^{n}\mathbf{T}_k(\mu_k, \sigma_k)\right). \tag{16}$$

This gives us a quadratic form for the quantization distortion D_q,

$$D_q \approx \Delta\boldsymbol{\mu}^{\mathsf{T}}\mathbf{Q}\Delta\boldsymbol{\mu}, \tag{17}$$

where the coupling matrix $\mathbf{Q}$ is quadratic in $\mathbf{x}^{(t)}$, i.e.

$$Q_{kl} = \mathbf{x}^{(t)\mathsf{T}}\left(\mathbf{M}^{k\mathsf{T}}\mathbf{M}^l\right)\mathbf{x}^{(t)}. \tag{18}$$

IV. Experimental results

A. Implementation of learning Lie group transformations

Transformation models were trained on video clips extracted from BBC's documentary series *Animal World*. They can be obtained from Hans Van Hateren's repository at *http://hlab.phys.rug.nl/vidlib*. The video clips contain footage of animals dwelling in grassland and rivers. The video contains many types of motion, including camera panning, tracking, object motion, and occlusion.

The training data was 17×17 pixel image patch pairs cropped randomly from consecutive video frames. A 4 pixel buffer was applied during learning, meaning that the mean squared reconstruction error (the first term in Eq. 8) was evaluated only within the central 9×9 region of the images. The rest of the patch represents the search window. Hence, the model can use information inside the buffer region to reconstruct the central region, without penalizing reconstruction error in the buffer region. This setup is used to provide a search region of 9×9 pixels in the reference frame for the 9×9 image patch.

We trained models with a number of additional operators, alongside hard-coded horizontal and vertical translation operators (easily described in analytic form). We chose to hard-code translation partly because it has been shown to account for a majority of the transformation in natural video [10]–[12]. Additionally it allows us to make a meaningful comparison of the proposed video coding scheme based on translations + learned transformations against standard video prediction that uses motion estimation based purely on translations. In a separate test we trained 15 transformations starting from random initial conditions on the same data set. Several of those transformations learned translation along multiple directions, reinforcing the idea that translation is a reasonable operator to hard code. The remainder became intensity scaling, contrast scaling, and spatially localized translation operators, as well as a few operators which defy simple interpretation.

B. Evaluation of the proposed prediction model

To test the performance of different learned models, 2000 image patch pairs were extracted randomly from sequences in *Animal World* that were not used during training. We first evaluated the Peak-Signal to Noise Ratio (PSNR) between the second patch in a chosen pair and its prediction from the first patch, using different prediction models: 1) block-based motion estimation using exhaustive search with single, half and quarter pixel accuracy; and 2) Lie group transformations with two continuous translation operators and 1, 2, 3 and 4 learned transformation operators. Figure 2 shows the average PSNR comparison for all these models. The continuous translation model outperforms exhaustive search methods even at quarter pixel resolution. This is expected, since the continuous transformation coefficients have double precision, unlike motion vectors with limited (quarter-pixel at most) precision. It is also important to note that the PSNR increases when the σ variable is introduced in the inference of the Lie group transformations (note the difference between the fourth and the fifth bar in Figure 2). Recall that σ smoothes the image patch in an operator specific fashion - in this case along the direction of translation. It therefore allows the higher spatial frequencies in the horizontal or vertical direction to be attenuated in the transformed patch when this will lead to a better reconstruction.

The most important observation is that the PSNR of the prediction increases as a function of the number of transformations used for prediction. This is an important result,

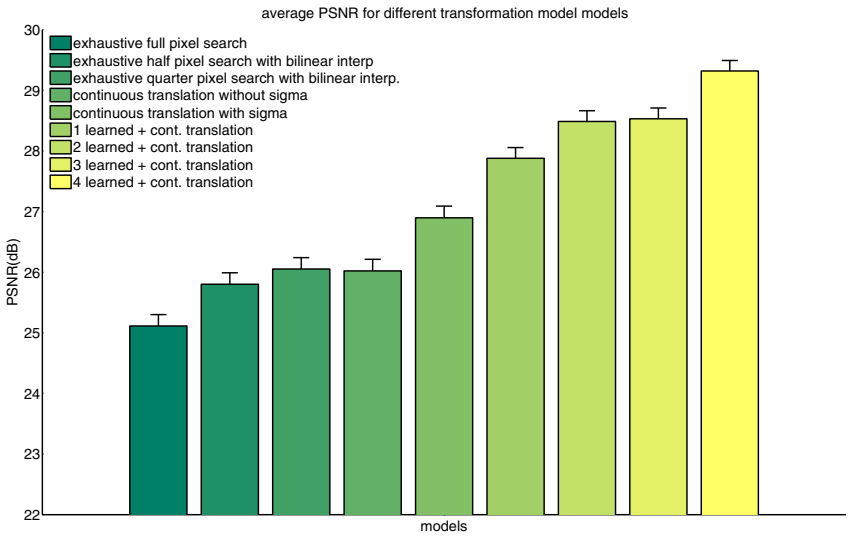

Fig. 2. PSNR of the reconstruction of 2000 image patch pairs extracted from natural video. The first 3 bars correspond to exhaustive search rigid translation model at full pixel, half pixel and quarter pixel resolution. The rest of the 6 bars correspond to continuous transformation models with different numbers of operators.

which shows that even though simple translations account for most of the correlations in video, there is still a significant amount of redundancy left to exploit using more complex, adapted models. However, we still need to evaluate the coding cost introduced by the learned transformations, which is done in the next section.

C. Rate-distortion performance of the proposed prediction model

The coding costs for different learned models were evaluated on the first 40 frames of the *Foreman* sequence. We used the structure I-P-I-P-... . The transformation coefficients were quantized with Lloyd-Max quantizers, which were optimized for distributions of the corresponding coefficients (σ, μ). The bound on the bit-rate for the quantized coefficients was evaluated using the second order conditional entropy, where the quantized coefficients were organized in a raster scan fashion. Specifically, the entropy of the current symbol was computed conditioned on the previous symbol in the symbol stream. We did not implement a specific entropy encoder in order to see the best case performance of all prediction methods.

Figure 3 shows the average number of bits per pixel required to code the motion model for a given PSNR. The PSNR is obtained by averaging over the P-frames for the first 40 frames of the *Foreman* sequence. The quantization for μ ranges from 2 to 8 bits and the quantization for σ ranges from 1 to 6 bits. The rate-distortion curves represents the outer-hull of the all combinations of these parameter settings. Therefore, these plots correspond to the case where the strategy for bit-allocation among parameters is achieved by exhaustive search of the parameter space.

The most striking result is that the bit cost per pixel of our analytically coded continuous translation model is lower than the traditional exhaustive search motion estimation at rates above 0.06 bits per pixel, even though our model has four parameters per block. This

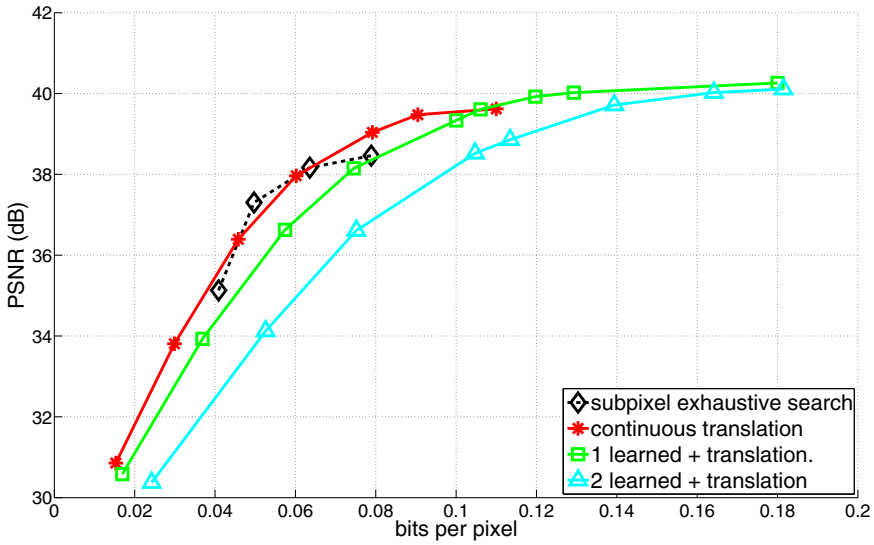

Fig. 3. The coding cost of different motion models versus the PSNR of model reconstruction. This plot is obtained by averaging over the P-frames of the first 40 frames of *Foreman*.

suggests that the inferred coefficients are more consistent between neighboring blocks, which leads to smaller second order entropy and more efficient encoding. This probably means that the motion field defined by the inferred continuous translation coefficients is more spatially regular, where the regularization comes from the smoothing operator introduced in the model. The most complex models that incorporates one and two learned transforms in additional to the translation operators (green and cyan curve) outperform the other models at higher rates.

D. Performance of the hybrid video coder

Finally, we have evaluated the coding performance of a full hybrid video coder using learned transformation models. The transformation model coefficients of the P-frames are quantized and coded as described in the previous section. The DCT coefficients of the transformation compensated residual are quantized uniformly and their coding cost is computed with empirical entropy.

In the hybrid coding scheme, there is a trade off between the coding cost for the motion model and the coding cost for the residual. In general, better motion models lead to lower energy in the motion compensated residual and to lower coding cost for the residual. On the other hand, a more complex motion model with a larger number of coefficients is more expensive to code. Figure 5 shows the rate distortion of the first 40 frames of *Foreman* coding with different motion models. For each of the motion models, we vary the number of quantization bins for μ, σ and the DCT coefficients of the residual. The curves shows the outer hull of the rate distortion of each model. The continuous translation model introduced in this paper outperforms the traditional subpixel exhaustive search motion model at higher bit-rates by up to 1dB, while it performs competitively at lower bit rates.

The coding cost increases as the motion model becomes more complex than continuous translation (i.e., when including more learned transformations). We believe this to be largely caused by two factors. First, the quantization based on Lloyd-Max optimization is suboptimal because the image distortion is not equal to the coefficient distortion, as shown in Section III-A. Second, the residue after transform compensation has different statistics than in the case of translations only, which suggests that a different residual encoding strategy might prove useful. Figure 4 shows the power spectrums of the residue after applying different prediction models. Those models with additional learned operators have less power at low frequencies, suggesting that a DCT basis may not be well suited to them. These are important questions to address for the complete coder design, and we believe that more advanced quantizer and encoder models will lead to even better rate-distortion performance than the continuous translation model.

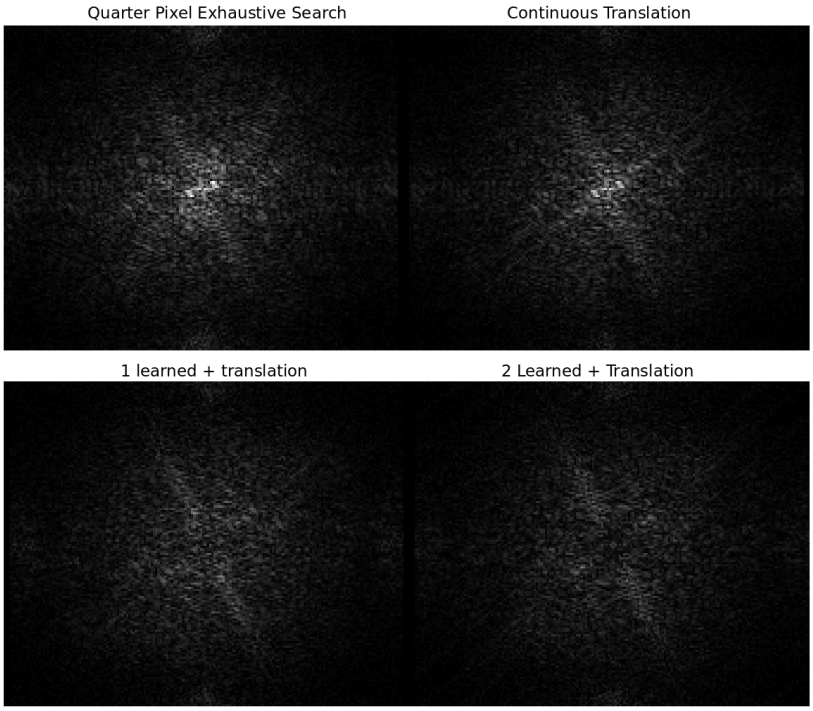

Fig. 4. The 2d power spectrum of the motion compensated residual of frame 14 of *Foreman*. Low frequencies lie in the center of the image, and the DC component has been removed.

V. Conclusion

We have proposed a method for learning transformation operators in natural movies, which are then exploited for removing the inter-view redundancy in a predictive video coding scheme. The efficiency of the learned models has been evaluated with respect to the quality of the prediction and the rate-distortion efficiency of a hybrid video coding scheme. We have shown that the coder based on learned transformations outperforms the standard coder based on prediction with the exhaustive search motion estimation model at rates above 0.06 bpp. These results suggest that the proposed method has potential for designing video coders that are able to efficiently compress data characterized by a

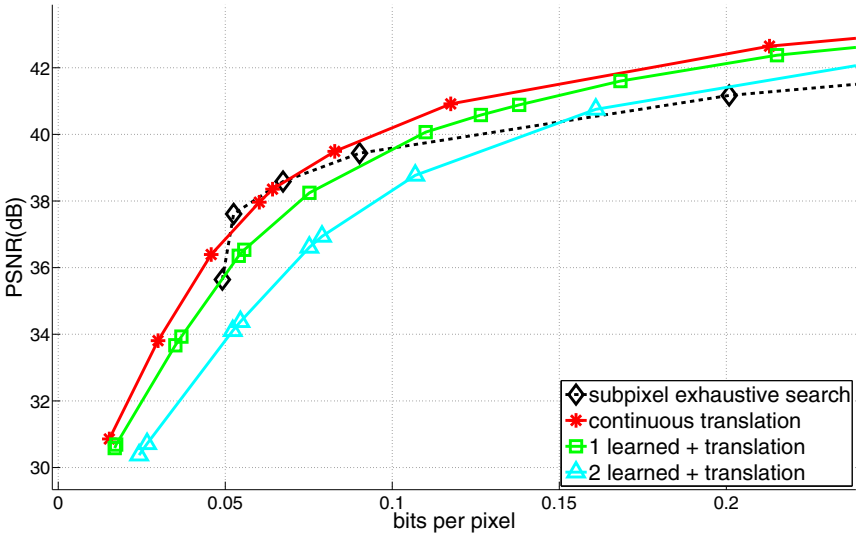

Fig. 5. The outer hull of the rate-distortion curves for the coder in Figure 1, and the for the hybrid video coder based on motion estimation. This plot is obtained by averaging over the P-frames of the first 40 frames of *Foreman*.

wide variability of transformation structures. Likewise, learning transforms in multi-view video data offers an interesting perspective on the problem of exploiting temporal and inter-camera redundancy in multi-view video coding, where simple translational models are not sufficient to model the correlation inherent to multi-view geometry.

REFERENCES

[1] F. Dufaux and F. Moscheni, "Background mosaicking for low bit rate video coding," in *International Conference on Image Processing*, vol. 1, Sep. 1996, pp. 673 –676.

[2] Y. Yokoyama, Y. Miyamoto, and M. Ohta, "Very low bit rate video coding using arbitrarily shaped region-based motion compensation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 500 –507, Dec. 1995.

[3] T. Wiegand, E. Steinbach, and B. Girod, "Affine multipicture motion-compensated prediction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 15, no. 2, pp. 197 – 209, 2005.

[4] R. Rao and D. Ruderman, "Learning lie groups for invariant visual perception," *Advances in Neural Information Processing Systems 11*, pp. 810–816, 1999.

[5] J. Sohl-Dickstein, J. C. Wang and B. A. Olshausen, "An unsupervised algorithm for learning lie group transformations," *CoRR*, vol. abs/1001.1027, 2010.

[6] D. Arathorn, *Map-seeking circuits in visual cognition: a computational mechanism for biological and machine vision*. Stanford University Press, 2002.

[7] N. Vasconcelos and A. Lippman, "Multiresolution tangent distance for affine invariant classification," *Proceedings of Neural Information Processing Systems 10*, 1997.

[8] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Proc. of the Fourth European Conference on Computer Vision*, pp. 320–342, 1996.

[9] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," *Proceedings of Imaging understanding workshop*, pp. 121–130, 1981.

[10] R. Memisevic and G. Hinton, "Learning to represent spatial transformations with factored higher-order boltzmann machines," *Neural Computation*, Jan 2010.

[11] B. Olshausen, C. Cadieu, B. Culpepper, and D. Warland, "Bilinear models of natural images," *SPIE Proceedings vol. 6492: Human Vision Electronic Imaging XII (B.E. Rogowitz, T.N. Pappas, S.J. Daly, Eds.)*, 2007.

[12] C. Cadieu and B. Olshausen, "Learning transformational invariants from natural movies," *Advances in Neural Information Processing Systems 21*, pp. 209–216, 2008.