MITSUBISHI ELECTRIC RESEARCH LABORATORIES
http://www.merl.com

Graph Transformation for Keypoint Trajectory Coding

Tian, D.; Sun, H.; Vetro, A.

TR2016-157 December 2016

Abstract

In contrast to still image analysis, motion information offers a powerful means to analyze video. In particular, motion trajectories determined from keypoints have become very popular in recent years for a variety of video analysis tasks, including search, retrieval and classification. Additionally, cloud-based analysis of media content has been gaining momentum, so efficient communication of salient video information to perform the necessary analysis of video at the cloud server is needed. In this paper, we propose a novel graph transformation to efficiently represent the keypoint trajectories, motivated by the fact that keypoints are distributed irregularly across the images. Compared to conventional DCT-like transformation, it is easier for graph transform to compact the energy and make the coding efficiently. Experimental results on several popular datasets including Stanford MAR, Hopkin155, KITTI, etc. demonstrate a significant rate saving between 26% and 42% with our proposed trajectory coding approaches relative to a DCT based transformation approach, provided that the coding errors are between 2 pixels to 4 pixels.

IEEE Global Conference on Signal and Information Processing (GlobalSIP)

GRAPH TRANSFORMATION FOR KEYPOINT TRAJECTORY CODING

Dong Tian, Huifang Sun, Anthony Vetro

Mitsubishi Electric Research Labs (MERL)
Cambridge, Massachusetts, USA
{tian, hsun, avetro}@merl.com

ABSTRACT

In contrast to still image analysis, motion information offers a powerful means to analyze video. In particular, motion trajectories determined from keypoints have become very popular in recent years for a variety of video analysis tasks, including search, retrieval and classification. Additionally, cloud-based analysis of media content has been gaining momentum, so efficient communication of salient video information to perform the necessary analysis of video at the cloud server is needed. In this paper, we propose a novel graph transformation to efficiently represent the keypoint trajectories, motivated by the fact that keypoints are distributed irregularly across the images. Compared to conventional DCT-like transformation, it is easier for graph transform to compact the energy and make the coding efficiently. Experimental results on several popular datasets including Stanford MAR, Hopkin155, KITTI, etc. demonstrate a significant rate saving between 26% and 42% with our proposed trajectory coding approaches relative to a DCT based transformation approach, provided that the coding errors are between 2 pixels to 4 pixels.

Index Terms— Keypoint trajectory, video analysis, interframe prediction, graph transformation, CDVA

1. INTRODUCTION

For image analysis and understanding tasks, it is common to identify and extract keypoints with accompanying feature descriptors in an image. Such descriptors need to be invariant to translation, rotation, scaling and illumination so as to efficiently and robustly recognize objects and/or scenes, e.g. scale-invariant feature transform (SIFT) [1], speeded-up robust features (SURF) [2], histogram of oriented gradients (HoG) [3] [4], etc. For applications that perform image analysis tasks over cloud servers, the feature descriptors need then to be transmitted over networks and hence a requirement arises to efficiently code those descriptors. *Compact Descriptor for Visual Analysis* (CDVS) is an ISO/IEC standard addressing such a compression need that was released recently [5]. One limitation of this standard is that it only handles the coding of descriptors associated with a single image frame.

Compared to image analysis, the motion of objects in a scene is very important for video analysis problems, such as object tracking, action recognition /detection [6], mobile robotics [7] and autonomous driving, as well as motion segmentation [8]. Similar to image analysis schemes that are based upon keypoints, motion trajectories are often determined for keypoints; the coding of these keypoint trajectories for the purpose of enabling efficient video analysis is the primary focus of this paper.

Earlier work that was done in the context of MPEG-7 defined several types of motion descriptors, that described camera motion, motion activity for video segments, as well as motion trajectories for regions of the scene [9]. The MPEG-7 motion trajectory descriptor was fairly high-level in that a single trajectory was associated with one object of the scene. Since the number of objects in a scene usually is rather limited, it was not demanding to efficiently code each trajectory.

A naive method to represent the descriptors extracted for a video is to treat each image independently. As the approach does not exploit the similarities among the features from successive images frames, it would lead to a very redundant representation. One approach to remove such redundancy is to make use of the motion of those descriptors through the video sequence via an affine transform between neighboring pictures [10] [11]. However, those methods are limited in using a single affine transformation which may oversimplify the motion for many use cases, thereby reducing the accuracy of the analysis results.

Low-rank non-negative matrix factorization [12] demonstrated the capability to perform object/scene clustering using a small percentage of visual descriptors. Hence, if all descriptors from an image have been coded, no significant bits are expected to code their corresponding visual descriptors in the successive image frames. However, this approach was not able to provide a representation for motion over time, which is the major motivation of our work.

In our work in [13], a framework was proposed to represent keypoint trajectories across pictures of a video from utilizing interframe prediction. In this paper, we extend our previous work by appending a novel graph transformation for efficient compression of the motion trajectories to support a good range of video analysis tasks.

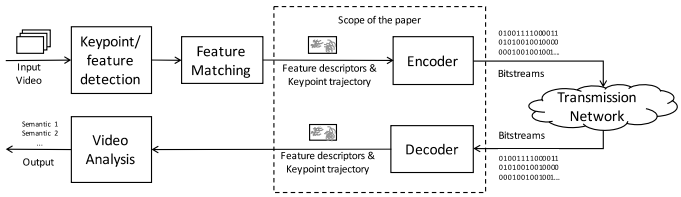

Fig. 1. System diagram [13]

The remainder of the paper is organized as follows. The next section reviews the overall system framework proposed in [13]. Section 3 proposes a graph transform to efficiently encode the trajectory shapes. Section 4 presents experimental results comparing the proposed methods against a benchmark method using DCT transform to show the superiority of the proposed methods. Finally, Section 5 provides concluding remarks.

2. SYSTEM OVERVIEW

As a low-level motion information, keypoint trajectories are useful for many high-level motion analysis tasks. To the best of our knowledge, there is no work done to efficiently compress the keypoint trajectories before our previous work [13].

In a typical scenario as shown in Fig. 2, trajectory information for each keypoint is extracted and then encoded at the client device. We rely on existing methods to generate the keypoint trajectories. At the cloud server, the keypoint trajectories are reconstructed then fed to a motion-based video analysis module to perform the desired video analysis tasks.

A merit to the above framework is that such a video analysis system could be compatible with the image descriptors and analysis systems, where the descriptors associated with *key pictures* are coded in a conventional way. Further, a *Group of Pictures* (GOP) structure for every n pictures is composed of a key picture and several other non-key pictures.

More formally, a *keypoint trajectory* is represented by a sorted set of positions over time,

$$\{\mathbf{p}(t_i) = (\mathbf{p}_x(t_i), \mathbf{p}_y(t_i)),\ t_i \in \{t_1, t_2, .., t_n\}\}. \quad (1)$$

Assume that there are m keypoints with associated trajectories. Let $\mathbf{p}^c,\ c \in [1, m]$ denote the position in the c-th trajectory. As the trajectory represents the travel path of a keypoint, the associated feature descriptors of the same keypoint in the subsequent picture are likely unchanged and would be skipped for actual coding. Therefore, the major challenge to extend the still image analysis toward video analysis is on the coding $\{\mathbf{p}^c(t_i),\ c \in [1, m],\ i \in [2, n]\}$, given picture t_1 as a key picture that has been previously coded. In addition, let $\hat{\mathbf{p}}$ denote the reconstructed trajectory from a coded bitstream.

The major contribution in [13] was to code the keypoint trajectories in a differential way instead of coding their absolute positions. As a continuation of our previous work, in the next section, we propose graph transformation for efficiently represent the residual trajectories.

3. GRAPH TRANSFORM CODING

3.1. General Graph Transform Coding

The field of the graph-based signal processing is emerging in recent years [14]. An undirected graph $G = (V, E)$ consists of a collection of nodes $V = \{1, 2, ..., N\}$ connected by a set of links $E = \{(i, j, w_{ij})\}$, $i, j \in V$ where (i, j, w_{ij}) denotes the link between nodes i and j having weights w_{ij}. The adjacency matrix $\mathbf{W}$ of the graph is an $N \times N$ matrix, the degree d_i of a node i is the sum of link weights connected to node i. The degree matrix $\mathbf{D} := diag\{d_1, d_2, ..., d_N\}$ is a diagonal matrix, and the combinatorial Laplacian matrix is $\mathcal{L} := \mathbf{D} - \mathbf{W}$. The normalized Laplacian matrix $\mathbf{L} := \mathbf{D}^{-1/2}\mathcal{L}\mathbf{D}^{-1/2}$ is a symmetric positive semi-definite matrix. Therefore, it has eigendecomposition $\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^t$, where $\mathbf{U} = \{\mathbf{u}_1, ..., \mathbf{u}_N\}$ is an orthogonal set of eigenvectors and $\mathbf{\Lambda} = diag\{\lambda_1, ..., \lambda_N\}$ is its corresponding eigenvalue matrix.

The eigenvectors and eigenvalue of the Laplacian matrix provide a spectral interpretation of the graph signals. The eigenvalues $\{\lambda_1, ..., \lambda_N\}$ can be treated as graph frequencies and are always situated in the interval $[0, 2]$ on the real line.

A *graph Fourier transform* (GFT) is defined as a projection of a signal $\mathbf{x}$ onto the eigenvectors $\mathbf{U}$ of the graph:

$$\mathbf{y} := \mathbf{U}^t\mathbf{x}. \quad (2)$$

Since a graph spectral domain is introduced with GFT, it is possible to utilize it for transformation coding on a signal. Supposing q to be the quantization step, we have it applied on the graph coefficients $\mathbf{y}$,

$$\tilde{\mathbf{y}} = round(\mathbf{y}/q), \quad (3)$$

where $round(\cdot)$ converts a floating value to its closest integer value and $\tilde{\mathbf{y}}$ is finally entropy coded to produce the bitstream.

On the decoder side, the signal $\mathbf{x}$ can be reconstructed after de-quantization,

$$\hat{\mathbf{y}} = \tilde{\mathbf{y}}q, \quad (4)$$

and inverse GFT transformation,

$$\hat{\mathbf{x}} = \mathbf{U}\hat{\mathbf{y}}. \quad (5)$$

3.2. Graph Transform for Keypoint Trajectory Coding

As described in Section 2, reference images are used to predict the keypoint location of a trajectory in a non-key image, that is, instead of coding $\mathbf{p}(t_i)$ directly, but coding its gradient $\mathbf{v}(t_i)$ relative to a reference image,

$$\mathbf{v}(t_i) = \mathbf{p}(t_i) - \hat{\mathbf{p}}(t_r), \quad (6)$$

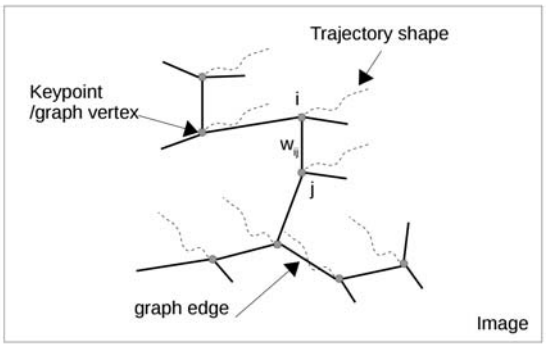

Fig. 2. k-NN graph built on keypoint trajectories, with $k = 3$

where t_i is the current image and t_r corresponds to its reference image. Note that the locations from the reference t_r are selected as its reconstructed values to ensure that the encoding process will use the same predictor as in the decoding process.

Because the keypoint trajectories typically distribute irregularly within an image, one could expect a traditional linear transform, e.g. DCT, may be less efficient if being utilized. As an extension of traditional discrete signal processing from linear to general graphs, graph signal processing provides special benefits when dealing with points in feature spaces that typically reside on an irregular grid. On the contrary, previous work in [15] used graph on compression of image/video signal that reside on a regular grid. Here, we are motivated to explore using a novel graph based transform to code keypoint trajectories.

We first propose to build a graph based on the trajectory locations in the reference image. Each keypoint in the reference image would be treated as a vertex in the constructed graph and is then connected to its k nearest keypoints. Euclidean distance d_{ij} between connected keypoints i and j is calculated to determine the edge weights, as follows,

$$w_{ij} = \exp\left(-\frac{\left\|\mathbf{p}^i - \mathbf{p}^j\right\|_2^2}{2\sigma^2}\right) \tag{7}$$

where σ controls the sensitivity of graph weights relative to Euclidean distances. Once the graph is constructed, the GFT transform $\mathbf{U}$ is determined as the eigenvectors of graph Laplacian $\mathbf{L}$ as described in Section 3.1.

Here, we are basically motivated by the assumption that graph structure keeps being similar between neighboring images along temporal direction. Hence, it makes sense to use a reference image to build a graph and then apply the derived graph to process the current image.

We do not rely on the distance in feature domain since the motion or trajectory is believed more correlated to keypoint positions. The way to assign the graph weights in Eqn. 7 reflects the strength of correlation between the underlying keypoint motions. In addition, Euclidean distance involves only 3 dimensional signals while feature spaces involve much higher dimensions, e.g. 128 for a typical SIFT feature.

Fig. 3. Keypoint trajectories of example video sequences

In the end, we treat the motion trajectories $\mathbf{v}$ to be encoded in Eqn. 6 as a graph signal $\mathbf{x}$ in Eqn. 2. Hence, we could perform the steps from GFT transform, quantization, de-quantization and inverse GFT as shown from Eqn. 2 to 5.

4. EXPERIMENTS AND DISCUSSIONS

4.1. Setting-up

In this section, we will verify the efficiency of the proposed k-NN graph transform on coding keypoint trajectories. We selected 11 sequences covering different motion complexity levels. One sequence from Stanford MAR [11] has limited motion. Two sequences from Hopkin155 [16] were captured by a hand-held camera with multiple moving object in addition to a camera motion. Three indoor sequences from TUM [17] involves faster camera motion with moving people. Two KITTI sequences [7] are from cameras mounted on a moving car. In addition, three sequences from MPEG CDVA dataset were selected.

First, the trajectories are extracted based on Wang's method in [6]. Then 300 trajectories evenly distributed over each picture are selected for the coding experiments. The GOP size is set to 15 pictures. Keypoint trajectories of the 4 example sequences are shown in Fig. 3.

An immediate previous image is selected to be used as reference to predict future trajectory locations, and the derived motion vectors $\mathbf{v}$ as per Eqn. 6 are to be transform coded. Since the coding of key images is not the scope of this work, we will inspect the coding performance of the trajectory locations in the non-key images. The bitrate is collected in terms of bits per picture, and the distortion is evaluated as the distance away from the original locations measured in pixels.

We have the horizontal and vertical component of $\mathbf{v}$ vectorized respectively in one image, as $\mathbf{v}_x$ and $\mathbf{v}_y$. Then the same GFT transform $\mathbf{U}^t$ will be applied on them separately. Since the GFT can be derived from the decoded anchor image, no transmission overhead is required for GFT transform.

As a benchmark approach, we apply 1D DCT on $\mathbf{v}_x$ and $\mathbf{v}_y$. We do not enforce a 2D transform as the keypoints are not in aligned with any regular 2D grid.

After GFT or DCT transformation, we have the coefficients quantized using a series of different steps, $q = 2^i$, $i \in [0, 1, .., 6]$. The same quantization step is shared between the coefficients from $\mathbf{v}_x$ and $\mathbf{v}_y$.

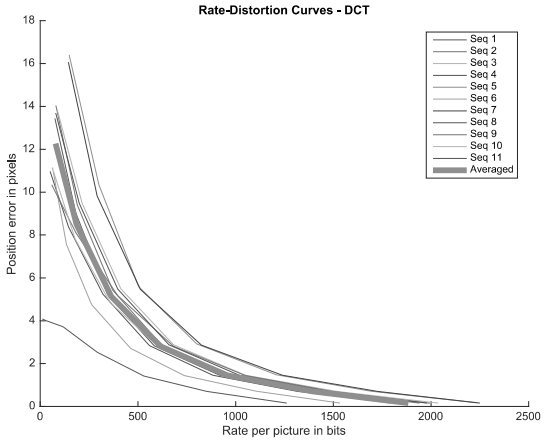

Fig. 4. RD curves for DCT coding

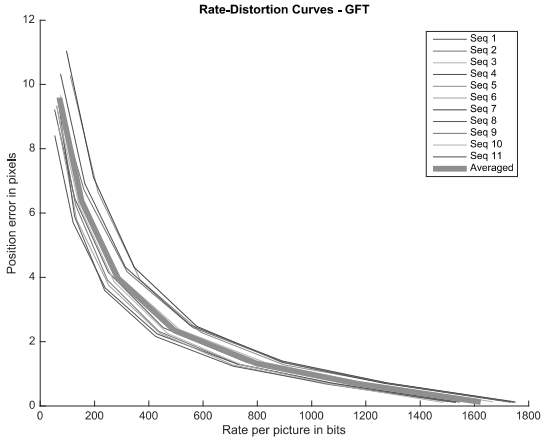

Fig. 5. RD curves for GFT coding

4.2. Experimental Results and Discussions

RD curves of each sequence are plotted with different quantization steps in Fig. 4 for DCT and Fig. 5 for GFT, respectively. The curves appearing lower in the figures at a given bitrate (horizontal axes) indicates smaller distortion and is more favorable. The comparison of the average results over all sequences is put together within Fig. 6 for easier verification. Observations on the results are elaborated below.

When checking the RD curve shapes in Fig. 4 and Fig. 5, we note that the variation of DCT method between sequences is obviously larger than GFT. This demonstrates the content adaptivity property of GFT, while DCT does not have such a property as DCT has fixed transform basis. For GFT, as the graph weights are assigned dynamically based on the keypoint locations in the reference image, the transform basis could be tuned on the fly.

Except for one sequence from Stanford MAR dataset with very simple motion, GFT significantly outperforms DCT in general. Averagely speaking, given the location precision to be 2 pixels, 3 pixels and 4 pixels, the bitrate savings of GFT relative to DCT are 26%, 30%, and 42%, which is a good cut on the transmission or storage burden.

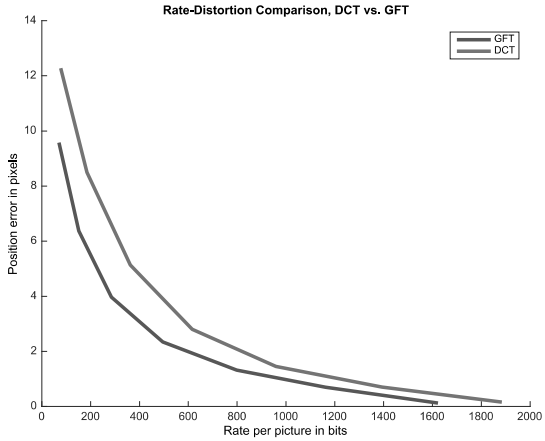

Fig. 6. RD curves comparison, GFT vs DCT coding

In addition to the relative savings, about 600 bits, 400 bits and 280 bits are reported in Fig. 6 to code one image at different trajectory precision of 2, 3 and 4 pixels, or equivalently 2 bits, 1.3 bits, 0.9 bits per keypoint location, considering 300 trajectories being coded. Depending on end usage of the keypoint trajectories, the bitrate can be adaptively chosen.

In our previous work in [13], which is a no transform scheme with lossless coding, i.e. at integer pixel precision, it would consume about 1400 bits per pictures if being averaged over the same set of sequences. On contradictory, GFT would cost around 960 bits per picture for precision less than 1 pixel, that is about 30% bitrate saving compared to the no transform scheme in [13].

Last, as we have mentioned earlier, the spatial feature descriptors along a trajectory are likely to change mildly. However, if it is not the case and they need to be coded, the proposed GFT method could be applied on descriptor coding in the same manner, which is subject to a future work.

5. CONCLUSION

This paper describes a novel graph method to do transform coding for keypoint trajectories of videos for the purpose of analysis. A k-NN graph was proposed to be built based on the keypoint trajectory locations in a reference image, which would lead to an efficient graph transform to code the trajectory locations in a current image. This coding scheme is able to operate at varying bitrates via selection of quantization steps depending on different trajectory location precision. Experimental results demonstrate that the proposed methods significantly outperform a conventional DCT method or a no transform coding scheme. The work will be extended to use the graph transform to encode the feature descriptors along the trajectories in the future.

6. REFERENCES

[1] David G Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[2] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, "Surf: Speeded up robust features," in *Computer vision–ECCV 2006*, pp. 404–417. Springer, 2006.

[3] Navneet Dalal and Bill Triggs, "Histograms of oriented gradients for human detection," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, 2005, vol. 1, pp. 886–893.

[4] Pedro Felzenszwalb, David McAllester, and Deva Ramanan, "A discriminatively trained, multiscale, deformable part model," in *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008, pp. 1–8.

[5] ISO/IEC JTC 1/SC29, "Information technology - multimedia content description interface - part 13: Compact descriptors for visual search," in *Standards*. MPEG, N14956, 2014.

[6] Heng Wang, Dan Oneata, Jakob Verbeek, and Cordelia Schmid, "A robust and efficient video representation for action recognition," *International Journal of Computer Vision*, pp. 1–20, 2015.

[7] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun, "Vision meets robotics: The KITTI dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[8] Gabriel J Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla, "Segmentation and recognition using structure from motion point clouds," in *Computer Vision–ECCV 2008*, pp. 44–57. Springer, 2008.

[9] Sylvie Jeannin and Ajay Divakaran, "MPEG-7 visual motion descriptors," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 6, pp. 720–724, Jun 2001.

[10] Zhangshuai Huang, Ling-Yu Duan, Jie Lin, Shiqi Wang, Siwei Ma, and Tiejun Huang, "An efficient coding framework for compact descriptors extracted from video sequence," in *Image Processing (ICIP), 2015 IEEE International Conference on*, Sept 2015, pp. 3822–3826.

[11] Mina Makar, Vijay Chandrasekhar, Shauhyuarn Sean Tsai, David Chen, and Bernd Girod, "Interframe coding of feature descriptors for mobile augmented reality," *Image Processing, IEEE Transactions on*, vol. 23, no. 8, pp. 3352–3367, 2014.

[12] Hassan Mansour, Shantanu Rane, Petros T Boufounos, and Anthony Vetro, "Video querying via compact descriptors of visually salient objects," in *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE, 2014, pp. 2789–2793.

[13] Dong Tian, Huifang Sun, and Anthony Vetro, "Keypoint trajectory coding on compact descriptor for video analysis," in *Image Processing (ICIP), 2016 IEEE International Conference on*. IEEE, 2016, p. accepted.

[14] D.I. Shuman, S.K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *Signal Processing Magazine, IEEE*, vol. 30, no. 3, pp. 83–98, 2013.

[15] Cha Zhang and Dinei Florêncio, "Analyzing the optimality of predictive transform coding using graph-based models," *IEEE Signal Processing Letters*, vol. 20, no. 1, pp. 106–109, 2013.

[16] Roberto Tron and Rene Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," in *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, June 2007, pp. 1–8.

[17] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 573–580.