LBP-TOP: a Tensor Unfolding Revisit

Xiaopeng Hong*, Yingyue Xu*, Guoying Zhao

Center for Machine Vision and Signal Analysis, University of Oulu

Abstract. Local Binary Pattern histograms from Three Orthogonal Planes (LBP-TOP) has shown its promising performance on facial expression recognition as well as human activity analysis, as it extracts features from spatial-temporal information. Originally, as the calculation of LBP-TOP has to traverse all the pixels in the three dimensional space to compute the LBP operation along XY, YT and XT planes respectively, the frequent use of loops in implementation shapely increases the computational costs. In this work, we aim to fasten the computational efficiency of LBP-TOP on spatial-temporal information and introduce the concept of tensor unfolding to accelerate the implementation process from three-dimensional space to two-dimensional space. The spatial-temporal information is interpreted as a 3-order tensor, and we use tensor unfolding method to compute three concatenated big matrices in two-dimensional space. LBP operation is then performed on the three unfolded matrices. As the demand for loops in implementation is largely down, the computational cost is substantially reduced. We compared the computational time of the original LBP-TOP implementation to that of our fast LBP-TOP implementation on both synthetic and real data, the results show that the fast LBP-TOP implementation is much more time-saving than the original one. The implementation code of the proposed fast LBP-TOP is now publicly available[1].

1 Introduction

Feature extraction, for a long time, plays an important role in image processing and pattern recognition. Specifically, feature extraction constructs dimensionality-reduced values from the large amount of original data to describe the statistical characters, and facilitates recognition tasks such as facial expression recognition, object detection, texture classification, *etc.* There are a variety of methods for feature description, including geometric feature-based methods, shape-based methods and appearance-based methods [1], among which appearance-based methods are the most widely used ones. Appearance-based methods extract the image features in spatial domain and learn the feature extraction scheme based on the relationship between the components or points within the space [2].

* These two authors contributed equally.

[1] The implementation code of the proposed fast LBP-TOP can be downloaded at http://www.ee.oulu.fi/research/imag/cmvs/files/code/Fast_LBPTOP_Code.zip

Local Binary Pattern (LBP) [1] is one of the most popular and efficient appearance-based feature descriptors [3, 4]. It has proven to be highly discriminative and its invariance to monotonic gray-level discrepancies makes it a robust feature descriptor in two-dimensional space. In general, an LBP operator measures each pixel of a given image by thresholding its neighborhood with the value of the center pixel and forms the results into a binary pattern. Then, the occurrence histogram based on the resulted binary patterns can be computed over an image or a region of the image, which is proven to be a powerful feature descriptor. Now, due to its discriminative power and computational simplicity, LBP is broadly utilized in image pattern recognition. Moreover, it receives tremendous success in facial expression recognition as it is insensitive to illumination variations and well describes subtle appearance details of the local features on human faces [5].

As LBP is proven to be of high performance as well as low computational cost, it is frequently applied in pattern recognition on static images. Following the steps of LBP, Zhao *et al.* [6] started to explore appearance-based feature descriptors on dynamic or temporal information, which combines appearance and motion. Hence, Local Binary Pattern histograms from Three Orthogonal Planes (LBP-TOP) was proposed. LBP-TOP is an extension of LBP from two-dimensional space to three-dimensional space including spatial and time domain. More specifically, LBP-TOP regards the pixel in a three dimensional space with spatial and temporal properties and computes the LBP of each pixel on three orthogonal planes, and finally formulates three occurrence histograms corresponding to the three orthogonal planes. LBP-TOP not only inherits merits from LBP which is insensitive to illumination variations, translations or rotations, but also extends its applications to high dimensional video feature analysis. Moreover, LBP-TOP has shown its promising performance on facial expression recognition as well as human activity analysis, as it is capable to analyse appearance changes from a sequence of images [7–11].

However, as LBP-TOP operates on three dimensional spatial-temporal information, the computational cost also sharply increases compared to LBP on two dimensional static images. Suppose there is an image with frame width W and height H, the total number of LBP operations over all the pixels on the image is $(W \times H)$ times. Now we have another sequence of dynamic images with the same image size of width W and height H and the number of frames T, LBP-TOP needs to be applied to this three dimensional matrix. Then the total number of LBP operations over all the pixels climbs to $(3 \times W \times H \times T)$ times. Moreover, as LBP-TOP traverses all the pixels to compute LBP operations on XY, YT and XT planes respectively, it results in a frequent usage of nested loops in implementation. The use of complicated nested loops in implementation already heavily affects the computational time of LBP-TOP, let along the steep rise in computational costs, especially when we need to analyse a long sequence of images.

In this paper, we aim to fasten the computational efficiency of LBP-TOP for feature extraction on spatial-temporal information and introduce the con-

cept of tensor unfolding to accelerate the implementation process from three-dimensional space to two-dimensional space. Therefore, we propose a fast LBP-TOP implementation method which unfolds the 3-order tensor of spatial-temporal information to 2-order tensors and performs LBP operations over the 2-order tensors. The proposed fast LBP-TOP implementation method benefits from optimized codes with reduced nested loops and largely saves the computational costs compared to the original implementation.

The contribution of this paper are two folds:

1. We propose a fast LBP-TOP implementation method that takes the advantage of tensor unfolding method to simplify the LBP-TOP implementation from a three-dimensional space to a two-dimensional space. The tensor unfolding method largely reduces the demand of loops in codes, such that the computational cost is substantially reduced.

2. We perform experimental comparisons of the proposed fast LBP-TOP implementation method to the original one. The results show significant improvements in terms of computational cost.

2 Related Work

2.1 Local Binary Patterns

The local binary patterns (LBP) has proven to be a simple yet very efficient operator for feature description. The LBP operator is derived from a general definition of texture in a local neighborhood, and was firstly proposed by Ojala *el al.* in 1996 [12]. It can be regarded as a unifying methodology other than traditionally divergent statistical and structural models of texture analysis. Now it is broadly utilized in fields such as face expression recognition and analysis [13, 14], biomedical image processing [15] and texture analysis [16].

Given a texture or an image, suppose that we computes the LBP operation over all the pixels. For each pixel, a certain range of its neighborhood is pre-defined for the LBP operator. In this work, P counts for the number of neighboring pixels around the central one, while R refers to the radius of a circle with the P equally spaced neighbors surrounding the central pixel. Figure 1 presents the distributions of the neighborhood of a given pixel with different settings of P and R.

Now, we perform a basic LBP operation over a pixel with its neighborhood setting as $P = 8$ and $R = 1$ as is shown in the example in Figure 2. Then, we assume that the intensity value of the center pixel is g_c, while the intensity values of its circular neighborhood are $g_p(p = 0, \ldots, P-1)$. Here we threshold the neighbourhood by the intensity value of its center, namely

$$B = b(s(g_0 - g_c), s(g_1 - g_c), \ldots, s(g_{P-1} - g_c)), \tag{1}$$

where

$$s(n) = \begin{cases} 1, & n \geq 0 \\ 0, & n < 0 \end{cases} \tag{2}$$

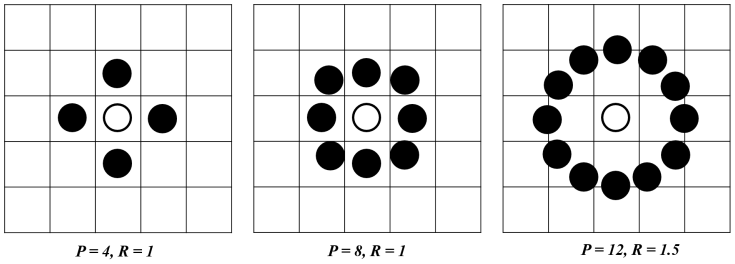


Fig. 1. Examples of central pixel with different settings of circular neighborhoods. Each grid represents a pixel and the pixel values are bilinearly interpolated if the sampling point is not right in the center of a pixel. The white points represent the central pixel and the black points refer to its neighboring pixels being selected.

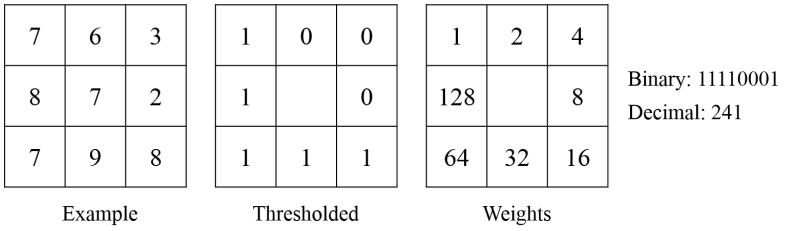


Fig. 2. LBP operation with neighborhood settings as $P = 8$ and $R = 1$. The example block is shown on the left with intensity values and the neighborhood points are thresholded by the central pixel. Then the results are formulated into a bit-wise binary pattern, and the corresponding weights are used to compute the decimal value of the local binary pattern.

After thresholding the neighborhood using Equation 1 and Equation 2, we formulate the results into a bit-wise binary pattern as shown in Figure 2. Thus, this $(P = 8, R = 1)$ circular neighborhood results in a final binary pattern as 11110001. Then we calculate the decimal value of the local binary pattern as follow:

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_c)2^p \tag{3}$$

After computing all the LBP decimal values over all the pixels on the image, we finally construct the occurrence histogram with all the results. The computed LBP histogram adopts uniform patterns so that the histogram has a separate bin for every uniform pattern and all nonuniform patterns are assigned to a single bin.

2.2 Local Binary Patterns from Three Orthogonal Planes

Following the steps of LBP on static image analysis, Zhao *et al.* [6] proposed the Local Binary Patterns from Three Orthogonal Planes (LBP-TOP) to analyse videos with motions.

The spatial-temporal information can be regarded as a set of volumes in the (X, Y, T) space, where X and Y represent the spatial coordinates, while T denotes the frame index (time) in temporal domain. Hence, the neighborhood of each pixel no longer fall in a two dimensional space, where LBP operation can be used to extract features into histograms. Instead, we need to compute feature descriptor in the three dimensional space (X, Y, T). Thus, LBP-TOP is proposed to describe the spatial-temporal information in the three dimensional space.

Similar to LBP, LBP-TOP also computes the local binary patterns of a center pixel by thresholding its neighborhood. However, as the spatial-temporal information falls in a three dimensional space, LBP-TOP decomposes the three dimensional volume into three orthogonal planes: XY, XT and YT as is shown in Figure 3. The XY plane represents the appearance feature in spatial domain, while the XT plane describes a visual impression of one row changing with time and YT captures the features of motion for one column in temporal space. Then, LBP values are extracted for all pixels from the XY, XT and YT planes, denoted as XY-LBP, XT-LBP and YT-LBP. In such a representation, a sequence of images are encoded by the appearance (XY-LBP) and two spatial temporal (XT-LBP and YT-LBP) co-occurrence statistics. Finally, the LBP-TOP is computed by concatenating the histograms from all the three orthogonal planes including XY, XT and YT into a single histogram.

Note that when computing LBP-TOP over the spatial-temporal information, we also define a neighborhood in the three dimensional space for LBP coding. R_X, R_Y and R_T denote the radius of the sampling points surrounding the cental pixel along X, Y and T direction respectively. And P_{XY}, P_{XT} and P_{YT} count for the number of neighborhood points sampled on XY plane, XT plane and YT plane respectively. In the experiments in Section 3, we will use these denotations to differentiate various parameter settings of the defined neighborhood for LBP-TOP implementation.

2.3 Revisit LBP-TOP from a Tensor Point of View

A tensor refers to a multi-dimensional matrix, or array of numbers. The order of a tensor is defined by the number of dimensionality of the matrix required to describe the data [17, 18]. That is, an one-dimensional array is an 1-order tensor, a two-dimensional matrix is a 2-order tensor and so forth.

Given a sequence of images with spatial-temporal information, they are usually regarded as a three-dimensional matrix, of which two coordinates label the spatial information and the third represents the time span. We can also consider the three-dimensional matrix as a 3-order tensor, of which the components include spatial information as well as temporal information. This facilitates

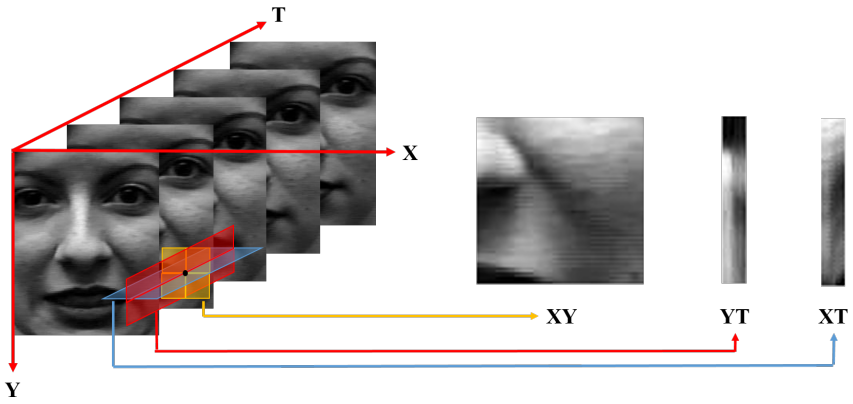


Fig. 3. Visualization of XY, XT and YT orthogonal planes on the spatial-temporal information with volume $183 \times 229 \times 11$ based on LBP-TOP algorithm. The image in XY plane is a 70×70 clip, the image in XT plane is a 11×70 clip when $y = 140$, and the image in YT plane is a 11×70 clip when $x = 140$ (the starting point is on the left-top of the image).

spatial-temporal information analysis from a tensor point of view. Many previous works achieve video analysis with tensor theory, for instance, Wang *et al.* [19, 20] proposed a tensor independent color space to analyze micro-expression recognition on spatial-temporal information over different color channels, while Kim *et al.* [21] proposed tensor canonical correlation analysis for action classification.

Given an N-order tensor $\mathcal{T} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the element inside $\mathcal{T}$ is represented as $\tau_{i_1 i_2 \cdots i_N}$, where $i_1, i_2, \cdots, i_N$ denote the coordinate positions. In practice, the spatial-temporal information are usually a video with a sequence of frames. Thus, we denote the video as a 3-order tensor $\mathcal{T}$, which is a $I_1 \times I_2 \times I_3$ matrix. I_1 is the height of the frame, I_2 refers to the width of the frame, and I_3 denotes the number of frames in temporal domain. Then, LBP-TOP traverses all the elements $\tau_{i_1 i_2 i_3}$ inside the tensor $\mathcal{T}$ to compute LBP on the three orthogonal planes $I_1 I_2$, $I_2 I_3$ and $I_1 I_3$ respectively.

2.4 Fast LBP-TOP Implementation Based on Tensor Unfolding

Given an N-order tensor $T \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the tensor unfolding [22–24] $\mathcal{T}_n \in \mathbb{R}^{I_n \times (I_1 I_2 \cdots I_{n-1} I_{n+1} I_{n+2} \cdots I_N)}$ contains the element $\tau_{i_1 i_2 \cdots i_n i_{n+1} \cdots i_N}$ at the position with row number i_n and column number that is equal to $[(i_{n+1} - 1) I_{n+2} \cdots I_P I_1 \cdots I_{n-1}] + [(i_{n+2} - 1) I_{n+3} \cdots I_P I_1 \cdots I_{n-1}] + \cdots + [(i_2 - 1) I_3 I_4 \cdots I_{n-1}] + \cdots + i_{n-1}$. Figure 4 visualizes the unfolded results of the 3-order tensor. Tensor unfolding is one of the simplest way to reduce the number of dimensions of a matrix.

Then we apply tensor unfolding on the 3-order matrix, and receive three 2-order unfolded tensors as visualized in Figure 4.

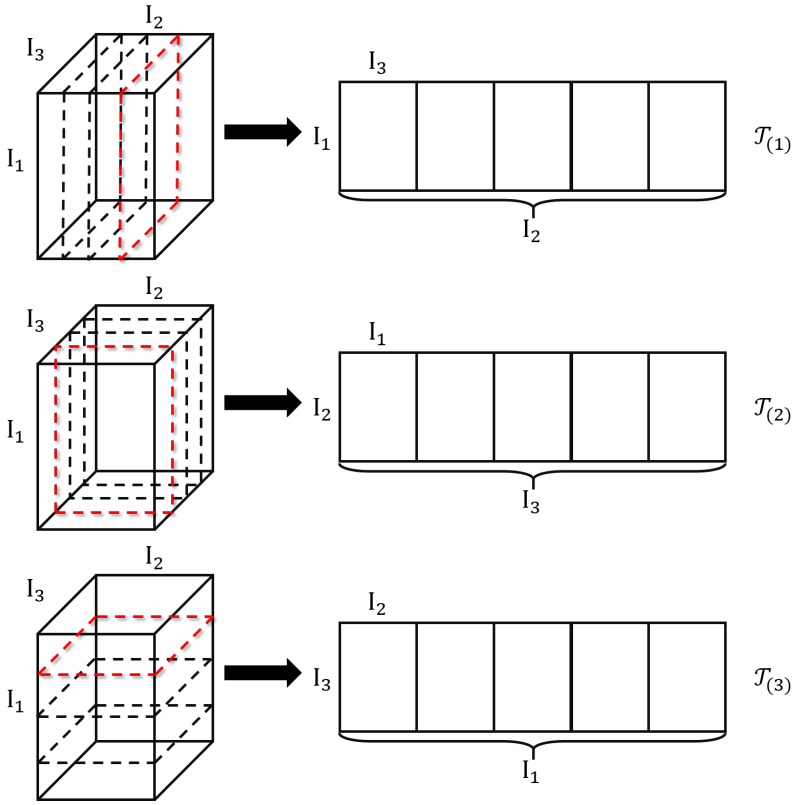


Fig. 4. Visualization of the three unfoldings of a 3-order tensor.

Apparently, the unfolded 2-order tensor $\mathcal{T}_{(1)}$ concatenates all the YT planes into a large two-dimensional matrix. Similarly, the tensor $\mathcal{T}_{(2)}$ connects all the XY planes and the tensor $\mathcal{T}_{(3)}$ concatenates all the XT planes.

Originally, LBP-TOP has to traverse all the pixels in the 3-order tensor to compute the local binary patterns along XY, YT and XT planes respectively. The huge demand of nested looping in implementation sharply increases the computation costs. In this work, as we unfold the 3-order tensor to three two-dimensional concatenated matrices, we can effectively accelerate the computation process by optimizing the codes with vectorization. Then, we reformulate the unfolded tensors back to 3-order tensor and computes occurrence histograms for each orthogonal planes. The pseudo code of the original LBP-TOP implementation is illustrated in Algorithm 1, while that of the proposed fast LBP-TOP implementation is shown in Algorithm 2. Apparently, the proposed fast LBP-TOP implementation largely reduces the usage of nested loops and optimizes the codes through vectorization that uses matrix and vector operations.

Algorithm 1: Original LBP-TOP implementation.

Data: Video data V, where $[H, W, T] = size(V)$. R_X, R_Y and R_T are radius of neighborhood along X, Y and T direction respectively, and P_{XY}, P_{XT} and P_{YT} are numbers of neighborhood points on XY, XT and YT plane respectively.
Result: LBP occurrence histograms on XY, XT and YT plane, namely $HIST_{XY}$, $HIST_{XT}$, and $HIST_{YT}$ respectively.

```
for i = R_T to T − R_T − 1 do
    for j = −R_T to R_T do
        Frame_{H×W×(j+R_T+1)} = V_{H×W×t}, where t = i + j + 1;
    end
    for yc = 1 to H do
        for xc = 1 to W do
            CenterVar = Frame(yc, xc, R_T);
            // Compute LBP on XY plane for center pixel with neighborhood
               (P_XY, R_X, R_Y).
            LBP_CenterVar =LBP_PIXEL(CenterVar, P_XY, R_X, R_Y);
            Update HIST_XY with the value of LBP_CenterVar;
            // Compute LBP on XT plane for center pixel with neighborhood
               (P_XT, R_X, R_T).
            LBP_CenterVar =LBP_PIXEL(CenterVar, P_XY, R_X, R_T);
            Update HIST_XT with the value of LBP_CenterVar;
            // Compute LBP on YT plane for center pixel with neighborhood
               (P_YT, R_Y, R_T).
            LBP_CenterVar =LBP_PIXEL(CenterVar, P_XY, R_Y, R_T);
            Update HIST_YT with the value of LBP_CenterVar;
        end
    end
end
```

3 Experiments

Usually, LBP-TOP is performed over the spatial-temporal information for feature extraction and the extracted features are utilized for further pattern recognition or classification. In most of the cases, the extracted LBP-TOP features facilitate data training process for pattern recognition. Thus, in the experiment, we imitate the feature extraction process for data training to evaluate the performance of the original LBP-TOP implementation and the proposed fast LBP-TOP implementation. More specifically, we extract LBP-TOP features on spatial-temporal information with the same extraction strategy but different implementations (original LBP-TOP and fast LBP-TOP), to evaluate the performance improvements of the proposed fast LBP-TOP implementation. We implement the fast LBP-TOP on Matlab R2014b. The original implementation of LBP-TOP is provided by the corresponding authors in Matlab version[2].

We compared the computational time of the original LBP-TOP implementation and our fast LBP-TOP implementation on both synthetic and real data. Firstly, we compared the computational time of the original LBP-TOP implementation to our fast LBP-TOP implementation on synthetic data. Firstly, we evaluate the computational time by averaging 50 randomly simulated video samples with varying settings of the neighborhood, as is shown in Table 1. Apparently, the proposed fast LBP-TOP implementation effectively improves the

[2] The original implementation code of LBP-TOP method can be downloaded at http://www.cse.oulu.fi/CMV/Downloads/LBPMatlab

Algorithm 2: Fast LBP-TOP implementation.

Data: Video data V, where $[H, W, T] = size(V)$. R_X, R_Y and R_T are radius of neighborhood along X, Y and T direction respectively, and P_{XY}, P_{XT} and P_{YT} are numbers of neighborhood points on XY, XT and YT plane respectively.
Result: LBP occurrence histograms on XY, XT and YT plane, namely $HIST_{XY}$, $HIST_{XT}$, and $HIST_{YT}$ respectively.

```
// Simple zero padding is used but not limited in this pseudo code.
// 1. Operations on XY plane.
Plane_XY =zeros (H, 2 × R_T + W × R);
// Unfold the 3-order tensor to 2-order tensor.
Plane_XY(:, R_T + 1 : end − R_T) = Unfold (V, [H, W × T]);
/* Unfolding can be done through permute and reshape in Matlab for example.   */
// Compute LBP on XY plane with neighborhood (P_XY, R_X, R_Y).
LBP_XY =LBP_PLANE(Plane_XY, P_XY, R_X, R_Y);
/* LBP on all the pixels on a 2D plane is implemented through vectorization instead of
   loops.                                                                     */
// Reformulate the 2-order tensor to 3-order tensor.
LBP_XY =Reformulate(LBP_XY, [H − 2 × R_Y, W, T]);
HIST_XY =Histogram(LBP_XY);
// 2. Operations on YT plane.
Plane_YT =zeros (T, 2 × R_X + W × H);
Plane_YT(:, R_X + 1 : end − R_X) = Unfold (V, [T, W × H]);
// Compute LBP on YT plane with neighborhood (P_YT, R_Y, R_T).
LBP_YT =LBP_PLANE(Plane_YT, P_YT, R_Y, R_T);
LBP_YT =Reformulate(LBP_YT, [T − 2 × R_T, H, W]);
HIST_YT =Histogram(LBP_YT);
// 3. Operations on XT plane.
Plane_XT =zeros (W, 2 × R_H + W × T);
Plane_XT(:, R_H + 1 : end − R_H) = Unfold (V, [W, T × H]);
// Compute LBP on XT plane with neighborhood (P_XT, R_X, R_T).
LBP_XT =LBP_PLANE(Plane_XT, P_XT, R_X, R_T);
LBP_XT =Reformulate(LBP_XT, [T − 2 × R_T, W, H]);
HIST_XT =Histogram(LBP_XT);
```

computational costs, especially when the sizes of the videos are large or the neighborhood settings are complicated. Further, we randomly formulate video clips of different frame size and time length to evaluate the computational cost of the original LBP-TOP implementation and the proposed fast LBP-TOP implementation. The results are illustrated in Figure 5. From Figure 5(a), it is easy to perceive that as the frame size increases, the computational time of the original LBP-TOP implementation sharply increases while that of the fast LBP-TOP implementation has no remarkable change. Similarly, in Figure 5(b), when the length of the video clip increases, our fast LBP-TOP implementation reveals significant advances on computational time than the original LBP-TOP implementation. Hence, the fast LBP-TOP implementation is proven to be much more time-saving than the original LBP-TOP implementation in all cases especially for spatial-temporal information of high volume.

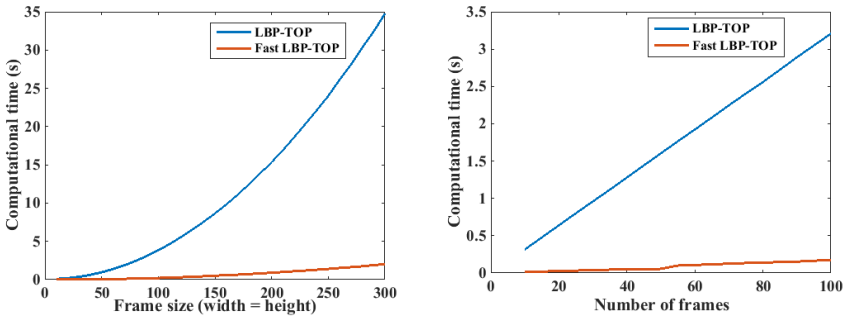
Then, we evaluate the computational time on real data based on three databases: CASME II [25], SMIC [26] and Cohn-Kanade [27]. All of the three databases are video clips with spatial-temporal information. CASME II is a micro-expression database of micro facial movements with high temporal and spatial resolution. SMIC is a spontaneous micro-expression database for analyzing people's deceitful behaviors. The Cohn-Kanade AU-Coded Facial Expression

Table 1. Average computational time of the original LBP-TOP implementation and the fast LBP-TOP (FLBP-TOP) on 50 randomly simulated video samples. The numbers of sampled points of the neighborhood P_{XY}, P_{YT} and P_{XT} are set equal (denoted as P), and the radius R_X ,R_Y and R_T are set equal as well (denoted as R). 'Inc.' refers to the increasing rate of computational time from LBP-TOP to FLBP-TOP.

W	H	T	P	R	LBP-TOP(s)	FLBP-TOP(s)	Inc.
64	64	30	8	1	1.58	0.06	26.85
64	64	30	8	2	1.38	0.05	26.59
64	64	30	8	3	1.19	0.05	22.56
64	64	30	16	2	7.66	0.10	78.72
64	64	30	16	3	7.43	0.10	72.69
64	64	60	8	1	3.17	0.18	17.59
64	64	60	8	2	2.88	0.17	16.91
64	64	60	8	3	2.60	0.17	15.71
64	64	60	16	2	9.65	0.38	25.36
64	64	60	16	3	9.28	0.39	23.57
64	64	180	8	1	9.59	0.56	17.21
64	64	180	8	2	8.92	0.56	15.94
64	64	180	8	3	8.23	0.54	15.31
64	64	180	16	2	17.42	1.21	14.42
64	64	180	16	3	16.61	1.21	13.75
128	128	30	8	1	6.25	0.36	17.41
128	128	30	8	2	5.69	0.35	16.15
128	128	30	8	3	5.05	0.32	15.92
128	128	30	16	2	13.21	0.77	17.19
128	128	30	16	3	12.45	0.69	18.14
128	128	60	8	1	12.88	0.74	17.43
128	128	60	8	2	12.09	0.71	17.07
128	128	60	8	3	11.31	0.72	15.67
128	128	60	16	2	21.66	1.54	14.07
128	128	60	16	3	20.77	1.53	13.56
128	128	180	8	1	39.71	2.26	17.60
128	128	180	8	2	37.64	2.18	17.29
128	128	180	8	3	36.15	2.17	16.68
128	128	180	16	2	54.82	4.66	11.76
128	128	180	16	3	52.93	4.63	11.43
256	256	500	8	1	455.80	29.71	15.34
256	256	500	8	2	448.61	27.09	16.56
256	256	500	8	3	438.66	25.74	17.04
256	256	500	16	2	586.64	55.33	10.60
256	256	500	16	3	578.37	54.78	10.56

database is for research in automatic normal facial expression analysis and synthesis and for perceptual studies.

On each database, we randomly select 5 patches from each facial expression video clip of each subject, and Table 2 shows the computational time of the original LBP-TOP implementation and the proposed fast LBP-TOP implementation. In the table, we evaluate the performance of the original LBP-TOP implementation versus the proposed fast LBP-TOP implementation on three databases with different settings of frame width and height. As we perform LBP-TOP on all the video clips over each database, the temporal lengths are always the same such that we eliminate the setting of parameter T in the table. The parameters of neighborhood including the number of sampling points P_{XY}, P_{XT} and P_{YT} and radius R_X, R_Y and R_T are set according to Table 2. From Table 2, it is obvious that the fast LBP-TOP implementation dramatically improves the computational time by 31.19 times on average to the original LBP-TOP implementation.



(a) (b)

Fig. 5. Comparisons of computational time of the original LBP-TOP implementation and the fast LBP-TOP implementation. (a) shows the computational time of LBP-TOP and Fast LBP-TOP when frame size increases, of which the x-axis represents the equal lengths of width and height, and the neigborhood settings are $P_{XY} = P_{XT} = P_{YT} = 8$, $R_X = R_Y = R_T = 1$, and the video length $T = 30$. (b) illustrates the computational time of LBP-TOP and Fast LBP-TOP when video length increases, where the neighborhood settings are $P_{XY} = P_{XT} = P_{YT} = 8$, $R_X = R_Y = R_T = 1$ and the frame size is 30×30.

4 Discussions

The proposed fast LBP-TOP implementation takes the advantage of tensor unfolding to reformulate the tree-dimensional matrix to two-dimensional matrices. In future, this tensor unfolding method can be applied to any other descriptors that extract features from spatial-temporal information, such as LGBP-TOP [28] and SIFT-TOP [29]. Moreover, we implement the fast LBP-TOP codes on Matlab as an example. Further, the fast LBP-TOP can be implemented on other platforms that optimize codes through vectorization instead of nested loops, such as Python, Octave, R, *etc.*

5 Conclusions

In this work, we propose a fast LBP-TOP implementation method to fasten the computation efficiency of LBP-TOP for feature extraction on spatial-temporal information. We introduce the concept of tensor unfolding to accelerate the implementation process from three-dimensional space to two-dimensional space. The proposed fast LBP-TOP implementation method benefits from the optimization of codes with less nested loops and largely reduces the computational cost compared to the original implementation. We compare the computational time of the original LBP-TOP implementation and our fast LBP-TOP implementation on both synthetic and real data. The results show that our fast LBP-TOP implementation is quite time-saving than the original one. In future, as parallel

Table 2. Total computational time of the original LBP-TOP implementation and the fast LBP-TOP (FLBP-TOP) implementation on three databases: CASME II [25], SMIC [26] and Cohn-Kanade [27]. As LBP-TOP is performed by selecting 5 patches on every video clip of each database, the temporal lengths are the same for each comparison group such that video lengths T are not listed. The numbers of sampled points of the neighborhood P_{XY}, P_{YT} and P_{XT} are set equal (denoted as P), and the radius R_X ,R_Y and R_T are set equal as well (denoted as R). The column 'Clips' shows the total number of video clips of each database. 'Inc.' refers to the increasing rate of computational time from LBP-TOP to FLBP-TOP.

Database	W	H	Clips	P	R	LBP-TOP(s)	FLBP-TOP(s)	Inc.
CASME II	30	30	1285	4	1	834.14	20.58	20.58
	30	30	1285	4	2	704.42	18.74	18.74
	30	30	1285	8	1	1001.94	38.37	38.37
	30	30	1285	8	2	850.17	35.88	35.88
	50	50	1285	4	1	2357.59	59.02	59.02
	50	50	1285	4	2	2130.38	55.33	55.33
	50	50	1285	8	1	2805.47	138.59	138.59
	50	50	1285	8	2	2495.50	128.60	128.60
Cohn-Kanade	30	30	2415	4	1	462.69	12.71	12.71
	30	30	2415	4	2	367.43	11.07	11.07
	30	30	2415	8	1	569.51	25.79	25.79
	30	30	2415	8	2	465.13	22.53	22.53
	50	50	2415	4	1	1174.95	28.14	28.14
	50	50	2415	4	2	970.90	25.08	25.08
	50	50	2415	8	1	1402.63	54.44	54.44
	50	50	2415	8	2	1154.99	48.33	48.33
SMIC	30	30	1640	4	1	527.54	13.71	13.71
	30	30	1640	4	2	434.42	11.82	11.82
	30	30	1640	8	1	638.04	25.77	25.77
	30	30	1640	8	2	533.85	23.56	23.56
	50	50	1640	4	1	1429.49	32.33	32.33
	50	50	1640	4	2	1244.76	30.48	30.48
	50	50	1640	8	1	1705.84	62.66	62.66
	50	50	1640	8	2	1471.51	58.51	58.51

computing can be performed on two dimensional matrices, the computational time of the fast LBP-TOP implementation can be further saved.

Acknowledgement. This work is sponsored by the Academy of Finland, Infotech Oulu, the post-doc fellow postion of Infotech Oulu, and Tekes Fidipro Program. Moreover, Xiaopeng Hong is partly supported by the Natural Science Foundation of China under the contract No. 61572205.

References

1. Wolf, L.: Face recognition, geometric vs. appearance-based. Encyclopedia of Biometrics (2015) 495–500
2. Gross, R., Matthews, I., Baker, S.: Appearance-based face recognition and light-fields. IEEE Transactions on Pattern Analysis and Machine Intelligence **26** (2004) 449–465
3. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on pattern analysis and machine intelligence **24** (2002) 971–987
4. Ojala, T., Pietikäinen, M., Mäenpää, T.: Gray scale and rotation invariant texture classification with local binary patterns. In: European Conference on Computer Vision, Springer (2000) 404–420

5. Huang, X., Wang, S.J., Zhao, G., Piteikainen, M.: Facial micro-expression recognition using spatiotemporal local binary pattern with integral projection. In: Proceedings of the IEEE International Conference on Computer Vision Workshops. (2015) 1–9
6. Zhao, G., Pietikainen, M.: Dynamic texture recognition using local binary patterns with an application to facial expressions. IEEE transactions on pattern analysis and machine intelligence **29** (2007) 915–928
7. de Freitas Pereira, T., Anjos, A., De Martino, J.M., Marcel, S.: Lbp- top based countermeasure against face spoofing attacks. In: Asian Conference on Computer Vision, Springer (2012) 121–132
8. Kellokumpu, V., Zhao, G., Pietikäinen, M.: Human activity recognition using a dynamic texture based method. In: BMVC. Volume 1. (2008) 2
9. Wang, Y., Yu, H., Stevens, B., Liu, H.: Dynamic facial expression recognition using local patch and lbp-top. In: 2015 8th International Conference on Human System Interaction (HSI), IEEE (2015) 362–367
10. Chen, Y., Guo, X., Klein, D.: Orthogonal combination of local binary patterns for dynamic texture recognition. In: Ninth International Symposium on Multispectral Image Processing and Pattern Recognition (MIPPR2015), International Society for Optics and Photonics (2015) 98130R–98130R
11. Qi, X., Li, C.G., Zhao, G., Hong, X., Pietikäinen, M.: Dynamic texture and scene classification by transferring deep image features. Neurocomputing **171** (2016) 1230–1241
12. Ojala, T., Pietikäinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern recognition **29** (1996) 51–59
13. Ahonen, T., Hadid, A., Pietikainen, M.: Face description with local binary patterns: Application to face recognition. IEEE transactions on pattern analysis and machine intelligence **28** (2006) 2037–2041
14. Ahonen, T., Hadid, A., Pietikäinen, M.: Face recognition with local binary patterns. In: European conference on computer vision, Springer (2004) 469–481
15. Murala, S., Wu, Q.J.: Local mesh patterns versus local binary patterns: biomedical image indexing and retrieval. IEEE journal of biomedical and health informatics **18** (2014) 929–938
16. Mäenpää, T.: The local binary pattern approach to texture analysis: extensions and applications. Oulun yliopisto (2003)
17. Kolecki, J.C.: An introduction to tensors for students of physics and engineering. (2002)
18. Kolda, T.G., Bader, B.W.: Tensor decompositions and applications. SIAM review **51** (2009) 455–500
19. Wang, S., Yan, W.J., Li, X., Zhao, G., Fu, X.: Micro-expression recognition using dynamic textures on tensor independent color space. In: ICPR, Citeseer (2014) 4678–4683
20. Wang, S.J., Yan, W.J., Li, X., Zhao, G., Zhou, C.G., Fu, X., Yang, M., Tao, J.: Micro-expression recognition using color spaces. IEEE Transactions on Image Processing **24** (2015) 6034–6047
21. Kim, T.K., Wong, S.F., Cipolla, R.: Tensor canonical correlation analysis for action classification. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition, IEEE (2007) 1–8
22. Meyer, C.D.: Matrix analysis and applied linear algebra. Volume 2. Siam (2000)

23. Kuang, L., Hao, F., Yang, L.T., Lin, M., Luo, C., Min, G.: A tensor-based approach for big data representation and dimensionality reduction. IEEE Transactions on emerging Topics in Computing **2** (2014) 280–291
24. Manolopoulos, P.S.A.N.Y.: Tag recommendations based on tensor dimensionality reduction. (2008)
25. Yan, W.J., Li, X., Wang, S.J., Zhao, G., Liu, Y.J., Chen, Y.H., Fu, X.: Casme ii: An improved spontaneous micro-expression database and the baseline evaluation. PloS one **9** (2014) e86041
26. Li, X., Pfister, T., Huang, X., Zhao, G., Pietikäinen, M.: A spontaneous micro-expression database: Inducement, collection and baseline. In: Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on, IEEE (2013) 1–6
27. Tian, Y.I., Kanade, T., Cohn, J.F.: Recognizing action units for facial expression analysis. IEEE Transactions on pattern analysis and machine intelligence **23** (2001) 97–115
28. Almaev, T.R., Valstar, M.F.: Local gabor binary patterns from three orthogonal planes for automatic facial expression recognition. In: Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on, IEEE (2013) 356–361
29. Sun, B., Li, L., Zhou, G., He, J.: Facial expression recognition in the wild based on multimodal texture features. Journal of Electronic Imaging **25** (2016) 061407–061407