Fox's Video Library (Version 1.0)

Ludmila I Kuncheva

June 8, 2018

1 Introduction

This library was developed in the process of preparing several publications on video processing and summarisation.

2 Feature extraction

```
function x = fox_get_features(im,fstr,blocks,bins)
%=========================================================================
% (c) Fox's Video Toolbox                                          ^--^
% 01.11.2017 ------------------------------------------------------  \oo/
%                                                                  -\/-%
% Input
%
% im: RGB image                    = input image
% fstr: RGB, HSV, CHR, OHT, H      = feature string
% blocks: 9, 4, 1                  = number of blocks
% bins: e.g., 8, 16, 32 (or any)   = number of bins for histogram features
%
% Output
%
% x: vector-row with the extracted features
%=========================================================================
```

The following feature spaces are available from this function:

1. *RGB moments.* (`fstr = 'RBG'`) The function returns the mean and standard deviation of the red (R), green (G) and blue (B) channels for each the image ($n = 6$ features).
2. *HSV space.* (`fstr = 'HSV'`) The function returns the mean and standard deviations of the three HSV components ($n = 6$ features).
3. *Chrominance.* (`fstr = 'CHR'`) The function returns the mean and standard deviation of Chrominance components C_1 and C_2 ($n = 4$ features) calculated as:

$$C_1 = \frac{R}{q}, \;\; C_2 = \frac{G}{q}, \;\; q = \sqrt{R^2 + G^2 + B^2},$$

4. *Ohta space.* (`fstr = 'OHT'`) The function returns the mean and standard deviation of features I_1, I_2' and I_3' of Ohta space ($n = 6$ features) calculated as

$$\begin{aligned} I_1 &= \frac{1}{3}(R+G+B) \\ I_2' &= R-B \\ I_3' &= \frac{1}{2}(2G-R-B) \end{aligned}$$

The image can be processed as a whole (`'blocks' = 1`) or split into $K \times K$ uniformly sized cells (`'blocks' = K^2`). The number of elements of `x` is $K^2 \times n$, where n if the number of features for a single image. So, for $K = 3$ (`'blocks' = 9`), and feature space RGB (`fstr = 'RBG'`) the total number of features is $9 \times 6 = 54$.

The returned vector `x` contains the means followed by the standard deviations. For example, a 4-block RGB feature vector will be:

$$\begin{aligned} \mathbf{x} = [&m_{r1}, m_{g1}, m_{b1}, s_{r1}, s_{g1}, s_{b1}, m_{r2}, m_{g2}, m_{b2}, s_{r2}, s_{g2}, s_{b2}, ... \\ &m_{r3}, m_{g3}, m_{b3}, s_{r3}, s_{g3}, s_{b3}, m_{r4}, m_{g4}, m_{b4}, s_{r4}, s_{g4}, s_{b4}], \end{aligned}$$

where m denotes mean, s denotes standard deviation, rgb stand for red, green, blue, and the number tag is the block number.

For feature string 'H', the function will return a hue value histogram in the specified number of bins. The bins span the whole range of possible hue values. Thus, a solid red image will be represented with a histogram containing 1 in the first bin (after scaling to sum 1), and zeros elsewhere.

`FoxsVideoToolboxTester_Features` demonstrates feature extraction through function `fox_get_features`. The code uploads an image, extracts RGB features using 9 blocks (3 by 3), and plots the colours of the respective blocks as in Figure 1. The numbering of the cells is also shown. Next, a 16-bin histogram of the hue values is extracted and shown in the bottom right sub-figure.

3 Frame matching

```
function [match, value] = fox_match_two_vectors(a,b,threshold,d)
%=========================================================================
% (c) Fox's Video Toolbox                                          ^--^
% 08.06.2018 ------------------------------------------------------  \oo/
%                                                                  -\/-%
% Input
%
% a,b: vectors 1-by-n = image *representations* to match
% threshold: match is declared if distance is <= threshold
% d: 1 Euclidean, 2 Minkowski (L1 norm), 3 Cosine = distance type
%
% Output
%
% match: 0 no match, 1 match (logical variable)
% value: distance
%=========================================================================
```

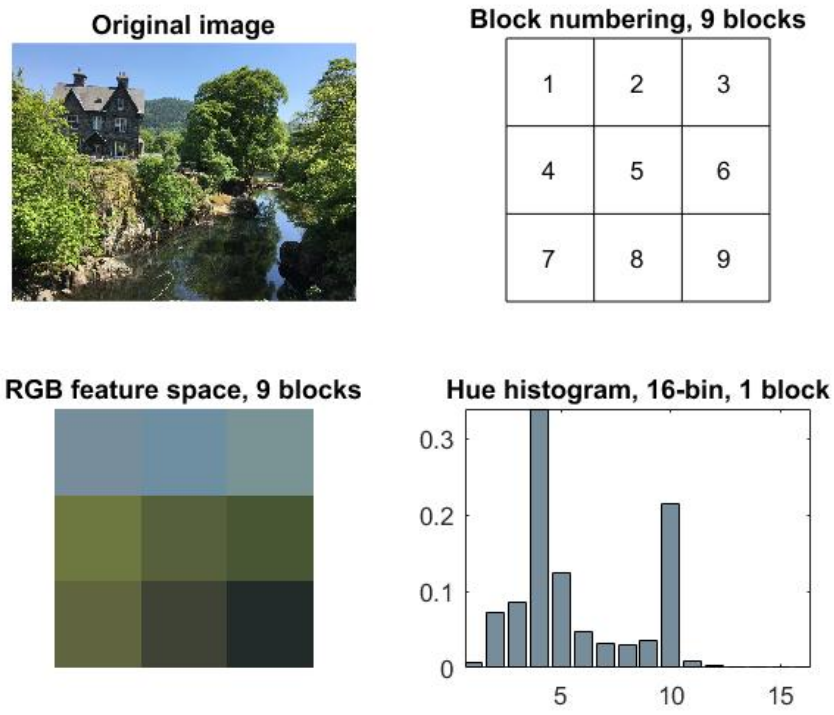


Figure 1: Example of feature extraction.

Function `fox_match_two_vectors` matches two frames, $\mathbf{a} = [a_1, \ldots a_n]^T$ and $\mathbf{b} = [b_1, \ldots b_n]^T$, represented as vectors in some n-dimensional feature space. The two vectors must have the same number of elements. Each vector can be either a row or a column. The following distances are included:

- (1) Euclidean

$$v = \sqrt{\sum_{i=1}^{n} (a_i - b_i)^2},$$

- (2) Minkowski

$$v = \sum_{i=1}^{n} |a_i - b_i|,$$

- (3) Cosine (angular distance)

$$v = \frac{2}{\pi} \cos^{-1}\left(\frac{\mathbf{a}^T\mathbf{b}}{||\mathbf{a}||\,||\mathbf{b}||}\right).$$

Function `fox_match_two_frames_surf` matches two frames, A and B, represented as images. SURF descriptors are used.

```
function [match, value] = fox_match_two_frames_surf(A,B,threshold,vis)
%==========================================================================
% (c) Fox's Video Toolbox                                            ^--^
% 08.06.2018 ------------------------------------------------------  \oo/
%                                                                    -\/-%
% Input
```

```
%
% A,B: images to match
% threshold: match is declared if distance is <= threshold
% d: 1 Euclidean, 2 Minkowski (L1 norm), 3 Cosine (angular), 4 SURF
% vis: 0/1 (optional, default 0) set to non-zero to display the two images
%
% Output
%
% match: 0 no match, 1 match (logical variable)
% value: distance
%
% Note: Needs the Computer Vision Toolbox
%=========================================================================
```

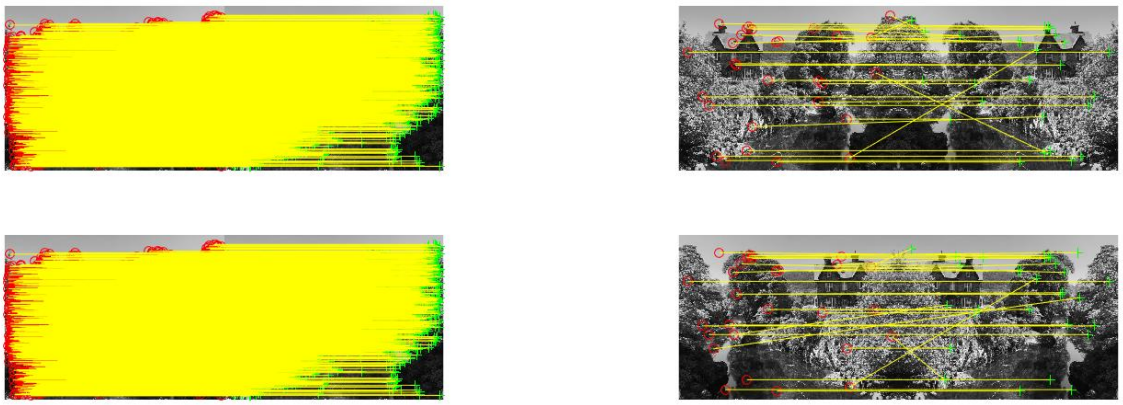
`FoxsVideoToolboxTester_MatchingFrames` demonstrates the use of the matching functions by comparing an image with its reverse, both with 1 block (there should be no difference), and 9 blocks. Note that calling the functions for every pairwise comparison is slow. For the experiments in the related papers, we wrote bespoke code which did the calculations in bulk. Due to its problem-woven nature, however, this code is not easily usable as a part of a library. The output is shown below and in Figure 2 for the SURF feature matching:

```
Testing Frame Matching (RGB), Euclidean

Threshold 1.2247
Original vs reversed - 1 block: 1, value 0.0000
Original vs reversed - 9 blocks: 1, value 0.6092

Testing Frame Matching (H-histograms, 32 bins) Minkowski

Threshold 0.5000
Original vs reversed - 1 block: 1, value 0.0000
Original vs reversed - 9 blocks: 0, value 0.7491


Testing Frame Matching SURF

Threshold 0.5000
Original vs Original 1, value 0.0000
Original vs Reverse 0, value 0.9885
```

4 Pairing (summary matching) methods

To compare two keyframe summaries, C (candidate) and GT (ground truth), we need a method to pair frames, one from each summary. Function `fox_pairing_frames` offers 6 methods [1, 2]. An example of the use of the function is given in `FoxsVideoToolboxTester_MatchingSummaries`.

```
function [F,number_of_matches,mcs,mgt] = fox_pairing_frames(...
```



(a) Matching self
(`match = 1, value = 0`)

(b) Matching reverse image
(`match = 0, value = 0.9885`)

Figure 2: SURF feature matching results.

```
    matchMatrix,threshold,pairingMethod)
%==========================================================================
% (c) Fox's Video Toolbox                                          ^--^
% 08.06.2018 ------------------------------------------------------ \oo/
%                                                                  -\/-%
% Input
%
% matchMatrix: matrix with the pairwise *distances* between the frames of
% summaries A and B. The size of the matrix is M-by-N where M is the number
% of frames in summary A, and N is the number of frames in summary B.
%
%+++++ Note: A is the candidate summary and B is the ground truth.+++++
%
% threshold: values in matchMatrix smaller than threshold are declared
%            matches
% pairingMethod: must be one of the following [1,2]
%                   1: Naive Matching
%                   2: Greedy Matching
%                   3: Hungarian Matching
%                   4: Mahmoud Method
%                   5: Kannappan Method
%                   6: Maximal Matching (Hopcroft-Karp)
% Output
%
% F: F-measure
% number_of_mathces: number of matched frames
% mcs: indices of matched frames from the candidate summary
% mgt: indices of matched frames from the groun truth
%
% ---------------
% [1] Kuncheva L. I., P. Yousefi and I. A. D. Gunn, On the Evaluation
% of Video Keyframe Summaries using User Ground Truth,
% arXiv:1712.06899, 2017.
%
% [2] Gunn I. A. D., L. I. Kuncheva, and P. Yousefi, Bipartite Graph
% Matching for Keyframe Summary Evaluation, arXiv:1712.06914, 2017.
%==========================================================================
```

Two hypothetical summaries in the 2D space are shown in the plots in Figure 3. The frames

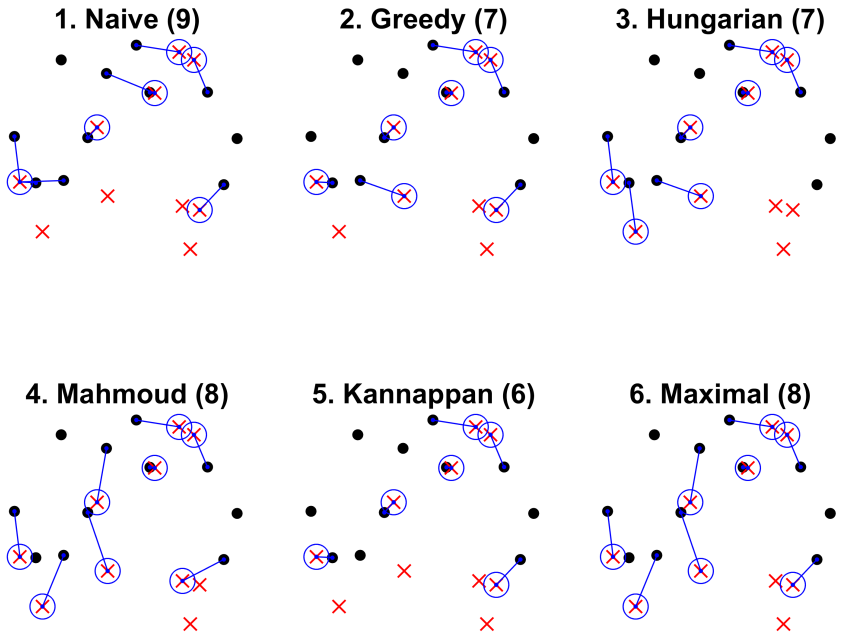


Figure 3: Results from the 6 matching (pairing) methods for two hypothetical summaries. The ground truth (10 frames) is shown with red crosses, and the candidate summary (11 frames), with black dots. The matching frames are joined with blue lines.

for the ground truth (10 frames) are plotted with red crosses, and the ones for the candidate summary (11 frames), with black dots. The matching frames are joined with blue lines. The Ground Truth frame of the matched pair is also circled. The number of matches is given in brackets next to the method's name in the title of the plot. The function returns the F-values too. It is clear that while some matching methods give high number of matches, their results (pairs that have been matched) are not very convincing.

5 Example of the pipeline

`FoxsVideoToolboxTester_Pipeline` demonstrates the work of the entire pipeline of evaluating the similarity between two keyframe summaries. We made the following choices (the same as De Avila et al. [3]):

1. Feature space: H histogram, 16 blocks.
2. Distance: Minkowski (same as L_1 norm, Manhattan).
3. Threshold for frame similarity: 0.5.
4. Pairing method: Greedy

The example compares the user #1 summary (ground truth, 11 frames) of video v21 from

Ground truth

Candidate summary

Figure 4: Results from the match of the two summaries. The 6 matched frames are shown at the front (blue rims).

the VSUMM database[1] and the summary obtained using the OV method (also included in the database, 9 frames). The frames of the two summaries are stored in folder `KeyframeSummaries`.

The number of matches is 6, $F = 0.6$. The two summaries are shown in Figure 4. The matched frames are arranged at the front of the summary (not chronologically). The matched frames have dark blue rims.

6 Auxiliary functions

- Plot a cell grid with short text in the middle of each cell

```
function fox_plot_grid(r,c,t,axesHandle,fontName)
%==========================================================================
% (c) Fox's Video Toolbox                                          ^--^
% 08.06.2018 ------------------------------------------------------  \oo/
%                                                                  -\/-%
% Input
%
% r: number of rows
% c: number of columns
% t: an r-by-c cell array with text in the cells (optional)
%    (No provision is made to centre long text within the cell.)
% axesHandle: handle of the axes to plot the grid (optional)
% fontName: string (optional)
%==========================================================================
```

- Visualise a summary as a montage, with an option to put a rim of different colour for each image.

```
function fox_montage(f,s,c,w)
%==========================================================================
% (c) Fox's Video Toolbox                                          ^--^
% 03.03.2017 ------------------------------------------------------  \oo/
%                                                                  -\/-%
% Input
%
```

[1] `https://sites.google.com/site/vsummsite/download`

```
% f: cell array with n images of the same size
% s: size [rows,columns] of the montage (optional)
% c: colour of the outside border of each image (optional)
% w: with of the border, proportion of the smaller dimension of the
%         image (optional, default = 0.08)
%
%  If 'c' is not specified, MATLAB 'montage' function is used (no gaps
%  between the images). If 'c' is of size (1,3), this is taken to be the
%  colour of all the borders. The values must be between 0 and 1. If c is
%  of size (n,3), each image will have a border with colour specified in
%  the corresponding row in 'c'.
%=========================================================================
```

References

[1] L. I. Kuncheva, P. Yousefi, I. A. D. Gunn, On the evaluation of video keyframe summaries using user ground truth, arXiv:1712.06899 (2017).

[2] I. A. D. Gunn, L. I. Kuncheva, P. Yousefi, Bipartite graph matching for keyframe summary evaluation, arXiv:1712.06914 (2017).

[3] S. E. F. De Avila, A. P. B. Lopes, A. Da Luz, A. De Albuquerque Araújo, VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method, Pattern Recognition Letters 32 (1) (2011) 56–68.