Library for Temporal Video Segmentation (TVSlib/TVSDLL) – Documentation

Version 2.0.1

Author: Christian Petersohn, petersohn@hhi.fraunhofer.de

General

TVSlib/TVSDLL is a C++ library. Given the frames of a video, TVSlib finds the temporal video segments on the scene, shot and sub-shot level.

That means the library has three main tasks:
1. shot detection - find the shot boundaries in a given video
2. sub-shot detection - find the sub-shot boundaries inside of shots
3. scene detection - cluster shots into scenes

Dongle: TVSDL will only work if the provided dongle is connected to the computer.

![Diagram showing the hierarchy of temporal video segments]

Fig. 1, Hierarchy of temporal video segments
Input:

Decoded video frames in YUV 4:2:0, plane-wise (first all Y-data, then all U-data, finally all V-data).

Output:

- shots: begin, end, key-frame number, shot transition types and durations in frame numbers
- sub-shots: begin, end in frame numbers
- scenes: begin, end in shot numbers

Definitions:

- Shot: sequence of frames in a video that have been recorded by a single camera act.
- Sub-shot: sequence of frames in a shot that are visually similar
- Scene: temporally uninterrupted sequence of shots showing one event in one setting in a continuous time interval

Short description of library usage:

First TVSlib has to be initialized. That includes specifying which segmentation levels (scenes, shots, sub-shots) are requested and whether shot detection should only detect hard cuts or additionally gradual transitions (fades, dissolves, wipes). By default all transition types are detected. Additionally, the size of the input video has to be specified. Then the decoded frames of the video have to be given to the TVSlib. Shot and sub-shot segmentation data is available after the last frame has been given to the TVSlib or possibly earlier after detection by TVSlib. Scene segmentation is available after a call of the method calcScenes() after all frames have been processed.

Interface

```c
#define SUBSHOTDETECTION 0x01
#define SHOTDETECTION 0x02
#define SCENEDETECTION 0x04

// Shot and Subshot-Detection Parameter Mode
// The higher the sensitivity, the more transitions are detected and the more transitions are falsely inserted. SENSITIVITY_MEDIUM is a good compromise between missed and falsely inserted transitions.
#define SENSITIVITY_HIGHEST 10
#define SENSITIVITY_HIGH 8
#define SENSITIVITY_MEDIUM 6
#define SENSITIVITY_LOW 4
#define SENSITIVITY_LOWEST 2

// ShotDetection return values
#define SD_ERROR -1
#define SD_NO_DATA 0
#define SD_VALID_DATA 1
```
// ShotDetection transition types
#define SD_CUT   0
#define SD_DISSOLVE  1
#define SD_FADE   2
#define SD_WIPE   3

typedef struct
{
    int begin;  // Number of first frame
    int keyframe; // Number of key frame (currently middle frame)
    int end;  // Number of last frame
    int beginTransitionType; // Type of shot transition from last shot
    int endTransitionType; // Type of shot transition to next shot
    int beginTransitionPostFNum; // Number of first frame after transition at beginning of shot
    int endTransitionPreFNum; // Number of last frame before transition at end of shot
    int numberOfSubShotBoundaries; // Number of sub-shot boundaries in the shot
    int * subShotBoundaryFrameNumbers; // Frame numbers of sub-shot boundaries
} t_ShotData;

typedef struct {
    int begin;  // Number of first shot in scene
    int key;  // Number of key shot in scene
    int end;  // Number of last shot in scene
} t_SceneData;

class TVSDLL_API CTemporalVideoSegmentationDLLInterface
{
public:
    CTemporalVideoSegmentationDLLInterface(void);
    ~CTemporalVideoSegmentationDLLInterface(void);

    /********************************************************************
    * Initialisation of TVSLib
    * detectionTypesNeeded tells the library which levels of segmentation are needed. Multiple levels may be 'OR-ed':
    * SUBSHOTDETECTION | SHOTDETECTION | SCENEDETECTION
    * shdParameterMode, ssdParameterMode and scdParameterMode define the parameter mode for each level (SENSITIVITY_HIGHEST,
    * SENSITIVITY_HIGH, SENSITIVITY_MEDIUM, SENSITIVITY_LOW,
    * SENSITIVITY_LOWEST)
    * dimx, dimy define the size of input frames in x- and y-direction
    * hardCutsOnly turns off gradual shot transition detection (e.g. for unedited video) hard cuts will still be detected, dissolves
    * (except one-frame dissolves), fades and wipes will not be detected
    *********************************************************************/

    void Initialize(int detectionTypesNeeded, int shdParameterMode, int ssdParameterMode, int scdParameterMode, int dimx, int dimy, bool hardCutsOnly=false);

    void DeInitialize(void);

    /*********************************************************************/
* processFrame() is called for every frame in the videosequence.
* frame is a decoded frame in YUV 4:2:0, planewise
* return values are SD_ERROR, SD_NO_DATA or SD_VALID_DATA
* SD_VALID_DATA means, that shot (and possibly sub-shot) data is available
*******************************************************************/
int processFrame(unsigned char * frame);

/********************************************************************
* getShotData() retrieves the shot (and possibly sub-shot) data from
* TVS.
* shotNo starts at 0.
* Returns NULL, if shot data is not available for this shotNo.
*******************************************************************/
t_ShotData * getShotData(int shotNo);

/********************************************************************
* Segmentation into scenes
*******************************************************************/
void calcScenes();

/********************************************************************
* getScene() retrieves the scene data from the TVS library.
* sceneNo starts at 0.
* Returns NULL, if scene data is not available for this sceneNo.
*******************************************************************/
t_SceneData * getScene(int sceneNo);