Musical Gestures Toolbox
Documentation

Balint Laczko

April 25, 2020

Contents

1 Class MgObject

Initializes Musical Gestures data structure from a video file.

Attributes

- filename : str
  Path to the video file.
- filtertype : {'Regular', 'Binary', 'Blob'}, optional
  The `filtertype` parameter for the `motion()` method. `Regular` turns all values below `thresh` to 0. `Binary` turns all values below `thresh` to 0, above `thresh` to 1. `Blob` removes individual pixels with erosion method.
- thresh : float, optional
  The `thresh` parameter for the `motion()` method. A number in the range of 0 to 1. Default is 0.05. Eliminates pixel values less than given threshold.

- starttime : int or float, optional
  Trims the video from this start time (s).
- endtime : int or float, optional
  Trims the video until this end time (s).
- blur : {'None', 'Average'}, optional
  The `blur` parameter for the `motion()` method. `Average` to apply a 10px * 10px blurring filter, `None` otherwise.
- skip : int, optional
  Time-shrinks the video by skipping (discarding) every n frames determined by `skip`.
- rotate : int or float, optional
  Rotates the video by a `rotate` degrees.
- color : bool, optional
  Default is `True`. If `False`, converts the video to grayscale and sets every method in grayscale mode.
- contrast : int or float, optional
  Applies +/- 100 contrast to video.
- brightness : int or float, optional
  Applies +/- 100 brightness to video.
- crop : {'none', 'manual', 'auto'}, optional
  If `manual`, opens a window displaying the first frame of the input video file, where the user can draw a rectangle to which cropping is applied. If `auto` the cropping function attempts to determine the area of significant motion and applies the cropping to that area.
- keep_all : bool, optional
  Default is `False`. If `True`, preserves an output video file after each used preprocessing stage.

2 MgObject methods

2.1 show

```
show( self, filename=None, key=None):
```

This function simply plays the current vidcap VideoObject. The speed of the video playback might not match the true fps due to non-optimized code.

Parameters

- filename : str, optional

  Default is `None`. If `None`, the current video to which the MgObject points is played. If filename is given, this file is played instead.
- key : {None, 'mgx', 'mgy', 'average', 'plot', 'motion', 'history', 'motionhistory', 'sparse', 'dense'}, optional

  If either of these shorthands is used the method attempts to show the (previously rendered) video file corresponding to the one in the MgObject.

2.2 motion

`motion( self, filtertype='Regular', thresh=0.05, blur='None', kernel_size=5, inverted_motionvideo=False, inverted_motiongram=False, unit='seconds', equalize_motiongram=True, save_plot=True, save_data=True, data_format="csv", save_motiongrams=True, save_video=True):`

Finds the difference in pixel value from one frame to the next in an input video, and saves the frames into a new video. Describes the motion in the recording.

Parameters

- filtertype : {'Regular', 'Binary', 'Blob'}, optional

  `Regular` turns all values below `thresh` to 0. `Binary` turns all values below `thresh` to 0, above `thresh` to 1. `Blob` removes individual pixels with erosion method.

- thresh : float, optional

  A number in the range of 0 to 1. Default is 0.05. Eliminates pixel values less than given threshold.

- blur : {'None', 'Average'}, optional

  `Average` to apply a 10px * 10px blurring filter, `None` otherwise.

- kernel_size : int, optional

  Default is 5. Size of structuring element.

- inverted_motionvideo : bool, optional

  Default is `False`. If `True`, inverts colors of the motion video.

- inverted_motiongram : bool, optional

  Default is `False`. If `True`, inverts colors of the motiongrams.

- unit : {'seconds', 'samples'}, optional

  Unit in QoM plot.

- equalize_motiongram : bool, optional

  Default is `True`. If `True`, converts the motiongrams to hsv-color space and flattens the value channel (v).

- save_plot : bool, optional

  Default is `True`. If `True`, outputs motion-plot.

- save_data : bool, optional

  Default is `True`. If `True`, outputs motion-data.

- data_format : {'csv', 'tsv', 'txt'}, optional

  Specifies format of motion-data.

- save_motiongrams : bool, optional

  Default is `True`. If `True`, outputs motiongrams.

- save_video : bool, optional

  Default is `True`. If `True`, outputs the motion video.

Outputs

- `filename`_motion.avi

  A video of the absolute difference between consecutive frames in the source video.

- `filename`_motion_com_qom.png

  A plot describing the centroid of motion and the quantity of motion in the source video.

- `filename`_mgx.png

  A horizontal motiongram of the source video.

- `filename`_mgy.png

  A vertical motiongram of the source video.

- `filename`_motion.csv

  A text file containing the quantity of motion and the centroid of motion for each frame in the source video with timecodes in milliseconds. Available formats: csv, tsv, txt.

Returns

- MgObject

  A new MgObject pointing to the output '_motion' video file. If `save_video=False`, it returns an MgObject pointing to the input video file.

2.3 history

`history( self, filename='', history_length=10):`

This function creates a video where each frame is the average of the n previous frames, where n is determined by `history_length`. The history frames are summed up and normalized, and added to the current frame to show the history.

Parameters

- filename : str, optional

  Path to the input video file. If not specified the video file pointed to by the MgObject is used.

- history_length : int, optional

  Default is 10. Number of frames to be saved in the history tail.

Outputs

- `filename`_history.avi

Returns

- MgObject A new MgObject pointing to the output '_history' video file.

2.4 motionhistory

`motionhistory( self, history_length=10, kernel_size=5, filtertype='Regular', thresh=0.05, blur='None', inverted_motionhistory=False):`

Finds the difference in pixel value from one frame to the next in an input video, and saves the difference frame to a history tail. The history frames are summed up and normalized, and added to the current difference frame to show the history of motion.

Parameters

- history_length : int, optional

  Default is 10. Number of frames to be saved in the history tail.

- kernel_size : int, optional

  Default is 5. Size of structuring element.

- filtertype : {'Regular', 'Binary', 'Blob'}, optional

  `Regular` turns all values below `thresh` to 0. `Binary` turns all values below `thresh` to 0, above `thresh` to 1. `Blob` removes individual pixels with erosion method.

- thresh : float, optional

  A number in the range of 0 to 1. Default is 0.05. Eliminates pixel values less than given threshold.

- blur : {'None', 'Average'}, optional

  `Average` to apply a 10px * 10px blurring filter, `None` otherwise.

- inverted_motionhistory : bool, optional

  Default is `False`. If `True`, inverts colors of the motionhistory video.

Outputs

- `filename`_motionhistory.avi

Returns

- MgObject

  A new MgObject pointing to the output '_motionhistory' video file.

2.5 average

`average( self, filename='', normalize=True):`

Finds and saves an average image of an input video file.

Parameters

- filename : str, optional

  Path to the input video file. If not specified the video file pointed to by the MgObject is used.

- normalize : bool, optional

  Default is `True`. If `True`, normalizes pixel values in the output image.

Outputs

- `filename`_average.png

Returns

- MgImage

  A new MgImage pointing to the output '_average' image file.

2.6 flow.sparse

```
sparse( self, filename='', corner_max_corners=100, corner_quality_level=0.3, corner_min_distance=7,
corner_block_size=7, of_win_size=(15, 15), of_max_level=2, of_criteria=(cv2.TERM_CRITERIA_EPS
| cv2.TERM_CRITERIA_COUNT, 10, 0.03)):
```

Renders a sparse optical flow video of the input video file using `cv2.calcOpticalFlowPyrLK()`. `cv2.goodFeaturesToTrack()` is used for the corner estimation. For more details about the parameters consult the cv2 documentation.

Parameters

- filename : str, optional

  Path to the input video file. If not specified the video file pointed to by the MgObject is used.

- corner_max_corners : int, optional

  Default is 100.

- corner_quality_level : float, optional

  Default is 0.3.

- corner_min_distance : int, optional

  Default is 7.

- corner_block_size : int, optional

  Default is 7.

- of_win_size : tuple (int, int), optional

  Default is (15, 15).

- of_max_level : int, optional

  Default is 2.

- of_criteria : optional

  Default is `(cv2.TERM_CRITERIA_EPS | cv2.TERM_CRITERIA_COUNT, 10, 0.03)`.

Outputs

- `filename`_flow_sparse.avi

Returns

- MgObject

  A new MgObject pointing to the output '_flow_sparse' video file.

2.7 flow.dense

`dense( self, filename='', pyr_scale=0.5, levels=3, winsize=15, iterations=3, poly_n=5, poly_sigma=1.2, flags=0, skip_empty=False):`

Renders a dense optical flow video of the input video file using `cv2.calcOpticalFlowFarneback()`. For more details about the parameters consult the cv2 documentation.

Parameters

- filename : str, optional

  Path to the input video file. If not specified the video file pointed to by the MgObject is used.

- pyr_scale : float, optional

  Default is 0.5.

- levels : int, optional

  Default is 3.

- winsize : int, optional

  Default is 15.

- iterations : int, optional

  Default is 3.

- poly_n : int, optional

  Default is 5.

- poly_sigma : float, optional

  Default is 1.2.

- flags : int, optional

  Default is 0.

- skip_empty : bool, optional

  Default is `False`. If `True`, repeats previous frame in the output when encounters an empty frame.

Outputs

- `filename`_flow_dense.avi

Returns

- MgObject

  A new MgObject pointing to the output '_flow_dense' video file.

3 Utility Functions

3.1 centroid

`centroid( image, width, height):`

Defined in *_centroid.py*.
Computes the centroid of an image or frame.

Parameters

- image : np.array(uint8)

  The input image matrix for the centroid estimation function.

- width : int

  The pixel width of the input video capture.

- height : int

  The pixel height of the input video capture.

Returns

- np.array(2)

  X and Y coordinates of the centroid of motion.

- int

  Quantity of motion: How large the change was in pixels.

3.2 mg_cropvideo

`mg_cropvideo( fps, width, height, length, of, fex, crop_movement='Auto', motion_box_thresh=0.1, motion_box_margin=1):`

Defined in *_cropvideo.py*.
Crops the video.

Parameters

- fps : int

  The FPS (frames per second) of the input video capture.

- width : int

  The pixel width of the input video capture.

- height : int

  The pixel height of the input video capture.

- length : int

  The number of frames in the input video capture.

- of : str

  'Only filename' without extension (but with path to the file).

- fex : str

  File extension.

- crop_movement : {'Auto','Manual'}, optional

  `Auto` finds the bounding box that contains the total motion in the video. Motion threshold is given by motion_box_thresh. `Manual` opens up a simple GUI that is used to crop the video manually by looking at the first frame.

- motion_box_thresh : float, optional

  Only meaningful if `crop_movement='Auto'`. Takes floats between 0 and 1, where 0 includes all the motion and 1 includes none.

- motion_box_margin : int, optional

  Only meaningful if `crop_movement='Auto'`. Adds margin to the bounding box.

3.3 filter_frame

`filter_frame( motion_frame, filtertype, thresh, kernel_size):`

Defined in *_filter.py*.
Applies a filter to an image or videoframe.

Parameters

- motion_frame : np.array(uint8)

  Input motion image.

- filtertype : {'Regular', 'Binary', 'Blob'}

  `Regular` turns all values below `thresh` to 0. `Binary` turns all values below `thresh` to 0, above `thresh` to 1. `Blob` removes individual pixels with erosion method.

- thresh : float

  A number in the range of 0 to 1. Eliminates pixel values less than given threshold.

- kernel_size : int

  Size of structuring element.

Returns

- np.array(uint8)

  Filtered frame.

3.4 mg_contrast_brightness

`mg_contrast_brightness( of, fex, vidcap, fps, length, width, height, contrast, brightness):`

Defined in *_videoadjust.py*.
Applies contrast and brightness to a video.

Parameters

- of : str
  'Only filename' without extension (but with path to the file).
- fex : str
  File extension.
- vidcap :
  cv2 capture of video file, with all frames ready to be read with `vidcap.read()`.
- fps : int
  The FPS (frames per second) of the input video capture.
- length : int
  The number of frames in the input video capture.
- width : int
  The pixel width of the input video capture.
- height : int
  The pixel height of the input video capture.
- contrast : int or float, optional
  Applies +/- 100 contrast to video.
- brightness : int or float, optional
  Applies +/- 100 brightness to video.

Outputs

- A video file with the name `of` + 'cb' + `fex`.

Returns

- cv2 video capture of output video file.

3.5 mg_skip_frames

`mg_skip_frames( of, fex, vidcap, skip, fps, length, width, height):`

Defined in *_videoadjust.py*.
Time-shrinks the video by skipping (discarding) every n frames determined by `skip`.

Parameters

- of : str

  'Only filename' without extension (but with path to the file).

- fex : str

  File extension.

- vidcap :

  cv2 capture of video file, with all frames ready to be read with `vidcap.read()`.

- skip : int

  Every n frames to discard. `skip=0` keeps all frames, `skip=1` skips every other frame.

- fps : int

  The FPS (frames per second) of the input video capture.

- length : int

  The number of frames in the input video capture.

- width : int

  The pixel width of the input video capture.

- height : int

  The pixel height of the input video capture.

Outputs

- A video file with the name `of` + '_skip' + `fex`.

Returns

- videcap :

  cv2 video capture of output video file.

- length : int

  The number of frames in the output video file.

- fps : int

  The FPS (frames per second) of the output video file.

- width : int

  The pixel width of the output video file.

- height : int

  The pixel height of the output video file.

3.6 mg_videoreader

`mg_videoreader( filename, starttime=0, endtime=0, skip=0, rotate=0, contrast=0, brightness=0, crop='None', color=True, keep_all=False, returned_by_process=False):`

Defined in *_videoreader.py*.
Reads in a video file, and optionally apply several different processes on it. These include:

- trimming,
- skipping,
- rotating,
- applying brightness and contrast,
- cropping,
- converting to grayscale.

Parameters

- filename : str

  Path to the input video file.

- starttime : int or float, optional

  Trims the video from this start time (s).

- endtime : int or float, optional

  Trims the video until this end time (s).

- skip : int, optional

  Time-shrinks the video by skipping (discarding) every n frames determined by `skip`.

- rotate : int or float, optional

  Rotates the video by a `rotate` degrees.

- contrast : int or float, optional

  Applies +/- 100 contrast to video.

- brightness : int or float, optional

  Applies +/- 100 brightness to video.

- crop : {'none', 'manual', 'auto'}, optional

  If `manual`, opens a window displaying the first frame of the input video file, where the user can draw a rectangle to which cropping is applied. If `auto` the cropping function attempts to determine the area of significant motion and applies the cropping to that area.

- color : bool, optional

  Default is `True`. If `False`, converts the video to grayscale and sets every method in grayscale mode.

- keep_all : bool, optional

  Default is `False`. If `True`, preserves an output video file after each used preprocessing stage.

Outputs

- A video file with the applied processes. The name of the file will be `filename` + a suffix for each process.

Returns

- length : int

  The number of frames in the output video file.

- width : int

  The pixel width of the output video file.

- height : int

  The pixel height of the output video file.

- fps : int

  The FPS (frames per second) of the output video file.

- endtime : float

  The length of the output video file in seconds.

- of: str

  The path to the output video file without its extension. The file name gets a suffix for each used process.

- fex : str

  The file extension of the output video file. Currently it is always 'avi'.