Musical Gestures Toolbox
Documentation

Frida Furmyr & Marcus Widmer

July 17, 2019

Contents

1 Class MgObject

Initializes Musical Gestures data structure from a given parameter video file.

Parameters:

filename (str): Name of input parameter video file.

method (str): Currently 'Diff' is the only implemented method.

filtertype (str): 'Regular', 'Binary', 'Blob' (see function motionfilter).

thresh (float): a number in [0,1]. Eliminates pixel values less than given threshold.

starttime (float): cut the video from this start time (min) to analyze what is relevant.

endtime (float): cut the video at this end time (min) to analyze what is relevant.

blur (str): 'Average' to apply a blurring filter, 'None' otherwise.

skip (int): When proceeding to analyze next frame of video, this many frames are skipped. NB: skip cannot exceed number of frames per second (fps).

color (bool): True does the analysis in RGB, False in grayscale.

contrast (float): apply +/- 100 contrast to video

brightness (float): apply +/- 100 brightness to video

crop (str): 'none', 'manual', 'auto' to select cropping of relevant video frame size

2 MgObject methods

2.1 mg_videoreader

mg_videoreader(filename, starttime = 0, endtime = 0, skip = 0, contrast = 0, brightness = 0, crop = 'none'):

Reads in a video file, and by input parameters user decide if it: trims the length, skips frames, applies contrast/brightness adjustments and/or crops image width/height.

Parameters:

filename (str): Name of input parameter video file.

starttime (float): cut the video from this start time (min) to analyze what is relevant.

endtime (float): cut the video at this end time (min) to analyze what is relevant.

skip (int): When proceeding to analyze next frame of video, this many frames are skipped.

contrast (float): apply +/- 100 contrast to video

brightness (float): apply +/- 100 brightness to video

crop (str): 'None', 'Auto' or 'Manual' to crop video.

Returns:

vidcap (VideoCapture object): cv2 video capture of editevideo file

length (int), fps(int), width(int), height(int): video attributes

of (str): only-filename - filename gets updated with what procedures it went through.

2.2 mg_motionvideo

mg_motionvideo(self, method = 'Diff', filtertype = 'Regular', thresh = 0.001, blur = 'None', kernel_size = 5, inverted_motionvideo = False, inverted_motiongram = True, unit = 'seconds', equalize_motiongram = 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. Outputs a video called filename +'_motion.avi'.

Parameters:

kernel_size (int): Size of structuring element.

method (str): Currently 'Diff' is the only implemented method.

filtertype (str): 'Regular', 'Binary', 'Blob'(see function motionfilter)

thresh (float): a number in [0,1]. Eliminate spixel values less than given threshold.

blur (str): 'Average' to apply a blurring filter, 'None' otherwise.

inverted_motionvideo (bool): Inverts colors of motionvideo

inverted_motiongram (bool): Inverts colors of motiongram

unit (str): Unit in QoM plot. 'seconds' or 'samples'

equalize_motiongram (bool): Converts the motiongram to hsv-color space and flattens the value channel (v).

Returns:

None

2.3 mg_cropvideo

mg_cropvideo(fps,width,height, length, of, crop_movement = 'auto', motion_box_thresh = 0.1, motion_box_margin = 1)

Crops the video.

Parameters:

crop_movement(str): 'auto' finds the bounding box that contains the total motion in the video. 'manual' opens up a simple GUI that is used to crop the video manually by looking at the first frame.

motion_box_thresh (float): Only meaningful is crop_movement = 'auto'. Takes floats between 0 and 1, where 0 includes all the motion and 1 includes none.

motion_box_margin (int): Only meaningful is crop_movement = 'auto'. Add margin to the bounding box.

Returns:

None

2.4 mg_input_test

mg_input_test(filename,method,filtertype,thresh,starttime,endtime,blur,skip):

Gives feedback to user if initialization from input went wrong. Ex: raise InputError(msg) msg = 'Please specify a filter type as str: Regular or Binary'

2.5 mg_motionhistory

mg_motionhistory(self, history_length = 20, kernel_size = 5, method = 'Diff', filtertype = 'Regular', thresh = 0.001, 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.
Outputs a video called filename + '_motionhistory.avi'.

Parameters:

history_length (int): How many frames will be saved to the history tail.

kernel_size (int): Size of structuring element.

method (str): Currently 'Diff' is the only implemented method.

filtertype (str): 'Regular', 'Binary', 'Blob' (see function motionfilter)

thresh (float): a number in [0,1]. Eliminates pixel values less than given threshold.

blur (str): 'Average' to apply a blurring filter, 'None' otherwise.

inverted_motionhistory (bool): Inverts the colors of motionhistory video

Returns:

None

2.6 mg_contrast_brightness

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

Edit contrast and brightness of the video.

Parameters:

of (str): filename without extension

vidcap (VideoCapture object): cv2 capture of video file, with all frames ready to read with vidcap.read().

fps (int), width (int), height (int) are simply info about vidcap

contrast (float): apply +/- 100 contrast to video

brightness (float): apply +/- 100 brightness to video

Returns:

vidcap (VideoCapture object): cv2 video capture of edited video file

2.7 mg_skip_frames

mg_skip_frames(of, vidcap, skip, fps, width, height):

Frame skip, convenient for saving time/space in an analysis of less detail looking at big picture movement. Skips the given number of frames, making a compressed version of the input video file.

Parameters:

of (str): filename without extension

vidcap (VideoCapture object): cv2 capture of video file, with all frames ready to read with vidcap.read().

fps (int), width (int), height (int) are simply info about vidcap

skip (int): When proceeding to analyze next frame of video, this many frames are skipped.

Returns:

vidcap (VideoCapture object): cv2 video capture of edited video file.

length (int), fps (int), width (int), height(int): new video attributes.

2.8 mg_show

mg_show(self, filename = 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): If left empty, the current vidcap object is played. If filename is given, this file is played instead.

Returns:

None

3 Functions

3.1 average_image

average_image(filename, normalize = True):

Post-processing tool. Finds and saves an average image of entire video.

Usage:
from _motionaverage import motionaverage
motionaverage('filename.avi')

Parameters:

filename (str): name of video

normalize (bool): Scales pixel values to lie between 0 and 255. Additionally the pixel value (in HSV color space) histogram is flattened.

Returns:

None

3.2 history

history(filename,history_length = 10):

This function creates a video where each frame is the average of the n previous frames, where n is determined from history_length. The history frames are summed up and normalized, and added to the current frame to show the history.
Outputs a video called filename + '_history.avi'.

Parameters:

history_length (int): How many frames will be saved to the history tail.

Returns:

None

3.3 centroid

centroid(image, width, height):

Computes the centroid of motion and quantity of motion of an image/frame.

Parameters:

image (numpy array(uint8))

width (int)

height (int)

Returns:

Centroid of motion ([X(int),Y(int)]): X,Y(origo in bottom left corner) coordinate of the maximum change in pixel value

Quantity of motion (int): How large was the change in pixel value

3.4 constrainNumber

constrainNumber(n, minn, maxn)

Constrains number to having a value between minn and maxn

Parameters:

n (int, float): number to constrain

minn (int, float): lower limit n can be

maxn (int, float): lower limit n can be

Returns:

Constrained number(int, float)

3.5 filter_frame

filter_frame(motion_frame, filtertype,thresh,kernel_size)

Apply a filter to a picture/videoframe

Parameters:

motion_frame (array(uint8)): input motion image

filtertype (str): '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): for 'Regular' and 'Binary' option, thresh is a value of threshold [0,1];

kernel_size(int): Size of structuring element

Returns:

filtered frame (array(uint8))