Program for Automated Motion Detection and Number Recognition

(in a repeated task setting)

Professor: Dr. Mili Shah

Student: Jongoh (Andy) Jeong

Affiliation: The Cooper Union for the Advancement of Science and Art

Date: July 22, 2019

Table of Contents

1 Python version

a. Functionalities

- The ultimate goal is to analyze task performance improvements in a repeated-task environment upon wearing exoskeletons/suit and not wearing them by a number of metrics. In order to measure the efficiency improvement of exoskeletons in such repetitive work settings, some observed metrics are held in interest in this program, including the time it takes to complete a cycle of the given task, heart rate changes and weight of applied force at the point of contact with the device.
- The duration of each repetition is detected in the program by motion detection, where the current running frame is compared against the frame at which the user specifies the region of interest (by mouse click and release). The underlying conditions for detection of this changes in the pixels rely on the structural similarity index and the number of different pixels between the two frames being compared. For this particular force measurement videos, these two measures give rather clearer distinction between the two frames than the peg drilling task, where shadows and light settings affect the view by imposing dark pixels on regions over which the drill moves – this difficulty in distinguishing whether another object is introduced in this region makes it a bit more difficult to extract differences simply from 2D color-dependent views from the video.
- For recognition of digits for the heart rate and applied force measurements, there were two general approaches considered in the process – Optical Character Recognition (OCR), and a deep-learning training model. The latter relies on a pre-trained model, where numerous images of digits are necessary to train and still may not be as effective, whereas the former is lighter in space and time complexity but is often more prone to error due to several factors, including, but not limited to, angle and resolution. After experimenting process with transformations on several isolated views, the heart rate was read correctly most of the time (set to 'default' mode), despite its slight difficulty distinguishing between 1 and 7, and also 3 and 5 at times. As for reading force measurements, the angle, light settings (brightness in the device) and difficulty in clear visibility of the digits even with bare eyes impose even further failures in reading the digits correctly. During several experiments, rotations and 4-point perspective transformations have been applied to zoom into the screen; however, from the available videos it was not possible to read all digit segments from the angle, and thus the current program does not take into account force readings.

b. Dependencies

- This program is written in Python language (for portability across systems without MATLAB, for instance) and uses a number of modules available for open-source:
  1) *Tesseract OCR (source: https://github.com/tesseract-ocr/tesseract/wiki)
  2) Pytesseract (OCR) in Python
  3) Tkinter (GUI tool)
  4) Matplotlib (Image Figures)
  5) Pillow (Image, ImageTk for Image Procecssing)
  6) Scikit-Image (Structural Similarity Index)
  7) OpenCV (Image Processing)
  8) Numpy/time/csv

  ** third-party software that requires installation on system paths; this software and trained files required*

c. Usage / Instructions

- The package is packaged and made available to run for Windows, Unix (Macintosh) and Linux executables.
- Upon opening the file, a screen (Fig. 1) asking for a video and two options (display and heart rate region selection modes) pops up. Checking the first option turns off displaying ongoing display windows on the side, and the second option allows the user to select a rectangular region for heart rate reading. Not checking the second box ('default') assumes that the heart rate reading device is positioned at the same location in the video; this mode is set such that the number is read as digits most of the time.
- Upon selecting the video, the user needs to define a rectangular region where the handheld device comes to contact. When selected the region, it marks with colors (Fig. 2) that represent the status: Red for not detecting any motion, Green for detected motion, Blue for finished reading the video. As the motion is detected, it will read in continuously the heart rate, and display the current frame, times, and recognized text if the first checkbox isn't selected at start. Upon completion, a pop up window will indicate its finished state of reading the video, and the selected region will be highlighted in blue.

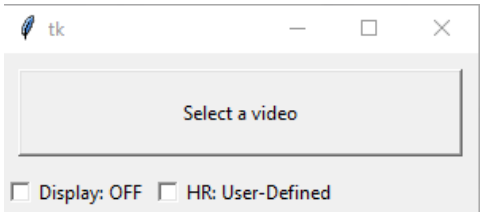


Figure 1. Upon Execution

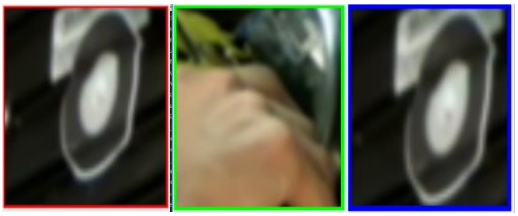


Figure 2. Selected Region by Status

d. Constraints / Limitations

- This program largely relies on available third party software, Tesseract. This neural-network-based model along with its training files (version 4.0) are necessary in the system path for text in image to be read in a proper manner (Follow README for instructions for installation).
- The region for motion detection (and heart rate) needs to be defined for the program to initiate. For videos recorded in the same settings as example videos, the default heart rate works better in recognition, since a user-defined region may have pixel differences from the default, and the optical character recognition technique may not necessarily work.
- At times due to the inherent nature of slight obstruction in the region in the video, it may seem to have detected motion when there was none; this is taken into account by updating the start time when a noticeably large obstruction comes into the region at a later time – this in addition considers the time it takes for the hand-held device to come in full contact to the object in the region. As far as the performance on the provided videos in prior, this gives a fairly good performance.
- To speed up the process, a few video frames are discarded, after some experimentation with sample videos where it is observed that there was not much difference between frames that are a few frames apart. Upon selecting region until any motion is detected, frames equal to FPS (frames per second) of the video divided by 4 are skipped, and after motion is detected, it speeds up faster by skipping by twice as many (FPS/2). FPS/2, or simply half a second, therefore is the error bound (+/- 0.5 second error) in detection. This didn't pose much trouble as differences in frames were large enough to be detected.

e. Output Files

- A CSV-formatted file containing the start time, end time, duration of each rep, and minimum, maximum, and average heart rate is produced at the end, all in a single row labeled by the iteration.

2 MATLAB version

a. Functionalities

- The general functionalities of the program are identical to that of the Python version; there exist some adjustments for the thresholds for structural similarity index and pixel differences due to window size.

b. Dependencies

- This MATLAB version requires installation of Computer Vision Toolbox for image-processing and OCR functions.

c. Usage / Instructions

- Upon running the file in MATLAB, a GUI will be displayed. Then load a video to scan, and follow the pop up message directions (highlight region of interest, and another region for heart rate reading). Then it will automatically run through the video, outputting cropped region/heart rate region, and a list of measurements (time and heart rate) for each rep below. One difference from the Python version is that the highlighted region for the number (heart rate) works good enough in most cases, and thus there is no option for 'default' region. There is also an optional manual button to override false motion detection, in case something passes by, and detects as a rep, when it should not. This number is accepted if in range [50, 200], particularly for practical heart rate readings.

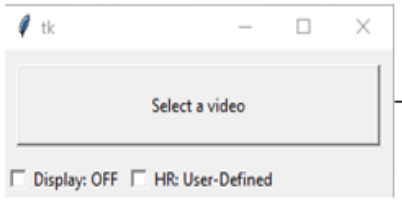
d. Constraints / Limitations

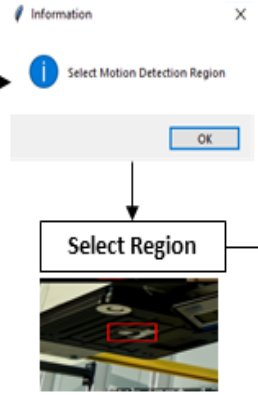
- Without skipping a couple video frames, the runtime of the program could take approximately as long as the original video run, depending on the machine. This program thus, in order to speed up the process a bit, skips 3 frames every time it loads a new frame. Since the movement in the video is not so fast that it is lost by skipping a few, this is within tolerance level in terms of detecting motions. Since not precisely coordinated, the time intervals may be different for different machines; it goes at a rate of one frame per almost every 0.2 seconds, which means it has error within bounds of +/- 0.1 seconds for time duration measurement.

e. Output Files

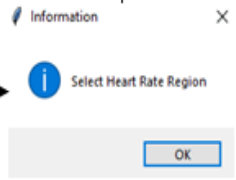
- In addition to what is shown in the window while running, a CSV-formatted file containing the start time, end time, duration of each rep, and minimum, maximum, and average heart rate is produced at the end, all in a single row labeled by the iteration. The very last row contains an average statistic of each column, labeled as iteration = 0.

Flow Diagram for the Program (Python)



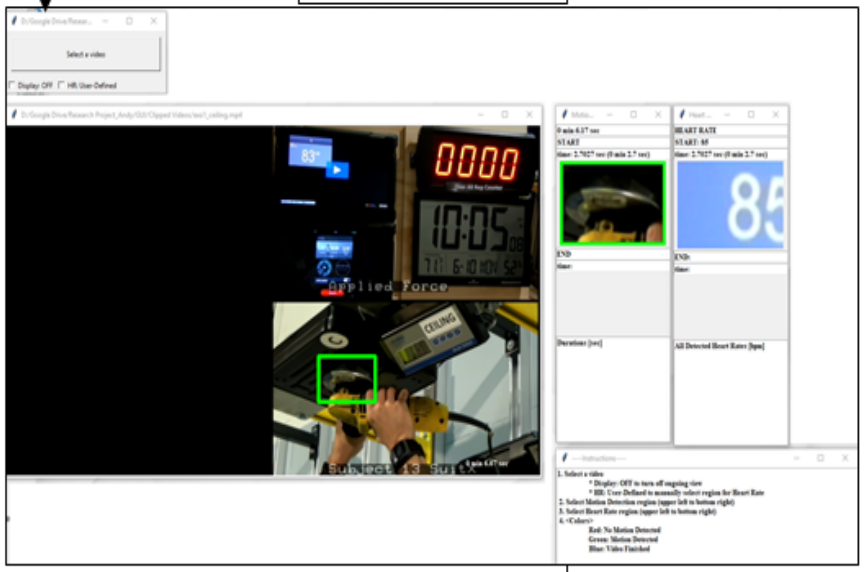


Upon selecting region, it will appear in red box



Select Region

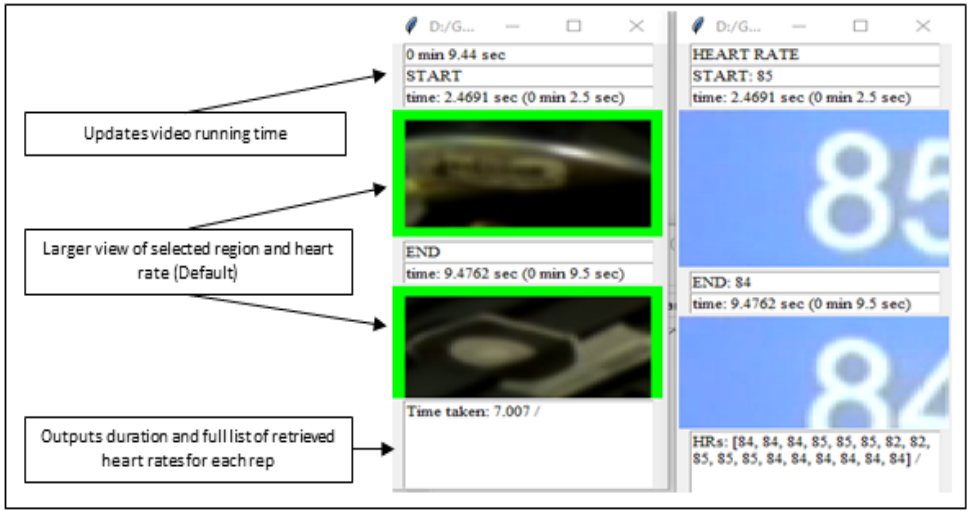
Sample View in Windows OS



Option2: User-defined / Default Heart Rate Region

* If heart rate region is not set to 'default', the user selects the motion detection region first, and then the region where heart rate is read from (in order)

Legend: *Nothing detected: Red / Detected: Green / Finished: Blue*



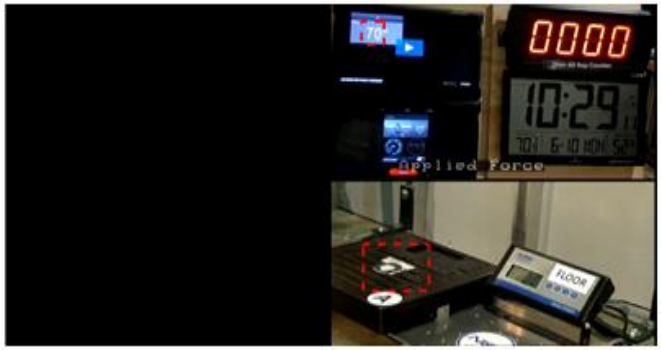
	A	B	C	D	E	F	G
1	ITERATIONS	TIME_START	TIME_END	DURATION	MIN_HR	MAX_HR	AVG_HR
2	1	5.9393	13.6804	7.7411	59	72	69.9412
3	2	16.7501	24.6247	7.8745	68	73	70.4146
4	3	27.9614	36.3698	8.4084	70	74	71.3523
5	4	39.3061	47.581	8.2749	69	74	71.5714
6	5	55.0552	63.3301	8.275	59	73	70.8136
7	6	66.5333	74.8083	8.275	71	80	77.3256
8	7	79.2127	88.155	8.9423	78	90	83.61
9	8	94.4279	102.7029	8.275	89	94	91.4679
10	9	106.44	114.448	8.008	87	91	88.4659
11	10	118.3186	126.727	8.4084	87	90	87.8043

Output (CSV)

Flow Diagram for the Program (MATLAB)

Select Region

Select HR Region



No motion detected

Count: 2 Status: START ONGOING Heart Rate: 71 Click here if START is mistaken.

Time 22.2556

	Iteration	TIME_START	TIME_END	DURATION	MIN_HR	MAX_HR	AVG_HR
1	1	5.9393	13.6804	7.7411	59	72	69.9412

Output per iteration

	A	B	C	D	E	F	G
1	ITERATIONS	TIME_START	TIME_END	DURATION	MIN_HR	MAX_HR	AVG_HR
2	1	5.9393	13.6804	7.7411	59	72	69.9412
3	2	16.7501	24.6247	7.8745	68	73	70.4146
4	3	27.9614	36.3698	8.4084	70	74	71.3523
5	4	39.3061	47.581	8.2749	69	74	71.5714
6	5	55.0552	63.3301	8.275	59	73	70.8136
7	6	66.5333	74.8083	8.275	71	80	77.3256
8	7	79.2127	88.155	8.9423	78	90	83.61
9	8	94.4279	102.7029	8.275	89	94	91.4679
10	9	106.44	114.448	8.008	87	91	88.4659
11	10	118.3186	126.727	8.4084	87	90	87.8043
12	0	0	0	8.2483	73.7	81.1	78.2767

Output at the end (csv)

Motion detected

* Shows cropped images while rep is in motion