Program for Automated Motion Detection and Number Recognition

(for Applied Force tasks)

Abstract

The goal of this program is to automate heart rate readings from quad-view videos of a subject performing a task on force plates, in the floor-wall-ceiling order. The heart rate measurements taken directly from a smartwatch and a smartphone application, whose screens are projected in one of the views in the video. Upon start of the program, a user is able to select regions of interest (ROIs) for motion changes and heart rate devices. The motion ROIs, in general, takes into account the pixel values per frame and structural similarity (SSIM) index values computed across the previous and current frames.

Professor: Dr. Mili Shah

Student: Jongoh (Andy) Jeong

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

Date: February 10, 2020

Table of Contents

1 MATLAB version

a. Functionalities

- SSIM index is a measure to score each pair of two frames by computing a numerical value for comparison, as in Eq. 1-a. The closer to 1 this metric is, the more similar the two frames are.

$$\mathrm{SSIM}(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Equation 1-a. Structural Similarity Index

μ_x: average x, σ: (co)variance, c_i: $(k_iL)^2$, variable to stabilize the division with weak denominator, L: dynamic range of the pixel-values (typically $2^{\#bits/pixel}-1$), k_1: 0.01, k_2: 0.03 by default.

From a different approach, this formula is based on luminance (l), contrast (c), and structure (s) determined by Eq. 1-b, where the SSIM score is the weighted sum:

$$\mathrm{SSIM}(x,y) = \left[l(x,y)^\alpha \cdot c(x,y)^\beta \cdot s(x,y)^\gamma\right]$$
$$l(x,y) = \frac{2\mu_x\mu_y + c_1}{\mu_x^2 + \mu_y^2 + c_1}$$
$$c(x,y) = \frac{2\sigma_x\sigma_y + c_2}{\sigma_x^2 + \sigma_y^2 + c_2}$$
$$s(x,y) = \frac{\sigma_{xy} + c_3}{\sigma_x\sigma_y + c_3}$$

Equation 1-b. Luminance, contrast, structure formulas, with $c_3 = c_2 / 2$

- For the force plate measurement videos, these two measures give rather clearer distinction between the two frames than the peg drilling task, where shadows and light settings directly affect the view largely 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.
- Before reading the text, a few processing on the image are taken. First, top-hat filtering with a structural element (disk of radius 15) is performed (subtracting the morphological opening of the image from the original). Then this filtered image is eroded with another structural element (line of length 10), which is then morphologically reconstructed under the first filtered image. This clears noisy pixels before sending off to ocr for reading (see page 6).

b. Dependencies

- This MATLAB version requires installation of *Computer Vision Toolbox* for image-processing and OCR functions. This is tested under R2018a ver.

c. Usage / Instructions

- Refer to README file.

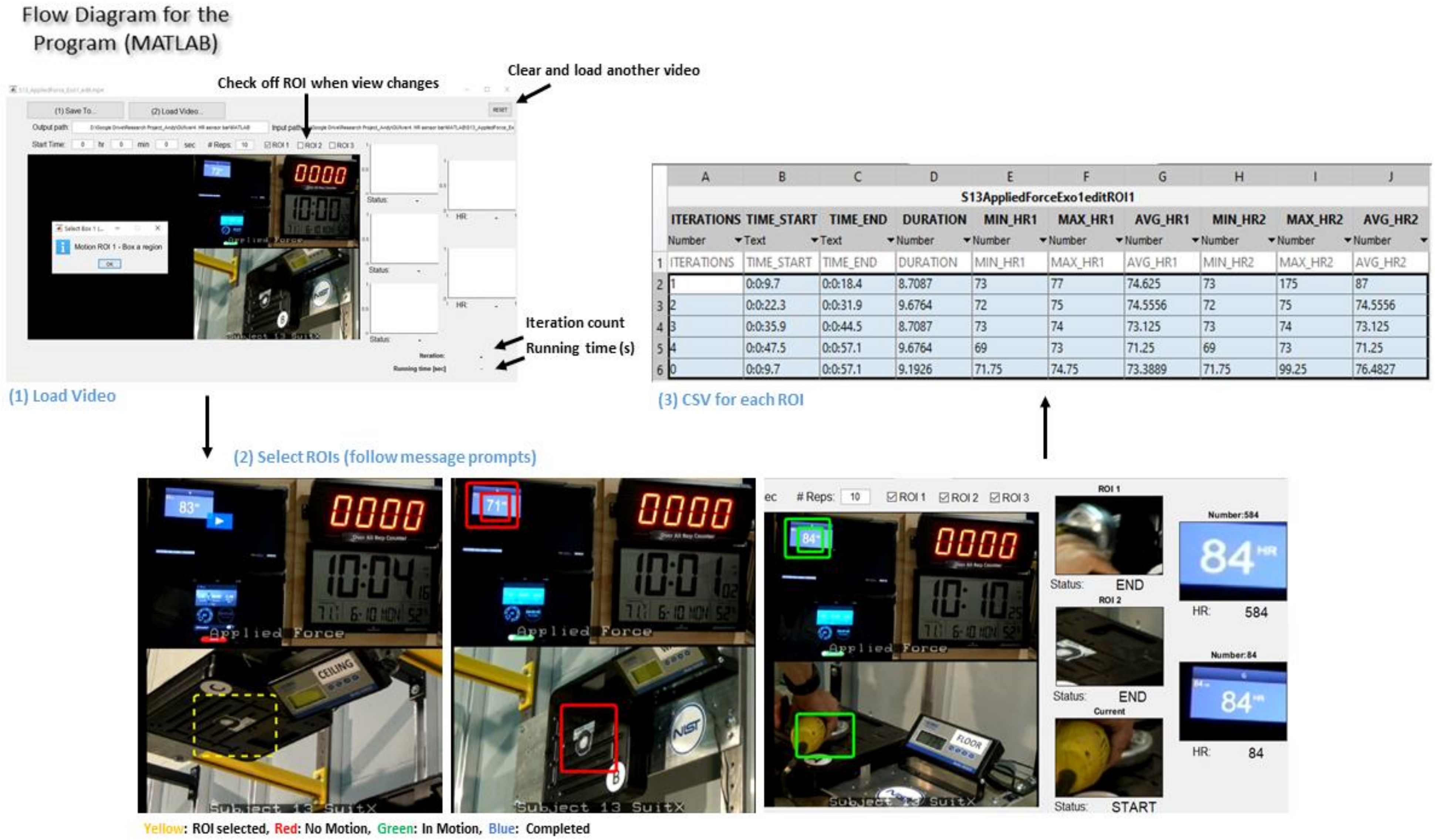
d. Constraints / Limitations

- 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 may still not be as effective, whereas the former is lighter in terms of 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. 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.
- 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. The user can change the variable *frame_speedup* value in the code to change how fast it processes frames.

e. Output Files

- In addition to what is shown in the window while running, CSV-formatted files containing the start time, end time, duration of each rep, and minimum, maximum, and average heart rate for each HR device are produced at the end, each row for each iteration. The very last row contains an average statistic of each column, labeled as iteration = 0 – only to note uniquely in the index.

Appendix A -1



	A	B	C	D	E	F	G	H	I	J
	S13AppliedForceExo1editROI1									
	ITERATIONS Number	TIME_START Text	TIME_END Text	DURATION Number	MIN_HR1 Number	MAX_HR1 Number	AVG_HR1 Number	MIN_HR2 Number	MAX_HR2 Number	AVG_HR2 Number
1	ITERATIONS	TIME_START	TIME_END	DURATION	MIN_HR1	MAX_HR1	AVG_HR1	MIN_HR2	MAX_HR2	AVG_HR2
2	1	0:0:9.7	0:0:18.4	8.7087	73	77	74.625	73	175	87
3	2	0:0:22.3	0:0:31.9	9.6764	72	75	74.5556	72	75	74.5556
4	3	0:0:35.9	0:0:44.5	8.7087	73	74	73.125	73	74	73.125
5	4	0:0:47.5	0:0:57.1	9.6764	69	73	71.25	69	73	71.25
6	0	0:0:9.7	0:0:57.1	9.1926	71.75	74.75	73.3889	71.75	99.25	76.4827

Appendix A -2

Processing before OCR

1) Top-hat Filtering, disk 15 	2) Eroding, line, 10 
3) Reconstruct (2) under (1) 	4) Binarized