Signal Averaging Algorithm for Utilization in ECG Monitoring

Alex Gilmore | asgilmore@wisc.edu

Introduction

Electrocardiogram (ECG) signals are commonly difficult to analyze due to contamination by many different types of noise. The signal-to-noise ratio (SNR) can often be improved by means of averaging many heart beats together. This does require the temporal location of the signal to be known, as well as having a repeating signal, and having the signal and noise be uncorrelated[4]. This improved signal can then be used to diagnose cardiac conditions in a clinic[2]. In this project I created an algorithm that is capable of identifying the R peak of a QRS complex and utilizing that peak to average multiple QRS-synced heart beats together.

Signal Processing Algorithm Description

The algorithm will detect heartbeats utilizing the Pan-Tompkins algorithm. These heartbeats will be synced by the R peak location and summed into an averaging buffer. The improvement of the SNR will be equal to the $\sqrt{n}$ where n is the max number of epochs averaged. Each time the max number of epochs is reached, the signal average will be output with each progressive heartbeat until a new average is calculated. A flowchart of my algorithm can be seen in Fig. 1.

Filters utilized:

- LPF (11Hz cutoff)
  - $H(z) = (1 - z^{-6})^2 / (1 - z^{-1})^2$
  - $y(nT) = 2y(nT - T) - y(nT - 2T) + x(nT) - 2x(nT - 6T) + x(nT - 12T)$
  - Group delay = -5T
- HPF (5.4Hz cutoff)
  - $H(z) = (- (1/32) + z^{-16} - z^{-17} - (1/32)\, z^{-32}) / (1 - z^{-1})$
  - $y(nT) = y(nT - T) - (1/32)\, x(nT) + x(nT - 16T)$
    - $x(nT - 17T) + (1/32)\, x(nT - 32T)$
  - Group delay = -16T
- Derivative
  - $H(z) = (⅛) (2 + z^{-1} - z^{-3} - 2z^{-4})$
  - $y(nT) = (¼)\, x(nT) + (⅛)\, x(nT - T) - (⅛)\, x(nT - 3T) - (¼)\, x(nT - 4T)$
  - Group delay = -2T
- MWI
  - $y(nT) = (1/32) (x(nT) + x(nT - T) + x(nT - 2T) + \ldots + x(nT - 31T)$
  - Group delay = -32T

New SNR = √n * Original SNR

Digiscope Implementation

The Pan-Tompkins algorithm can be simulated by cascading the built-in PanT_lowpass, PanT_highpass, PanT_Deriv, square function, PanT_MWI, and Pan-Tompkins Thresh function. The signal averager algorithm can be somewhat simulated in Digiscope through the averager function, although the noise does not mimic the noise found in the MIT-BIH files utilized in this project. I therefore didn't have any useful averaging simulations to structure my algorithm around.

Nucleo Implementation

To achieve R peak detection I utilized the Pan-Tompkins algorithm.This algorithm passes the ECG signal through a cascade of filters containing, in order of occurrence, a bandpass of 5.4-11Hz, a derivative filter, a squaring function, and a moving window integrator. The collective group delay of these filters is 55 samples, which will be used as a buffer size in the averager. This assures we will capture the QRS complex that occurs before the beat is officially detected. The moving window integrator output is then analyzed by an adaptive threshold function to classify each new peak as either an R peak or noise. When an R peak is detected, a flag is set to 1, which activates the averager function.

The averager function will activate each time a flag is set to 1. When this occurs the most recent averaged signal will be divided by the number of desired epochs and output to the oscilloscope. A buffer for the group delay of length 55 that collects each new sample, regardless of flag value, will be immediately added into the temporary buffer. This temporary buffer of length 200 will then add the remaining sample positions with each new incoming sample until full. This sequence occurs for each heartbeat detected, summing each sample in its respective position in the temporary buffer. Once the desired number of epochs is reached, the temporary buffer is copied to the signal averager buffer to be output. The temporary buffer and epoch count are then reset and ready to begin the next averaging sequence. Buffer details are shown in Fig. 2.

Testing and Results

The Pan-Tompkins algorithm is capable of detecting heartbeats in the noisy section of the MIT-BIH patient data relatively well even with an SNR of 1:1. The detection capability only increased as the original SNR increased. I did not calculate the PanT detection accuracy explicitly, however I output a blink to the oscilloscope when a beat was detected so I had an idea of what was being fed to the averager. I didn't test the PanT algorithm on Digiscope and I acknowledge I could have received useful statistics by doing this, however my algorithm was still able to function relatively well and in the interest of time I decided not to. I was unable to test the averaging algorithm on Digiscope unfortunately due to the built-in averaging algorithm

functioning differently. The output SNR was also not calculated due to time constraints and lack of an efficient method to do so. Algorithm performance here is therefore subjective.

In order to test my nucleo code I used patient 118 of the MIT-BIH Noise Stress Test Database with SNR ratios of 1:1 and 12:1, with some additional testing of SNR 6:1. The noise in these ECG signals was added in 2 minute intervals and contained baseline drift, EMG, and electrode motion artifact. This noise was identical except for amplitude modulation to artificially change the SNR and was uncorrelated to the signal. All trials were run with the same section of noise ECG from samples 108322 to 151648. All average outputs were accompanied by the live ECG signal and although it didn't match to exactly what was averaged it provided a helpful representation of the averager input.

I began by averaging the clean signal of 118e00 (1:1 SNR) with 4 epochs (SNR improvement of 2) and achieved a near identical signal over the sampled period as seen in Fig. 3. I then tested the noisy signal sections of 118e00. Utilizing epochs of length 16 and 32 (SNR improvement of 4, 5.66 respectively) I was able to sometimes isolate a QRS, although the morphology was less than ideal and still contained undesirable noise. At 100 epochs (SNR improvement of 10) the QRS amplitude could possibly be determined given its amplitude was similar to the clean averaged amplitude, but I don't believe any other information was reliable. Each of these epoch averages are seen in Fig. 4.

ECG 118e12 (12:1 SNR) revealed clear QRS complexes at epochs of 16, 32, and 100 reliably as seen in Fig. 5, although there was an occasional poor average and I would have to wait for the averager to rerun to receive a clean signal. I also tested 118e06 (6:1 SNR) at 100 epochs and was able to receive a clear QRS complex occasionally, although the amplitude was significantly reduced as seen in Fig. 6.

Discussion

My algorithm works well with an original SNR of 12:1 or higher. Once the SNR dips to 6:1 or lower it becomes unable to properly average with the noise provided in this ECG database. It should be considered as well that even with 12:1, sometimes a poor average would occur and the algorithm would have to rerun to achieve a readable waveform. This can be time consuming with higher epoch counts such as 32 and 100. I believe the electrode motion artifact had a significant role in this due to its mimicking of real QRS complexes. Future testing of this algorithm should utilize data absent of electrode motion artifact for additional metrics. Research into how to properly filter the electrode motion artifact should also be conducted.

My output buffer contains 55 samples that are never output by the algorithm as seen in Fig. 2. This is likely due to my use of two separate counters to track the temporary buffer and

output buffer positions. I believe these can be condensed into a single counter and / or my buffer lengths can be adjusted to remove any existing dead space.

In future implementations I would like to add the option to save and output timestamped histories of past averages. This is useful in clinical settings to see longitudinal changes in finer detail. This could possibly be achieved by saving multiple outputs to different arrays and then inputting into the console which history you'd like to view on the oscilloscope. I also would like to implement the ability to detect abnormal beats and average them separately according to their classification. This would require multiple filtering sequences and could be too intensive for the nucleo, but given the diagnostic role of ECG signal averaging this seems to be a necessity for any clinical application.

Conclusion

My signal averager is capable of averaging heartbeats with a SNR of 12:1 or greater utilizing the Pan-Tompkins algorithm for heartbeat detection. Once the SNR drops below this point results become increasingly trivial and at an SNR of 1:1 it appears even high epoch counts don't provide satisfactory data. Additional filtering may be implemented to avoid averaging electrode motion artifact in the future.

References

[1] M. Cain. “Signal-Averaged Electrocardiography”. JACC, vol. 27, pp 238-49, Jan 1996.
[2] K. Gatzoulis. “Signal-averaged electrocardiography: Past, present, and future”. Journal of Arrhythmia, Jan 2018.
[3] medscape.com. “What is monomorphic ventricular tachycardia (VT)?” Available: https://www.medscape.com/answers/159075-67701/what-is-monomorphic-ventricular-tachycardia-vt#:~:text=Monomorphic%20VT%20is%20most%20commonly,%2C%20hypertrophy%2C%20and%20muscle%20degeneration., Dec 05, 2017, [March 30, 2022].
[4]A. Nimunkar, W. Tompkins, Class Lecture, College of Engineering, University of Wisconsin - Madison, Madison, WI, Spring 2022.

Appendix

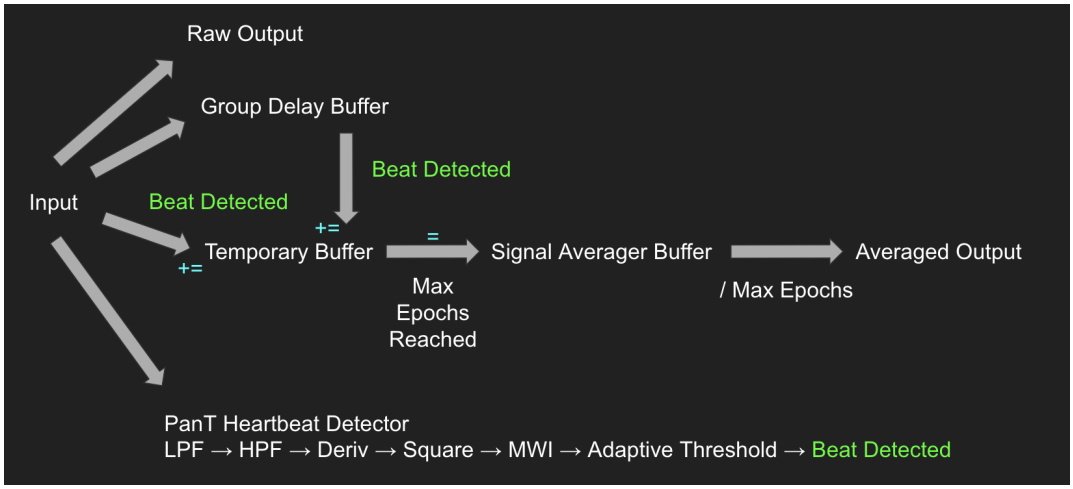

Fig. 1 Flowchart of signal averaging algorithm

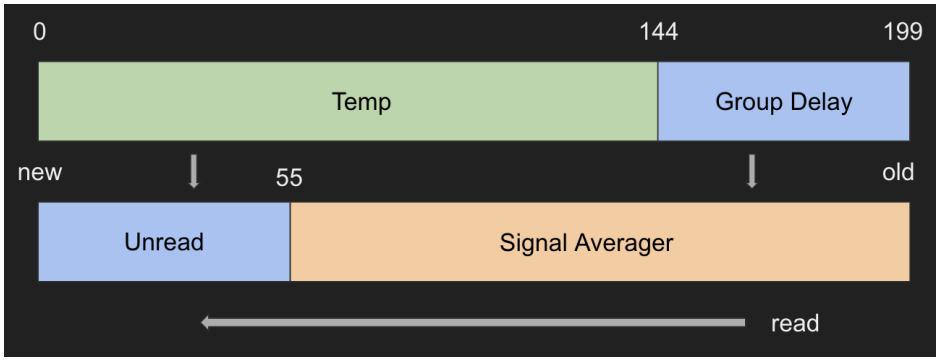

Fig. 2 Organization and flow of buffer system

Fig. 3 Clean signal averaged with 4 epochs
Yellow: averaged signal, Green: original signal, Blue: beat detection pulse

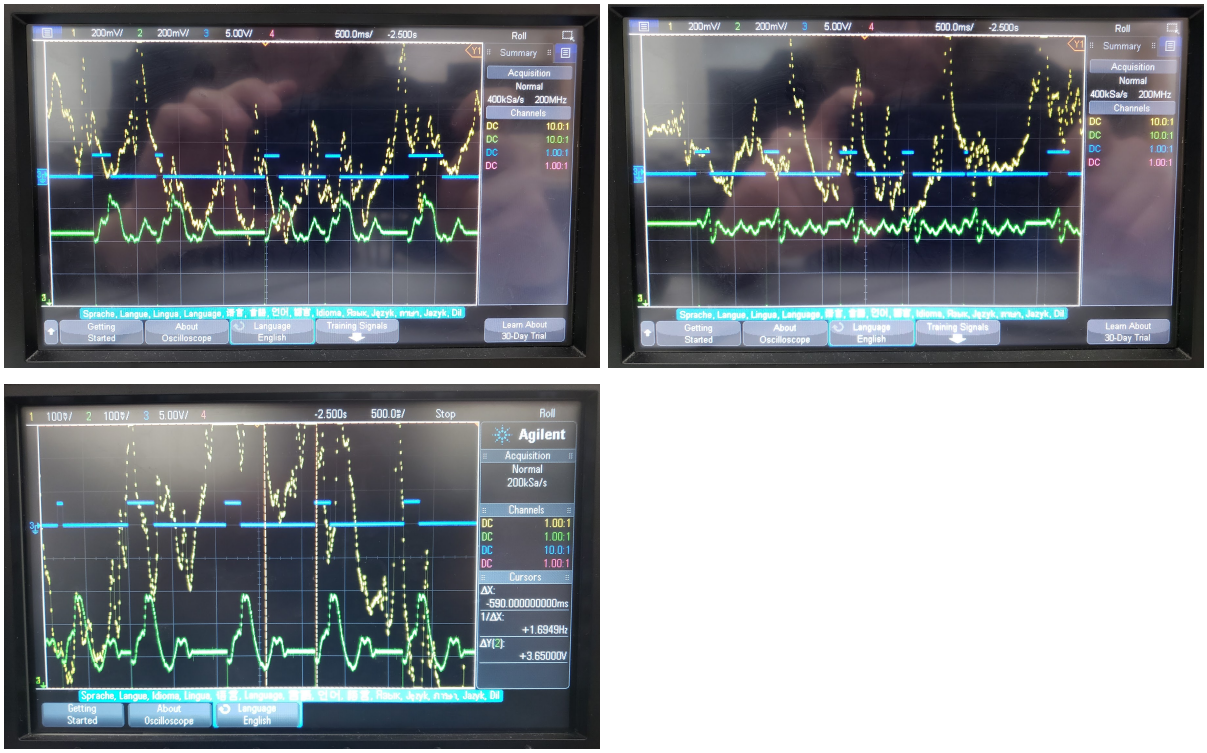

Fig. 4 1:1 SNR ECG averaged 16 (top left), 32 (top right), and 100 (bottom left) epochs. Yellow: original signal, Green: averaged signal, Blue: beat detection pulse

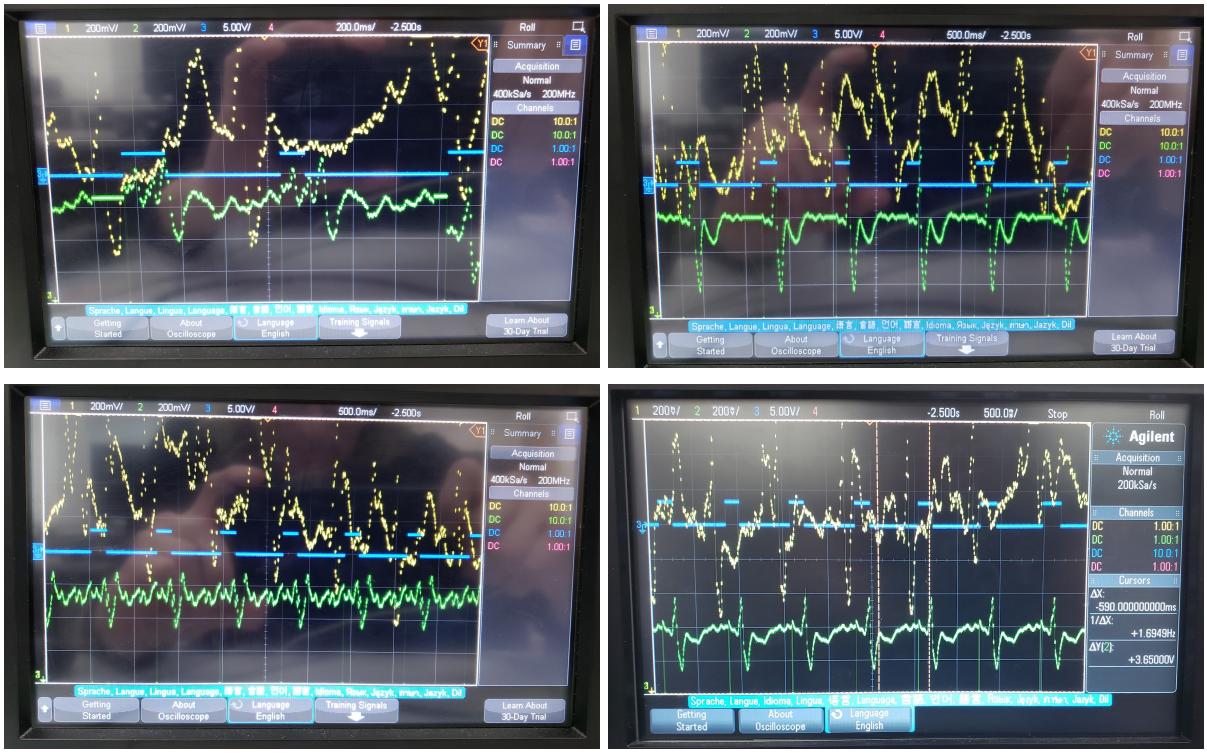

Fig. 5 12:1 SNR ECG averaged 16 (top left), 32 [good average] (top right), 32 [poor average] (bottom left), and 100 (bottom right) epochs.
Yellow: original signal, Green: averaged signal, Blue: beat detection pulse

Fig. 6 6:1 SNR ECG averaged 100 epochs, 2 examples
Yellow: original signal, Green: averaged signal, Blue: beat detection pulse