12

ECG QRS Detection

Valtino X. Afonso

Over the past few years, there has been an increased trend toward processing of the electrocardiogram (ECG) using microcomputers. A survey of literature in this research area indicates that systems based on microcomputers can perform needed medical services in an extremely efficient manner. In fact, many systems have already been designed and implemented to perform signal processing tasks such as 12-lead off-line ECG analysis, Holter tape analysis, and real-time patient monitoring. All these applications require an accurate detection of the QRS complex of the ECG. For example, arrhythmia monitors for ambulatory patients analyze the ECG in real time (Pan and Tompkins, 1985), and when an arrhythmia occurs, the monitor stores a time segment of the abnormal ECG. This kind of monitor requires an accurate QRS recognition capability. Thus, QRS detection is an important part of many ECG signal processing systems.

This chapter discusses a few of the many techniques that have been developed to detect the QRS complex of the ECG. It begins with a discussion of the power spectrum of the ECG and goes on to review a variety of QRS detection algorithms.

12.1 POWER SPECTRUM OF THE ECG

The power spectrum of the ECG signal can provide useful information about the QRS complex. This section reiterates the notion of the power spectrum presented earlier, but also gives an interpretation of the power spectrum of the QRS complex. The power spectrum (based on the FFT) of a set of 512 sample points that contain approximately two heartbeats results in a series of coefficients with a maximal value near a frequency corresponding to the heart rate.

The heart rate can be determined by multiplying together the normalized frequency and the sampling frequency. We can also get useful information about the frequency spectrum of the QRS complex. In order to obtain this information, the QRS complex of the ECG signal must be selected as a template and zero-padded prior to the power spectrum analysis. The peak of the frequency spectrum obtained corresponds to the peak energy of the QRS complex.

The ECG waveform contains, in addition to the QRS complex, P and T waves, 60-Hz noise from powerline interference, EMG from muscles, motion artifact from the electrode and skin interface, and possibly other interference from electrosurgery equipment in the operating room. Many clinical instruments such as a cardiotachometer and an arrhythmia monitor require accurate real-time QRS detection. It is necessary to extract the signal of interest, the QRS complex, from the other noise sources such as the P and T waves. Figure 12.1 summarizes the relative power spectra of the ECG, QRS complexes, P and T waves, motion artifact, and muscle noise based on our previous research (Thakor et al., 1983).

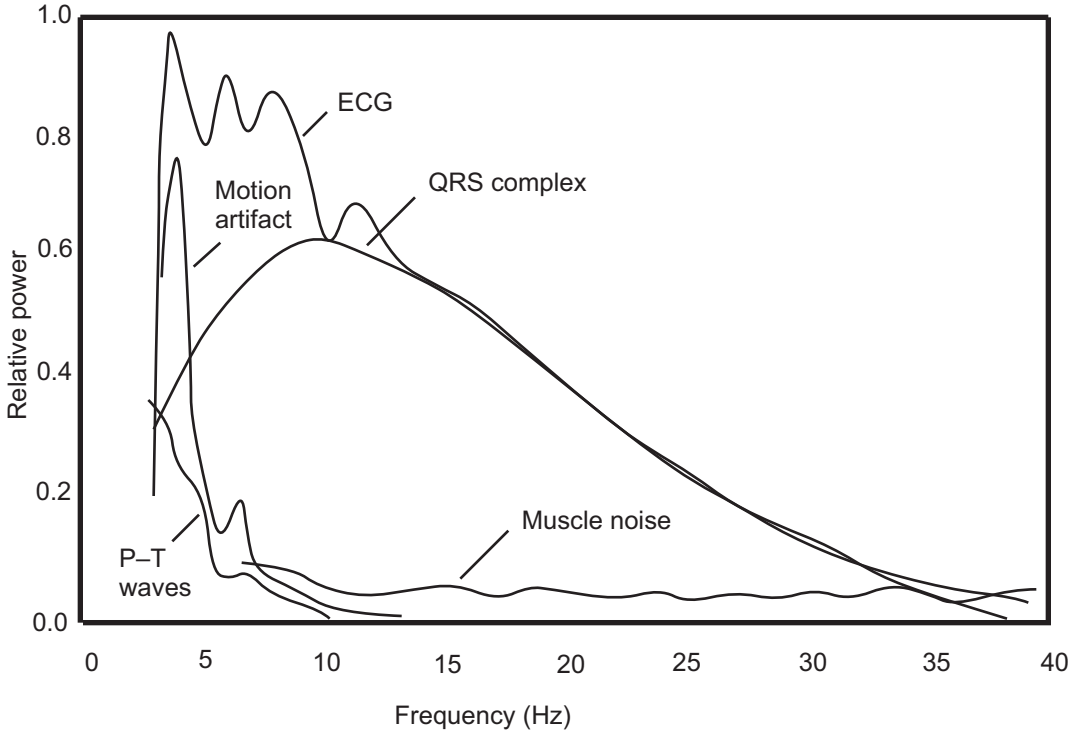


Figure 12.1 Relative power spectra of QRS complex, P and T waves, muscle noise and motion artifacts based on an average of 150 beats.

12.2 BANDPASS FILTERING TECHNIQUES

From the power spectral analysis of the various signal components in the ECG signal, a filter can be designed which effectively selects the QRS complex from the ECG. Another study that we performed examined the spectral plots of the ECG and the QRS complex from 3875 beats (Thakor et al., 1984). Figure 12.2 shows a plot of the signal-to-noise ratio (SNR) as a function of frequency. The study of the power spectra of the ECG signal, QRS complex, and other noises also revealed that a maximum SNR value is obtained for a bandpass filter with a center frequency of 17 Hz and a Q of 3. Section 12.3 and a laboratory experiment examine the effects of different values of the Q of such a filter.

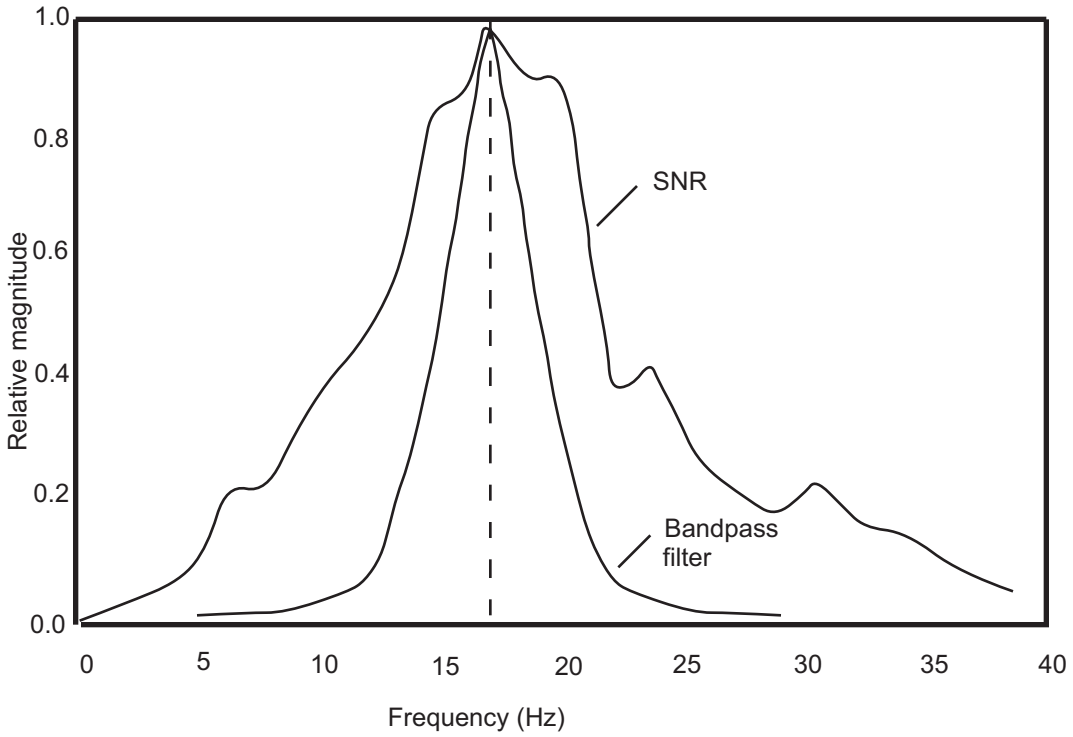


Figure 12.2 Plots of the signal-to-noise ratio (SNR) of the QRS complex referenced to all other signal noise based on 3875 heart beats. The optimal bandpass filter for a cardiotachometer maximizes the SNR.

12.2.1 Two-pole recursive filter

A simple two-pole recursive filter can be implemented in the C language to bandpass the ECG signal. The difference equation for the filter is

$$y(nT) = 1.875y(nT - T) - 0.9219y(nT - 2T) + x\,(nT) - x(nT - 2T) \qquad (12.1)$$

This filter design assumes that the ECG signal is sampled at 500 samples/s. The values of 1.875 and 0.9219 are approximations of the actual design values of 1.87635 and 0.9216 respectively. Since the coefficients are represented as powers of two, the multiplication operations can be implemented relatively fast using the shift operators in the C language. Figure 12.3 displays the code fragment that implements Eq. (12.1).

```
twoPoleRecursive(int data)
{
      static int xnt, xm1, xm2, ynt, ym1, ym2 = 0;
      xnt = data;

      ynt = (ym1 + ym1 >> 1 + ym1 >> 2 + ym1 >> 3) +
            (ym2 >> 1 + ym2 >> 2 + ym2 >> 3 +
             ym2 >> 5 + ym2 >> 6) + xnt - xm2;

      xm2 = xm1;
      xm1 = xnt;
      xm2 = ym1;
      ym2 = ym1;
      ym1 = ynt;
      return(ynt);
}
```

Figure 12.3 C-language code to implement a simple two-pole recursive filter.

Note that in this code, the coefficients 1.87635 and 0.9216 are approximated by

$$1.875 = 1 + \frac{1}{2} + \frac{1}{4} + \frac{1}{8}$$

and

$$0.9219 = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + \frac{1}{32} + \frac{1}{64}$$

12.2.2 Integer filter

An approximate integer filter can be realized using the general form of the transfer function given in Chapter 7. QRS detectors for cardiotachometer applications frequently bandpass the ECG signal using a center frequency of 17 Hz. The denominator of the general form of the transfer function allows for poles at 60˚, 90˚, and 120˚, and these correspond to center frequencies of a bandpass filter of $T/6$, $T/4$, and $T/3$ Hz, respectively. The desired center frequency can thus be obtained by choosing an appropriate sampling frequency.

Ahlstrom and Tompkins (1985) describe a useful filter for QRS detection that is based on the following transfer function:

$$H(z) = \frac{(1 - z^{-12})^2}{(1 - z^{-1} + z^{-2})^2} \tag{12.2}$$

This filter has 24 zeros at 12 different frequencies on the unit circle with poles at ±60°. The ECG signal is sampled at 200 sps, and then the turning point algorithm is used to reduce the sampling rate to 100 sps. The center frequency is at 16.67 Hz and the nominal bandwidth is ±8.3 Hz. The duration of the ringing is approximately 240 ms (the next section explains the effects of different filter Qs). The difference equation to implement this transfer function is

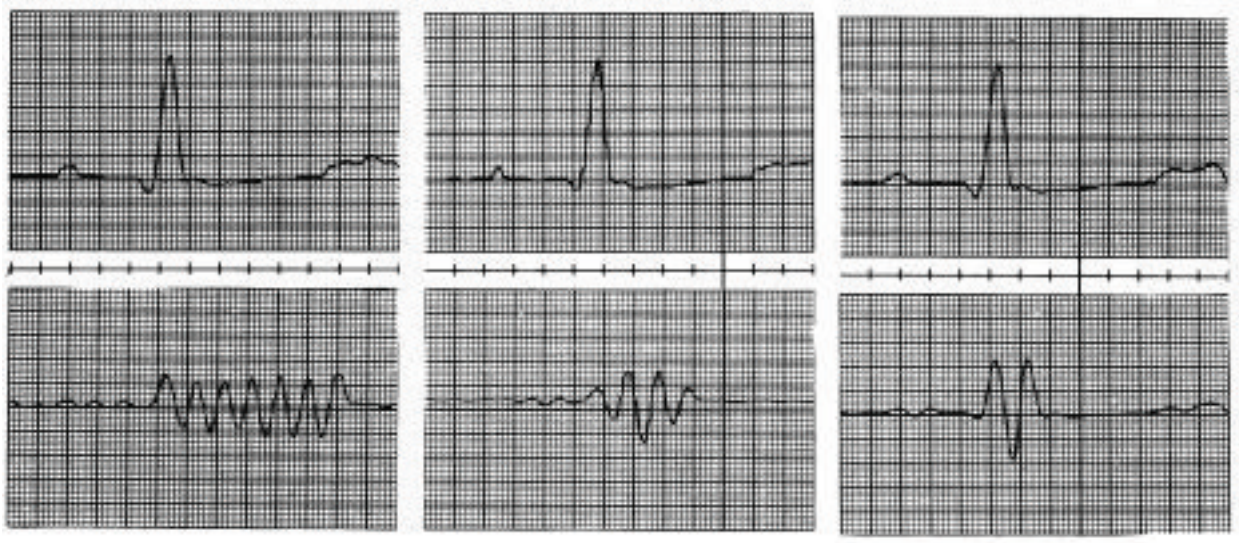
$$y(nT) = 2y(nT - T) - 3y(nT - 2T) + 2y(nT - 3T) - y(nT - 4T) + x(nT) - 2x(nT - 12T) + x(nT - 24T) \quad (12.3)$$

12.2.3 Filter responses for different values of *Q*

The value of Q of the bandpass filter centered at f_C = 17 Hz determines how well the signal of interest is passed without being attenuated. It is also necessary to increase the SNR of the signal of interest; that is, the QRS complex. The Q of the filter is calculated as

$$Q = \frac{f_C}{BW} \quad (12.4)$$

A value of Q that is too high will result in a very oscillatory response (Thakor et al., 1984). The ripples must die down within 200 ms. This is necessary so that the ripples from one QRS complex do not interfere with the ripples from the next one. With a center frequency of 17 Hz, the maximal permissible Q was found to be 5. Figure 12.4 shows the effect of different values of Q. For a bandpass filter with f_C = 17 Hz, a Q value of 5 was found to maximize the SNR (Thakor et al., 1984).



(a) (b) (c)

Figure 12.4 Effects of different values of Q. A higher Q results in a oscillatory transient response. (a) Q = 8. (b) Q = 3. (c) Q = 1.

12.3 DIFFERENTIATION TECHNIQUES

Differentiation forms the basis of many QRS detection algorithms. Since it is basically a high-pass filter, the derivative amplifies the higher frequencies characteristic of the QRS complex while attenuating the lower frequencies of the P and T waves.

An algorithm based on first and second derivatives originally developed by Balda et al. (1977) was modified for use in high-speed analysis of recorded ECGs by Ahlstrom and Tompkins (1983). Friesen et al. (1990) subsequently implemented the algorithm as part of a study to compare noise sensitivity among certain types of QRS detection algorithms. Figure 12.5 shows the signal processing steps of this algorithm.

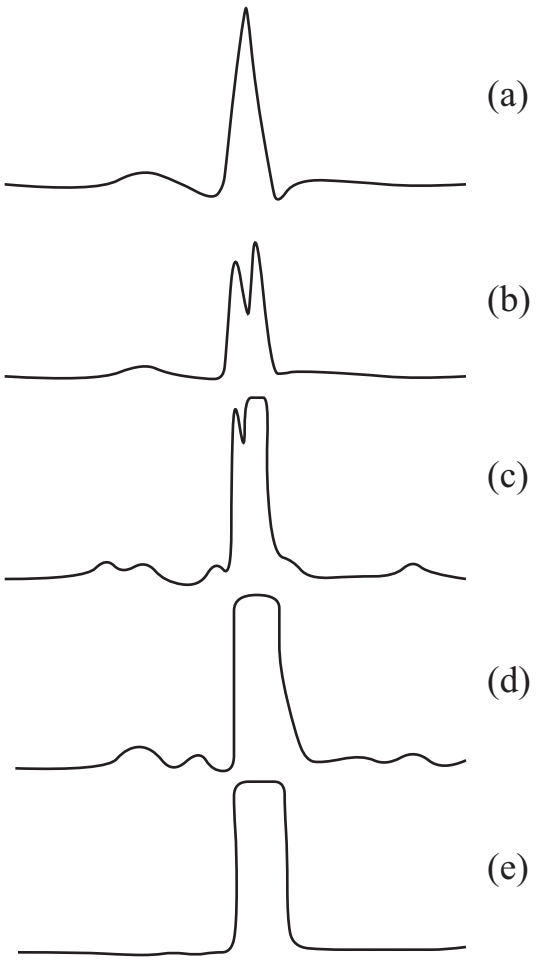


Figure 12.5 Various signal stages in the QRS detection algorithm based on differentiation. (a) Original ECG. (b) Smoothed and rectified first derivative. (c) Smoothed and rectified second derivative. (d) Smoothed sum of (b) and (c). (e) Square pulse output for each QRS complex.

The absolute values of the first and second derivative are calculated from the ECG signal

$$y_0(nT) = | x(nT) - x(nT - 2T) | \tag{12.5}$$

$$y_1(nT) = | x(nT) - 2x(nT - 2T) + x(nT - 4T) | \tag{12.6}$$

These two data buffers, $y_0(nT)$ and $y_1(nT)$, are scaled and then summed

$$y_2(nT) = 1.3y_0(nT) + 1.1y_1(nT) \tag{12.7}$$

The data buffer $y_2(nT)$ is now scanned until a certain threshold is met or exceeded

$$y_2(iT) \geq 1.0 \tag{12.8}$$

Once this condition is met for a data point in $y_2(iT)$, the next eight points are compared to the threshold. If six or more of these eight points meet or exceed the threshold, then the segment might be part of the QRS complex. In addition to detecting the QRS complex, this algorithm has the advantage that it produces a pulse which is proportional in width to the complex. However, a disadvantage is that it is particularly sensitive to higher-frequency noise.

12.4 TEMPLATE MATCHING TECHNIQUES

In this section we discuss techniques for classifying patterns in the ECG signal that are quite related to the human recognition process.

12.4.1 Template crosscorrelation

Signals are said to be correlated if the shapes of the waveforms of two signals match one another. The correlation coefficient is a value that determines the degree of match between the shapes of two or more signals. A QRS detection technique designed by Dobbs et al. (1984) uses crosscorrelation.

This technique of correlating one signal with another requires that the two signals be aligned with one another. In this QRS detection technique, the template of the signal that we are trying to match stores a digitized form of the signal shape that we wish to detect. Since the template has to be correlated with the incoming signal, the signal should be aligned with the template. Dobbs et al. describe two ways of implementing this.

The first way of aligning the template and the incoming signal is by using the fiducial points on each signal. These fiducial points have to be assigned to the signal by some external process. If the fiducial points on the template and the signal are aligned, then the correlation can be performed.

Another implementation involves continuous correlation between a segment of the incoming signal and the template. Whenever a new signal data point arrives, the oldest data point in time is discarded from the segment (a first-in-first-out data structure). A correlation is performed between this signal segment and the template segment that has the same number of signal points. This technique does not require processing time to assign fiducial points to the signal. The template can be thought of as a window that moves over the incoming signal one data point at a time. Thus, alignment of the segment of the signal of interest must occur at least once as the window moves through the signal.

The value of the crosscorrelation coefficient always falls between +1 and –1. A value of +1 indicates that the signal and the template match exactly. As mentioned earlier, the value of this coefficient determines how well the *shapes* of the two waveforms under consideration match. The magnitude of the actual signal samples does not matter. This shape matching, or recognizing process of QRS complexes, conforms with our natural approach to recognizing signals.

12.4.2 Template subtraction

Figure 12.6 illustrates a template subtraction technique. This is a relatively simple QRS detection technique as compared to the other ones described in this chapter.

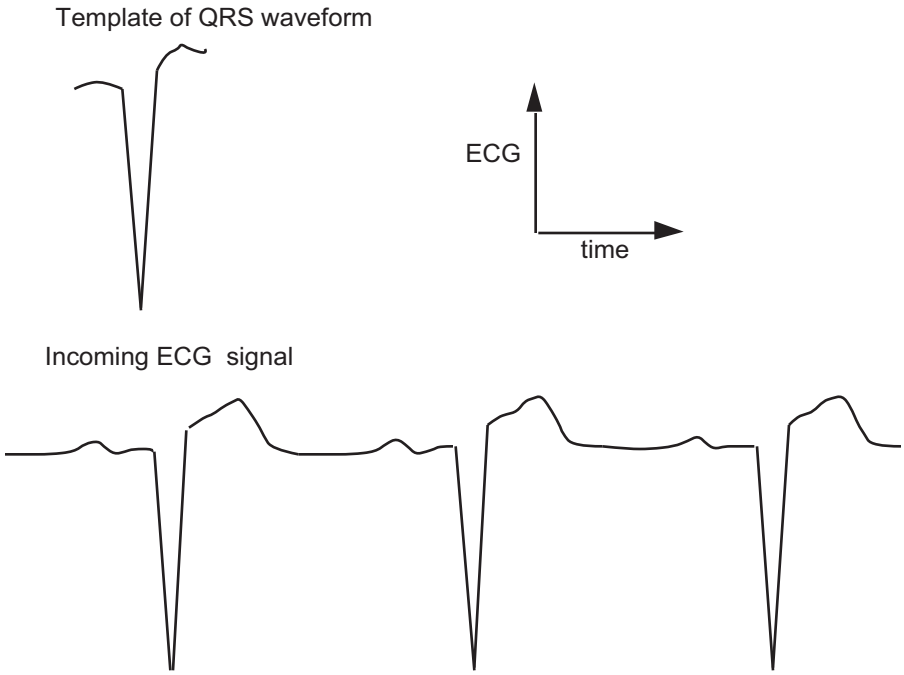


Figure 12.6 In simple template matching, the incoming signal is subtracted, point by point, from the QRS template. If the two waveforms are perfectly aligned, the subtraction results in a zero value.

The algorithm begins by saving a segment of the incoming ECG signal that corresponds to the QRS waveform. This segment or template is then compared with the incoming ECG signal. Each point in the incoming signal is subtracted from the corresponding point in the template. When the template is aligned with a QRS waveform in the signal, the subtraction results in a value very close to zero. This algorithm uses only as many subtraction operations as there are points in the template.

12.4.3 Automata-based template matching

Furno and Tompkins (1982) developed a QRS detector that is based on concepts from automata theory. The algorithm uses some of the basic techniques that are

common in many pattern recognition systems. The ECG signal is first reduced into a set of predefined tokens, which represent certain shapes of the ECG waveform.

Figure 12.7 shows the set of tokens that would represent a normal ECG. Then this set of tokens is input to the finite state automaton defined in Figure 12.8. The finite state automaton is essentially a state-transition diagram that can be implemented with IF … THEN control statements available in most programming languages. The sequence of tokens is fed into the automaton. For example, a sequence of tokens such as *zero*, *normup*, *normdown*, and *normup* would result in the automaton signaling a normal classification for the ECG.

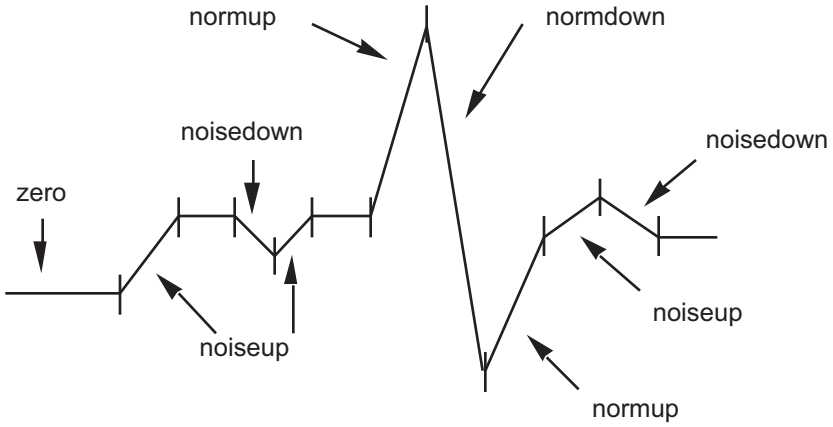


Figure 12.7 Reduction of an ECG signal to tokens.

The sequence of tokens must be derived from the ECG signal data. This is done by forming a sequence of the differences of the input data. Then the algorithm groups together those differences that have the same sign and also exceed a certain predetermined threshold level. The algorithm then sums the differences in each of the groups and associates with each group this sum and the number of differences that are in it.

This QRS detector has an initial learning phase where the program approximately determines the peak magnitude of a normal QRS complex. Then the algorithm detects a normal QRS complex each time there is a deflection in the waveform with a magnitude greater than half of the previously determined peak. The algorithm now teaches the finite state automaton the sequence of tokens that make up a normal QRS complex. The number and sum values (discussed in the preceding paragraph) for a normal QRS complex are now set to a certain range of their respective values in the QRS complex detected.

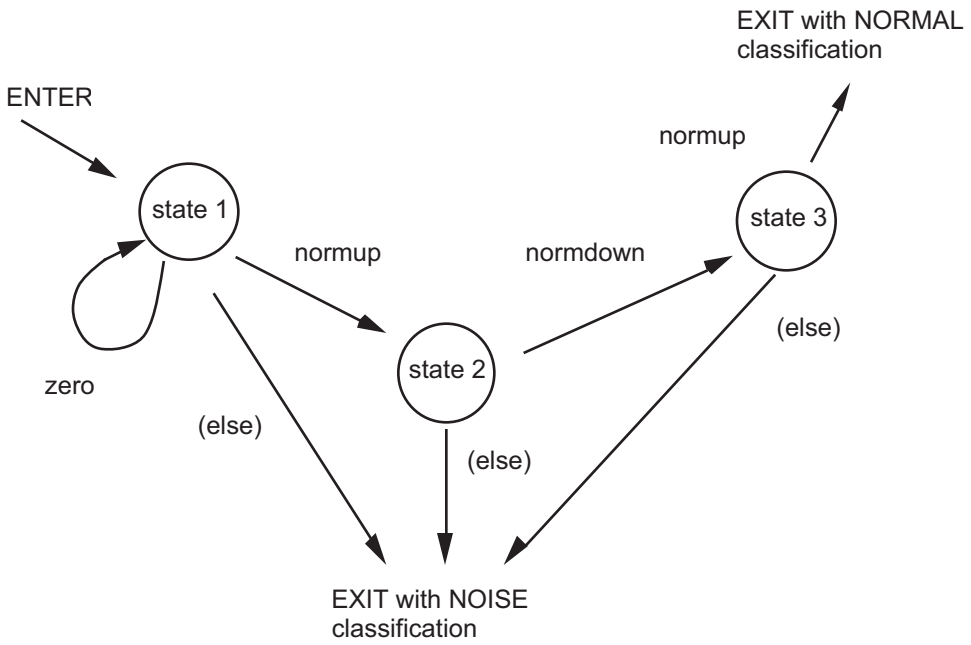


Figure 12.8 State-transition diagram for a simple automaton detecting only normal QRS complexes and noise. The state transition (else) refers to any other token not labeled on a state transition leaving a particular state.

The algorithm can now assign a waveform token to each of the groups formed previously based on the values of the number and the sum in each group of differences. For example, if a particular group of differences has a sum and number value in the ranges (determined in the learning phase) of a QRS upward or downward deflection, then a *normup* or *normdown* token is generated for that group of differences. If the number and sum values do not fall in this range, then a *noiseup* or *noisedown* token is generated. A *zero* token is generated if the sum for a group of differences is zero. Thus, the algorithm reduces the ECG signal data into a sequence of tokens, which can be fed to the finite state automata for QRS detection.

12.5 A QRS DETECTION ALGORITHM

A real-time QRS detection algorithm developed by Pan and Tompkins (1985) was further described by Hamilton and Tompkins (1986). It recognizes QRS complexes based on analyses of the slope, amplitude, and width.

Figure 12.9 shows the various filters involved in the analysis of the ECG signal. In order to attenuate noise, the signal is passed through a bandpass filter composed of cascaded high-pass and low-pass integer filters. Subsequent processes are differentiation, squaring, and time averaging of the signal.

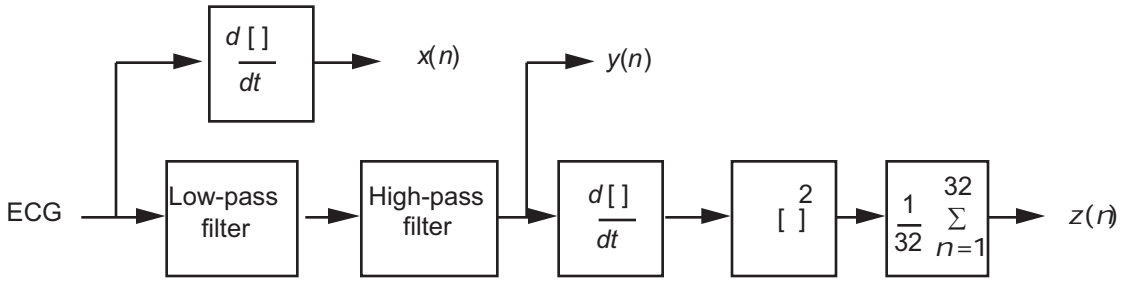


Figure 12.9 Filter stages of the QRS detector. $z(n)$ is the time-averaged signal. $y(n)$ is the bandpassed ECG, and $x(n)$ is the differentiated ECG.

We designed a bandpass filter from a special class of digital filters that require only integer coefficients. This permits the microprocessor to do the signal processing using only integer arithmetic, thereby permitting real-time processing speeds that would be difficult to achieve with floating-point processing. Since it was not possible to directly design the desired bandpass filter with this special approach, the design actually consists of cascaded low-pass and high-pass filter sections. This filter isolates the predominant QRS energy centered at 10 Hz, attenuates the low frequencies characteristic of P and T waves and baseline drift, and also attenuates the higher frequencies associated with electromyographic noise and power line interference.

The next processing step is differentiation, a standard technique for finding the high slopes that normally distinguish the QRS complexes from other ECG waves. To this point in the algorithm, all the processes are accomplished by linear digital filters.

Next is a nonlinear transformation that consists of point-by-point squaring of the signal samples. This transformation serves to make all the data positive prior to subsequent integration, and also accentuates the higher frequencies in the signal obtained from the differentiation process. These higher frequencies are normally characteristic of the QRS complex.

The squared waveform passes through a moving window integrator. This integrator sums the area under the squared waveform over a 150-ms interval, advances one sample interval, and integrates the new 150-ms window. We chose the window's width to be long enough to include the time duration of extended abnormal QRS complexes, but short enough so that it does not overlap both a QRS complex and a T wave.

Adaptive amplitude thresholds applied to the bandpass-filtered waveform and to the moving integration waveform are based on continuously updated estimates of the peak signal level and the peak noise. After preliminary detection by the adaptive thresholds, decision processes make the final determination as to whether or not a detected event was a QRS complex.

A measurement algorithm calculates the QRS duration as each QRS complex is detected. Thus, two waveform features are available for subsequent arrhythmia analysis—RR interval and QRS duration.

Each of the stages in this QRS detection technique are explained in the following sections. Figure 12.10 is a sampled ECG that will serve as an example input signal for the processing steps to follow.

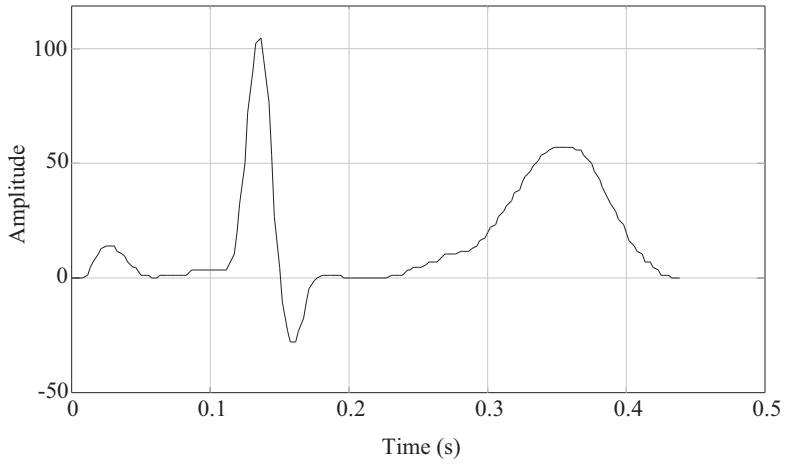


Figure 12.10 Electrocardiogram sampled at 200 samples per second.

12.5.1 Bandpass integer filter

The bandpass filter for the QRS detection algorithm reduces noise in the ECG signal by matching the spectrum of the average QRS complex. Thus, it attenuates noise due to muscle noise, 60-Hz interference, baseline wander, and T-wave interference. The passband that maximizes the QRS energy is approximately in the 5–15 Hz range, as discussed in section 12.1. The filter implemented in this algorithm is a recursive integer filter in which poles are located to cancel the zeros on the unit circle of the z plane. A low-pass and a high-pass filter are cascaded to form the bandpass filter.

Low-pass filter

The transfer function of the second-order low-pass filter is

$$H(z) = \frac{(1 - z^{-6})^2}{(1 - z^{-1})^2} \tag{12.9}$$

The difference equation of this filter is

$$y(nT) = 2y(nT - T) - y(nT - 2T) + x(nT) - 2x(nT - 6T) + x(nT - 12T) \tag{12.10}$$

The cutoff frequency is about 11 Hz, the delay is five samples (or 25 ms for a sampling rate of 200 sps), and the gain is 36. Figure 12.11 displays the C-language program that implements this low-pass filter. In order to avoid saturation, the output is divided by 32, the closest integer value to the gain of 36 that can be implemented with binary shift arithmetic.

```
int LowPassFilter(int data)
{
      static int y1 = 0, y2 = 0, x[26], n = 12;
      int y0;

      x[n] = x[n + 13] = data;
      y0 = (y1 << 1) - y2 + x[n] - (x[n + 6] << 1) + x[n + 12];
      y2 = y1;
      y1 = y0;
      y0 >>= 5;
      if(--n < 0)
            n = 12;

      return(y0);
}
```

Figure 12.11 C-language program to implement the low-pass filter.

Figure 12.12 shows the performance details of the low-pass filter. This filter has purely linear phase response. Note that there is more than 35-dB attenuation of the frequency corresponding to 0.3 f/f_S. Since the sample rate is 200 sps for these filters, this represents a frequency of 60 Hz. Therefore, power line noise is significantly attenuated by this filter. Also all higher frequencies are attenuated by more than 25 dB.

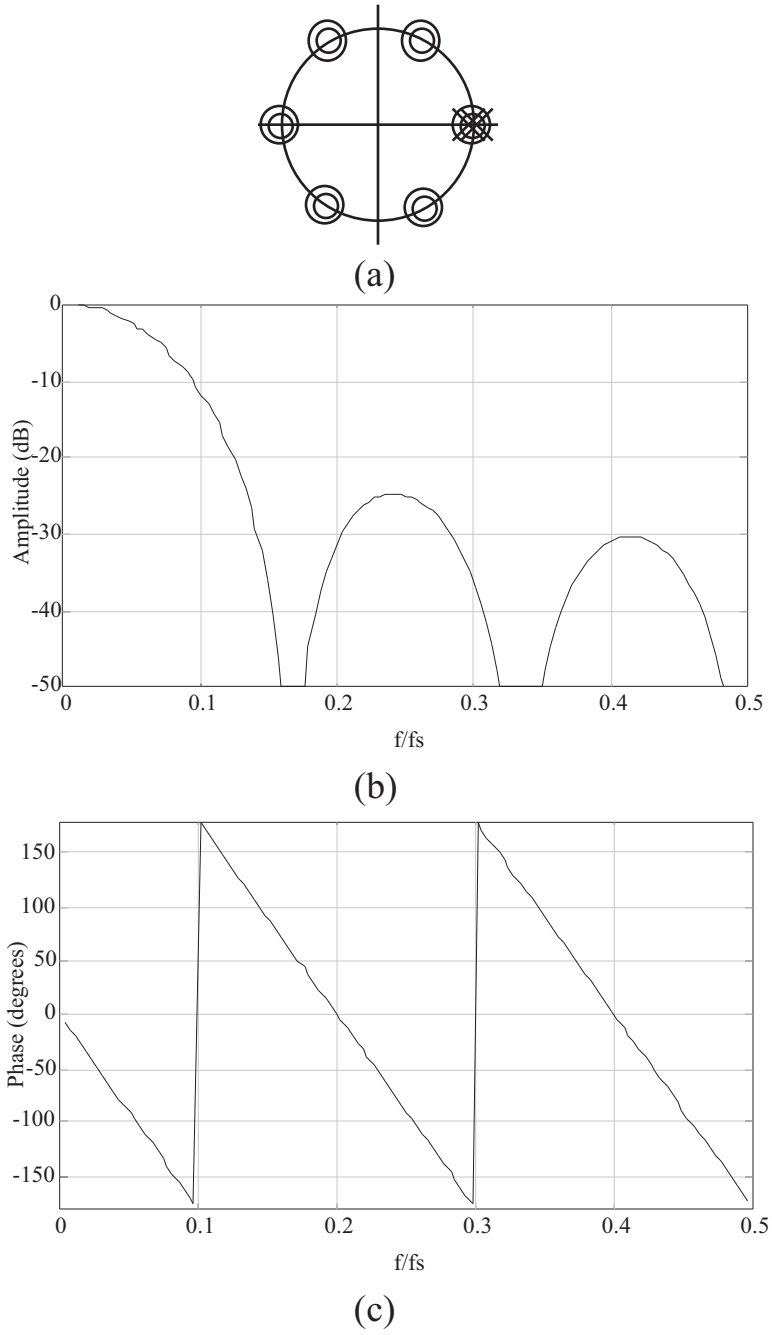


Figure 12.12 Low-pass filter. a) Pole-zero plot. b) Amplitude response. c) Phase response.

Figure 12.13 shows the ECG of Figure 12.10 after processing with the low-pass filter. The most noticeable result is the attenuation of the higher frequency QRS complex. Any 60-Hz noise or muscle noise present would have also been significantly attenuated.

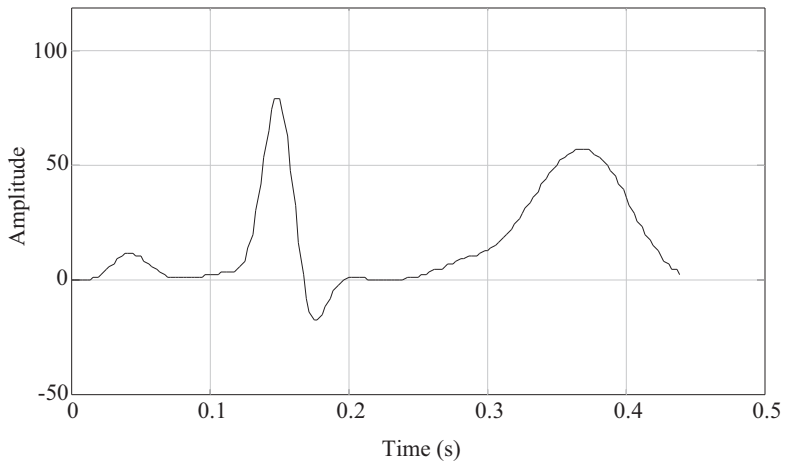


Figure 12.13 Low-pass filtered ECG.

High-pass filter

Figure 12.14 shows how the high-pass filter is implemented by subtracting a first-order low-pass filter from an all-pass filter with delay. The low-pass filter is an integer-coefficient filter with the transfer function

$$H_{lp}(z) = \frac{Y(z)}{X(z)} = \frac{1 - z^{-32}}{1 - z^{-1}} \tag{12.11}$$

and the difference equation

$$y(nT) = y(nT - T) + x(nT) - x(nT - 32T) \tag{12.12}$$

This filter has a dc gain of 32 and a delay of 15.5 samples.

The high-pass filter is obtained by dividing the output of the low-pass filter by its dc gain and then subtracting from the original signal. The transfer function of the high-pass filter is

$$H_{hp}(z) = \frac{P(z)}{X(z)} = z^{-16} - \frac{H_{lp}(z)}{32} \tag{12.13}$$

The difference equation for this filter is

$$p(nT) = \mathrm{x}(nT - 16T) - \frac{1}{32}\,[y(nT - T) + x(nT) - x(nT - 32T)] \tag{12.14}$$

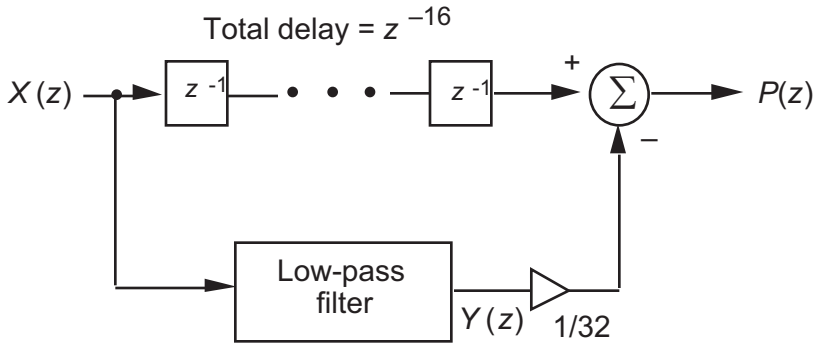


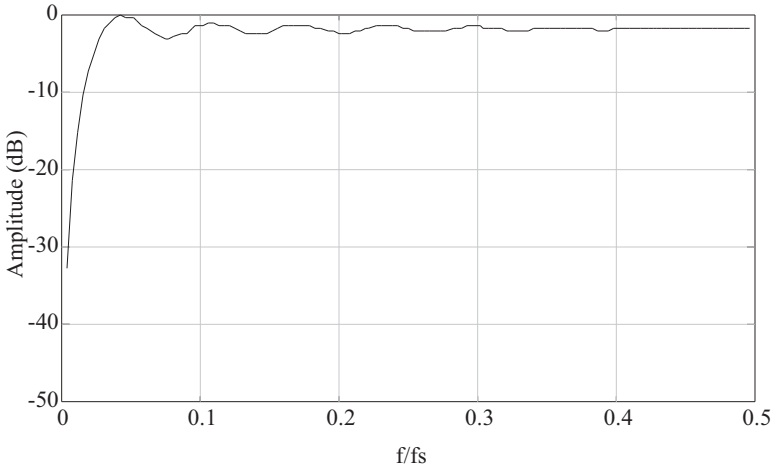
Figure 12.14 The high-pass filter is implemented by subtracting a low-pass filter from an all-pass filter with delay.

The low cutoff frequency of this filter is about 5 Hz, the delay is about 16*T* (or 80 ms), and the gain is 1. Figure 12.15 shows the C-language program that implements this high-pass filter.

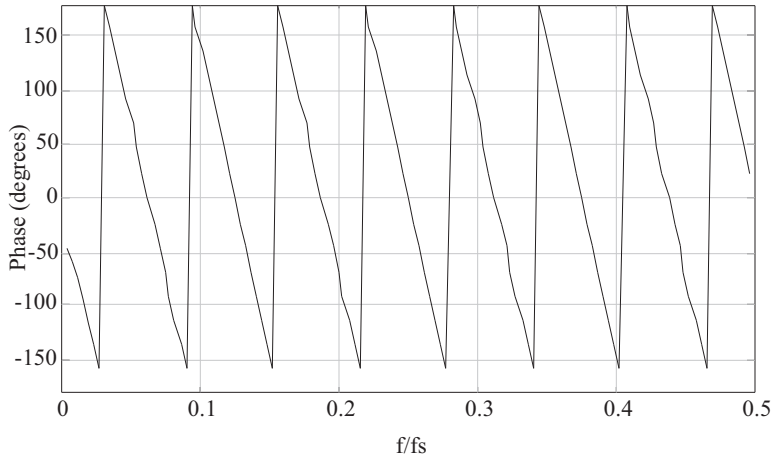
```
int HighPassFilter(int data)
{
      static int y1 = 0, x[66], n = 32;
      int y0;

      x[n] = x[n + 33] = data;
      y0 = y1 + x[n] - x[n + 32];
      y1 = y0;
      if(--n < 0)
            n = 32;

      return(x[n + 16] - (y0 >> 5));
}
```

Figure 12.15 C-language program to implement the high-pass filter.

Figure 12.16 shows the performance characteristics of the high-pass filter. Note that this filter also has purely linear phase response.



(a)



(b)

Figure 12.16 High-pass filter. a) Amplitude response. b) Phase response.

Figure 12.17 shows the amplitude response of the bandpass filter which is composed of the cascade of the low-pass and high-pass filters. The center frequency of the passband is at 10 Hz. The amplitude response of this filter is designed to approximate the spectrum of the average QRS complex as illustrated in Figure 12.1. Thus this filter optimally passes the frequencies characteristic of a QRS complex while attenuating lower and higher frequency signals.

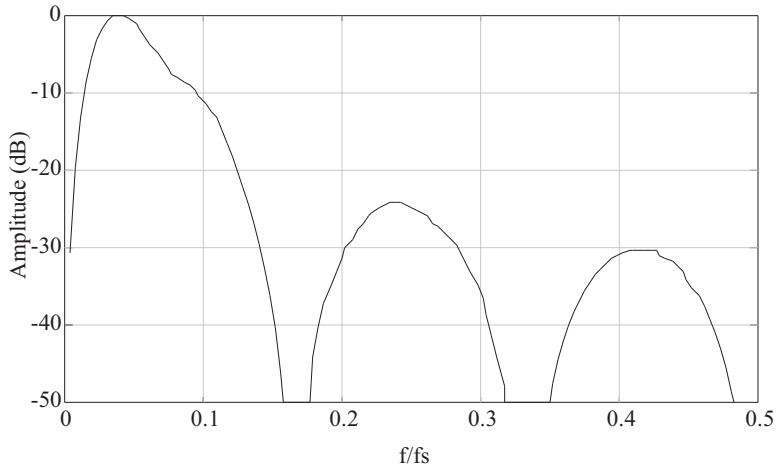


Figure 12.17 Amplitude response of bandpass filter composed of low-pass and high-pass filters.

Figure 12.18 is the resultant signal after the ECG of Figure 12.10 passes through the bandpass filter. Note the attenuation of the T wave due to the high-pass filter.

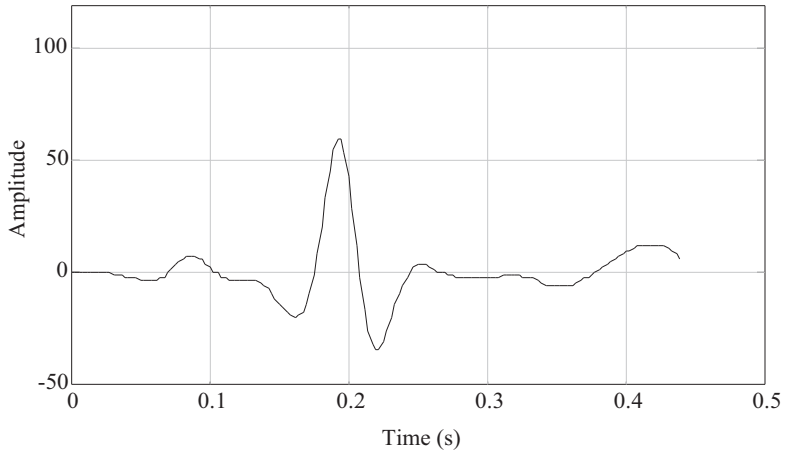


Figure 12.18 Bandpass-filtered ECG.

12.5.2 Derivative

After the signal has been filtered, it is then differentiated to provide information about the slope of the QRS complex. A five-point derivative has the transfer function

$$H(z) = 0.1\,(2 + z^{-1} - z^{-3} - 2z^{-4}) \tag{12.15}$$

This derivative is implemented with the difference equation

$$y(nT) = \frac{2x(nT) + x(nT - T) - x(nT - 3T) - 2x(nT - 4T)}{8} \tag{12.16}$$

The fraction 1/8 is an approximation of the actual gain of 0.1. Throughout these filter designs, we approximate parameters with power-of-two values to facilitate real-time operation. These power-of-two calculations are implemented in the C language by shift-left or shift-right operations.

This derivative approximates the ideal derivative in the dc through 30-Hz frequency range. The derivative has a filter delay of $2T$ (or 10 ms). Figure 12.19 shows the C-language program for implementing this derivative.

```
int Derivative(int data)
{
      int y, i;
      static int x_derv[4];

      /*y = 1/8 (2x(  nT) + x( nT - T) - x(  nT - 3T) - 2x(  nT -
4T))*/

      y = (data << 1) +   x_derv[3] -  x_derv[1] - ( x_derv[0] <<
1);

      y >>= 3;
      for (i = 0; i < 3; i++)
            x_derv[i] = x_derv[i + 1];
      x_derv[3] = data;

      return(y);
}
```

Figure 12.19 C-language program to implement the derivative.

Figure 12.20 shows the performance characteristics of this derivative implementation. Note that the amplitude response approximates that of a true derivative up to about 20 Hz. This is the important frequency range since all higher frequencies are significantly attenuated by the bandpass filter.

Figure 12.21 is the resultant signal after passing through the cascade of filters including the differentiator. Note that P and T waves are further attenuated while the peak-to-peak signal corresponding to the QRS complex is further enhanced.

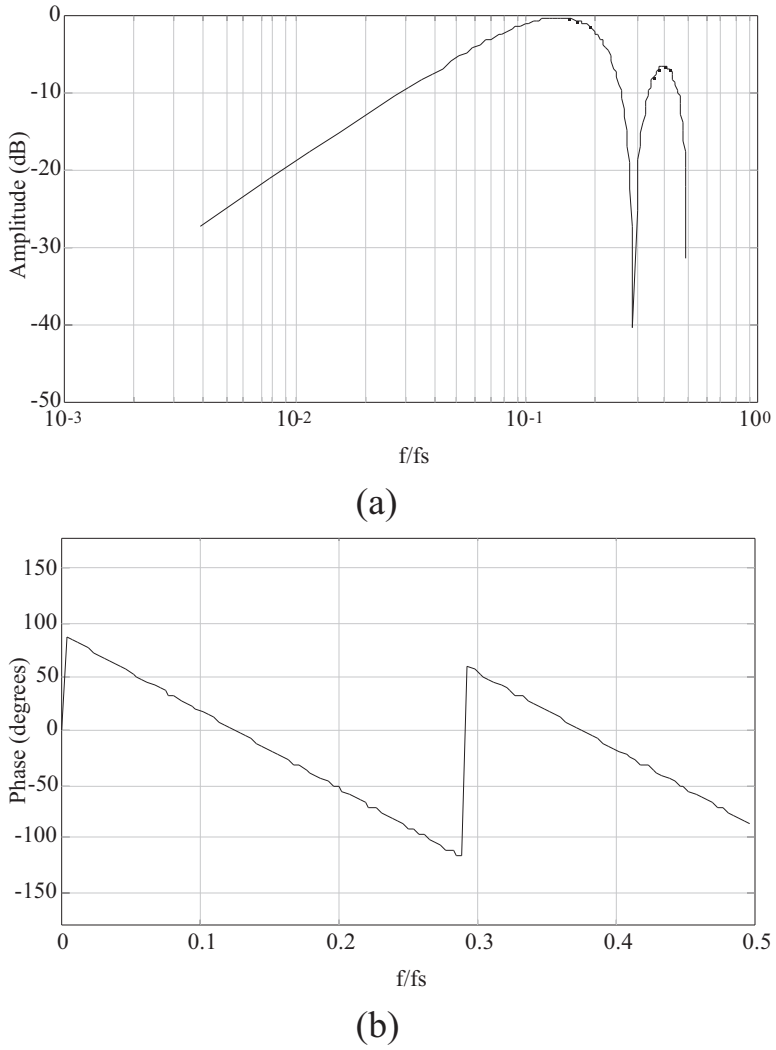


Figure 12.20 Derivative. a) Amplitude response. b) Phase response.

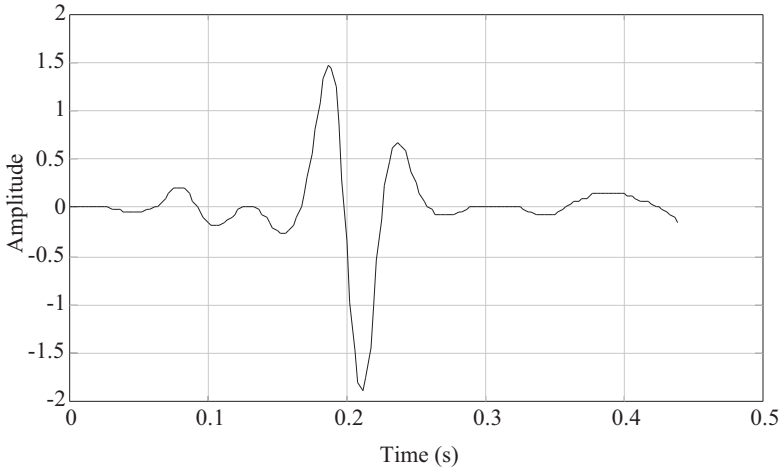


Figure 12.21 ECG after bandpass filtering and differentiation.

12.5.3 Squaring function

The previous processes and the moving-window integration, which is explained in the next section, are linear processing parts of the QRS detector. The squaring function that the signal now passes through is a nonlinear operation. The equation that implements this operation is

$$y(nT) = [x(nT)]^2 \tag{12.17}$$

This operation makes all data points in the processed signal positive, and it amplifies the output of the derivative process nonlinearly. It emphasizes the higher frequencies in the signal, which are mainly due to the QRS complex. A fact to note in this operation is that the output of this stage should be hardlimited to a certain maximum level corresponding to the number of bits used to represent the data type of the signal. Figure 12.22 shows the results of this processing for our sample ECG.

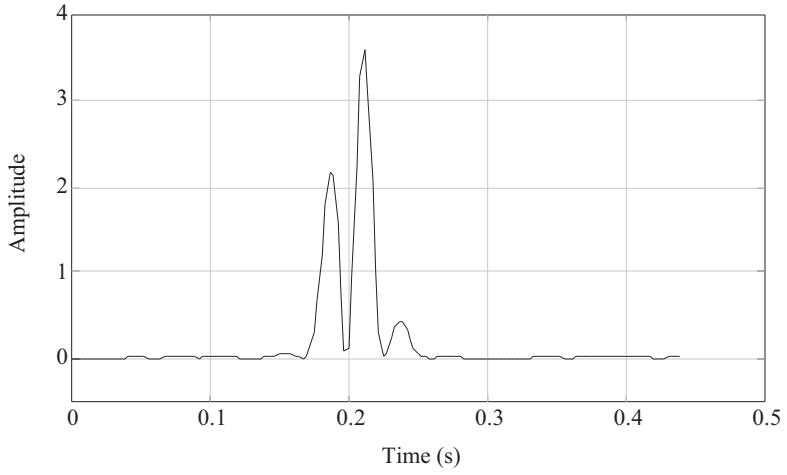


Figure 12.22 ECG signal after squaring function.

12.5.4 Moving window integral

The slope of the R wave alone is not a guaranteed way to detect a QRS event. Many abnormal QRS complexes that have large amplitudes and long durations (not very steep slopes) might not be detected using information about slope of the R wave only. Thus, we need to extract more information from the signal to detect a QRS event.

Moving window integration extracts features in addition to the slope of the R wave. It is implemented with the following difference equation:

$$y(nT) = \frac{1}{N}\,[x(nT - (N-1)T) + x(nT - (N-2)T) + \ldots + x(nT)] \tag{12.18}$$

where N is the number of samples in the width of the moving window. The value of this parameter should be chosen carefully.

Figure 12.23 shows the output of the moving window integral for the sample ECG of Figure 12.10. Figure 12.24 illustrates the relationship between the QRS complex and the window width. The width of the window should be approximately the same as the widest possible QRS complex. If the size of the window is too large, the integration waveform will merge the QRS and T complexes together. On the other hand, if the size of the window is too small, a QRS complex could produce several peaks at the output of the stage. The width of the window should be chosen experimentally. For a sample rate of 200 sps, the window chosen for this algorithm was 30 samples wide (which corresponds to 150 ms).

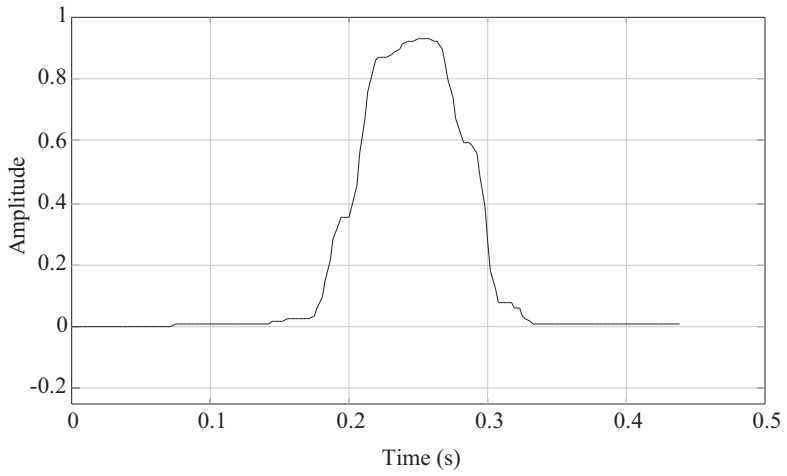


Figure 12.23 Signal after moving window integration.

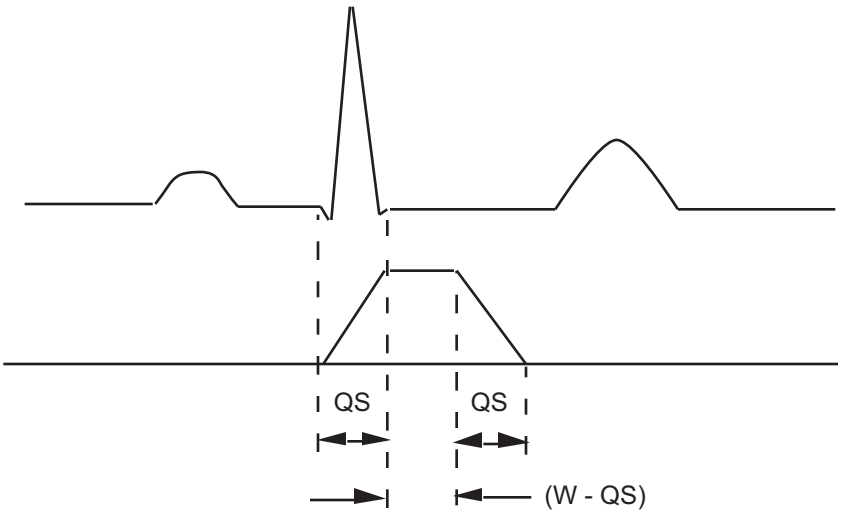


Figure 12.24 The relationship of a QRS complex to the moving integration waveform. (a) ECG signal. (b) Output of moving window integrator. QS: QRS width. W: width of the integrator window.

Figure 12.25 shows the C-language program that implements the moving window integration.

```
int MovingWindowIntegral(int data)
{
      static int x[32], ptr = 0;
      static long sum = 0;
      long ly;
      int y;

      if(++ptr == 32)
            ptr = 0;
      sum -= x[ptr];
      sum += data;
      x[ptr] = data;
      ly = sum >> 5;
      if(ly > 32400)       /*check for register overflow*/
            y = 32400;
      else
            y = (int) ly;

      return(y);
}
```

Figure 12.25 C-language program to implement the moving window integration.

Figure 12.26 shows the appearance of some of the filter outputs of this algorithm. Note the processing delay between the original ECG complexes and corresponding waves in the moving window integral signal.

12.5.5 Thresholding

The set of thresholds that Pan and Tompkins (1985) used for this stage of the QRS detection algorithm were set such that signal peaks (i.e., valid QRS complexes) were detected. Signal peaks are defined as those of the QRS complex, while noise peaks are those of the T waves, muscle noise, etc. After the ECG signal has passed through the bandpass filter stages, its signal-to-noise ratio increases. This permits the use of thresholds that are just above the noise peak levels. Thus, the overall sensitivity of the detector improves.

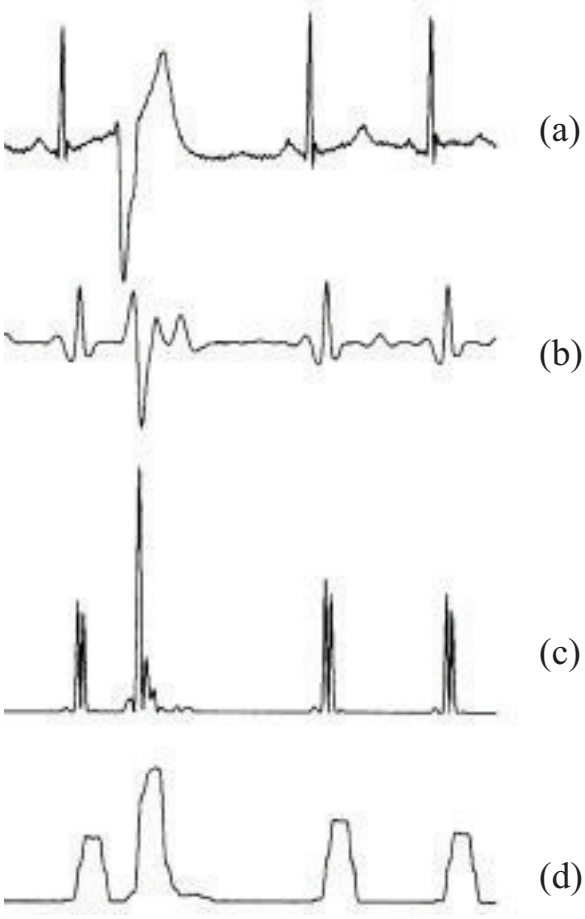


Figure 12.26 QRS detector signals. (a) Unfiltered ECG. (b) Output of bandpass filter. (c) Output after bandpass, differentiation, and squaring processes. (d) Final moving-window integral.

Two sets of thresholds are used, each of which has two threshold levels. The set of thresholds that is applied to the waveform from the moving window integrator is

$$SPKI = 0.125\ PEAKI + 0.875\ SPKI \qquad \text{if } PEAKI \text{ is the signal peak}$$

$$NPKI = 0.125\ PEAKI + 0.875\ NPKI \qquad \text{if } PEAKI \text{ is the noise peak}$$

$$THRESHOLD\ I1 = NPKI + 0.25\ (SPKI - NPKI)$$

$$THRESHOLD\ I2 = 0.5\ THRESHOLD\ I1$$

All the variables in these equations refer to the signal of the integration waveform and are described below:

PEAKI is the overall peak.
SPKI is the running estimate of the signal peak.
NPKI is the running estimate of the noise peak.
THRESHOLD I1 is the first threshold applied.
THRESHOLD I2 is the second threshold applied.

A peak is determined when the signal changes direction within a certain time interval. Thus, *SPKI* is the peak that the algorithm has learned to be that of the QRS complex, while *NPKI* peak is any peak that is not related to the signal of interest. As can be seen from the equations, new values of thresholds are calculated from previous ones, and thus the algorithm adapts to changes in the ECG signal from a particular person.

Whenever a new peak is detected, it must be categorized as a noise peak or a signal peak. If the peak level exceeds *THRESHOLD I1* during the first analysis of the signal, then it is a QRS peak. If searchback technique (explained in the next section) is used, then the signal peak should exceed *THRESHOLD I2* to be classified as a QRS peak. If the QRS complex is found using this second threshold level, then the peak value adjustment is twice as fast as usual:

$$SPKI = 0.25\ PEAKI + 0.75\ SPKI$$

The output of the final filtering stages, after the moving window integrator, must be detected for peaks. A peak detector algorithm finds peaks and a detection algorithm stores the maximum levels at this stage of the filtered signal since the last peak detection. A new peak is defined only when a level that is less than half the height of the peak level is reached. Figure 12.27 illustrates that this occurs halfway down the falling edge of the peak (Hamilton and Tompkins, 1986).

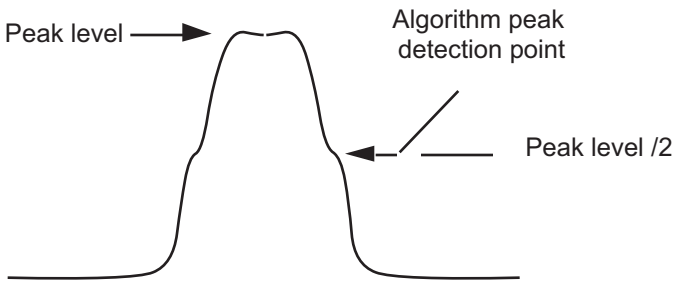


Figure 12.27 Output after the moving window integrator, with peak detection point.

12.5.6 Searchback technique

To implement the searchback technique, this algorithm maintains two RR-interval averages. One average, *RR AVERAGE1*, is that of the eight most recent heartbeats.

The other average, *RR AVERAGE2*, is the average of the eight most recent beats which had *RR* intervals that fell within a certain range.

$$RR\ AVERAGE1 = 0.125\ (RR_{n-7} + RR_{n-6} + \ldots + RR_n)$$
$$RR\ AVERAGE2 = 0.125\ (RR'_{n-7} + RR'_{n-6} + \ldots + RR'_n)$$

The RR'_n values are the *RR* intervals that fell within the following limits:

$$RR\ LOW\ LIMIT = 92\% \times RR\ AVERAGE2$$
$$RR\ HIGH\ LIMIT = 116\% \times RR\ AVERAGE2$$

Whenever the QRS waveform is not detected for a certain interval, *RR MISSED LIMIT*, then the QRS is the peak between the established thresholds mentioned in the previous section that are applied during searchback.

$$RR\ MISSED\ LIMIT = 166\ \% \times RR\ AVERAGE2$$

The heart rate is said to be normal if each of the eight most recent RR intervals are between the limits established by *RR LOW LIMIT* and *RR HIGH LIMIT*.

12.5.7 Performance measurement

We tested the performance of the algorithm on the 24-hour annotated MIT/BIH database, which is composed of half-hour recordings of ECGs of 48 ambulatory patients (see references, MIT/BIH ECG database). This database, available on CD ROM, was developed by Massachusettsa Institute of Technology and Beth Israel Hospital. The total error in analyzing about 116,000 beats is 0.68 percent, corresponding to an average error rate of 33 beats per hour. In fact, much of the error comes from four particular half-hour tape segments (i.e., two hours of data from the total database).

Figure 12.28 shows the effect of excluding the four most problematic half-hour tapes from the overall results. Notice that the false-positive errors decrease much more than do the false negatives. This difference indicates that this algorithm is more likely to misclassify noise as a QRS complex than it is to miss a real event. Elimination of these four half-hour tape segments reduces the error rate below 10 beats per hour. Another available ECG database was developed by the American Heart Association (see references, AHA ECG database).

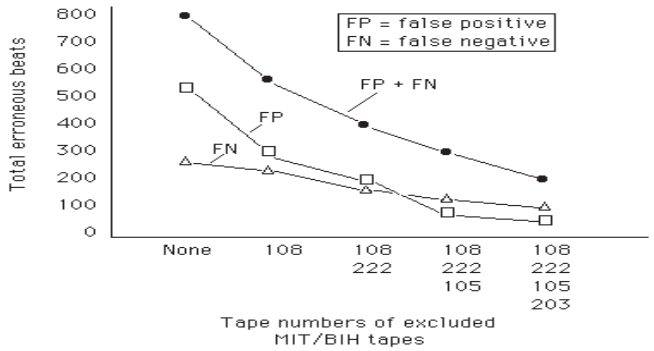


Figure 12.28 Performance of QRS detection software. The total error of the QRS detection algorithm can be substantially reduced by selectively eliminating problem tapes in the database.

12.6 LAB: REAL-TIME ECG PROCESSING ALGORITHM

This lab lets you "look inside" the inner workings of the algorithm QRS detection algorithm developed by Pan and Tompkins (1985) that is described in section 12.5. Load UW DigiScope, select **`ad(V) Ops`**, then **`(Q)RS detect`**.

12.6.1 QRS detector algorithm processing steps

Observe the output of each of the stages in the QRS detector. Sketch or print one cycle of the original ECG signal and the outputs of the low-pass, bandpass, derivative, squaring, and moving window integrator stages. Note the filter delay at each of these stages.

12.6.2 Effect of the value of the *Q* of a filter on QRS detection

Implement several two-pole recursive filters with 17-Hz center frequencies to observe the effects of different values of Q on the ECG, as in section 12.2.1. What value of r produces the most desirable response for detecting the QRS complex?

12.6.3 Integer filter processing of the ECG

Use **`(G)enwave`** to generate an ECG signal sampled at 100 Hz. Process this signal with a filter having the following difference equation.

$$y(nT) = 2y(nT - T) - 3y(nT - 2T) + 2y(nT - 3T) - y(nT - 4T) + x(nT) - 2x(nT - 12T) + x(nT - 24T)$$

Observe the output and note the duration of the ringing.

12.7 REFERENCES

AHA ECG database. Available from Emergency Care Research Institute, 5200 Butler Pike, Plymouth Meeting, PA 19462.
Ahlstrom, M. L. and Tompkins, W. J. 1983. Automated high-speed analysis of Holter tapes with microcomputers. *IEEE Trans. Biomed. Eng.*, **BME-30**: 651–57.
Ahlstrom, M. L. and Tompkins, W. J. 1985. Digital filters for real-time ECG signal processing using microprocessors. *IEEE Trans. Biomed. Eng.*, **BME-32**: 708–13.
Balda R. A., Diller, G., Deardorff, E., Doue, J., and Hsieh, P. 1977. The HP ECG analysis program. *Trends in Computer-Processed Electrocardiograms.* J. H. vanBemnel and J. L. Willems, (eds.) Amsterdam, The Netherlands: North Holland, 197–205.
Dobbs, S. E., Schmitt, N. M., Ozemek, H. S. 1984. QRS detection by template matching using real-time correlation on a microcomputer. *Journal of Clinical Engineering*, **9**: 197–212.
Friesen, G. M., Jannett, T. C., Jadallah, M. A., Yates, S. L., Quint, S. R., Nagle, H. T. 1990. A comparison of the noise sensitivity of nine QRS detection algorithms. *IEEE Trans. Biomed. Eng.*, **BME-37**: 85–97.
Furno, G. S. and Tompkins, W. J. 1982. QRS detection using automata theory in a battery-powered microprocessor system. *IEEE Frontiers of Engineering in Health Care*, **4**: 155–58.
Hamilton, P. S. and Tompkins, W. J. 1986. Quantitative investigation of QRS detection rules using the MIT/BIH arrhythmia database. *IEEE Trans. Biomed. Eng.* **BME-33**: 1157–65.
MIT/BIH ECG database. Available from: MIT-BIH Database Distribution, Massachusetts Institute of Technology, 77 Massachusetts Avenue, Room 20A-113, Cambridge, MA 02139.
Pan, J. and Tompkins, W. J. 1985. A real-time QRS detection algorithm. *IEEE Trans. Biomed. Eng*. **BME-32**: 230–36,.
Thakor, N. V., Webster, J. G., and Tompkins, W. J. 1983. Optimal QRS detector. *Medical and Biological Engineering*, 343–50.
Thakor, N. V., Webster, J. G., and Tompkins, W. J. 1984. Estimation of QRS complex power spectra for design of a QRS filter. *IEEE Trans. Biomed. Eng.*, **BME-31**: 702–05.

12.8 STUDY QUESTIONS

12.1 How can ectopic beats be detected using the automata approach to QRS detection?
12.2 How can QRS complexes in abnormal waveforms be detected using the crosscorrelation method?
12.3 In the moving window integrator of the algorithm in section 12.5, how should the width of the window be chosen? What are the effects of choosing a window width that is too large or too small?
12.4 In the QRS detection algorithm explained in section 12.5, how should the first threshold in each set of thresholds be changed so as to increase the detection sensitivity of irregular heart rates?
12.5 What are the effects of bandpass filter Q on the QRS-to-noise ratio in the ECG?
12.6 Design an algorithm that obtains the fiducial point on the ECG.
12.7 As an implementation exercise write a program using the C language, to detect QRS complexes in the ECG signal using any of the techniques described in this chapter.
12.8 Suggest a QRS detection algorithm, based on some of the techniques explained in this chapter or in other related literature, that can detect QRS complexes from the ECG in real time.
12.9 Experiments to determine the frequency characteristics of the average QRS complex have shown that the largest spectral energy of the QRS complex occurs at approximately what frequency?
12.10 A filter with the difference equation, $y(nT) = (y(nT - T))^2 + x(nT)$, is *best* described as what traditional filter type?

12.11 The center frequency of the optimal QRS bandpass filter is not at the location of the maximal spectral energy of the QRS complex. (a) What function is maximized for the optimal filter? (b) What is the center frequency of the optimal QRS filter for cardiotachometers? (c) If this filter has the proper center frequency and a $Q = 20$, will it work properly? If not, why not?

12.12 In addition to heart rate information, what QRS parameter is provided by the QRS detection algorithm that is based on the first and second derivatives?

12.13 The derivative algorithm used in a real-time QRS detector has the difference equation: $y(nT) = 2x(nT) + x(nT - T) - x(nT - 3T) - 2x(nT - 4T)$. (a) Draw its block diagram. (b) What is its output sequence in response to a *unit step* input? Draw the output waveform.

12.14 Write the equations for the amplitude and phase responses of the derivative algorithm used in a real-time QRS detector that has the transfer function

$$H(z) = \frac{-2z^{-2} - z^{-1} + z^{1} + 2z^{2}}{8}$$

12.15 A moving window integrator integrates over a window that is 30 samples wide and has an overall amplitude scale factor of 1/30. If a unit impulse (i.e., 1, 0, 0, 0, …) is applied to the input of this integrator, what is the output sequence?

12.16 A moving window integrator is five samples wide and has a unity amplitude scale factor. A pacemaker pulse is described by the sequence: (1, 1, 1, 1, 0, 0, 0, 0, …). Application of this pulse to the input of the moving window integrator will produce what output sequence?

12.17 The transfer function of a filter used in a real-time QRS detection algorithm is

$$H(z) = \frac{(1 - z^{-6})^2}{(1 - z^{-1})^2}$$

For a sample rate of 200 sps, this filter *eliminates* input signals of what frequencies?