Real-time Heart Monitoring and ECG Signal Processing

Fatima Bamarouf, Claire Crandell, and Shannon Tsuyuki

Advisors: Drs. Yufeng Lu and Jose Sanchez

Department of Electrical and Computer Engineering
Bradley University
November 24, 2015
Contents

• Introduction and Overview
• Progress
• Summary and Conclusions
Introduction and Overview

• Problem Description
• Project Objectives
• System Block Diagram
• Division of Labor
Problem Description

• Arrhythmias
  • Are irregular heartbeats caused by defective electrical signals in the heart [1]
  • Include premature ventricular contractions (PVCs)
  • PVCs may lead to ventricular tachycardia (VT)

Figure 1. Electrocardiogram with “V” labels for PVCs [2]
Problem Description

- An electrocardiogram (ECG) describes the heart’s electrical activity
- An ECG can be recorded using a Holter monitor or event monitor

Figure 2. Features of a normal ECG [3]
Problem Description

• Holter and event monitors are limited in functionality
  • Utilize some in-platform signal processing for diagnostic assistance
  • Must perform some signal processing offline
  • Are unable to address medical issues in real time
System Block Diagram

Unprocessed Heart Data → Real-time Heart Monitor System → Wireless Message

Figure 3. Overall heart monitoring system diagram
Project Objectives

• Develop a low-power, stand-alone embedded system for continuous heart monitoring that will
  • Process ECG data in real time
  • Detect PVCs accurately and consistently
  • Alert the patient’s doctor wirelessly of ventricular tachycardia
Project Objectives

• Real-time ECG signal processing
• On-board signal processing computations
• Battery-powered functionality
Division of Labor

• MATLAB Simulation (PVC detection)
  • Shannon/Fatima

• C Programming (PVC detection)
  • Claire/Shannon

• Wi-Fi Communication
  • Fatima/Claire/Shannon
Contents

• Introduction and Overview

• Progress
  • Fatima
  • Shannon
  • Claire

• Summary and Conclusions
Problem Approach

Figure 4. High-level flowchart for algorithms
Figure 5. Flowchart for T1 and T2 generation
Figure 6. MATLAB plot of QRS detection results. The marked peaks were placed into an array.
Template Generation Algorithm

Figure 7. Flowchart for T1 and T2 generation
Removing Extremities (3\(\sigma\))

- An amplitude \(A\) is kept if it is fulfills the criterion

\[
\bar{A} - 3\sigma \leq A \leq \bar{A} + 3\sigma
\]

where \(\bar{A}\) is the mean of the amplitude range
Figure 8. Flowchart for T1 and T2 generation
Template Generation Algorithm

Figure 9. Flowchart for T1 and T2 generation
Equal Sets

• sort_series=[0 1 1.5 1.7 2 3 5 5 7 9 10]
• Step=(Max-Min)/5=2
• Five equal sets: [0 1 1.5 1.7 2] [3] [5 5] [7] [9 10]
• Set one has maximum data points
Template Generation Algorithm

Figure 10. Flowchart for T1 and T2 generation
Template Generation Algorithm

Figure 11. Flowchart for T1 and T2 generation
Data between Two Fiducial Points

Figure 12. MATLAB plot of QRS detection results. The marked RR-intervals were placed into an array.
Template Generation Algorithm

Figure 13. Flowchart for T1 and T2 generation
Templates T1 and T2

Figure 14. T1 and T2 generation
Template-Matching Algorithm

Figure 15. Flowchart for template-matching algorithm
T1 Alignment

Figure 16. T1 template and PVC QRS complex aligned to T1’s R peak index
Template-Matching Algorithm

QRS Data (30min)

k-th Beat Aligned to T1

Correlation of Aligned k-th Beat to Template T1

Stretch/Compress k-th to k+1 Data Vector to Aligned with T2

Correlation of Data Between Two Fiducial Points to T2

Combine Both Correlation Using Exponential Function Correlation

Compare the Correlation result to the k-th Beat Aligned to T1 Threshold

PVC or not PVC

Figure 17. Flowchart for template-matching algorithm
Normalized Correlation Coefficient

\[ x_k = \frac{\sum_{n=0}^{L-1} [b_k(n) - \bar{b}_k][N(n) - \bar{N}]}{\sqrt{\sum_{n=0}^{L-1} [b_k(n) - \bar{b}_k]^2[N(n) - \bar{N}]^2}} \]

Where \( b_k(n) \) is the QRS complex in the \( k^{th} \) beat and \( L \) is the length of the predetermined \( T1(N(n)) \).
Template-Matching Algorithm

QRD Data (30min)

k-th Beat Aligned to T1

Correlation of Aligned k-th Beat to Template T1

Stretch/Compress k-th to k+1 Data Vector to Aligned with T2

Correlation of Data Between Two Fiducial Points to T2

Combine Both Correlation Using Exponential Function Correlation

Compare the Correlation Result to the T k-th Beat Aligned to T1 Threshold

PVC or not PVC

Figure 18. Flowchart for template-matching algorithm
Exponential Function Correlation

\[ z_k = f(x_k, y_k) = \frac{e^{x_k r} + e^{y_k r}}{2e} \]

Where \( r \) determines the increasing rate of the slope and \( z_k \geq z_{thre} \).
Figure 19. Gantt chart for fall semester
Future Schedule

Figure 20. Gantt chart up to the second progress presentation
Contents

• Introduction and Overview

• **Progress**
  • Fatima
  • Shannon
  • Claire

• Summary and Conclusions
Template-Matching Algorithm

QR Data (30min)

k-th Beat Aligned to T1

Correlation of Aligned k-th Beat to Template T1

Stretch/Compress k-th to k+1 Data Vector to Aligned with T2

Correlation of Data Between Two Fiducial Points to T2

Combine Both Correlation Using Exponential Function Correlation

Compare the Correlation result to the k-th Beat Aligned to T1 Threshold

PVC or not PVC

Figure 21. Flowchart for template-matching algorithm
Figure 22. T2 (interval between R peaks) and RR-interval signal for PVC beat
Interpolation and Extrapolation

• Let $b_k$ be the data between $k^{th}$ beat and $k + 1$ beat.

• If $\text{length}(b_k) \leq \text{Length}(T2)$, we do extrapolation based on

$$\text{new}_b k = b_k [1 + \alpha(n - 1)] \quad (n = 1, 2, \ldots, L)$$

$$\alpha = \frac{\text{length}(b_k)}{\text{length}(T2)}$$
Example (Extrapolation)

\[ L_k = 10, \quad L = 21, \quad \alpha_k = \frac{10}{21} \approx 0.476 \]

<table>
<thead>
<tr>
<th>( n ) value</th>
<th>Calculated Index</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1.48</td>
</tr>
<tr>
<td>3</td>
<td>1.95</td>
</tr>
<tr>
<td>4</td>
<td>2.43</td>
</tr>
<tr>
<td>5</td>
<td>2.90</td>
</tr>
</tbody>
</table>
Algorithm Efficacy, 100s

Figure 23. Performance of template-matching algorithm MATLAB simulation
Alternative QRS Detection [5]

Figure 24. Flowchart for alternative QRS detection method
Converting MATLAB to C

• Template-Matching Functions
  • normIntRange.m
  • normal_amplt.m
  • Corr_Coeff.m
  • Z_Corr.m
Figure 25. Gantt chart for fall semester
Future Schedule

Figure 26. Gantt chart up to the second progress presentation
Contents

• Introduction and Overview

• Progress
  • Fatima
  • Shannon
  • Claire

• Summary and Conclusions
Template Generation Algorithm

Figure 27. Flowchart for T1 and T2 generation
Testing of QRS Detection Phase

• Pan-Tompkins algorithm implemented on CC3200

Figure 28. CC3200 LaunchPad [6]
UDMA Diagram

Figure 29. UART data flow between the PC and the CC3200
UART Data Transfer

Obtain the next sample from the MIT-BIH database

Convert the sample into a sequence of characters

Add the characters to the output buffer

198 samples?

Y

Pad the output buffer to a length of 990 characters

Send out the buffer using UART, reset number of samples

N

Figure 30. UART data transfer flowchart (PC)
UART Data Transfer

UDMA Controller

- Data available?
  - N: Trigger UART interrupt
  - Y: Place buffer contents in memory

UART Interrupt

- Set bRxDone
- Clear UART interrupt

Main Function

- Is bRxDone set?
  - N: Clear bRxDone
  - Y: Convert character buffer to integer buffer
    - Use the Pan-Tompkins algorithm to find peaks

Figure 31. UART data transfer flowcharts (CC3200)
Pan-Tompkins Algorithm

Heart Data

Pan-Tompkins Algorithm

R Peak Locations

Figure 32. Pan-Tompkins block diagram
Pan-Tompkins Algorithm

1. Raw ECG Data
2. Low-Pass Filter
3. High-Pass Filter
4. Derivative
5. Squaring Function
6. Moving-Window Integration
7. QRS Complex Data

Figure 33. Flowchart for Pan-Tompkins algorithm
### Results

**TABLE II. PERFORMANCE OF PAN-TOMPKINS C IMPLEMENTATION**

<table>
<thead>
<tr>
<th>Record</th>
<th>QRS Sensitivity (%)</th>
<th>QRS Positive Predictivity (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>97.83</td>
<td>100.00</td>
</tr>
<tr>
<td>102</td>
<td>96.69</td>
<td>100.00</td>
</tr>
<tr>
<td>103</td>
<td>99.43</td>
<td>100.00</td>
</tr>
<tr>
<td>105</td>
<td>99.76</td>
<td>99.76</td>
</tr>
<tr>
<td>106</td>
<td>96.67</td>
<td>99.69</td>
</tr>
</tbody>
</table>
Conversion of MATLAB Code

• Conversion of template-matching MATLAB code to C code
  • `Normal_amplt.m`
  • Integrated code project
C Implementation Results

Figure 34. T1 (QRS complex) template generated on the CC3200
C Implementation Results

Figure 35. T2 (RR interval) template generated on the CC3200
Figure 36. Gantt chart for fall semester
Future Schedule

Figure 37. Gantt chart up to the second progress presentation
Contents

• Introduction and Overview
• Progress
  • Fatima
  • Shannon
  • Claire
• **Summary and Conclusions**
Summary and Conclusions

• PVCs are irregular heartbeats that may lead to VT

• An embedded device is proposed that will detect PVCs in real time and wirelessly alert the patient’s doctor of VT
Summary and Conclusions

• Design must include real-time ECG signal processing, on-board signal processing computations, and battery-powered functionality
Summary and Conclusions

• Current Tasks
  • MATLAB simulation of the PVC detection method
  • C implementation of completed PVC detection code
Summary and Conclusions

• Future Tasks
  • Complete simulation and testing of all MATLAB code
  • Test the integrated template-matching/Pan-Tompkins code
  • Research wireless communication methods
Real-time Heart Monitoring and ECG Signal Processing

Fatima Bamarouf, Claire Crandell, and Shannon Tsuyuki

Advisors: Drs. Yufeng Lu and Jose Sanchez

Department of Electrical and Computer Engineering
Bradley University
November 24, 2015
References


Envelope Signal [5]

Figure 38. Envelope of the initial ECG signal
Figure 39. Auxiliary signal of the envelope
Figure 40. State diagram for heart monitoring system

1. Start
2. Store heart data into memory
3. Transmit a message to the doctor (for VT)
4. Perform preprocessing
5. Determine if VT is present
6. Classify each beat as PVC or non-PVC
### Figure 41. Gantt chart for the MATLAB simulation (PVC algorithm) phase of the project
Figure 42. Gantt chart for the C implementation (PVC algorithm) phase of the project
### Figure 43. Gantt chart for the wireless development phase of the project

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Research best protocol/technique to use for web</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>application (sockets, TCP, ...)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Upload the data to a web server</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Send heart data from a PC to the board using wireless</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create figures for the web server</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Specificity and Sensitivity [7]

\[
SP = \frac{TN}{TN + FP} \quad \text{SE} = \frac{TP}{TP + FN}
\]

- **TP (True Positive)**: detected QRS complex that is present in the signal
- **TN (True Negative)**: data point between QRS complexes that does not contain a QRS peak
- **FP (False Positive)**: incorrect identification of QRS peak
- **FN (False Negative)**: QRS peak that was not detected by the algorithm
Memory Requirements

• Sampling rate for ECG signal (MIT-BIH arrhythmia database): 360 Hz
• Number of samples required for 30 seconds of ECG data: 10,800
• Amount of memory required: 21 kB
<table>
<thead>
<tr>
<th>Task</th>
<th>Duration (hours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PVC Algorithm (MATLAB)</td>
<td>65</td>
</tr>
<tr>
<td>PVC Algorithm (C)</td>
<td>100</td>
</tr>
<tr>
<td>Wi-Fi Communication</td>
<td>150</td>
</tr>
<tr>
<td>Progress Report I</td>
<td>80</td>
</tr>
<tr>
<td>Progress Report II</td>
<td>80</td>
</tr>
<tr>
<td>Final Presentation</td>
<td>80</td>
</tr>
<tr>
<td>Final Report</td>
<td>80</td>
</tr>
</tbody>
</table>
Figure 44. Gantt chart for the fall semester
Figure 45. Gantt chart for the spring semester
Pan-Tompkins Algorithm [8]

1. Low-pass Filter
   - 11 Hz cut-off frequency
   - 5-sample delay
   - Gain of 36
     \[ y(n) = 2y(n - 1) - y(n - 2) + x(n) - 2x(n - 6) + x(n - 12) \]

2. High-pass Filter
   - 5 Hz cut-off frequency
   - 29-sample delay
   - Gain of 1
     \[ y(n) = y(n - 1) - \frac{1}{32} x(n) + x(n - 16) - x(n - 17) + \frac{1}{32} x(n - 32) \]
Pan-Tompkins Algorithm

3. Derivative
   - Provides information about QRS slope
   - Approximates derivative from 0-30 Hz
   - Has a 4-sample delay
     \[ y(n) = \frac{1}{8} [2x(n) + x(n - 1) - x(n - 3) - 2x(n - 4)] \]

4. Squaring Function
   - Emphasizes higher frequencies of the ECG (caused by QRS complexes)
     \[ y(n) = x^2(n) \]
Pan-Tompkins Algorithm

5. Moving-Window Integration
   - Detects long-duration and large-amplitude QRS complexes
   \[ y(nT) = \frac{1}{N} \left[ x(nT - (N - 1)T) + x(nT - (N - 2)T + \cdots + x(nT) \right] \]
Algorithm Efficacy, 200s

Figure 47. Performance of template-matching algorithm MATLAB simulation
WFDB Library (PC side)

• `isigopen()`: open a specific WFDB record
• `getvec()`: get the next sample in the record
WFDB Library (PC side)

- `isigopen()`: open a specific WFDB record
- `getvec()`: get the next sample in the record
WFDB Toolbox (MATLAB)

- `rdsamp()`: place samples from a WFDB record into a vector
- `rdann()`: place annotations (characters) from a WFDB record into a vector
WFDB Toolbox (MATLAB)

• `wrsann()`: write experimental annotations into a vector

• `bxb()`: generate a report (with accuracy and positive predictivity data) using experimental annotations
Sample BXB Report

Beat-by-beat comparison results for record mitdb/100
Reference annotator: atr
Test annotator: test

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>n</th>
<th>s</th>
<th>v</th>
<th>f</th>
<th>q</th>
<th>o</th>
<th>x</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>356</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
</tr>
<tr>
<td>S</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>V</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Q</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>O</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>X</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

QRS sensitivity: 97.83% (360/368)
QRS positive predictivity: 100.00% (360/360)
VEB sensitivity: - (0/0)
VEB positive predictivity: - (0/0)
SVEB sensitivity: 0.00% (0/4)
SVEB positive predictivity: - (0/0)
RMS RR interval error: 168.87 ms

Figure 48. Text file generated using the \texttt{bxb()} function in the WFDB toolbox
DSP library for CC3200

• Corr_Coeff.c
  • arm_sub_f32()
  • arm_mult_f32()

• Normal_amplt.c, normIntRange.c
  • arm_mean_f32()
  • arm_std_f32()
Sorting the Amplitude Series

• Let $S_{\text{amp}}=[2 \ 1 \ 10 \ 5 \ 9 \ 1.5 \ 3 \ 0 \ 1.7 \ 5 \ 7]$

• After sorting,

  $\text{sort\_series}=[0 \ 1 \ 1.5 \ 1.7 \ 2 \ 3 \ 5 \ 5 \ 7 \ 9 \ 10]$
QRS Detection

Figure 49. MATLAB plot of QRS detection results
Alternative QRS Detection [5]

- Uses statistics to accurately locate QRS onset and offset
- Can be used to determine abnormal QRS complexes

Figure 50. Flowchart for alternative QRS detection method
Alternative QRS Detection

- Use the Hilbert transform to obtain the envelope

Figure 51. Flowchart for alternative QRS detection method
Alternative QRS Detection

- Estimate the derivative using a parabola:
  \[
  h'(k) = \frac{1}{10} \left( 2(h(k + 2r) - h(k - 2r)) + h(k + r) - h(k - r) \right)
  \]

Figure 52. Flowchart for alternative QRS detection method
Alternative QRS Detection

- Calculate a cumulative mean for the QRS onset and offset windows
- Determine the probability density functions

Figure 53. Flowchart for alternative QRS detection method