

Matlab®Code for ECG peak detection

```
1 clear all
2 close all
3 clc
4
5 %% Connect to Arduino
6 delete(instrfindall)
7 s = serial('COM1');
8 set(s,'BaudRate',57600);
9 try
10     fopen(s);
11 catch err
12     fclose(instrfind);
13     error('Make sure you select the correct COM Port where the
14         Arduino is connected.');
```

14 end

```
15
16 %% input parameters
17 Fs=200; % sampling frequency
18 N=3; % order of bandpass
19 filter
20 band=[5 15]; % bandpass frequency
21
22 %% calculate filter coefficients
23 [B,A] = butter(N,band/(Fs/2),'bandpass');
24 bbuffer = zeros(length(B),1); % buffer for bandpass
25 filter
26
27 if Fs ~= 200
28     int_c = (5-1)/(Fs*1/40);
29     b = interp1(1:5,[1 2 0 -2 -1].*(1/8)*Fs,1:int_c:5);
30 else
31     b = [1 2 0 -2 -1].*(1/8)*Fs;
32 end
33
34 dbuffer = zeros(length(b),1); % buffer for derivative
35 filter
36
37 MA=0.15*Fs;
38 MA=ones(1,MA)./MA;
39 mbuffer=zeros(length(MA),1); % buffer for moving
40 average filter
41
42 %% define parameters
43
44 i =0;
45 fs = 200;
46 trained = 0;
47 leng = round(0.075*fs);
48 MAecg_section = zeros(1,leng);
49 filtecg_section = zeros(1,0.15*fs); %0.15
50 global THR_SIG THR_NOISE SIG_LEV NOISE_LEV THR_SIG1 THR_NOISE1
51 SIG_LEV1 NOISE_LEV1
52 Beat_C = 0;
53
54 % Raw Beats
55 Beat_C1 = 0;
56
57 % Filtered Beats
58 skip = 0;
```

```

49 m_selected_RR = 0;
50 mean_RR = 0;
51 locs = 0;
52 pks = 0;
53 Slope1 = 0;
54 Slope2 = 0;
55 slope_all = [];
56 fil_ecg_train = [];
57 MA_ecg_train = [];
58 locs_all = [];
59 pks_all = [];
60 count_down = 0;
61 QRSPEAK_all = [];
62
63 if rem(round(0.03*fs),2) == 0
64     mov_win_size = round(0.03*fs)+1;
65 else
66     mov_win_size = round(0.03*fs);
67 end
68
69 mov_win = zeros(1,mov_win_size);
70 MA2=ones(1,mov_win_size)./mov_win_size;
71 m2buffer=zeros(mov_win_size,1);
72 Ts = 1/Fs;
73 TMAX = 40;
74 ecg = 0;
75 t = 0;
76 tic
77 j = 1;
78 while toc <= TMAX
79
80     while j<500
81         out = fgetl(s);
82         ecg = str2double(out);
83         trash(j) = ecg;
84         j = j+1;
85     end
86
87     i = i+1;
88     out = fgetl(s);
89     ecg = str2double(out);
90     ecg_store(i)=ecg;
91
92     % bandpass
93     [fil_ecg bbuffer_new] = BandPass(B, A, ecg,bbuffer);
94     bbuffer_new(1,end)=fil_ecg;
95     bbuffer=bbuffer_new;
96     fil_ecg_all(i) = fil_ecg;
97
98     % derivative and square
99     [deri_ecg dbuffer_new]= Derivative(fil_ecg,b,dbuffer);
100    dbuffer_new(1,end)=deri_ecg;
101    dbuffer=dbuffer_new;
102    deri_ecg_all(i) = deri_ecg;
103
104    % moving average
105    [MA1_ecg mbuffer_new]= MavgFilter(deri_ecg,MA,mbuffer);
106    mbuffer_new(1,end)=MA1_ecg;
107    mbuffer=mbuffer_new;
108    MA1_ecg_all(i) = MA1_ecg;

```

```

109
110 % MA2
111 [MA_ecg m2buffer_new]= MavgFilter(MA1_ecg,MA2,m2buffer);
112 m2buffer_new(1,end)=MA_ecg;
113 m2buffer=m2buffer_new;
114 MA_ecg_all(i) = MA_ecg;
115
116 MAecg_section = [MAecg_section(2:end) MA_ecg];
117 filtecg_section = [filtecg_section(2:end) fil_ecg];
118
119 if trained ==1 && count_down==0
120
121     mov_win = [mov_win(2:end) MA_ecg];
122     if mov_win((mov_win_size-1)/2)<mov_win((mov_win_size+1)/2)
123         && mov_win((mov_win_size+3)/2)<mov_win((mov_win_size
124             +1)/2) %local maximum
125
126         locs = i-(mov_win_size+1)/2+1;
127         pks = mov_win((mov_win_size+1)/2);
128         locs_all = [locs_all locs];
129         pks_all = [pks_all pks];
130         [y,x] = max(filtecg_section); %locate the
131             corresponding peak in the filtered signal
132
133     end
134
135     %% ===== update the heart_rate
136     %% ===== %%
137     if Beat_C >= 9
138         diffRR = diff(qrs_i(Beat_C-8:Beat_C)); % calculate RR
139             interval
140         mean_RR = mean(diffRR); %
141             calculate the mean of 8 previous R waves interval
142         comp =qrs_i(Beat_C)-qrs_i(Beat_C-1); % latest RR
143
144         if comp <= 0.92*mean_RR || comp >= 1.16*mean_RR
145             % ----- lower down thresholds to detect better in MVI
146             % ----- %
147             THR_SIG = 0.5*(THR_SIG);
148             THR_SIG1 = 0.5*(THR_SIG1);
149         else
150             m_selected_RR = mean_RR; % The
151                 latest regular beats mean
152
153     end
154
155     end
156
157     %% ===== find noise and QRS peaks
158     %% ===== %%
159     if pks >= THR_SIG
160         % ----- if No QRS in 360ms of the previous QRS See
161             if T wave -----%
162         if Beat_C >= 3
163             if (i-qrs_i(Beat_C)) <= round(0.3600*fs) %
164                 Slope1 = mean(diff(MAecg_section)); %
165                     mean slope of the waveform at that
166                     position

```

```

153
154         if abs(Slope1) <= abs(0.5*(Slope2))
                                                    % slope less
                then 0.5 of previous R
155                 skip = 1;

                % T wave identification
156                 % ----- adjust noise levels ----- %
157                 NOISE_LEV1 = 0.125*y + 0.875*NOISE_LEV1;
158                 NOISE_LEV = 0.125*pks + 0.875*NOISE_LEV;
159             else
160                 skip = 0;
161             end
162
163         end
164     end
165     %----- skip is 1 when a T wave is detected
166     %----- %
167     if skip == 0
168         Beat_C = Beat_C + 1;
169         qrs_i(Beat_C) = locs;
170     %----- bandpass filter check threshold
171     %----- %
172     if y >= THR_SIG1
173         Beat_C1 = Beat_C1 + 1;
174         count_down = round(0.2*fs);
175         QRSPEAK = 1;
176         fprintf(s, '%c', QRSPEAK);
                                                    %QRS
177         detected, generate pulse
178         Slope2 = mean(diff(MAecg_section)); % mean
179         slope of previous R wave
180         SIG_LEV1 = 0.125*y + 0.875*SIG_LEV1;
181         % adjust threshold
182         for bandpass filtered sig
183             ecg_section_all{i} = filtecg_section;
184         end
185         SIG_LEV = 0.125*pks + 0.875*SIG_LEV ;
186         % adjust Signal level
187     end
188     elseif (THR_NOISE <= pks) && (pks < THR_SIG)
189         NOISE_LEV1 = 0.125*y + 0.875*NOISE_LEV1;
190         % adjust Noise level in
191         filtered sig
192         NOISE_LEV = 0.125*pks + 0.875*NOISE_LEV;
193         % adjust Noise level in
194         MVI
195     elseif pks < THR_NOISE && pks~=0
196         NOISE_LEV1 = 0.125*y + 0.875*NOISE_LEV1;
197         % noise level in
198         filtered signal
199         NOISE_LEV = 0.125*pks + 0.875*NOISE_LEV;
200         % adjust Noise level in
201         MVI
202     end
203     %% ===== adjust the threshold with SNR
204     %% ===== %%
205     if NOISE_LEV ~= 0 || SIG_LEV ~= 0
206         THR_SIG = NOISE_LEV + 0.25*(abs(SIG_LEV -
207             NOISE_LEV));

```

```

191         THR_NOISE = 0.5*(THR_SIG);
192     end
193     %----- adjust the threshold with SNR for bandpassed
        signal ----- %
194     if NOISE_LEV1 ~= 0 || SIG_LEV1 ~= 0
195         THR_SIG1 = NOISE_LEV1 + 0.25*(abs(SIG_LEV1 -
            NOISE_LEV1));
196         THR_NOISE1 = 0.5*(THR_SIG1);
197     end
198     %----- take a track of thresholds of smoothed signal
        -----%
199     SIGL_buf(i) = SIG_LEV;
200     NOISL_buf(i) = NOISE_LEV;
201     THRS_buf(i) = THR_SIG;
202     %----- take a track of thresholds of filtered
        signal ----- %
203     SIGL_buf1(i) = SIG_LEV1;
204     NOISL_buf1(i) = NOISE_LEV1;
205     THRS_buf1(i) = THR_SIG1;
206     % ----- reset parameters
        ----- %
207     skip = 0;
208     not_nois = 0;
209     ser_back = 0;
210     pks = 0;
211     locs = 0;
212     QRSPEAK_all(i) = QRSPEAK;
213 end
214
215 QRSPEAK = 0;
216
217 if length(fil_ecg_all)>5*fs && length(MA_ecg_all)>5*fs &&
    trained==0
218     training(MA_ecg_all(1,2*fs:end),fil_ecg_all(1,2*fs:end),
        fs);
219     trained = 1;
220 end
221
222
223 if count_down ~= 0
224     count_down = count_down -1;
225 end
226
227
228 end
229
230 fclose(s)
231 delete(s)
232
233
234 %% Filter Functions
235 function [data bbuffer_new] = BandPass(b, a,value,bbuffer)
236     k = 1;
237     while(k<length(b))
238         bbuffer(k) = bbuffer(k+1);
239         k=k+1;
240     end
241     bbuffer(length(b)) = 0;
242     k = 1;
243     while(k<(length(b)+1))

```

```

244         bbuffer(k) = bbuffer(k) + value .* b(k);
245         k=k+1;
246     end
247
248     k = 1;
249     while(k<length(b))
250         bbuffer(k+1) = bbuffer(k+1) - bbuffer(1) * a(k+1);
251         k=k+1;
252     end
253
254     data = bbuffer(1);
255     bbuffer_new=bbuffer;
256 end
257
258 % Derivative and square
259 function [data,dbuffer_new] = Derivative(value,b,dbuffer)
260     k = 1;
261     N = length(b);
262     while(k<N)
263         dbuffer(k) = dbuffer(k+1);
264         k=k+1;
265     end
266     dbuffer(N) = value;
267     k = 1;
268     while(k<N+1)
269         dbuffer(k) = dbuffer(k) + value .* b(k);
270         k=k+1;
271     end
272     dbuffer_new=dbuffer;
273     data = dbuffer(1)^2;
274 end
275
276 function [data mbuffer_new] = MavgFilter(value,MA,mbuffer)
277     k = 1;
278     N=length(MA);
279     while(k<N)
280         mbuffer(k) = mbuffer(k+1);
281         k=k+1;
282     end
283     mbuffer(N) = 0;
284     k = 1;
285     while(k<N+1)
286         mbuffer(k) = mbuffer(k) + value.*MA(k);
287         k=k+1;
288     end
289     mbuffer_new=mbuffer;
290     data = mbuffer(1);
291 end
292
293 function training (ecg_m,ecg_h,fs)
294
295 global THR_SIG THR_NOISE SIG_LEV NOISE_LEV THR_SIG1 THR_NOISE1
296     SIG_LEV1 NOISE_LEV1
297
298 %% initialize the training phase (2 seconds of the signal) to
299     determine the THR_SIG and THR_NOISE
300 THR_SIG = max(ecg_m(1:2*fs))*1/3;
301                                     % 0.25 of the max
302                                     amplitude

```

```

299 THR_NOISE = mean(ecg_m(1:2*fs))*1/2;           % 0.5 of the mean signal
           is considered to be noise
300 SIG_LEV= THR_SIG;
301 NOISE_LEV = THR_NOISE;
302
303 %% Initialize bandpath filter threshold(2 seconds of the bandpass
           signal)
304 THR_SIG1 = max(ecg_h(1:2*fs))*1/3;             % 0.25 of the max
           amplitude
305 THR_NOISE1 = mean(ecg_h(1:2*fs))*1/2;
306 SIG_LEV1 = THR_SIG1;                           % Signal
           level in Bandpassed filter
307 NOISE_LEV1 = THR_NOISE1;                       % Noise level in Bandpassed
           filter
308
309 end

```