Elektrotehnički fakultet Sarajevo

Drugi ciklus studija

Odsjek za računarstvo i informatiku

Biomedicinski signali i sistemi

Seminarski rad

Razvoj aplikacije za rad sa EKG signalom: prikaz signala, detekcija QRS kompleksa i rad sa anotacijama

Radili:
Hajdarević Adnan
Hidić Adnan
Zubanović Damir

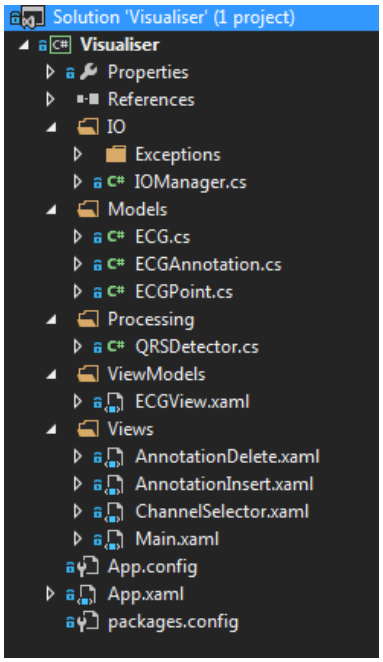
Ljetni semestar ak. 2013/2014. godine

1. Zahtijevi na funkcionalnost aplikacije

1. Aplikacija treba omogućiti učitavanje EKG signala preuzetog sa Physioneta
2. Aplikacija treba omogućiti iscrtavanje učitanog EKG signala na standardnom grafikonu (jedan kvadratić po x-osi označava 0.04 sekundi, dok po y-osi označava 0.1 mV, s tim da je svaki peti podebljan i označava 0.2 sekundi i 0.5 mV respektivno)
3. Aplikacija treba omogućiti dodavanje i uklanjanje anotacija
4. Aplikacija treba podržati standardne Physionetove anotacije, kao i korisnički definirane anotacije
5. Aplikacija treba omogućiti detekciju QRS kompleksa i određivanje srčanog ritma
6. Aplikacija treba omogućiti toggling anotacija

2. Detalji implementacije

Aplikacija je implementirana u programskom jeziku C#, .NET framework za Windows OS
Aplikacija je implementirana koristeći Visual Studio 2013 i .NET framework 4.5
Za izradu grafičkog korisničkog sučelja korišten je WPF
Aplikacija pored .NET frameworka zavisi i o OxyPlot biblioteci za rad sa grafikonima [1]
U izradi aplikacije korišten je programski kôd klasa objavljenih na coursewareu predmeta.
Struktura projekta:



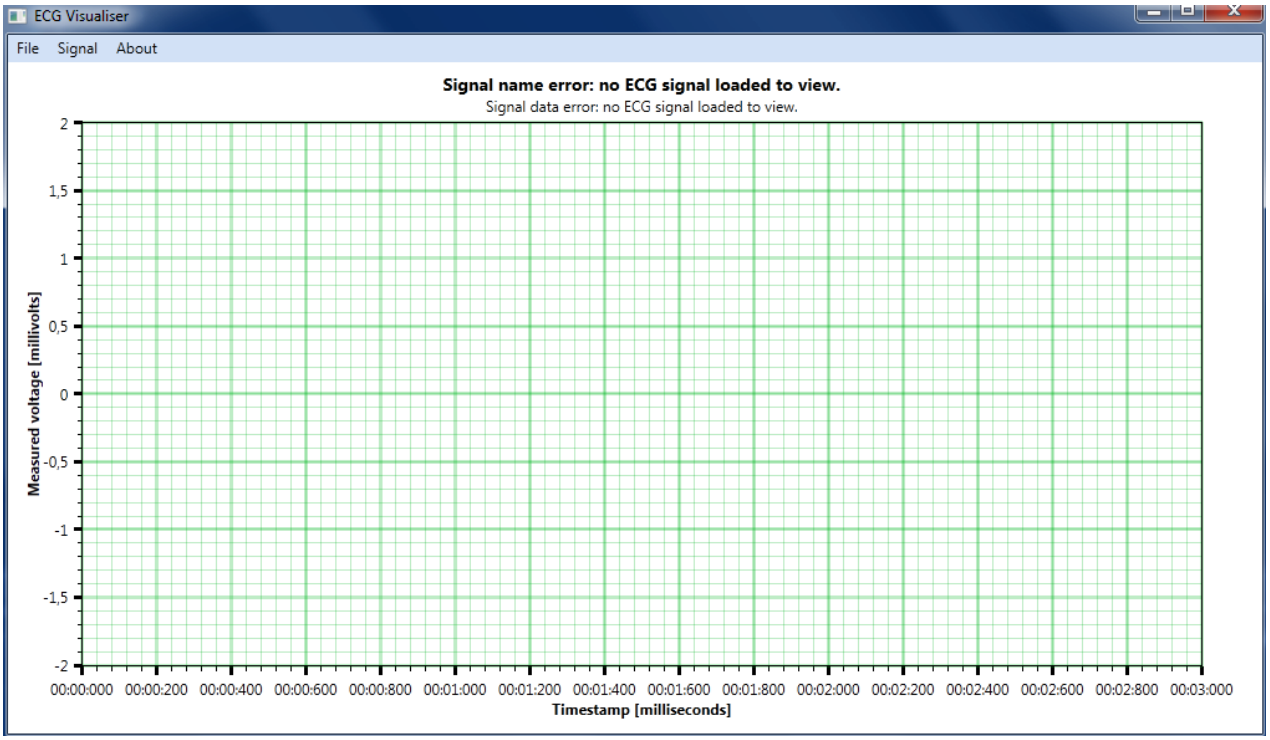
Slika 1 - struktura projekta. IO imenski prostor sadrži klase i metode za ulazno-izlazne operacije rada sa signalom, Models imenski prostor sadrži podatkovne modele ECG signala, uključujući i anotacije (ECGAnnotation) i pojedinačna mjerenja (ECGPoint). Processing imenski prostor sadrži klasu odgovornu za detekciju QRS kompleksa, ViewModels imenski prostor sadrži korisničku kontrolu za prikaz EKG signala i manipulacije nad anotacijama, dok su u Views imenskom prostoru raspoređeni prozori za interakciju sa aplikacijom.

Izvorni kod detekcije QRS kompleksa u modelu ECG signala [3]:

```
/// <summary>
        /// Performs QRS-complex detection on ECG signal provided as call argument.
        /// </summary>
        /// <param name="signal">ECG signal object</param>
        /// <returns>List of ECG points determined to be R spikes and Decimal value for measured heart-
rate</returns>
        public static Tuple<List<ECGPoint>,double> QRS_Detect(ECG signal)
        {
            // extract voltages from ecg signal points
            double[] voltages = signal.Points.Select(point => point.Value).ToArray();
            // perform QRS detection
            List<int> spikeIndices = SoAndChan(voltages);
            // map indices of R spikes into corresponding ecg signal points
            List<ECGPoint> spikes = spikeIndices.Select(index => signal.Points[index]).ToList();

            // if there are not enough spikes to determine HR
            if (spikes.Count < 2)
                return new Tuple<List<ECGPoint>, double>(spikes, -1);

            // get the seconds between first and the last spike
            double sumTimesBetweenSpikes= spikes[spikes.Count-1].TimeIndex - spikes[0].TimeIndex;
            // determine the HR
            double heartRate = 60 * spikes.Count/ sumTimesBetweenSpikes;

            return new Tuple<List<ECGPoint>, double>(spikes, heartRate);

        }

        private const double THRESHOLD_PARAM = 8;
        private const double FILTER_PARAMETER = 16;
        private const int SAMPLE_RATE = 250;


        // originalni rad:
http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=754529&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2
Fabs_all.jsp%3Farnumber%3D754529
        // izvorni kod: http://blog.kenliao.info/2012/03/c-so-and-chan-method-r-peak-detection.html
        // obrazloženje algoritma: http://researcher.most.gov.tw/public/8905780/Attachment/86518531071.ppt
        // prilagodio: Hidić Adnan
        public static List<int> SoAndChan(double[] voltages)
        {
            // initial maxi should be the max slope of the first SAMPLE_RATE points.
            double initial_maxi = -2 * voltages[0] - voltages[1] + voltages[3] + 2 * voltages[4];
            for (int i = 2; i < SAMPLE_RATE; i++)
            {
                double slope = -2 * voltages[i - 2] - voltages[i - 1] + voltages[i + 1] + 2 * voltages[i +
2];
                if (slope > initial_maxi)
                    initial_maxi = slope;
            }

            List<int> rIndices = new List<int>();

            // set initial maxi
            double maxi = initial_maxi;
            bool first_satisfy = false;
            bool second_satisfy = false;
            int onset_point = 0;
            int R_point = 0;

            // I want a way to plot all the r dots that are found...
            int[] rExist = new int[voltages.Length];
            // First two voltages should be ignored because we need rom length
            for (int i = 2; i < voltages.Length - 2; i++)
            {
```

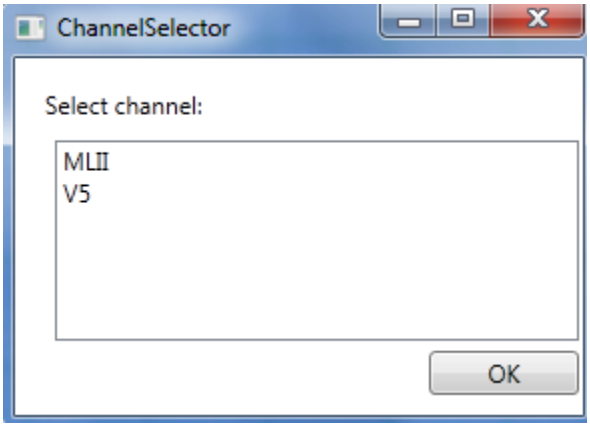
```
                // Last two voltages should be ignored too
                if (!first_satisfy || !second_satisfy)
                {
                    // Get Slope:
                    double slope = -2 * voltages[i - 2] - voltages[i - 1] + voltages[i + 1] + 2 *
voltages[i + 2];
                    // Get slope threshold
                    double slope_threshold = (THRESHOLD_PARAM / 16) * maxi;
                    // We need two consecutive data that satisfy slope > slope_threshold
                    if (slope > slope_threshold)
                    {
                        if (!first_satisfy)
                        {
                            first_satisfy = true;
                            onset_point = i;
                        }
                        else
                        {
                            if (!second_satisfy)
                            {
                                second_satisfy = true;
                            }
                        }
                    }
                }
                // We found the ONSET already, now we find the R point
                else
                {
                    if (voltages[i] < voltages[i - 1])
                    {
                        rIndices.Add(i - 1);
                        R_point = i - 1;

                        // Since we have the R, we should reset
                        first_satisfy = false;
                        second_satisfy = false;

                        // and update maxi
                        double first_maxi = voltages[R_point] - voltages[onset_point];
                        maxi = ((first_maxi - maxi) / FILTER_PARAMETER) + maxi;
                    }
                }
            }
            return rIndices;
        }
```

Osnovni izgled korisničkog sučelja:

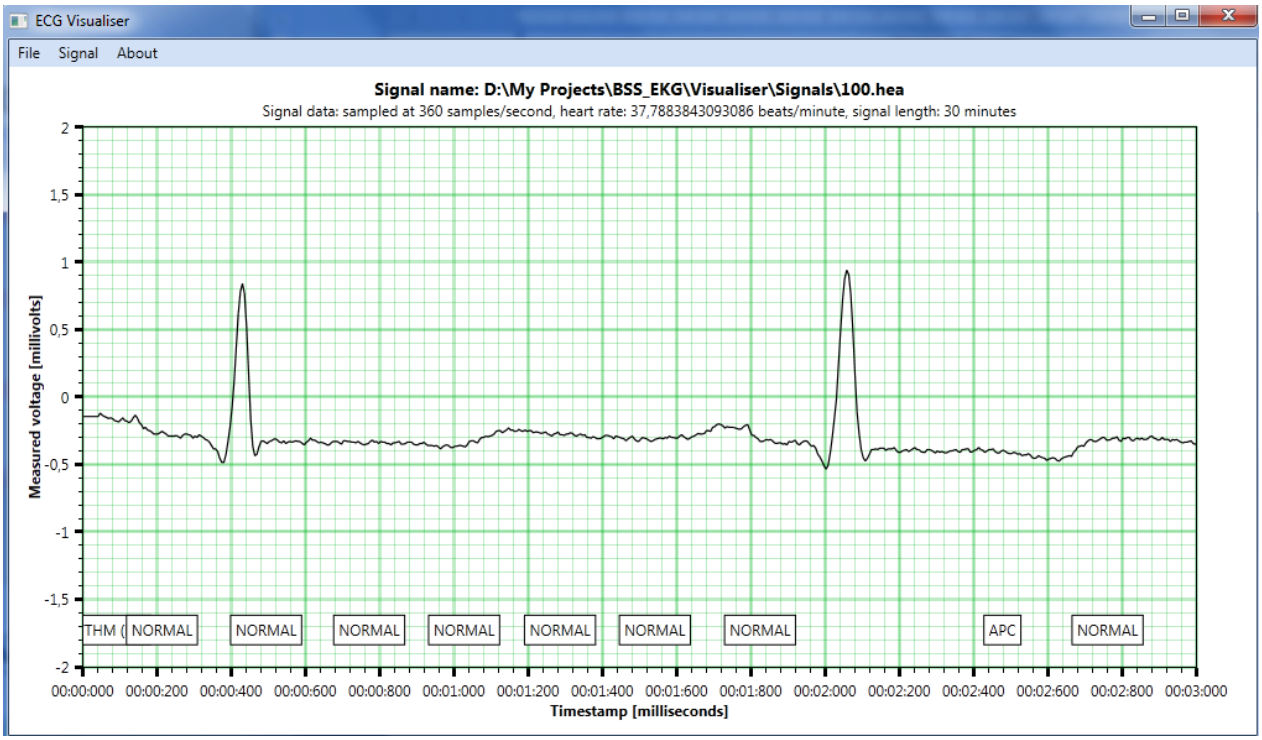


Izborom opcije File→ Open bira se .HEA datoteka u folderu kojem se još nalazi obavezno .DAT ili .TXT datoteka sa željenim ECG signalom (istog imena kao i .HEA), dok opcionalno mogu biti prisutne i anotacije. Anotacije mogu biti standardne Physionetove i nalaziti se u .ATR datoteci, ili mogu biti specifične za razvijenu aplikaciju kada se nalaze u .CUST datoteci.

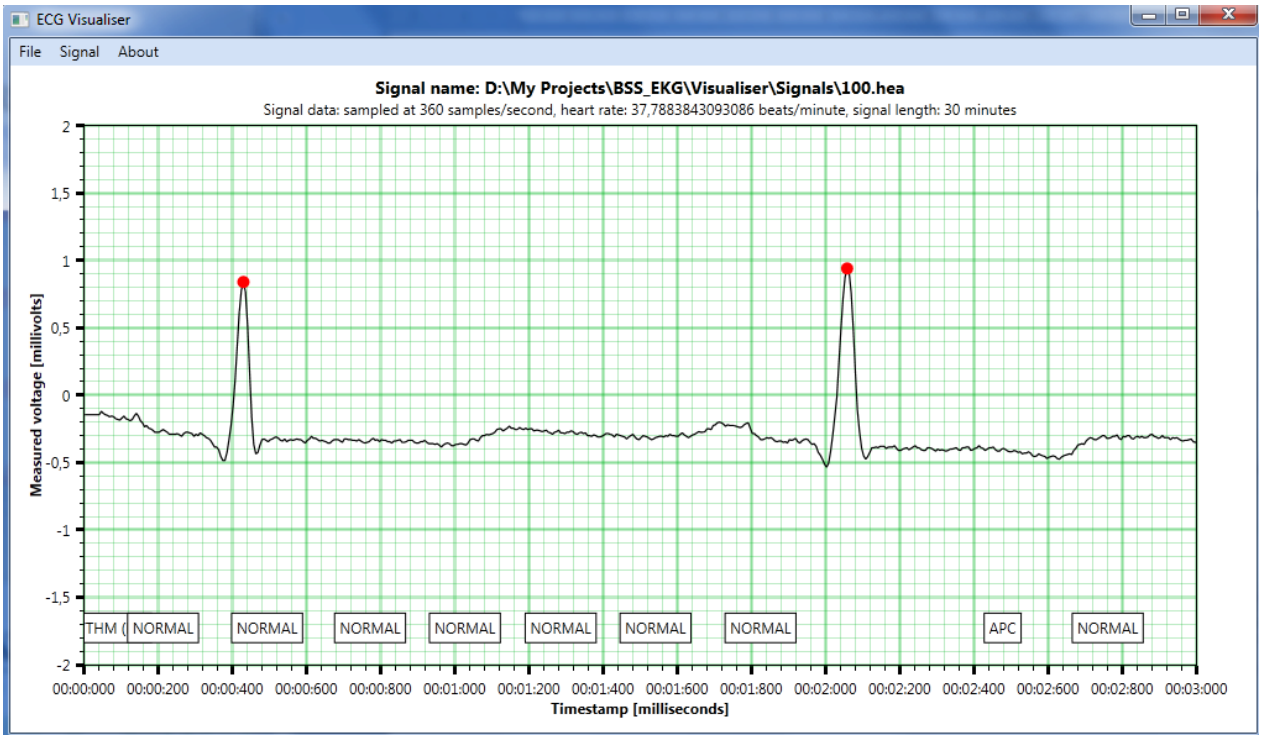
Izborom odgovarajuće .HEA datoteke dobija se prozor za izbor kanala:



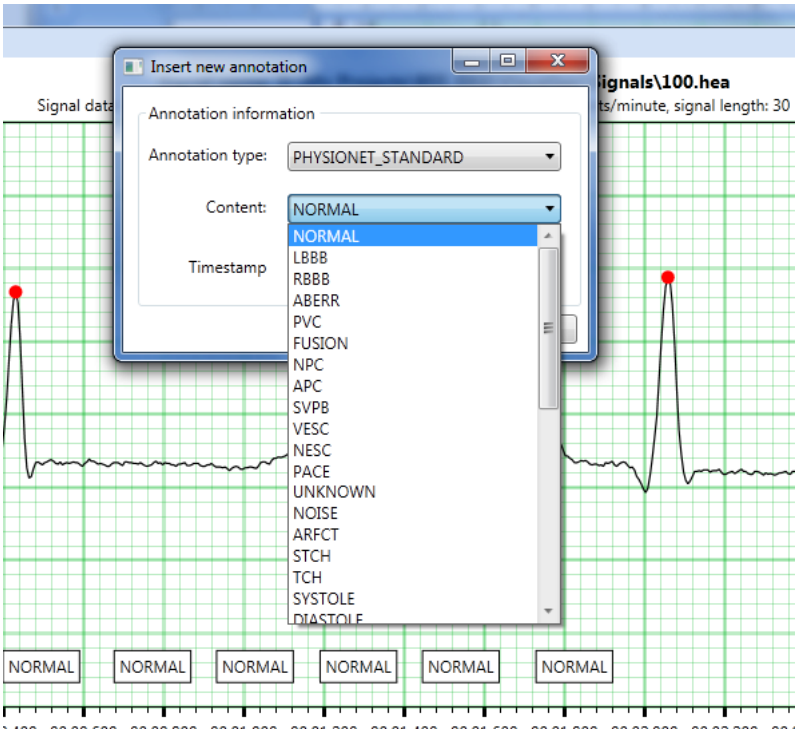
Nakon izbora odgovarajućeg kanala, učitava se signal i prikazuje skupa sa anotacijama:



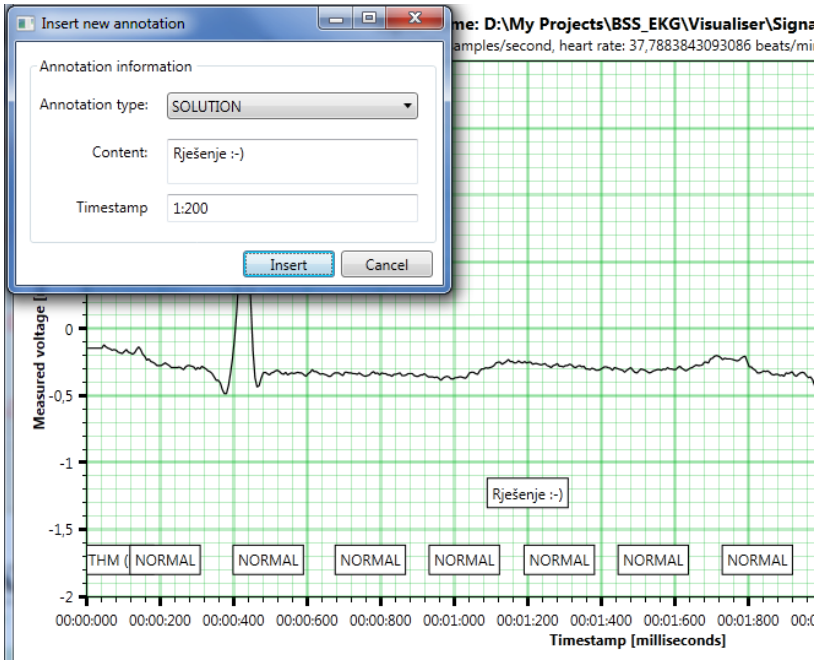
Opcija Signal→Detect omogućava mjerenje srčanog ritma i prikazuje QRS kompleks:



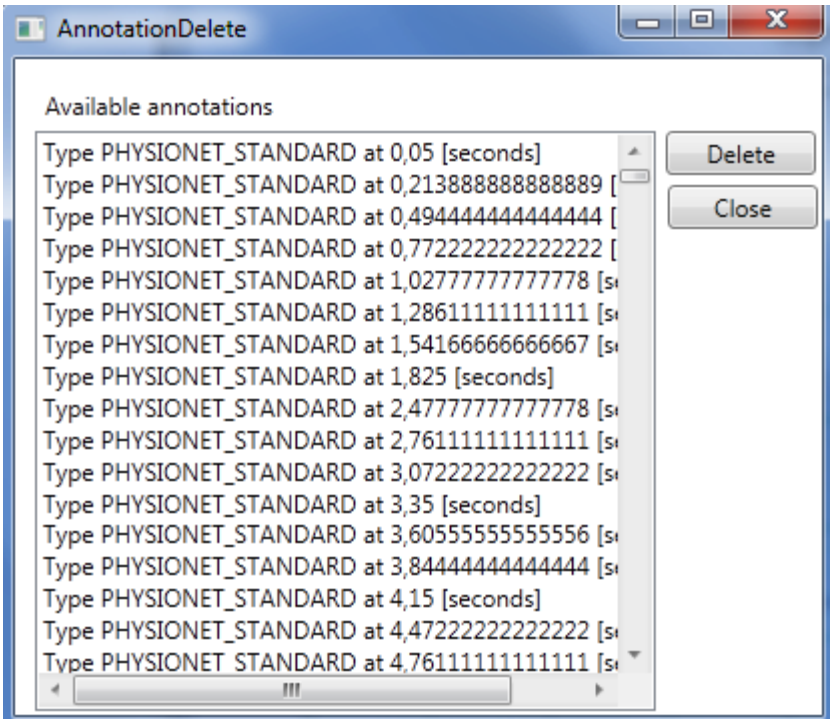
Opcija Signal→Insert annotation otvara prozor za ubacivanje anotacija. Potrebni podaci su tip anotacije, sadržaj anotacije i trenutak na koji se postavlja anotacija. Može biti zadat u formatu msec, sec:msec, mm:sec:msec.



Nakon što su uneseni ispravni podaci, anotacija se prikazuje na grafikonu na specificiranoj poziciji:



Anotacije je moguće i obrisati izborom opcije Signal→Delete annotations:



NAPOMENA: grafikon se pomjera držanjem desne tipke miša pritisnutom i pomjerajući miš lijevo ili desno. Zbog jednostavnosti je smatrano da je EKG signal u rasponu od -2mV do 2mV iako amplitude mogu biti i znatno veće od navedenih vrijednosti (apsolutno pozicioniranje anotacija).

Izvorni kôd projekta na raspolaganju je na sljedećem online repozitoriju:
https://github.com/AdnanHidic/bss_ecg

3. Izvori

[1] OxyPlot biblioteka za rad sa grafikonima: https://oxyplot.codeplex.com/

[2] So i Chanov metod detekcije QRS kompleksa: "Developoment of QRS detection method for real-time ambulatory cardiac monitor", So, H.H.; Chan, K.L. , Engineering in Medicine and Biology Society, 1997, Proceedings of the 19th Annual International Conference of the IEEE, Chicago, IL, USA
(više na linku:
http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=754529&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D754529)

[3] Obrazloženje algoritma, autora Ren-Guey Lee (članak iz [2] je iza IEEE paywalla):
http://researcher.most.gov.tw/public/8905780/Attachment/86518531071.ppt

[4] Izvorni kod C# implementacije [2] na raspolaganju je besplatno na sljedećem linku (autor Ken Liao):
http://blog.kenliao.info/2012/03/c-so-and-chan-method-r-peak-detection.html