

УДК 004.021

Разработка алгоритма измерения частоты пульса человека с помощью камеры мобильного телефона

Лаурэ Д. А.^{*}, Лагутина Н. С.^{*}, Парамонов И. В.^{** 1}

^{*}Ярославский государственный университет им. П. Г. Демидова
150000 Россия, г. Ярославль, ул. Советская, 14

^{**}ООО «ЭверестМД»
150008 Россия, г. Ярославль, ул. 2-я Ляпинская, 32а

e-mail: {den.a.laure, lagutinans}@gmail.com, ilya.paramonov@fruct.org

получена 26 августа 2014

Ключевые слова: mHealth, мобильный телефон, камера, пульс

В настоящее время существует большое количество различных способов измерения частоты пульса человека. Один из таких способов связан с использованием камеры мобильного телефона. Он удобен и прост с точки зрения пользователя и не требует дополнительных знаний или покупки специальных устройств для измерения пульса. Все, что необходимо, — это мобильный телефон со встроенной камерой и вспышкой. Основная идея данного способа заключается в том, чтобы детектировать изменения цвета кожи пальца руки, которые возникают из-за пульсации крови. Процесс измерения выглядит очень просто: пользователь прикладывает палец к камере, после чего приложение на мобильном телефоне начинает захватывать и анализировать кадры, полученные с камеры. Анализируя средние значения красной компоненты кадров с камеры мобильного телефона, содержащих изображение участка кожи, можно сделать вывод о частоте пульса.

В данной статье авторы делают обзор существующих алгоритмов для измерения частоты пульса с помощью мобильного телефона и предлагают собственный алгоритм, который более эффективен, чем рассмотренные.

Введение

Пульс — это периодические толчкообразные колебания стенок кровеносных сосудов, обусловленные сокращениями сердца. Частота пульса измеряется количеством ударов в минуту. Она помогает выявить такие заболевания, как тахикардия и брадикардия, а также является важным показателем общего состояния человека. Слежение за частотой пульса во время выполнения физических упражнений позволяет избежать перегрузок, а также проводить оценку уровня физической подготовки [1].

¹Работа выполнена при частичной поддержке гранта Фонда содействия развитию малых форм предприятий в научно-технической сфере. Контракт № 13-1-Н2.7-0080-1-С.

Частоту пульса можно измерить методом пальпации, а также с помощью различных медицинских приборов, таких как электрокардиографы, мониторы сердечного ритма и пульсоксиметры. Монитор сердечного ритма, или пульсометр, — это переносное устройство, измеряющее частоту пульса в режиме реального времени. Пульсометр состоит из двух составных частей: нагрудного ремня-датчика, который измеряет частоту пульса, и приёмника, который отображает результат измерения. Приёмник обычно выполнен в виде наручного браслета. В качестве приёмника также может выступать мобильный телефон. В настоящее время существуют пульсоксиметры в виде часов, которые вместо нагрудного датчика используют встроенный. Пульсометры получили широкое распространение, особенно среди спортсменов, поскольку позволяют следить за частотой пульса во время занятий спортом, когда измерение частоты пульса методом пальпации невозможно [2].

Иногда возникают ситуации, когда человеку необходимо узнать частоту своего пульса, но он не знает, как это сделать, и у него отсутствует пульсометр или другой прибор для измерения. В подобных случаях ему может помочь мобильный телефон. Сегодня существуют мобильные приложения, позволяющие измерять пульс пользователя с помощью встроенной в мобильный телефон камеры. Примерами таких приложений являются Instant Heart Rate² и Cardiograph³ для мобильных платформ Android и iOS.

Процесс измерения пульса в этом случае выглядит следующим образом. Пользователь запускает приложение, прикладывает палец руки к объективу камеры так, чтобы он полностью закрывал объектив и вспышку камеры, и нажимает кнопку в приложении. После этого приложение включает вспышку и начинает захватывать кадры с камеры мобильного телефона. Поскольку сердечные сокращения вызывают изменения яркости рисунка кровеносных сосудов, анализ этой яркости на получаемых кадрах позволяет определить частоту пульса пользователя.

В научной литературе описано несколько алгоритмов для измерения частоты пульса с помощью мобильного телефона. Однако их эффективность является недостаточной для использования в реальных приложениях. В данной статье авторы предлагают улучшенный алгоритм измерения частоты пульса человека с помощью мобильного телефона, эффективность которого выше, чем у аналогов.

1. Измерение частоты пульса с помощью мобильного телефона

В качестве входного сигнала для вычисления частоты пульса используется средняя интенсивность красной компоненты кадра. В нашем случае входной сигнал считается дискретным с частотой дискретизации равной частоте кадров, получаемых с камеры мобильного устройства. Примеры графиков входного сигнала изображены на рис. 1. Данный сигнал содержит «острые» локальные максимумы (пики), соответствующие сокращениям сердца. Зная количество пиков и длительность измерения, можно легко посчитать частоту пульса.

²<http://www.azumio.com/apps/heart-rate/>

³<http://macropinch.com/cardiograph/>

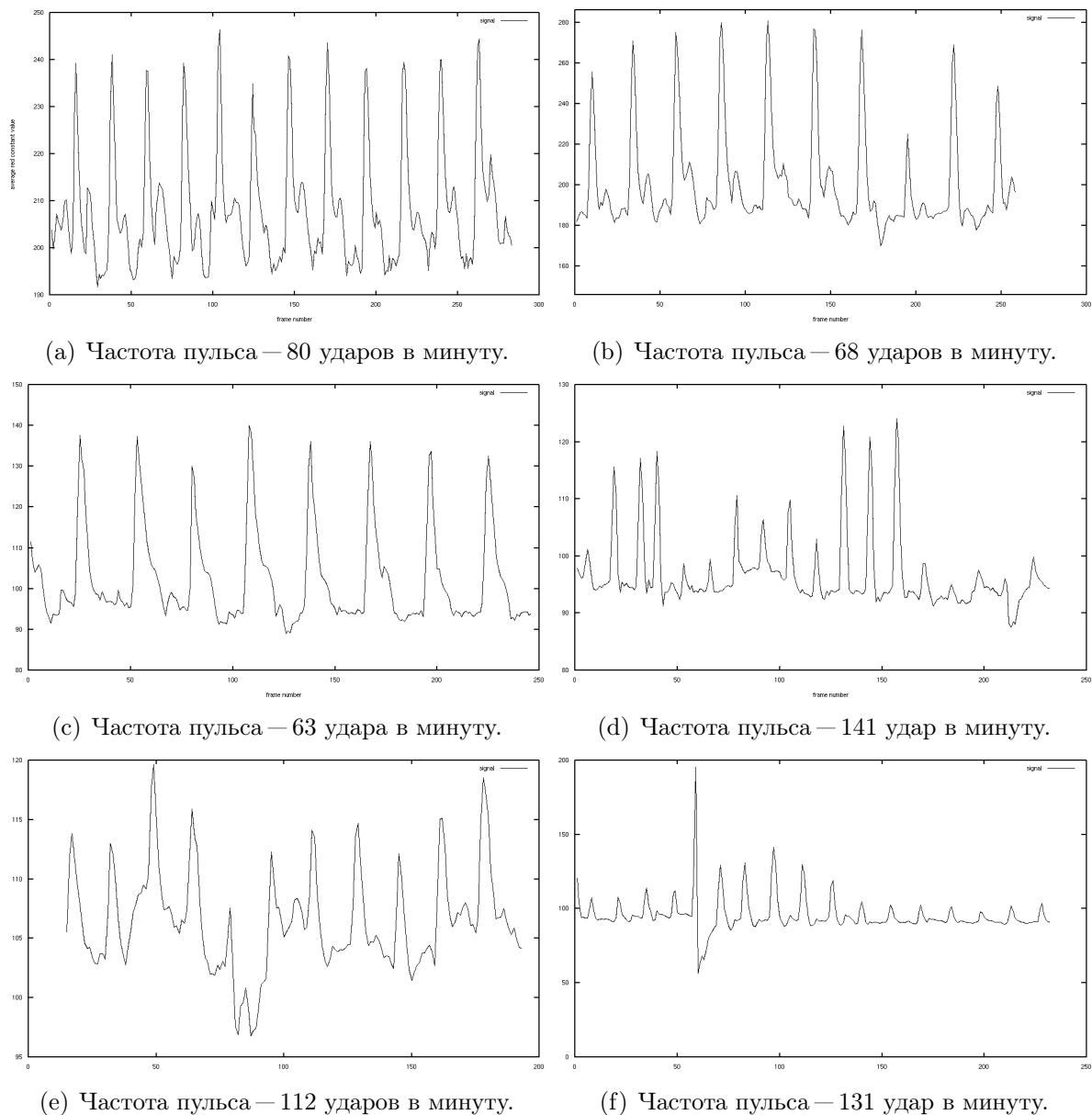


Рис. 1. Примеры графиков среднего значения красной компоненты кадров

Однако необработанный сигнал содержит много шумов, и в нём могут присутствовать лишние пики или отсутствовать пики, соответствующие некоторым сокращениям сердца. Поэтому при вычислении частоты пульса нельзя полагаться на количество пиков исходного сигнала, и необходим алгоритм, который выбирает из всех пиков такие, которые действительно соответствуют сердечным сокращениям. Разработка подобного алгоритма и составляет основную сложность решаемой задачи.

2. Существующие алгоритмы

Авторы рассмотрели две научные работы, в которых описывается измерение частоты пульса с помощью камеры мобильного телефона.

Алгоритм, описанный в [4], предполагает фильтрацию сигнала с помощью скользящего среднего. Отфильтрованный сигнал разбивается на окна фиксированной длины. Сигнал в каждом окне сравнивается с синусоидальным шаблоном. Если сигнал совпадает с шаблоном, то подсчитывается количество пиков в окне. Частота пульса вычисляется по формуле

$$HR = n \times \frac{60}{l}, \quad (1)$$

где HR — частота пульса, n — количество пиков, l — длина окна в секундах.

В работе [5] авторы предлагают нормализовать сигнал, используя сглаживающее дифференцирование (smooth differentiation). После чего производится подсчёт количества пиков нормализованного сигнала. Вычисление частоты пульса, так же как и в случае алгоритма, описанного в [4], осуществляется по формуле (1).

Для проверки оценки эффективности данных алгоритмов было записано 50 тестовых видеофайлов. В качестве устройства записи использовался мобильный телефон Nokia E7-00. Для тестирования был выбран наименьший возможный размер кадра (равный для данной модели телефона 176×144 пикселя), поскольку алгоритм требует лишь среднее значение красной компоненты кадров, которое практически не зависит от его размеров.

В качестве цветовой модели видеофайлов была выбрана модель YUV, так как она является стандартной для данной модели телефона. В цветовой модели YUV цвет пикселя представляется тройкой величин (Y, U, V) , где Y — яркость, а U и V соответствуют двум цветоразностным компонентам. Видеофайлы записывались в формате YUV422, в котором каждая пара пикселей описывается 4 битами. Это позволяет сократить объем записываемых файлов в 1.5 раза по сравнению со стандартным форматом RGB. В видеофайлах формата YUV422 биты расположены следующим образом: Y_1, U, Y_2, V . Данная последовательность из четырёх битов позволяет закодировать один макропиксель, состоящий из двух соседних горизонтальных пикселей. Получение значений красной компоненты, необходимое для алгоритма, осуществлялось по формулам

$$\begin{aligned} R_1 &= \text{clamp}((298 \cdot (Y_1 - 16) + 409 \cdot (U - 128) + 128) >> 8), \\ R_2 &= \text{clamp}((298 \cdot (Y_2 - 16) + 409 \cdot (U - 128) + 128) >> 8), \end{aligned}$$

где R_1, R_2 — значения красной компоненты первого и второго пикселей соответственно; $>>$ — операция побитового сдвига вправо, а функция $\text{clamp}()$ усекает передаваемое ей значение в диапазоне от 0 до 255^4 . Среднее значение красной компоненты кадра вычислялось по формуле:

$$\bar{R} = \frac{1}{\text{frame_size}} \sum_{i=1}^{\text{frame_size}} R_i,$$

где R_i — значение красной компоненты i -го пикселя кадра, frame_size — размер кадра (для тестовых видеофайлов это 176×144).

⁴<http://msdn.microsoft.com/en-us/library/ms893078.aspx>

3. Оценка эффективности работы алгоритмов

Для оценки эффективности работы алгоритмов использовался пакет GNU Octave⁵. Для реализации алгоритмов была применена функция *filter(b, a, x)* из стандартного набора функций Octave, которая возвращает решение линейного разностного уравнения

$$\sum_{k=0}^N a_{k+1} \cdot y_{n-k} = \sum_{k=0}^M b_{k+1} \cdot x_{n-k}, \quad 1 \leq n \leq L, \quad (2)$$

где a и b — векторы коэффициентов фильтра; x — вектор исходного сигнала; N , M , L — размерности векторов a , b и x соответственно.

Поскольку в работах [4] и [5] не представлено никакой информации о параметрах фильтров и способе детекции пиков, были выбраны параметры, при которых алгоритмы давали наименьшее количество ошибок. Для каждого из фильтров параметром фильтрации является количество точек исходного сигнала, используемых для вычисления значения фильтрованного сигнала, — величина M в уравнении (2). Оба алгоритма показали наилучшие результаты тестирования с параметром равным 13. Большие значения параметра слишком сильно сглаживают исходный сигнал, так что в нём не остается ярко выраженных пиков. Меньшие значения, наоборот, оставляют слишком большое количество ложных пиков.

Для алгоритма, описанного в [4], все коэффициенты равны 1/13, а вызов функции *filter* выглядит следующим образом:

$$filtered_signal = filter(ones(1, 13), 13, signal).$$

Здесь *signal* и *filtered_signal* — значения исходного и отфильтрованного сигналов соответственно, а вызов функции *ones* возвращает вектор из тринадцати единиц.

Для алгоритма, описанного в [5], коэффициенты вычислялись с помощью функции *smooth_diff*. Данная функция не является стандартной функцией Octave, она была написана Jianwen Luo специально для расчёта коэффициентов фильтра сглаживающего дифференцирования⁶. В качестве аргумента данной функции передаётся параметр фильтра. Вызов функции *filter* для алгоритма, описанного в [5], выглядит следующим образом:

$$filtered_signal = filter(-smooth_diff(13), 1, signal);$$

Для детекции пиков был использован Octave-скрипт, разработанный Eli Billauer⁷. Данный скрипт содержит функцию *peakdet*, которая определяет направленные вверх (локальные максимумы) и вниз (локальные минимумы) пики сигнала. В качестве аргументов данной функции передается сам сигнал и параметр δ . Функция возвращает два вектора. Первый вектор содержит локальные максимумы, значения которых отличаются от значений соседних точек на величину, большую δ . Второй вектор содержит локальные минимумы, значения которых отличаются от значений соседних точек на величину, большую δ .

⁵<http://www.gnu.org/software/octave/>

⁶<http://www.mathworks.com/matlabcentral/fileexchange/6170-smooth-differentiation>

⁷<http://www.billauer.co.il/peakdet.html>

Таблица 1. Эффективность алгоритмов вычисления частоты пульса

Алгоритм	Ошибочные вычисления, %	Относительная ошибка, %
Фильтр скользящего среднего [4]	20	8.5
Сглаживающее дифференцирование [5]	20	13.16
Первая версия предложенного алгоритма [3]	20	41.38
Окончательная версия предложенного алгоритма	8	10.93

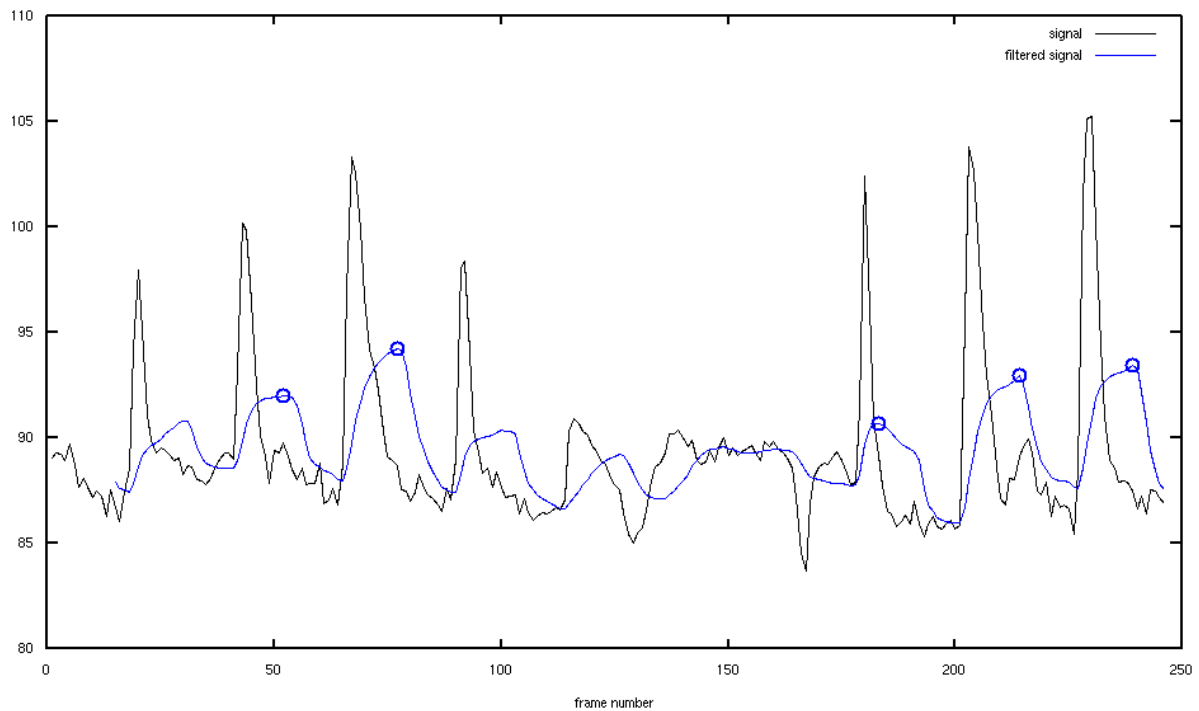


Рис. 2. Пример сигнала, в середине которого отсутствуют пики

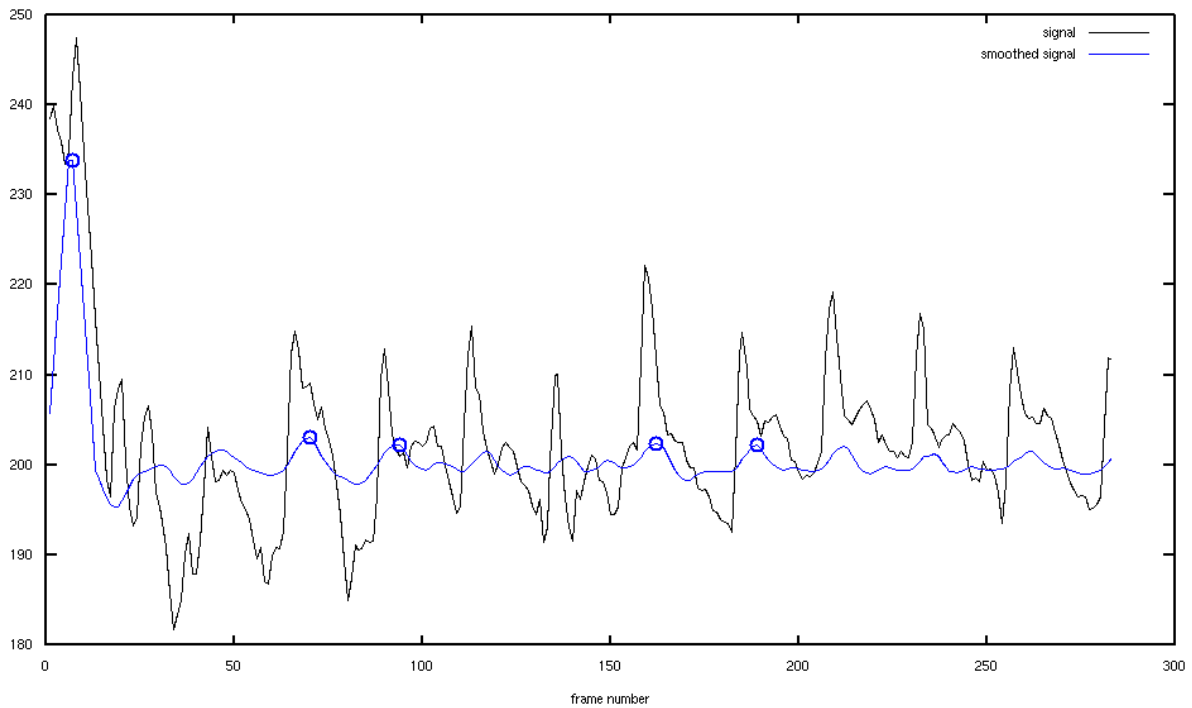
Для определения частоты пульса использовался лишь первый вектор. Параметр δ был выбран равным 3, так как именно при этом значении удавалось точно определить пики сигнала, но не детектировать ложные пики. Вызов данной функции выглядит следующим образом:

$$peaks = peakdet(filtered_signal, 3);$$

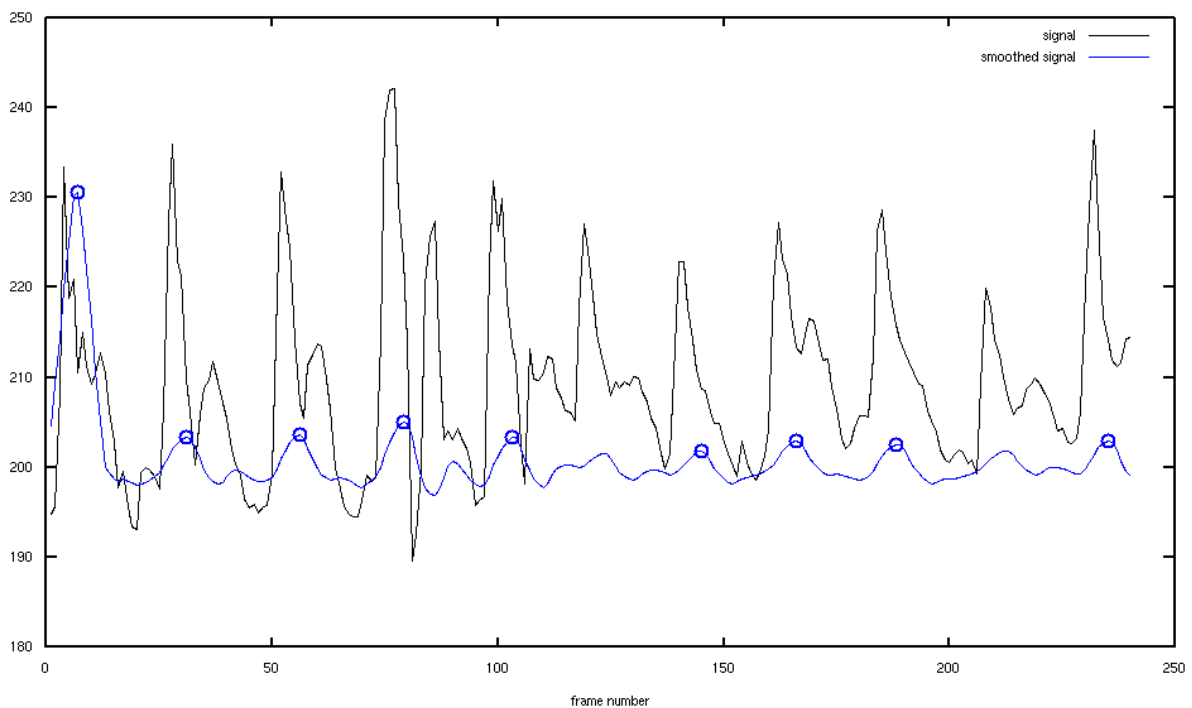
Здесь *filtered_signal* — вектор, содержащий значения отфильтрованного сигнала; *peaks* — вектор, содержащий индексы и соответствующие им значения сигнала в пиках, детектированных скриптом.

Результаты оценки эффективности приведены в первой и второй строках таблицы 1. Для обоих алгоритмов примерно 20 % результатов вычислений оказались ошибочными (такowymi мы считали результаты, отличающиеся от реальных значений более чем на 5 %).

Такой высокий процент ошибочных вычислений недопустим в реальных приложениях. Поэтому было решено разработать новый алгоритм вычисления частоты пульса. Анализ причин получения неправильных результатов показал, что нельзя



(a)



(b)

Рис. 3. Примеры отфильтрованных сигналов, на которых некоторые пики оказались сглаженными

полностью полагаться на фильтрацию, поскольку она не может избавить сигнал от проблемы отсутствия пиков (на рис. 2 изображен пример сигнала, в середине которого отсутствуют пики), а иногда даже может сгладить некоторые пики, соответствующие сокращениям сердца (на рис. 3 изображены два примера отфильтрованных сигналов, на которых некоторые пики оказались сглаженными). Следовательно, нужна другая идея для вычисления частоты пульса.

4. Предложенный алгоритм

Первая версия предложенного алгоритма была описана в статье [3]. Главная идея решения заключается в том, чтобы считать не количество пиков сигнала, а среднее расстояние между ними.

Алгоритм состоит из следующих шагов:

1. Дифференцирование сигнала. Данная операция позволяет выровнять сигнал так, что становится возможным сравнивать высоты пиков. Дифференцирование производится по формуле

$$f'(x_i) \approx \frac{f(x_{i+1}) - f(x_{i-1}))}{2},$$

где $f(x_i)$ и $f'(x_i)$ — значение сигнала и его производной для i -го кадра соответственно ($2 \leq i \leq \text{frame_number} - 1$), а frame_number — общее количество кадров.

2. Формирование наборов из k самых высоких пиков. Здесь k изменяется от 3 до n , где n — максимальное количество рассматриваемых пиков. Для сигнала длительностью 10 секунд используется n равное 20.
3. Вычисление дисперсии расстояний между соседними пиками для наборов, полученных на предыдущем шаге. Дисперсия для каждого набора определяется по формуле

$$\text{variance} = \frac{1}{k-1} \cdot \sum_{i=1}^{k-1} \left(l_i - \frac{1}{k-1} \cdot \sum_{j=1}^{k-1} l_j \right)^2,$$

где l_i — расстояние между i -м и $(i+1)$ -м пиками набора (здесь мы считаем, что пики в наборе упорядочены по времени их появления), а k — общее количество пиков в наборе.

4. Выбор подходящего набора пиков. На данном шаге выбирается набор пиков с наименьшим значением дисперсии расстояний.
5. Вычисление частоты пульса. Частота пульса определяется по формуле

$$HR = \frac{\text{frame_rate} \times 60}{\text{mean}},$$

где HR — частота пульса, frame_rate — частота кадров исходного видеофайла, mean — среднее значение разностей между соседними пиками выбранного на предыдущем шаге набора.

На тестовых наборах данных разработанный алгоритм показал достаточно хорошие результаты — около 10 % ошибок. Однако после реализации алгоритма в приложении Pulse Detector авторы получили значительное количество негативных отзывов от пользователей приложения. После этого было решено собрать больше тестовых данных. На тестовом наборе из 50 видеофайлов алгоритм показал 20 % ошибочных вычислений (третья строка таблицы 1), что эквивалентно результатам работы алгоритмов, рассмотренных выше, из чего был сделан вывод, что первоначально использованные наборы тестовых данных были не вполне репрезентативными.

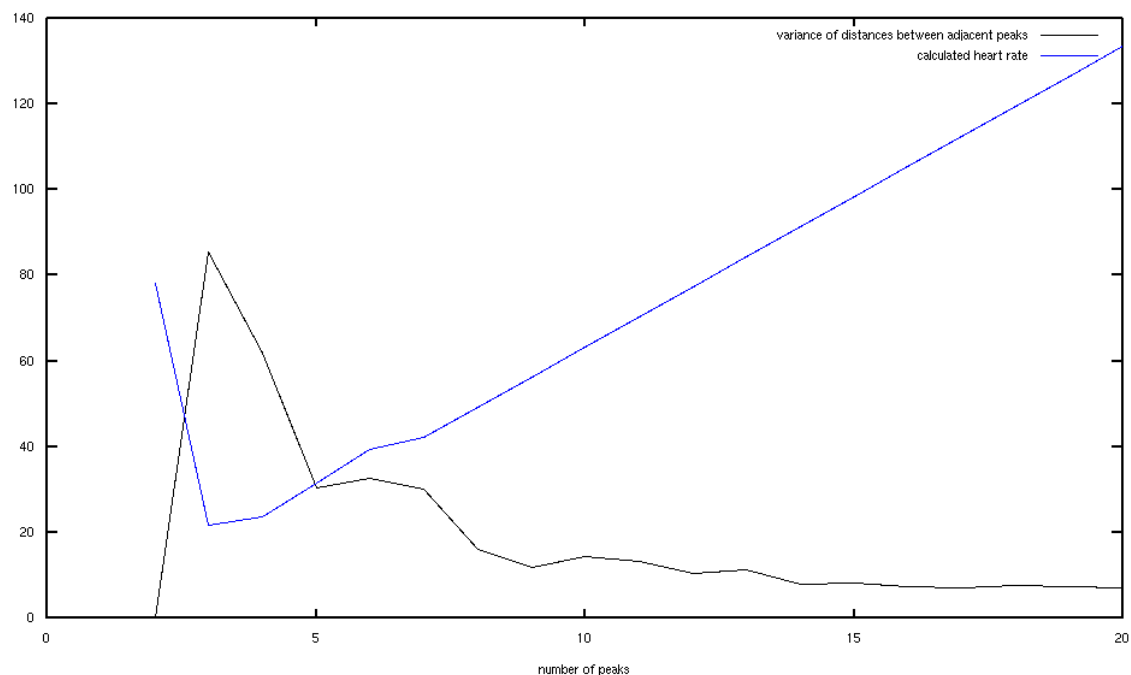
В результате более детального анализа работы алгоритма авторы определили, что алгоритм показывает хорошие результаты для сигналов, в которых отсутствуют некоторые пики. Однако лишние пики и шум иногда приводят к ситуации, в которой минимум дисперсий расстояний между соседними пиками достигается для набора с максимальным количеством пиков, тогда как количество настоящих (соответствующих сокращениям сердца) пиков сигнала меньше максимального. Пример такой ситуации изображен на рис. 4. Подобные случаи приводят к неверному определению частоты пульса, когда вычисленное алгоритмом значение выше, чем реальное.

Описанную проблему можно решить, уменьшив максимальное количество рассматриваемых пиков, однако это приведет к неправильным вычислениям при высоких частотах пульса, так как в этом случае сигнал будет содержать большее количество пиков. Поэтому такой подход неприменим.

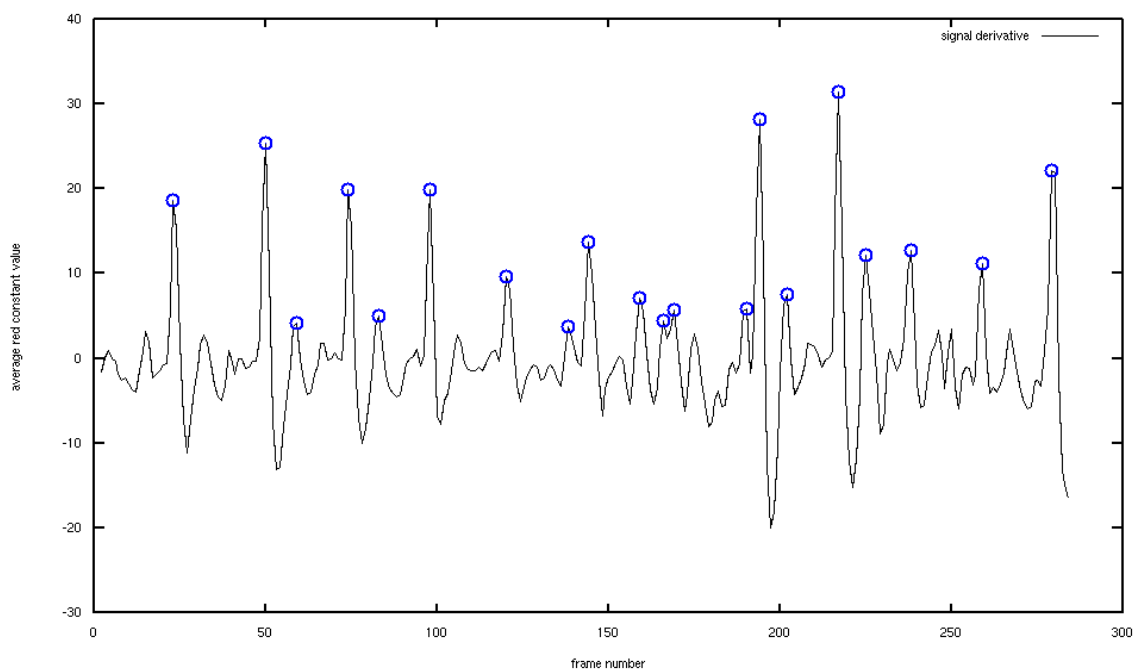
В результате анализа возможных способов решения поставленной задачи в алгоритм были внесены следующие изменения:

1. Минимальное количество рассматриваемых на втором шаге алгоритма пиков k увеличено с 3 до 5, так как 5 пиков при 10-секундном измерении соответствуют частоте пульса в 30 ударов в минуту.
2. На шаге 4 алгоритма, если минимум дисперсий расстояний достигается для набора с максимальным количеством пиков (n), то выбирается набор, дисперсия расстояний которого является локальным минимумом последовательности дисперсий.
3. Набор пиков, полученный на шаге 4, фильтруется путем удаления из него пиков, которые отстоят от одного из соседних пиков на расстояние, меньшее, чем минимальное допустимое расстояние. Минимальное допустимое расстояние равно 60 секундам, умноженным на частоту кадров и делённым на 200. Это расстояние соответствует 200 ударам в минуту. Меньшие расстояния соответствуют более высоким частотам пульса, которые недостижимы для обычного человека [6].
4. Полученный на предыдущем шаге набор пиков фильтруется еще раз путём удаления из него пиков, для которых расстояние до одного из соседних пиков отличается от среднего более чем на max_diff процентов. Для предложенного алгоритма значение max_diff равняется 25 %.

Окончательный результат на псевдокоде приведён в алгоритме 1 настоящей работы.



(а) Графики зависимости дисперсии расстояний между соседними пиками и вычисленными значениями частоты пульса от количества пиков в наборе. Минимум дисперсии достигается при максимальном количестве пиков.



(б) График производной сигнала. Выделенные пики соответствуют выбранному алгоритмом набору пиков.

Рис. 4. Пример ошибочного вычисления первой версии предложенного алгоритма. Реальная частота пульса: 77 ударов в минуту. Вычисленная алгоритмом частота пульса: 133 удара в минуту

Алгоритм 1 Окончательная версия алгоритма определения пульса.

Require: *signal* — последовательность значений красной компоненты кадров, полученных с камеры в процессе измерения; *frame_rate* — частота кадров, полученных в процессе измерения; *n* — максимально возможное количество пиков в выбранном наборе; *max_diff* — максимально допустимое отклонение расстояния между соседними пиками от среднего значения расстояний в наборе.

deriv = производная *signal*

for *value* $\in$ *signal* **do**

if *value* == max в ε -окрестности *value* **then**

peaks += *value*

end if

end for

for *k* = 5 : *n* **do**

distances = расстояния между соседними пиками для *k* самых высоких из *peaks*

variances += *distances*

end for

if min(*variances*) == *n* **then**

chosen_set = набор расстояний, дисперсия которых равна первому локальному минимуму *variances*

else

chosen_set = набор расстояний, дисперсия которых равна min(*variances*)

end if

удалить из *chosen_set* все значения, которые меньше чем $\text{frame_rate} \times 10/33$

repeat

R = значения *chosen_set*, отстающие от $\text{mean}(\text{chosen_set})$ более чем на *max_diff*

 удалить *R* из *chosen_set*

until *R* $\neq \emptyset$

return $60 \cdot \text{frame_rate} / \text{mean}(\text{chosen_set})$

5. Эффективность предложенного алгоритма

Две версии предложенного алгоритма, а также алгоритмы, рассмотренные в разделе 2, были протестированы на 50 видеофайлах. Результаты оценки эффективности данных решений приведены в таблице 1.

При проведении тестирования ошибочными считались вычисленные значения частоты пульса, отличающиеся от реальных значений более чем на 5 %. Для алгоритмов, описанных в [4] и [5], а также для первой версии предложенного алгоритма, доля ошибочных результатов составила 20 % от общего количества тестов. Для окончательной версии предложенного алгоритма данное значение составило всего 8 %.

Заключение

В данной статье проведён обзор существующих алгоритмов измерения частоты пульса человека с помощью мобильного телефона. Было обнаружено, что эффективность рассмотренных алгоритмов не позволяет применять их в реальных приложениях, а также были выявлены причины получения ошибочных результатов.

На основе проведённого анализа был предложен новый алгоритм, основанный на идее анализа расстояний между пиками сигнала. Первоначально эффективность разработанного алгоритма оказалась схожей с эффективностью других алгоритмов. Однако после внесения улучшений удалось повысить эффективность работы алгоритма более чем в два раза. Количество ошибочных результатов окончательной версии алгоритма составила 8 % от всех вычислений, что позволяет использовать данный алгоритм в различных персональных немедицинских приложениях.

Текущая версия алгоритма была успешно реализована в мобильном приложении Pulse Detector. Количество загрузок из официального магазина Nokia Store данного приложения на момент написания статьи составляет примерно 70 тысяч. Приложение доступно по адресу <http://store.oivi.com/content/314173>.

Список литературы

1. Karvonen J., Vuorimaa T. et al. Heart rate and exercise intensity during sports activities. practical application // Sports medicine (Auckland, NZ). 1988. Vol. 5, No. 5. P. 303.
2. Laukkanen R. M., Virtanen P. K. Heart rate monitors: state of the art // Journal of Sports Sciences. 1998. Vol. 16, No. suppl. P. 3–7.
3. Laure D. Heart rate measuring using mobile phone's camera // Proceedings of the 12th Conference of Open Innovations Association FRUCT and Seminar on e-Travel. Oulu, Finland, November 5–9, 2012. St.-Petersburg : SUAI, 2012. P. 272–273.
4. Dantu R. Measuring Vital Signs Using Smart Phones: Ph.D. thesis. University of North Texas, 2010.
5. A simple algorithm to monitor hr for real time treatment applications / K. Banitsas, P. Pelegris, T. Orbach et al. // Information Technology and Applications in Biomedicine, 2009. ITAB 2009. 9th International Conference on / IEEE. 2009. P. 1–5.
6. Tanaka H., Monahan K. D., Seals D. R. Age-predicted maximal heart rate revisited // Journal of the American College of Cardiology. — 2001. Vol. 37, No. 1. P. 153–156.

Development of an Algorithm for Heart Rate Measurement Using a Mobile Phone Camera

Laure D. A., Lagutina N. S., Paramonov I. V.

*P.G. Demidov Yaroslavl State University,
Sovetskaya str., 14, Yaroslavl, 150000, Russia EwerestMD LLC,
2nd Lyapinskaya Str., 32a, Yaroslavl, 150008, Russia*

Keywords: mHealth, pulse detection, heart rate, mobile phone, camera

Nowadays there exist many different ways to measure a person's heart rate. One of them assumes the usage of a mobile phone built-in camera. This method is easy to use and does not require any additional skills or special devices for heart rate measurement. It requires only a mobile cellphone with a built-in camera and a flash. The main idea of the method is to detect changes in finger skin color that occur due to blood pulsation. The measurement process is simple: the user covers the camera lens with a finger and the application on the mobile phone starts catching and analyzing frames from the camera. Heart rate can be calculated by analyzing average red component values of frames taken by the mobile cellphone camera that contain images of an area of the skin.

In this paper the authors review the existing algorithms for heart rate measurement with the help of a mobile phone camera and propose their own algorithm which is more efficient than the reviewed algorithms.

Сведения об авторах:

Лаурэ Денис Александрович,

Ярославский государственный университет им. П.Г. Демидова,
аспирант факультета ИВТ.

Лагутина Надежда Станиславовна,

Ярославский государственный университет им. П.Г. Демидова,
канд. физ.-мат. наук, доцент каф. вычислительных и программных систем.

Парамонов Илья Вячеславович,

Ярославский государственный университет им. П.Г. Демидова,
канд. физ.-мат. наук, доцент каф. компьютерных сетей;
ООО «ЭверестМД»,
менеджер программных проектов.