

РЕАЛИЗАЦИЯ АЛГОРИТМА ДЛЯ ОПРЕДЕЛЕНИЯ, ПОДСЧЁТА И ОЦЕНКИ ПУЛЬСА КАМЕРОЙ МОБИЛЬНОГО ТЕЛЕФОНА

Белойванов Максим Сергеевич

студент, Институт сферы обслуживания и предпринимательства (филиал) ДГТУ, РФ, г. Шахты

Притоманов Вадим Викторович

студент, Институт сферы обслуживания и предпринимательства (филиал) ДГТУ, РФ, г. Шахты

Грачев Георгий Андреевич

студент, Институт сферы обслуживания и предпринимательства (филиал) ДГТУ, РФ, г. Шахты

Для разработки алгоритма и, для его первоначального тестирования, нам надо было собрать тестовые данные. Для этого потребовалось написать мобильное приложение, которое бы могло работать на одной из известных мобильных платформ. А, также, умело записывать данные с камеры и отправлять их нам.

Для создания приложения мы выбрали платформу «Android»[3]. Этот выбор не случаен. Платформа «Android» пользуется большой популярностью в России[5], среди пользователей мобильных телефонов, к тому же, наша группа уже имела опыт программирования и работы с камерой на этой платформе. Стандартным языком для написания приложений на «Android», является язык «Java», но, для реализации данного приложения, мы выбрали язык «Kotlin» компании «JetBrains»[2]. Данный язык полностью совместим с языком «Java», это позволяет использовать его для написания «Android» приложений. Язык «Kotlin» имеет более короткий синтаксис и встроенную «null» безопасность, поэтому, разработка на нём Android приложений, гораздо удобнее и эффективнее.

Создание приложения.

При разработке приложения мы столкнулись с небольшой проблемой. Дело в том, что на платформе «Android», по умолчанию, обработка кадров с камеры происходит в основном потоке приложения. В результате, при обработке данных, камера начинает присылать кадры (с камеры) гораздо реже.

Пример: если, обычно, камера присылает 25-30 кадров в секунду то, когда мы начинаем обрабатывать каждый кадр, в основном потоке приложения (например, считать среднее значение красного канала на кадре), то количество кадров в секунду падает до 10-15.

Для решения такой проблемы в приложении был создан отдельный поток, в котором происходит приемка кадров с камеры и их обработка.

Стоит отметить, что стандартным форматом для кадров с камеры в платформе «Android» является-«YUV420SP», а не «RGB». Поэтому, каждый кадр, дополнительно, нужно переводить в цветовую палитру «RGB». Для отправки тестовых данных с камеры, мы решили использовать обычную электронную почту. В приложении создаётся запрос на отправку сообщения с прикрепленными данными, на заданный в приложении, E-mail.

Алгоритм.

Общая структура алгоритма показана на рисунке 1.

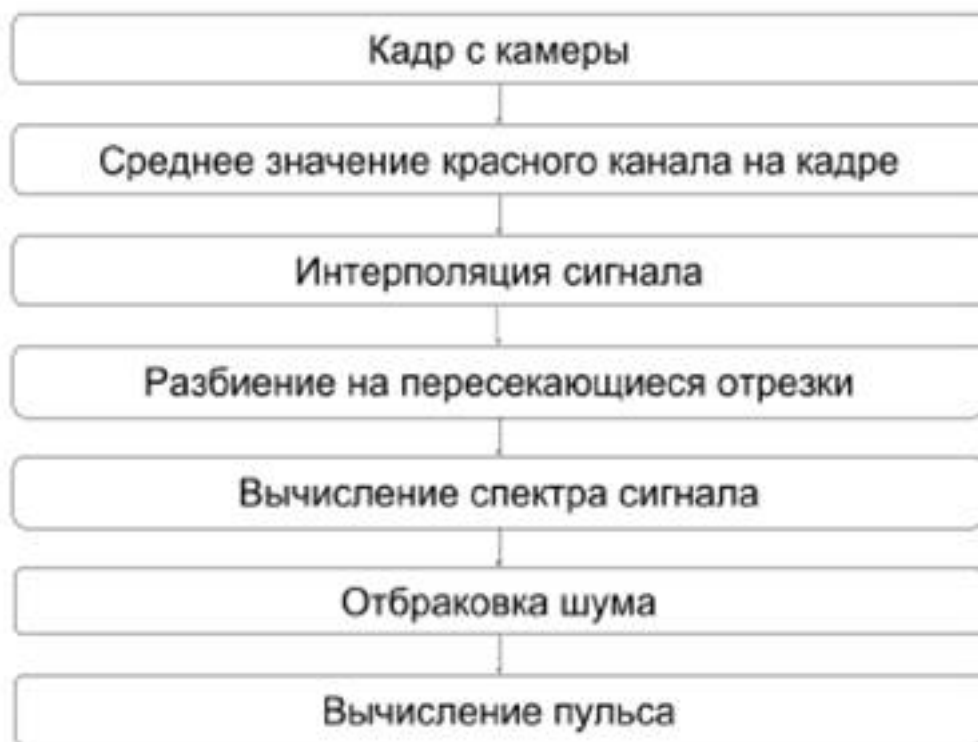


Рисунок1. Структура Алгоритма

Получение среднего значения красного канала.

Для каждого кадра, в качестве его числовой характеристики, используется среднее значение красного канала (рисунок 2). Среднее значение считается для всего кадра, и оно может меняться в пределах от 0 до 255.



Рисунок 2. Кадр с камеры с увеличенной контрастностью

Интерполяция сигнала.

Для получения спектра сигнала с помощью «БПФ», исходный сигнал должен иметь одинаковый и равномерный шаг дискретизации по времени. Однако, не все мобильные платформы, и «Android» в том числе, могут гарантировать приход кадров с камеры через равные промежутки времени, поэтому перед применением алгоритма «БПФ» проводится интерполяция исходного сигнала. При реализации алгоритма, одним из основных критериев - является скорость вычислений. Поэтому, в качестве интерполяции, была выбрана «интерполяция полиномом первой степени» (линейная интерполяция). Узлы сетки интерполяции выбирались с равномерным шагом, равным средней частоте оцифровки кадров камеры. Обычно, такой шаг составлял 30 - 34 м/с. Это, вполне, соответствует необходимой точности измерений.

Разбиение на пересекающиеся отрезки.

После интерполяции, полученный сигнал, разбивается на пересекающиеся отрезки. Длина каждого отрезка, составляет одну четверть от исходной длины измерения, а смещение - половину длины. Таким образом, измерение делится на семь пересекающихся отрезков.

Для измерения, длинной около 17 секунд, длина отрезка будет составлять примерно 4.25 секунды. Такой длины достаточно, чтобы предварительно оценить пульс, так как, даже при низком пульсе, равном 40 уд/мин, в такой отрезок попадёт информация не менее, чем о трёх притоках крови к пальцу, т.е. три периода сигнала.

Как уже говорилось ранее, разбиение на отрезки приводит к увеличению шага дискретизации амплитудного спектра. В данном примере, для измерения длиной - 512 сек, с частотой -30, шаг дискретизации спектра будет равен ≈ 14 .

Такая приблизительная точность результата, является неприемлемой для конечного результата. Но так как, разбиение на отрезки используется, исключительно для примерной оценки пульса, то такой точности нам будет достаточно.

Вычисление спектра сигнала

На каждом полученном отрезке, с помощью «БПФ», вычисляется дискретный амплитудный спектр сигнала. Частоты рассчитанного спектра сигнала, впоследствии, преобразуются в пульс - уд/мин.

Отбраковка шума

После предыдущего этапа мы получаем семь спектров. На этом этапе надо определить, какие из отрезков выбиваются из общей картины, т.е. являются шумом. Если большинство отрезков является мусором, нам придётся отбраковать всё измерение.

В каждом спектре находятся два наибольших пика (рисунок 3). Если у большинства отрезков совпадает хотя бы один из двух пиков - он принимается за предварительный результат и передаётся на следующий этап. А те отрезки, у которых нет данного пика на спектре, отбраковываются.

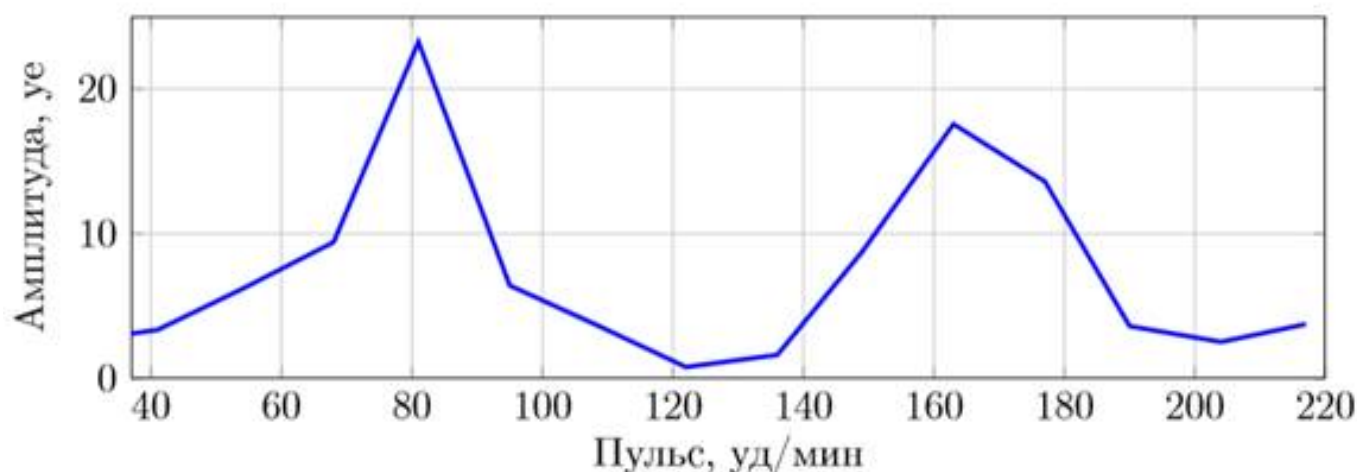


Рисунок 3. Участок спектра для выбранного отрезков

Если у четырёх из семи отрезков нет общего пика - измерение считается плохим и результат пользователю не выдаётся (рисунок 4).

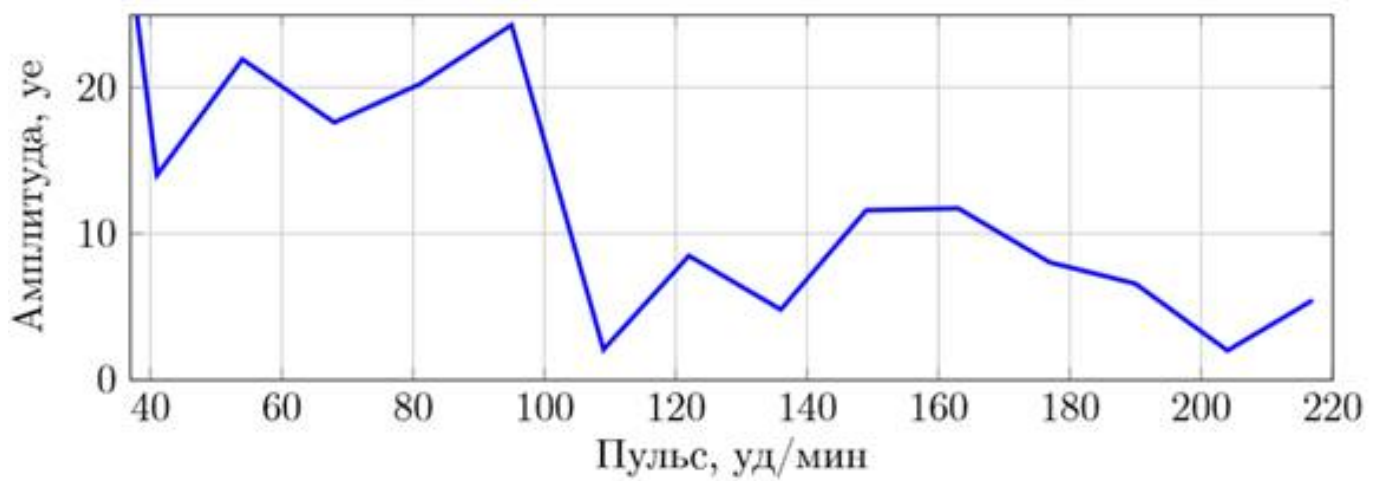


Рисунок 4. Участок спектра для отрезка, признанного, как шум

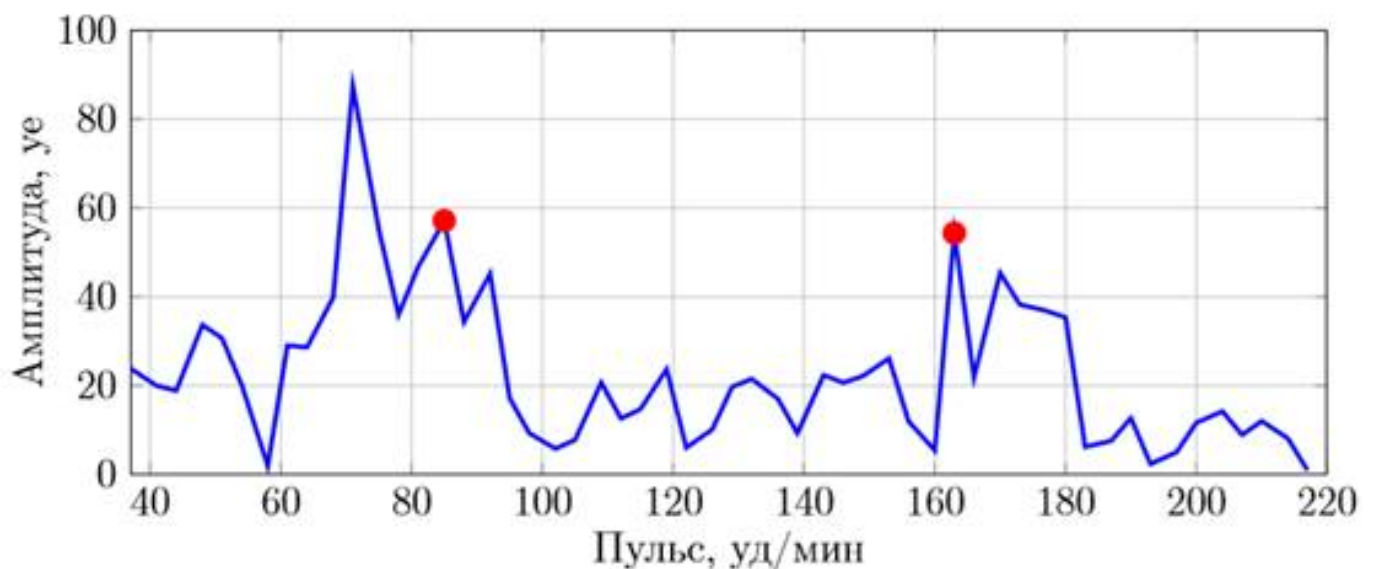
Вычисление пульса

Перед входом финальной стадией алгоритма, нами получены общие пики и отрезки, которые не являются шумом. Задача финального этапа - выдать пульс. Сначала с помощью «БПФ» мы строим спектр для всего измерения. Потом, на спектре всего измерения детектируются пики, которые соответствуют выбранным общим пикам в отрезках. Другими словами, выявляются пики на спектре всего измерения, расстояние между вершинами, которых, минимально, по отношению к соответствующим вершин, из прошлого этапа. Среди найденных таким образом пиков, выбирается наибольший по амплитуде.

Если, полученный таким образом пик, отклоняется от исходного пика больше, чем на половину шага дискретизации спектра для отрезков, (например, для спектров на рисунках 3 и 4 шаг дискретизации равен четырнадцати), то он дополнительно проверяется подсчётом спектра для той части измерения, которая содержит наименьшее количество шума.

Если результаты совпадают, то его выдают как ответ. Если не совпадают - измерение считается некачественным.

На рисунке 5 изображены пики, найденные на спектре всего измерения, которые соответствуют двум пикам со спектра, изображённого на рисунке 3.



Внедрение алгоритма в приложение.

Алгоритм был оформлен как отдельный класс и, как приложение, написан на языке «Kotlin». После измерения, все данные сохраняются, и, обязательно, передаются на вход алгоритму. Результат работы алгоритма отображается на экране.

Тестирование.

Полученный результат работы алгоритма, сравнивался с измерениями пульсоксиметра «Nonin-3150»[1]. Измерения проводились на людях европеоидной расы. Это считается важным, так как результаты работы, такого рода алгоритмов, могут зависеть от цвета кожи. Измерения проводились при неподвижном положении руки, поскольку движение руки, особенно на улице, может привести к изменению освещения с одинаковой частотой (например, движение руки при беге), это приведёт к искажению результата у алгоритма. В тестировании принимало участие около 20 человек, различных возрастов. В качестве итоговой характеристики работы алгоритма, выбран процент измерений, в которых алгоритм дал отклонение пульса меньше, чем на 5 уд/мин, т.е., где алгоритм достаточно точно оценил пульс. Из результатов тестирования можно заметить, что на точность алгоритма не влияет яркий солнечный свет, и точность алгоритма не зависит от размера реального пульса.

Внедрение алгоритма в платформу «MedM».

В нашем случае, выбор инструментов обусловлен архитектурой платформы «MedM». Алгоритм внедрён в платформу-независимую часть, а для этого мы переписали его на «C++». Это сделано для того, чтобы можно было использовать один алгоритм для подсчёта пульса на различных мобильных платформах.

Таблица 1.

Результаты тестирования алгоритма

Пульс	% измерений с относительной погрешностью < 5 уд/мин	
	В помещении	На улице
Обычный (<100 уд/мин)	86	85
Повышенный (<140 уд/мин)	85	84
Высокий (>140 уд/мин)	85	83

Работа с камерой на платформе «Android» внедрена в платформу-зависимую часть. Но, так как платформу-зависимая часть в платформе «MedM» записана на языке «Java», работу с камерой пришлось переписать с языка «Kotlin» на «Java».

Платформа «MedM», изначально, была создана для работы с внешними медицинскими устройствами, поэтому нашему алгоритму пришлось эмулировать такое устройство. Работа с внешним устройством состоит из четырёх этапов:

- Обнаружение устройства.
- Запоминание устройства.
- Соединение с устройством.
- Получение данных с устройства.

Общая схема взаимодействия пользователя платформы «MedM» с нашим алгоритмом, эмулирующим внешнее устройство, представлена на рисунке 6.

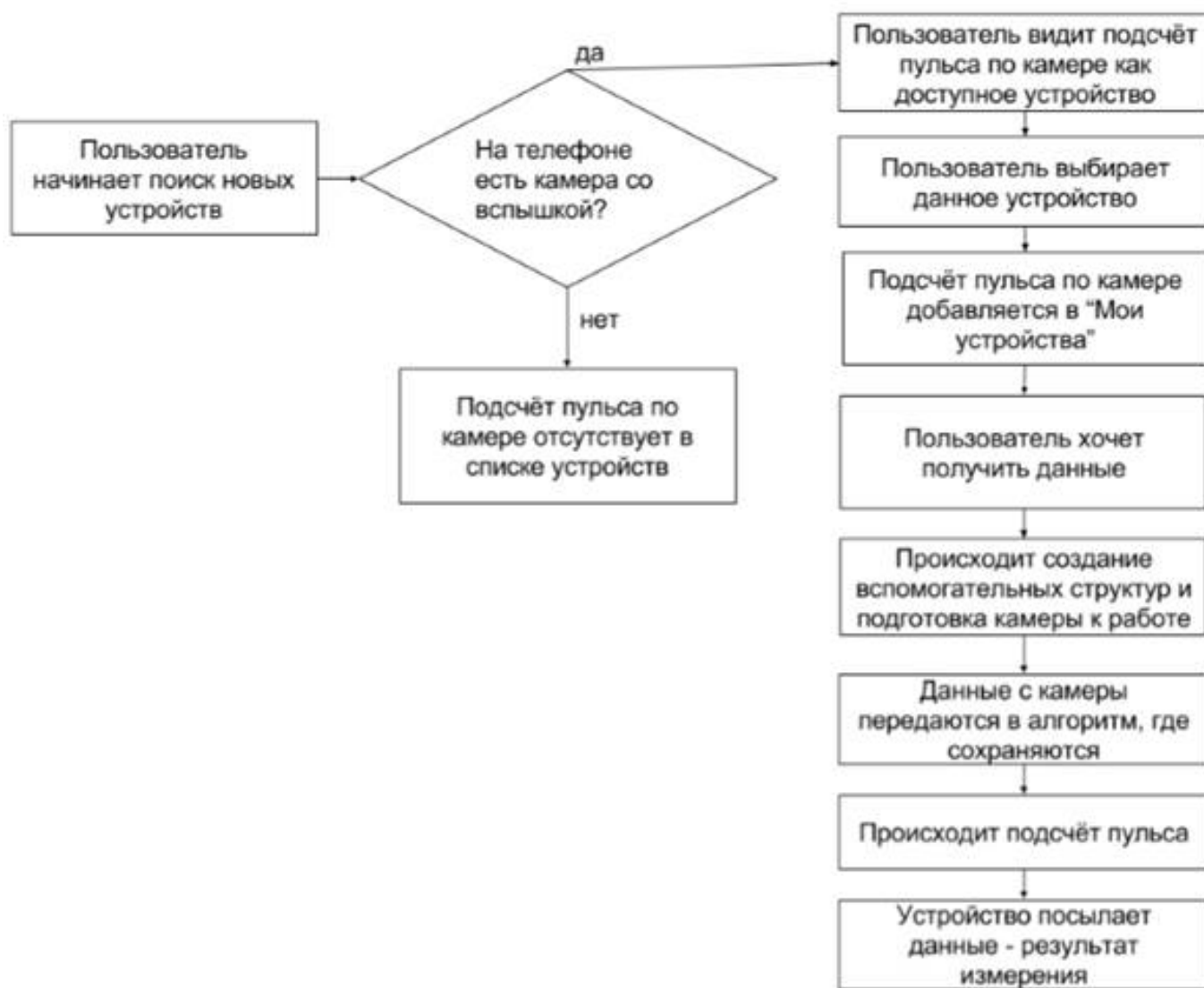


Рисунок 6. Схема взаимодействия пользователей платформы MedM с алгоритмом

Список литературы:

1. A Novel Method to Detect Heart Beat Rate Using a Mobile Phone /Arpan Pal, Aniruddha Sinha, Anirban Dutta Choudhury et al. -- 2013.
2. A Novel Method to Detect Heart Beat Rate Using a Mobile Phone /Pelegris P., Banitsas K., Orbach T., Marias K. -- 2010.

3. Improved heart rate detection using smart phone /Arpan Pal, Aishwarya Visvanathan, Anirban Dutta Choudhury,Aniruddha Sinha. -- 2014.
4. Fan Xiangmin, Wang Jingtao. BayesHeart: A Probabilistic Approachfor Robust, LowLatency Heart Rate Monitoring on Camera Phones. --2015.
5. Laure Denis, Paramonov Ilya. Improved Algorithm for Heart RateMeasure-ment Using Mobile Phone Camera. -- 2013.