Pulse Recognition by Video Project Development

Konstantin Stepanov
OSLL
St. Petersburg, Russia
ksstepanov92@gmail.com

Abstract

There are algorithms that can show changes on video, that are invisible by naked eye but not a video camera. Such algorithm was applied to recognize human pulse by a video, where there is a part of a skin. The algorithm detects invisible periodic changes of human skin color, amplifies them and shows to us. These algorithms haven’t yet become widely known, but they are going to do that. The goal of this project is to implement one of these algorithms as applied to the task of pulse recognition and developed an application for mobile devices, which could be simply used by everyone.

Index Terms: Pulse, Pulse recognition, Eulerian Video Magnification

I. Introduction

In 2012 group of researchers from MIT CSAIL and Quanta Research Cambridge suggested a new algorithm named “Eulerian Video Magnification” (EVM) of amplifying subtle changes in video stream, which are invisible to a naked eye [1]. They applied it to a problem of pulse recognition and developed a program, written in MATLAB, which amplified invisible pulse on video stream and show it to us. The goal of this project was to understand how the algorithm works and to implement it for mobile devices. Because of nowadays mobile devices market is growing rapidly, the aim was to make the application compatible with Android platform, and to have good performance on mobile hardware resources. So this article is a resume, describing application development, API products used in it and some problems appeared.

II. Project description

A. Description of the EVM algorithm

Here is a description in brief of the algorithm part concerning to a task of pulse recognition. It can be as follows:

- Reading an input video file and getting a sequence of frames from it.
- Spatial processing of frames: noise filtering.
- Temporal processing of frames sequence and getting pulse signal from it.
- Amplification of the pulse signal and sum it with the original frame sequence.
- Combining the result frames into a video.

Next there is a description of application code according to these tasks which shows how they are implemented in application.

B. Description of the application development.

As the application is planned to be used on mobile phone, it was developed in Qt Framework, using C++.

To get pulse signal as input algorithm requires only 2-3 seconds of video with human skin. Fig. 1 shows how the application works in general with video flow from camera.

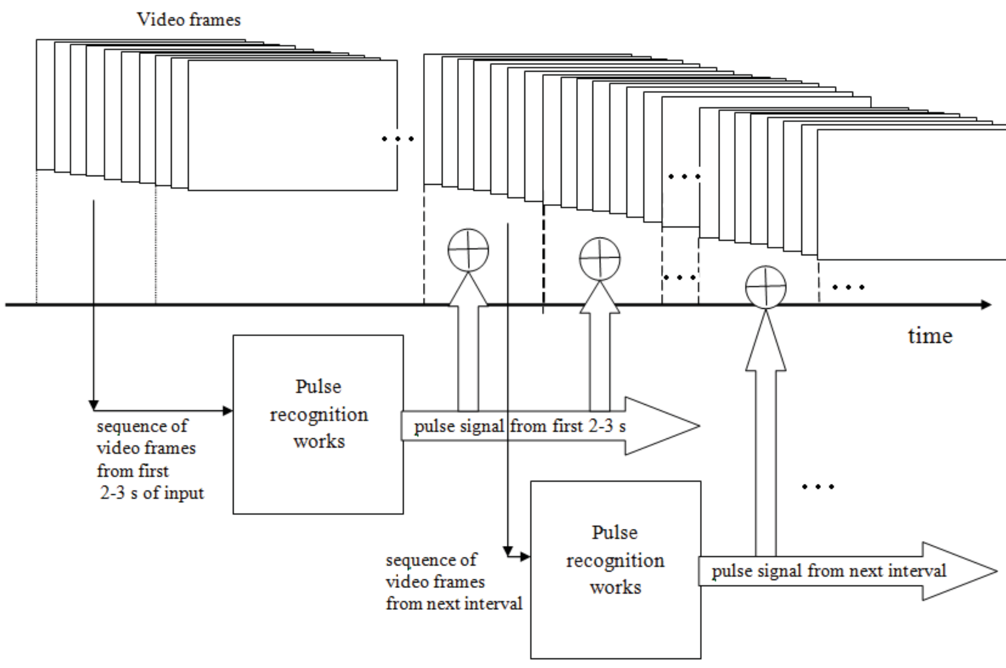
Video frames
time
Pulse recognition works
sequence of video frames from first 2-3 s of input
pulse signal from first 2-3 s
Pulse recognition works
sequence of video frames from next interval
pulse signal from next interval

Fig. 1. General algorithm of the application

Program takes sequence of frames from input video required to recognize pulse(2-3 seconds) and gives it as input to module of pulse recognition(which implements EVM algorithm). While first frames are processed no pulse is visible. Program gets pulse signal, which length corresponds to the length of input frames, which were taken to get this pulse. After program has got pulse signal, it adds this signal to current video frames. Since this moment pulse is visible like in real-time. After program has processed previous frame sequence interval it takes new one from current moment, gets pulse from it and then adds this new pulse signal to next input frames. In Fig. 2 there is a scheme of EVM algorithm, and then in 2.1 – 2.5 goes description of its implementation in this application.

2.1 Read an input video file and get a sequence of frames from it.

This task required an open source library which provide working with video codecs, so it was chosen FFMPEG. The last FFMPEG version was taken from git repository. It provided all functions for reading a video, searching for the needed video stream and getting frames. Frames were received in RGB format as arrays of unsigned char in specific FFMPEG container.

Next task was to decide how to store the frames in convenient way for next processing. For the next step - building a Gaussian stack for each frame was taken the OpenCV library, which provide this operation and frames were put in Mat - cv container. So in the end of this step was received an array of Mats with RGB frames.

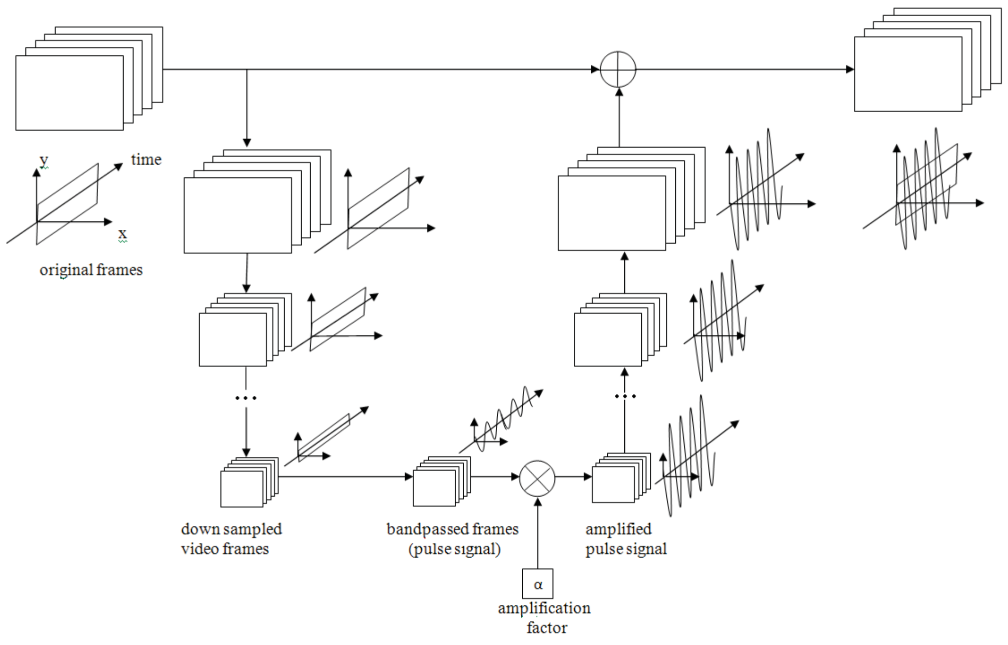

Fig 2. Eulerian Video Magnification algorithm in general.

2.2 Spatial processing of frames: noise filtering

This step is done for the following purposes:

- The linear amplification may not work for high pyramid levels with a lot of data, because they have different signal-to-noise ratios.
- For the next step – temporal filtering is better to reduce amplification on high levels to suppress artifacts.
- Increase speed of processing next steps due to the less data.

Building a Gaussian stack used OpenCV function named PyrDown, which downsamples image with 5x5 Gaussian filter. Now program contained 2 arrays of frames: original and blurred.

2.3 Temporal processing of frames sequence and getting pulse signal from it

This step is done with blurred frames. Main operation here is applying Fast Fourier transform to all sequences of frames for each color channel. First frames were reorganized from array of Mat to multidimensional array(in logical structure) of floating point data, where the lowest dimension was time and converted from RGB to YIQ color space. Then to perform FFT was used FFTW library. It is a very fast instrument of computing FFT. As there was a large number of FFT's and IFFTs to compute FFTW was a good choice. After that pulse rates were filtered from the spectrum by looking for frequencies correspond to bands where pulse can be(simply turning other frequencies to zero) and then performed IFFT. Bands for searching pulse frequency between are variables, and user can modify them. If these bands are modified program will compute pulse from next portion of frames (see fig 1) with new bands. Storing data not in Mats, but in array was necessary because computing FFT required not frames but sequences of pixel values of specific color channel along time, and Mat gives all pixels of each frame.

2.4 Amplification of the pulse signal and summing up it with the original frame sequence

This step consists of the following actions: multiplication the whole array of data with bandpassed pulse signal by amplification factor and resample bandpassed signal frames to original size. For summing up bandpassed signal with original frames last were reorganized from Mat to array and converted to YIQ color space as above. After that the sum was turned back to RGB and put in Mats.

2.5 Combine the result frames into a video

This step is as opposite to 3.1. An array of Mats with the help of FFMPEG functions was converted to a video stream.

III. Conclusion

To sum it up here are some positive and negative sights of the application. As for positive, it realizes new algorithm of pulse recognition. As it was tested the application can perform in real-time mode. Main causes of high performance of the application are as follows:

- Low computational complexity of EVM algorithm against other algorithms of pulse recognition by video (see [1], [2]).
- High speed of computing FFT with FFTW [3].
- Requirement of only 2-3 seconds of video stream to determine pulse, and as consequence general algorithm of application (Fig. 1), which model real-time working.

However some of the described positions of algorithm implementation lead to next application disadvantages:

- Three external libraries, used in the application have different data structures to work with, which leads to additional conversions of data between using some code from different libraries.
- The application doesn't seek for frequency bands by himself, and in some situations to get pulse it is necessary to move them to the sight of higher(if the processed pulse is too fast) or lower frequencies(opposite situation). Application provides interface for it.

References

[1] Hao-Yu Wu et all., "*Eulerian Video Magnification for Revealing Subtle Changes in the World,*" *ACM Trans. Graph.*, vol. 31, no 4, July 2012 art. 65. [Online]. Available: ACM Digital Library, doi:10.1145/2185520.2185561 2012.

[2] Hao-Yu Wu et all., "*Eulerian Video Magnification for Revealing Subtle Changes in the World Supplemental,*" MIT CSAIL, Nov. 2012. [Online]. Available: http://people.csail.mit.edu/mrub/vidmag/vidmag-supplemental.pdf [Accessed: 28 March 2013].

[3] M. Frigo, S.G. Johnson "*The design and implementation of FFTW3," Proc,* pp. 6-7. FFTW Nov. 25 2012. [Online]. Available: http://www.fftw.org/#documentation, [Accessed: 28 March 2012].