Computational Requirements for Video-Based Heart Rate Measurement Algorithms

Romir Kulshrestha[1]

Supervisor(s): J. Martinez[1]**, K. Rassels**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
January 26, 2025

Name of the student: Romir Kulshrestha
Final project course: CSE3000 Research Project
Thesis committee: J. Martinez, K. Rassels, C. Lofi

Computational Requirements for Video-Based Heart Rate Measurement Algorithms

Romir Kulshrestha
Delft University of Technology EEMCS
Delft, NL

***Abstract*—The measurement of the heart rate (HR) is of vital importance in modern medicine [1]. Advancements in medical technology have resulted in a myriad of techniques to measure and analyze these bio-signals, and the advent of telemedicine and the post-COVID-19 world has placed greater emphasis on contact-free measurement tools.**

Previous works have explored several methods of contact-free heart rate measurement. Remote Photo-Plethysmography (rPPG) [2] measures HR from RGB camera streams by detecting and analyzing the frequency of "micro-blushes" in the skin corresponding to pulse; this method can also be used to estimate SpO_2. Eulerian video magnification [3] instead attempts to detect subtle micro-movements caused by the pulse to measure HR as well as RR. Ballistocardiography tracks longitudinal movements of feature points to estimate HR [4]. However, there is little research on applying these (typically computationally-intensive) algorithms in the context of real-time, low-performance embedded systems.

This paper evaluates the computational requirements of algorithms used in extracting bio-signals using an RGB camera. It focuses on rPPG, a camera based method for extracting heart rate from video. Several rPPG implementations are tested on standard hardware and benchmarked on real-time performance and memory. The experiments were conducted on publicly available datasets. Additionally, region-of-interest selection algorithms are also compared. These results are of much use in developing embedded devices for remote monitoring of bio-signals and provide some insight into algorithms viable for use in real-time contexts.

***Index Terms*—photoplethysmography, Python, heart rate, embedded systems**

I. Introduction

Continuous monitoring of vital signs such as heart rate (HR), respiratory rate (RR), and blood oxygen saturation (SpO_2) is crucial for assessing an individual's health and well-being. These physiological parameters offer valuable insights into cardiovascular health, respiratory function, and overall physiological status [1], [5], [6]. The monitoring, detection, and analysis of these vital signs is crucial for timely medical intervention as well as the improvement of patient outcomes in a variety of settings, ranging from intensive care units (ICUs) to elderly care homes. However, conventional methods overwhelmingly rely on contact-based techniques that may cause discomfort or irritation in some patients or simply be too restrictive and impractical for long-term use [7]. This limitation underscores the need for non-contact measurement techniques that can provide accurate and continuous vital sign monitoring.

A. Prior Work

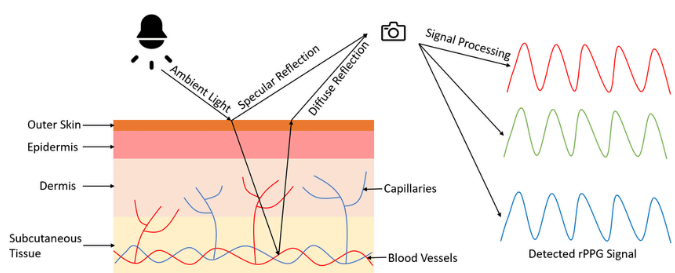

Fig. 1: Principle of remote photoplethysmography (rPPG) based on the dichromatic reflection model (DRM). Source: [8]

Modern imaging systems are now inexpensive, ubiquitous, and designed to be integrated into more complex systems. This provides an avenue for the development of low-cost, embedded systems that utilize an RGB camera to remotely monitor vital signals. A number of analysis methods have been developed involving a variation of techniques and approaches to extracting a signal from a video feed. Here, we summarize three well-known classes of techniques: remote photo-plethysmography (rPPG), ballistocardiography (BCG), and video magnification (VM).

a) *Remote photo-plethysmography:* Photo-plethysmography (PPG) exploits the relationship between blood volume changes and light absorption in the microvascular bed. The core idea can be understood as similar to the technique in widespread use in the form of pulse-oximeters: as the heart pumps blood throughout the body, it perfuses into the skin and causes subtle periodic changes in the color of the skin, detectable through visible light. Remote PPG techniques utilize this approach, analyzing subtle color variations in skin captured by cameras to extract physiological information (See Fig. 1). This is the approach the paper will focus on. [2]

b) *Video Magnification:* Video magnification techniques amplify subtle motions in video recordings, revealing physiological signals like pulse and respiration [3]. Blood pressure causes the skin to physically flex, revealing subtle changes that can be amplified and detected through motion-based video magnification.

c) *Ballistocardiography:* BCG extracts cardiac-induced head movements from video, tracking the longitudinal movements of feature points within a region of interest (ROI). Blood pressure changes over time as a result of pulse cause the head to react similar to an inverted pendulum, making subtle movements that can be tracked using a camera. [9]

B. Bio-Signal Monitoring on Embedded Systems

Much research has been done to determine the accuracy, reliability, and efficacy of these techniques in a wide range of scenarios [10]. However, there is little to no research at the present moment on the real-time computational cost of running these analysis methods, especially in the case of low-cost embedded devices that do not have access to high-performance computational power.

To fill this gap, we present a comparison of the computational requirements of existing remote monitoring techniques, and recommend strategies to enable the implementation of these techniques on low-performance hardware. With this work, we aim to answer the following research questions:

1) What are the computational requirements of algorithms used for video-based biometric signal extraction?
2) Which platforms, languages, and tools are the most suited to running these algorithms on embedded systems?

II. Evaluation Methods

This section describes the concepts and methods used to evaluate the computational requirements of algorithms used in the extraction of bio-signals from RGB camera streams. To do this, we implemented algorithms in Python and benchmarked them on real-time performance, accuracy, and memory usage while controlling for specific environmental factors. The experiments were conducted on publicly available datasets.

A. Ecosystems

Scientific computing thrives on robust ecosystems that provide the necessary tools and libraries for researchers and developers to tackle complex problems. While established languages like Python have dominated the field with their extensive libraries and user-friendly interfaces, newer languages like Julia and Rust are emerging as strong contenders, offering performance advantages and modern language features.

While Rust offers many advantages for embedded systems, some challenges hinder its widespread adoption in real-world applications. Firstly, the ecosystem is still relatively young, lacking the extensive range of mature and well-tested libraries found in C/C++. This can make it challenging to find readily available components for specific hardware or communication protocols, requiring developers to write more code from scratch or rely on less established libraries. Secondly, the learning curve for Rust can be steep, particularly for embedded developers accustomed to C/C++. This can increase development time and require additional training for teams transitioning to Rust. Lastly, debugging and tooling support for embedded Rust development, while improving, is not as comprehensive as that available for C/C++. This can make troubleshooting and optimization more challenging, especially for complex embedded systems.

Python is often preferred in many situations due to its ease of use, extensive ecosystem, and large community support. Its simple syntax and dynamic typing make it quicker to learn and write code, especially for beginners. The vast array of readily available libraries and frameworks (such as NumPy and OpenCV), particularly for data science and machine learning, significantly speeds up development time. The major downside is that since Python is an interpreted language, it is difficult to extract high performance. However, this is mitigated to a large extent by the fact that high-performance libraries are implemented in C/C++ and Rust (and occasionally FORTRAN).

Thus, this paper focuses on Python implementations.

B. Profiling

To evaluate the performance of different remote estimation algorithms, we employed a rigorous benchmarking methodology. This involved isolating the processing time by utilizing a consistent set of video frames across all implementations. Each algorithm was applied to the same frames, ensuring a controlled comparison of their computational efficiency. We measured execution time using the `timeit` and `cProfile` libraries in Python, which provided the total time taken to process a specified number of frames as well as time spent in specific functions. This approach allowed for a quantitative assessment of each algorithm's performance, enabling us to identify the most efficient solution for real-time video processing applications.

C. Used Datasets

For the purposes of this study, certain considerations were kept in mind while selecting datasets. Publicly available datasets were preferred where the following conditions were met (in no particular order):

- Face and upper body visible
- Ground truth data available (ECG, respiration amplitude, etc.)
- Low to no movement of the subject
- Stable lighting conditions
- Diverse range of skin-tones and ages

Accordingly, the UBFCrPPG Dataset 1 [11] was chosen. Fig. 2 shows an example of ground-truth data available in this dataset.

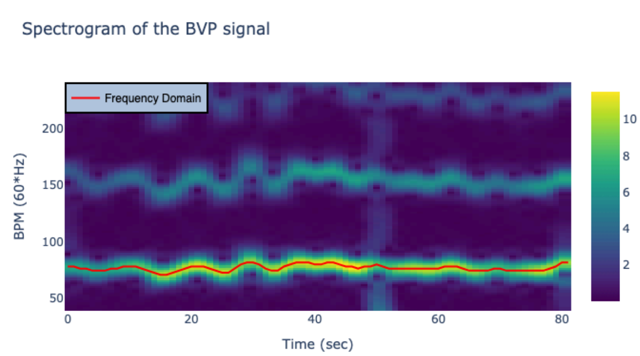

Fig. 2: Spectrogram of recorded ground truth from sample `11-gt` from the UBFCrPPG dataset.

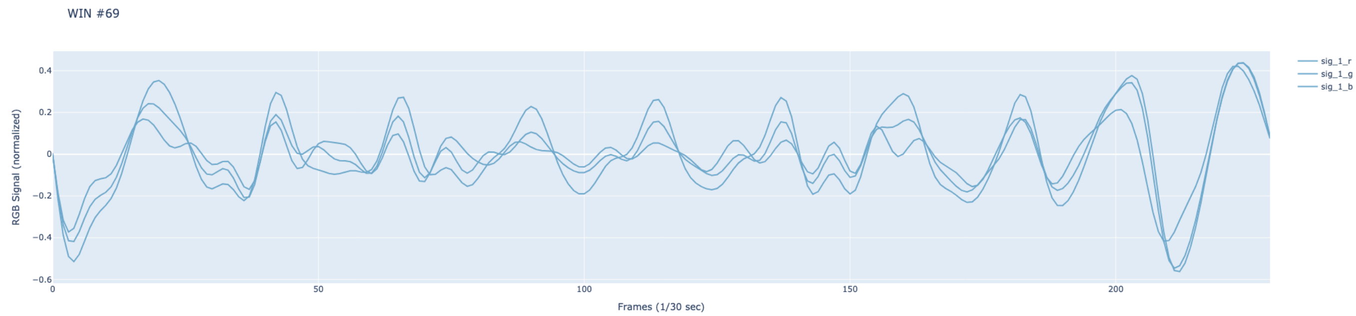

Fig. 3: Visualization of a window of an RGB signal. rPPG methods estimate HR by analyzing signals similar to the one shown.

D. *rPPG Variants*

Several techniques may be employed in the extraction of an rPPG signal from raw video data. Fig. 3 shows an example of an RGB signal that would be processed by rPPG.

This section explores and compares four prominent rPPG signal extraction variants used in this study: Plane-Orthogonal-to-Skin (POS), CHROM, Principal Component Analysis (PCA), and Blood Volume Pulse (PBV).

1) Plane-Orthogonal-to-Skin (POS)
The POS method capitalizes on the observation that skin-tone variations primarily lie within a 2D plane in the RGB color space. By projecting the RGB signals onto the plane orthogonal to this skin-tone plane, POS aims to isolate the pulsatile component, which is assumed to be perpendicular to skin-tone variations. This approach effectively separates the rPPG signal from motion-induced intensity changes, making it robust in scenarios with significant movement. [12]

$$S_{\text{POS}} = S - (S \cdot N)N \tag{1}$$

Where:

- S_{POS} is the rPPG signal extracted using POS.
- S is the original RGB signal (often represented as a 3D vector).
- N is the normal vector to the skin-tone plane.

2) CHROM
The CHROM method leverages the chrominance signals, which are derived from the RGB color space. It exploits the fact that blood volume changes predominantly affect the chrominance components while having minimal impact on luminance. CHROM utilizes a specific combination of chrominance signals, typically normalized by the green channel, to extract the rPPG signal. This method offers computational efficiency and has shown good performance in controlled environments. [13]

$$\begin{aligned} X &= \frac{R}{G} \\ Y &= \frac{B}{G} \end{aligned} \tag{2}$$

Where R, G, and B represent the red, green, and blue channels, respectively. Then, the rPPG signal can be expressed as

$$S_{\text{CHROM}} = aX + bY \tag{3}$$

Where a and b are empirically determined weights. A common choice is

$$S_{\text{CHROM}} = 3X - 2Y \tag{4}$$

.

3) Principal Component Analysis (PCA)
PCA is a widely used dimensionality reduction technique that identifies the principal components, which capture the maximum variance in the data. In the context of rPPG, PCA is applied to the RGB signals to extract the component that exhibits the strongest pulsatile signal. This component is often found to be associated with subtle color changes related to blood flow. PCA's effectiveness stems from its ability to identify the most relevant information in the data, even in the presence of noise. [14]

$$C = V \Lambda V^T \tag{5}$$

Where:

- C is the covariance matrix of the RGB signals.
- V is the matrix of eigenvectors.
- Λ is the diagonal matrix of eigenvalues.

Then,

$$S_{\text{PCA}} = S \cdot V_i \tag{6}$$

Where V_i is the eigenvector corresponding to the largest eigenvalue, representing the principal component with the most variance.

4) Blood Volume Pulse (PBV)
The PBV method is based on a physiological model that describes the relationship between blood volume changes and the reflected light from the skin. It utilizes a specific linear combination of RGB signals, weighted according to the absorption spectra of blood and melanin, to estimate the blood volume pulse. This model-based approach provides a physiological basis for rPPG signal extraction and has shown promising results in various applications. [15]

$$S_{\text{PBV}} = \alpha R + \beta G + \gamma B \tag{7}$$

Where α, β, and γ are weights derived from the optical properties of skin and blood. These weights are typically determined through empirical studies or based on a specific skin model (e.g., varying melanin concentrations).

E. *Noise Sources*

A number of factors can drastically impact the performance and accuracy of video-based HR estimation. Factors such as head movement or facial expressions or changes in lighting conditions can result in noise being introduced into the signal [16]. In order to mitigate this, bandpass filtering is used on both the raw and processed rPPG signals before estimating HR.

F. Region of Interest (ROI)

When performing heavy image processing in real-time on video streams, it is crucial to reduce the considered area of a frame to avoid spending unnecessary time on unusable portions of a frame. Thus, the ROI is identified and segmented out early on in the process. See Fig. 4.

There are a variety of methods of determining the ROI. Several remote estimation algorithms prefer to use a signal generated from the face of the subject [9], [17], [18], and others prefer to mask using the skin color [19]. Sometimes, the ROI is determined by hand [20]. In all cases the result is a mask containing (ideally) only those parts of a frame that will be analyzed in later stages.

III. Results

Several experiments were run to analyze and improve remote estimation methods. This section details these experiments and their results. A Raspberry Pi 5 and a 2024 M4 Mac Mini were used to run all experiments. The implementations were created using OpenCV, Numba, SciPy, NumPy, MediaPipe, and some plots were created using MatPlotLib.

A. ROI Selection

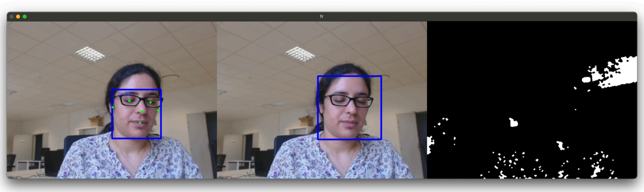

Fig. 4: ROI determined by the three tested methods. From left: Google MediaPipe (BlazeFace) [21], OpenCV Cascade Classifier (Viola-Jones) [22], Skin segmentation [19]. Note that in this sample the skin segmentation method has entirely failed to identify a usable ROI.

Determining and masking out the ROI can be a computationally heavy process. Three common methods—MediaPipe face detection [21], OpenCV Cascade Classifiers [22] and skin segmentation [19] — were evaluated on test hardware; the first is a neural network based face detection system, the second is an algorithmic face detection system, and the last is a color-based skin detection system. The results are shown in Fig. 5.

The neural network based system has vastly outperformed the other methods by orders of magnitude even without access to a GPU. However, it still takes considerable time to run: an average iteration duration of 9ms on the Raspberry Pi 5. Note that all of these methods strictly perform worse than manual or predetermined ROI selection, which takes zero time per iteration.

In situations where the subject's complexion is of a similar color to the background, the skin segmentation algorithm is unable to separate the subject from the background and therefore does not produce a usable ROI. See Fig. 4 for an example of this. This led to the algorithm not producing an ROI in 37.5% of the samples in the dataset.

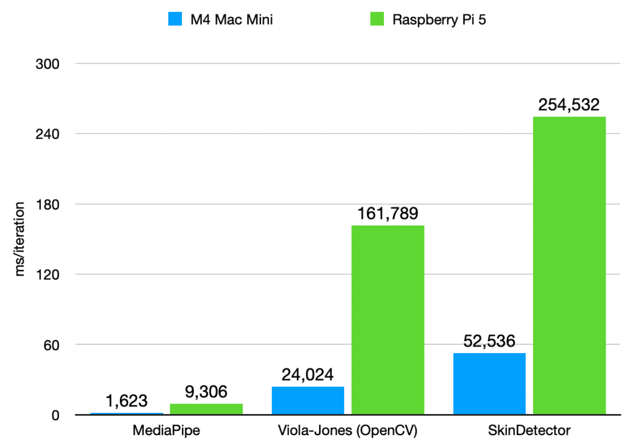

Fig. 5: Benchmarked average time per frame for the algorithms on test hardware expressed in milliseconds/iteration. Note that the comma in each data point is the European decimal point.

B. Runtime

This section compares the runtime performance of several prominent rPPG algorithms, examining their speed and resource demands. Understanding these performance characteristics is crucial for selecting the most suitable algorithm for real-time applications and resource-constrained environments.

Fig. 6 shows a breakdown of time spent on each processing step. On average, ROI selection takes up 96% of the processing time.

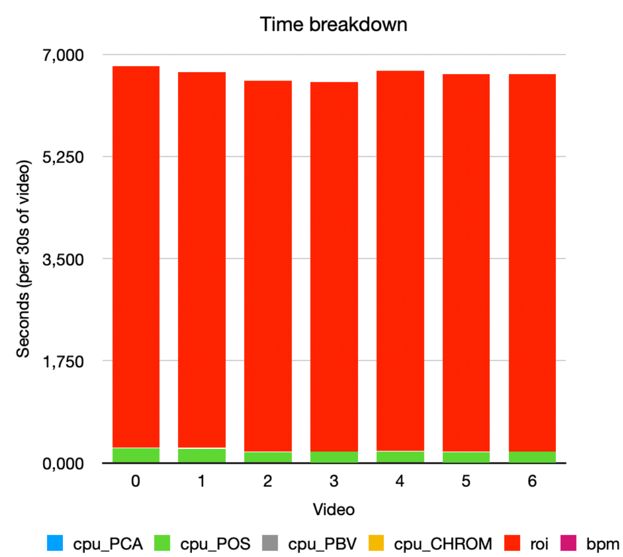

Fig. 6: Breakdown of time spent per 30 seconds of video on various processing steps across several videos in the dataset. Here, the ROI selection algorithm used is MediaPipe.

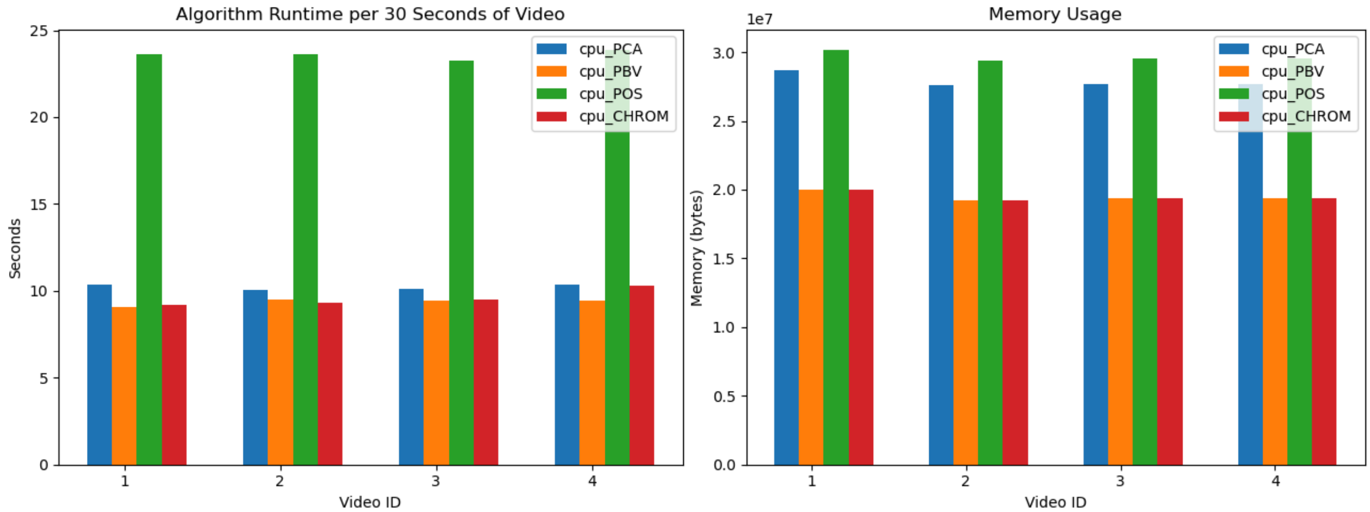

Fig. 7: Runtime and memory usage per 30 seconds of video for each of the studied algorithms.

Fig. 7 shows the comparative runtime and peak memory usage of PCA, PBV, POS and CHROM (discussed in Section II.D) on the UBFCrPPG Dataset when run on the M4 Mac Mini.

C. Accuracy

Fig. 8 shows the accuracy of each studied algorithm using Root Mean Squared Error (RMSE). POS shows the smallest mean and standard deviation (refer to Table I), indicating relatively higher reliability and accuracy.

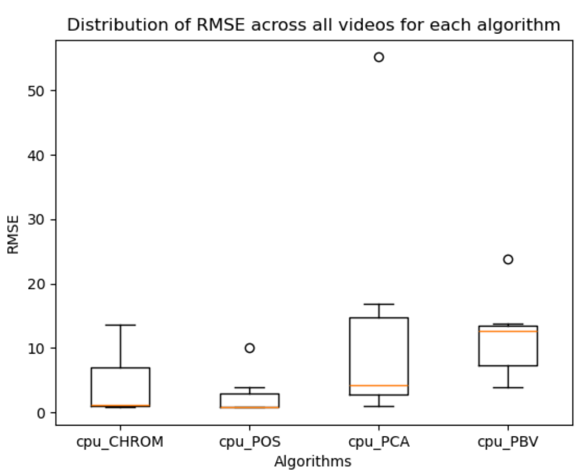

Fig. 8: Distribution of RMSE across all videos in the dataset for each algorithm.

TABLE I: Mean and Standard Deviation of RMSE Scores

Algorithm	Mean	Standard Deviation
CHROM	4.509	4.930
POS	**2.763**	**3.175**
PCA	13.606	17.840
PBV	11.678	6.250

IV. Discussion

1) *Skin segmentation is unreliable*: Skin segmentation did not produce a usable ROI in 37.5% of the samples.
2) *MediaPipe offers the lowest benchmarked average time*: per frame of 9ms on a Raspberry Pi 5 (in comparison to 161ms and 254ms respectively taken by OpenCV and skin segmentation).
3) *ROI selection is a significant bottleneck*: One key finding is that determining the Region of Interest (ROI) is critical to managing the computational load which in turn impacts the speed and accuracy of the final result. All the three methods offered some advantage but still pose significant challenges when considering real-time systems. Even when using MediaPipe, this step accounts for 96% of the computational load on average. In many cases, manual ROI selection may be preferable to avoid this cost.
4) *CHROM and POS may be good candidates for further analysis*: POS allocates the most memory and runs for longer than all of the other studied algorithms. However, it also offers relatively higher accuracy and reliability. CHROM offers much lower memory usage and runtime for only slightly lower accuracy and reliability.
5) *Python shows promise*: as a candidate for real time embedded devices due to rich scientific ecosystem and decades of optimization work in computing libraries.

V. Conclusion

This research provides insights into the challenges of deploying real-time contact-free HR measurement systems. By studying the computational requirements of various estimation methods, it provides direction on where future research might be concentrated.

Despite advances in contact-free estimation methods, achieving real-time performance on real-time embedded systems remains a challenge. ROI selection poses a significant bottleneck—therefore, optimizing ROI selection is a good first step to implementing contact-free HR measurement on embedded systems.

CHROM appears to offer a good balance of low computation cost and accuracy. POS could be a viable alternative as well. Future research may explore optimizing these algorithms to bring down computational cost while improving accuracy and reliability.

Future studies should expand the scope of algorithms and platforms considered, as this short study is limited to mainly Python-based implementations and a limited set of techniques. Additional work is required to test implementations in live scenarios, subject to unpredictable and dynamic real-world environments.

Additionally, any method chosen ultimately in the future should also address concerns related to privacy, and also country specific legislation limiting public usage.

VI. Ethical Considerations

Advancements in contact-free devices continue to test the boundaries of ethics and privacy especially in the area of healthcare where there are plausible patient data extraction opportunities without the provider or the patient realizing that such a breach may have occurred. This study has been carried out in accordance with the Netherlands Code of Conduct for Research Integrity [23].

A. Data Sensitivity

Due to the sensitive nature of personal medical information, no data was collected during this study; only existing publicly-available datasets were used and handled in accordance with best practices and regulations. A data management plan was created with the guidance of the TU Delft Data Steward and was adhered to through the course of this study.

B. Reproducibility

A copy of the source code used to create and run the experiments in this study will be made available online to allow for reproduction of all results.

Acknowledgements

I would like to acknowledge the contribution of all the people without whom this project would not have been possible. First, I would like to thank my parents and family for being a constant source of encouragement and motivation.

I would like to thank my Supervisor, Kianoush Rassels, and my Responsible Professor, Jorge Martinez, for their support and guidance throughout this study.

Last but not the least, I would like to thank my peers, Demetra Carata-Dejoianu and Vladimir Pechi, for their constant support to discuss various topics and ideas and for all the hours spent debating the direction of this project.

References

[1] C. Bruser, C. H. Antink, T. Wartzek, M. Walter, and S. Leonhardt, "Ambient and Unobtrusive Cardiorespiratory Monitoring Techniques," *IEEE Reviews in Biomedical Engineering*, vol. 8, pp. 30–43, 2015, doi: 10.1109/RBME.2015.2414661.

[2] W. Verkruysse, L. O. Svaasand, and J. S. Nelson, "Remote plethysmographic imaging using ambient light.," *Optics Express, Vol. 16, Issue 26, pp. 21434-21445*, vol. 16, no. 26, pp. 21434–21445, 2008, doi: 10.1364/OE.16.021434.

[3] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman, "Eulerian video magnification for revealing subtle changes in the world," *ACM Transactions on Graphics*, vol. 31, no. 4, pp. 1–8, 2012, doi: 10.1145/2185520.2185561.

[4] O. T. Inan, "Recent advances in cardiovascular monitoring using ballistocardiography," in *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, 2012. doi: 10.1109/EMBC.2012.6347125.

[5] M. Hassan *et al.*, "Heart rate estimation using facial video: A review," *Biomedical Signal Processing and Control*, vol. 38, pp. 346–360, 2017, doi: 10.1016/j.bspc.2017.07.004.

[6] H. H. J. van Noort, F. L. Becking-Verhaar, W. Bahlman-van Ooijen, M. Pel, H. van Goor, and G. Huisman-de Waal, "Three Years of Continuous Vital Signs Monitoring on the General Surgical Ward: Is It Sustainable? A Qualitative Study," *Journal of Clinical Medicine*, vol. 13, no. 2, p. 439, 2024, doi: 10.3390/jcm13020439.

[7] Y. J. Choo, G. W. Lee, J. S. Moon, and M. C. Chang, "Noncontact Sensors for Vital Signs Measurement: A Narrative Review," *Medical Science Monitor*, vol. 30, 2024, doi: 10.12659/MSM.944913.

[8] C.-H. Cheng, K.-L. Wong, J.-W. Chin, T.-T. Chan, and R. H. Y. So, "Deep Learning Methods for Remote Heart Rate Measurement: A Review and Future Research Agenda," *Sensors*, vol. 21, no. 18, p. 6296, 2021, doi: 10.3390/s21186296.

[9] G. Balakrishnan, F. Durand, and J. Guttag, "Detecting pulse from head motions in video," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 3430–3437, 2013, doi: 10.1109/CVPR.2013.440.

[10] P. V. Rouast, M. T. Adam, R. Chiong, D. Cornforth, and E. Lux, "Remote heart rate measurement using low-cost RGB face video: a technical literature review," *Frontiers of Computer Science*, vol. 12, no. 5, pp. 858–872, 2018, doi: 10.1007/S11704-016-6243-6.

[11] S. Bobbia, R. Macwan, Y. Benezeth, A. Mansouri, and J. Dubois, "Unsupervised skin tissue segmentation for remote photoplethysmography," *Pattern Recognition Letters*, vol. 124, pp. 82–90, 2019, doi: 10.1016/j.patrec.2017.10.017.

[12] W. Wang, A. C. Den Brinker, S. Stuijk, and G. De Haan, "Algorithmic Principles of Remote PPG," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 7, pp. 1479–1491, 2017, doi: 10.1109/TBME.2016.2609282.

[13] G. de Haan and V. Jeanne, "Robust Pulse Rate From Chrominance-Based rPPG," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 10, pp. 2878–2886, 2013, doi: 10.1109/TBME.2013.2266196.

[14] M. Lewandowska and others, *Measuring Pulse Rate with a Webcam-a Non-contact Method for Evaluating Cardiac Activity*. 2011, pp. 405–410.

[15] G. de Haan and A. van Leest, "Improved motion robustness of remote-PPG by using the blood volume pulse signature," *Physiological Measurement*, vol. 35, no. 9, pp. 1913–1926, 2014, doi: 10.1088/0967-3334/35/9/1913.

[16] P. Pirzada, A. Wilde, G. Doherty, and D. Harris-Birtill, "Remote Photoplethysmography (rPPG): A State-of-the-Art Review." [Online]. Available: http://medrxiv.org/lookup/doi/10.1101/2023.10.12.23296882

[17] A. K. Maity, J. Wang, A. Sabharwal, and S. K. Nayar, "RobustPPG: camera-based robust heart rate estimation using motion cancellation," *Biomedical Optics Express*, vol. 13, no. 10, p. 5447, 2022, doi: 10.1364/BOE.465143.

[18] W. Wang, S. Stuijk, and G. de Haan, "A Novel Algorithm for Remote Photoplethysmography: Spatial Subspace Rotation," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 9, pp. 1974–1984, 2016, doi: 10.1109/TBME.2015.2508602.

[19] F. Gasparini and R. Schettini, "Skin segmentation using multiple thresholding," S. Santini, R. Schettini, and T. Gevers, Eds., 2006, p. 60610. doi: 10.1117/12.647446.

[20] J. C. Cobos-Torres, M. Abderrahim, and J. Martínez-Orgado, "Non-Contact, Simple Neonatal Monitoring by Photoplethysmography," *Sensors 2018, Vol. 18, Page 4362*, vol. 18, no. 12, p. 4362, 2018, doi: 10.3390/S18124362.

[21] V. Bazarevsky, Y. Kartynnik, A. Vakunov, K. Raveendran, and M. Grundmann, "BlazeFace: Sub-millisecond Neural Face Detection on

Mobile GPUs," 2019, [Online]. Available: http://arxiv.org/abs/1907.05047
[22] P. Viola and M. J. Jones, "Robust Real-Time Face Detection," *International Journal of Computer Vision*, vol. 57, no. 2, pp. 137–154, 2004, doi: 10.1023/B:VISI.0000013087.49260.fb.
[23] "Netherlands Code of Conduct for Research Integrity," 2018.