README for rPPG Heart Rate Estimation MATLAB Project

Anna Simeone

July 4, 2025

Project Overview

This MATLAB project provides a comprehensive pipeline for remote photoplethysmography (rPPG) heart rate (HR) estimation and evaluation. It compares multiple rPPG extraction methods against a ground truth blood volume pulse (BVP) reference signal across multiple patients.

The project consists of scripts and functions for data loading, preprocessing, rPPG signal extraction, HR estimation, spectral quality analysis, statistical comparison, and result visualization.

Folder and File Structure

```
BSP_Final/
 routine/
    align_signals.m
    analyze_agreement.m
    analyze_spectral_quality.m
    compute_hr_fft.m
    compute_rppg_methods.m
    extract_ica_pca.m
    extract_reference_hr.m
    load_patient_data.m
    minmax.m
    postprocess_rppg.m
    preprocess_rgb.m
    run_pipeline.m
 data/
    RGB/
       rgb_raw_11_p1.csv
       rgb_raw_12_p1.csv
       rgb_raw_13_p1.csv
```

```
      ... (other RGB data files)
   Timestamp_p1/
      video_11_28-06-2022_12-41_time.csv
      video_12_28-06-2022_17-46_time.csv
      video_13_28-06-2022_18-40_time.csv
      ... (other timestamp files)
   bvp_p1.csv
   green_corrected_p1.csv
results/
   overall_results/
   11/
   12/
   13/
   ... (results folders per patient)
main.m
results.m
window_optimization.m
```

Data Files

- **bvp_p1.csv:** Reference BVP signal (ground truth for HR estimation). Columns correspond to patients.
- **green_corrected_p1.csv:** Preprocessed green channel values extracted from videos.
- **RGB raw data files:** Located in `data/RGB/`, named as `rgb_raw_<patientID>_p1.csv`, containing raw Red, Green, Blue channel values.
- **Timestamp files:** Located in `data/Timestamp_p1/`, named as `video_<patientID>_*_time.csv`, providing timestamps per video frame for resampling.

Routine Functions

The `routine/` folder contains all auxiliary functions and scripts used throughout the analysis pipeline. The core processing is managed by `run_pipeline.m`, which is invoked by `main.m` for each patient.

Brief descriptions of main functions:

- `load_patient_data.m`: Loads RGB signals, reference BVP, and timestamps for a patient.
- `preprocess_rgb.m`: Corrects jumps, interpolates to uniform sampling, low-pass filters RGB channels.

- `compute_rppg_methods.m`: Computes rPPG signals using multiple methods with normalization and spectral filtering.
- `extract_ica_pca.m`: Applies ICA and PCA on sliding windows to extract pulse-related components.
- `postprocess_rppg.m`: Bandpass filters signals and optionally plots them.
- `align_signals.m`: Cross-correlates and aligns rPPG and reference signals.
- `minmax.m`: Min-max normalization utility.
- `extract_reference_hr.m`: Detects peaks and estimates HR from BVP.
- `compute_hr_fft.m`: Estimates HR via windowed FFT of rPPG signals.
- `analyze_spectral_quality.m`: Computes spectral SNR of rPPG methods and of BVP.
- `analyze_agreement.m`: Performs Bland-Altman and agreement analysis between HR estimates and BVP reference.
- `run_pipeline.m`: Executes the complete rPPG processing pipeline for a single patient, taking as input a configuration structure that specifies the patient ID, sampling frequency, and relevant folder paths. The function is designed to reproduce the entire signal processing flow, from raw data loading to statistical validation. The pipeline begins by loading the signals. RGB channels are preprocessed an then multiple rPPG methods are then applied. Once the rPPG signals are extracted, they are bandpass filtered, and aligned with the reference BVP signal via cross-correlation using the $S_{X_s - \alpha Y_s}$ signal as alignment reference. All rPPG signals are trimmed to remove edge artifacts and are shifted according to the computed lag to ensure temporal synchronization with BVP. After alignment, the pipeline estimates the reference heart rate by detecting peaks in the BVP signal, computing RR intervals. Then, each rPPG method is used to estimate the heart rate via FFT peak detection in sliding windows. Following the HR estimation, the pipeline performs a spectral quality analysis by computing the signal-to-noise ratio (SNR) for each method. Finally, a statistical agreement analysis is conducted comparing the rPPG-derived HR signals against the BVP-derived reference, including Bland–Altman plots, correlation analysis, and error metrics. The function returns a structured output containing all aligned signals, estimated and reference HR curves, SNR values, and agreement statistics. It also supports optional plotting and saving of figures to a specified results directory.

Main Scripts

`main.m`

Primary entry point. For each patient, it sets up paths, configurations (patient ID, sampling frequency, plotting flags, etc.), creates output folders, and calls `run_pipeline.m` to process

data and compute metrics such as SNR.

It displays progress messages for transparency and reproducibility.

`results.m`

The `result.m` script performs the global evaluation of the rPPG heart rate estimation methods by loading saved patient-specific results from the `results/` subdirectories. It automatically scans the folder structure, loads the relevant `.mat` files, and aggregates the data for statistical analysis and visualization.

The script is fully automatic and provides an overview of method performance across all subjects.

Specifically, it performs the following operations:

- **Dynamic loading of patient results:** It searches for patient folders named with numeric IDs under `results/`, then loads the corresponding `.mat` structures containing the outputs of `main.m`.
- **Mean SNR computation:** It extracts signal-to-noise ratio (SNR) values for each rPPG method across all patients, computes the average SNR per method, and saves a summary bar chart under `results/overall_results/`.
- **Agreement metrics computation:** For each method, it calculates:
  - Pearson correlation coefficient between estimated and reference HR;
  - Regression slope between the two signals;
  - Standard deviation and RMSE of the difference signal.
- **Bland–Altman analysis:** For each method, it generates and saves a Bland–Altman plot showing bias and limits of agreement (LoA). It also computes the percentage of instances within ± 5 BPM.
- **Outlier rate analysis:** It flags windows outside the Bland–Altman LoA for each patient and method, computes the outlier rate per subject and method, and saves a corresponding heatmap.

All outputs are saved in the folder `results/overall_results/`.

`window_optimization.m`

The `window_optimization.m` script evaluates the effect of the spectral window length on the accuracy of heart rate (HR) estimation from rPPG signals.

The core of the script consists in iteratively varying the length of the window used for normalization of the parameter α, ranging from 1.6 to 25.6 seconds. For each window size, heart rate is estimated using FFT peak detection within the physiological frequency band (40–240 BPM), and then compared to the reference. The absolute error between the estimated and reference HR is computed over all windows.

For each patient, a curve showing the error trend as a function of window duration is plotted, and the optimal window length is reported.

Usage Instructions

1. Ensure the folder structure and data files described above are correctly organized.
2. Open `main.m` in MATLAB.
3. Run `main.m` to process the data and save results. For each patient, a dedicated folder is created in the `results/` directory. This folder contains a `.mat` structure that stores all relevant information for that patient, including: heart rate estimates for each method, rPPG signals extracted using different algorithms, ground truth BVP signal and signal-to-noise ratio computed for each method and window.
4. Run `results.m` to analyze and visualize the results across patients.
5. Optionally, run `window_optimization.m` to refine analysis window size.

Customization Options

- **Sampling frequency:** Default is 115 Hz; can be changed in configuration within `main.m` according to the actual frequency of the data.
- **Filter parameters:** Cutoff frequencies and filter order can be adjusted in preprocessing routines.
- **Window lengths:** Sliding window sizes for normalization and FFT can be customized.
- **Plotting:** Enable/disable plots and figure saving in configuration parameters.

Troubleshooting

- **File or folder not found:** Verify file names and folder paths match exactly.
- **Missing patient data:** Confirm that data for selected patients exist in `data/RGB` and `data/Timestamp_p1`.
- **Unexpected results or noisy signals:** Inspect intermediate plots and consider adjusting filter parameters or window lengths.
- **MATLAB warnings/errors:** Check for missing dependencies or required MATLAB toolboxes.