



**Ecole Polytechnique de l'Université de Tours**

Département Informatique

64 avenue Jean Portalis

37200 Tours, France

Tél. +33 (0)2 47 36 14 14

[polytech.univ-tours.fr](http://polytech.univ-tours.fr)

**Projet Recherche & Développement**

**2020-2021**

# **Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos**



**POLYTECH<sup>®</sup>**  
**TOURS**

**Entreprise**

**Polytech**



**Tuteur entreprise**

**Donatello CONTE**

**Étudiant**

**Florian GIGOT (DI5)**

**Tuteur académique**

**Donatello CONTE**

# Liste des intervenants

## Entreprise

Polytech  
64 avenue Jean Portalis  
37200 Tours, France  
[polytech.univ-tours.fr](http://polytech.univ-tours.fr)



Nom	Email	Qualité
Florian GIGOT	<a href="mailto:florian.gigot@etu.univ-tours.fr">florian.gigot@etu.univ-tours.fr</a>	Étudiant DI5
Donatello CONTE	<a href="mailto:donatello.conte@univ-tours.fr">donatello.conte@univ-tours.fr</a>	Tuteur académique, Département Informatique
Donatello CONTE	<a href="mailto:donatello.conte@univ-tours.fr">donatello.conte@univ-tours.fr</a>	Tuteur entreprise



# Avertissement

Ce document a été rédigé par Florian GIGOT susnommé l'auteur.

L'entreprise Polytech est représentée par Donatello CONTE susnommé le tuteur entreprise.

L'Ecole Polytechnique de l'Université de Tours est représentée par Donatello CONTE susnommé le tuteur académique.

Par l'utilisation de ce modèle de document, l'ensemble des intervenants du projet acceptent les conditions définies ci-après.

L'auteur reconnaît assumer l'entière responsabilité du contenu du document ainsi que toutes suites judiciaires qui pourraient en découler du fait du non respect des lois ou des droits d'auteur.

L'auteur atteste que les propos du document sont sincères et assume l'entière responsabilité de la véracité des propos.

L'auteur atteste ne pas s'approprier le travail d'autrui et que le document ne contient aucun plagiat.

L'auteur atteste que le document ne contient aucun propos diffamatoire ou condamnable devant la loi.

L'auteur reconnaît qu'il ne peut diffuser ce document en partie ou en intégralité sous quelque forme que ce soit sans l'accord préalable du tuteur académique et de l'entreprise.

L'auteur autorise l'école polytechnique de l'université François Rabelais de Tours à diffuser tout ou partie de ce document, sous quelque forme que ce soit, y compris après transformation en citant la source. Cette diffusion devra se faire gracieusement et être accompagnée du présent avertissement.



## Pour citer ce document

Florian GIGOT, *Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos*, Projet Recherche & Développement, Ecole Polytechnique de l'Université de Tours, Tours, France, 2020-2021.

```
@mastersthesis{
  author={GIGOT, Florian},
  title={Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse
    de vidéos: },
  type={Projet Recherche \& Développement},
  school={Ecole Polytechnique de l'Université de Tours},
  address={Tours, France},
  year={2020-2021}
}
```

# Table des matières

Liste des intervenants	<b>a</b>
Avertissement	<b>b</b>
Pour citer ce document	<b>c</b>
Table des matières	<b>i</b>
Table des figures	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1 Acteurs, enjeux et contexte .....	1
2 Objectifs .....	1
3 Hypothèses .....	2
4 Bases méthodologiques.....	2
<b>2 Description générale</b>	<b>3</b>
1 Environnement du projet .....	3
2 Caractéristiques des utilisateurs .....	3
3 Fonctionnalités du système .....	3
4 Structure générale du système.....	4
Architecture du code .....	4
Description générale du processus de traitement du Framework .....	5
<b>3 Etude de l'existant</b>	<b>6</b>
1 Prise en main du code existant .....	6
1.1 Installation de pyVHR.....	6
1.2 Démonstrations .....	6

1.2.1	Détection et Extraction d'un visage .....	6
1.2.2	Régions d'intérêt (Process ROIs) .....	7
1.2.3	Gestion des ensembles de données.....	8
1.2.4	Application de méthodes rPPG .....	10
1.2.5	Analyse Statistique.....	11
<b>4</b>	<b>Etat de l'art / Veille technologique</b> .....	<b>12</b>
1	Les méthodes rPPGs .....	12
2	Les méthodes traditionnelles .....	13
2.1	Avant-propos .....	13
2.2	Description des méthodes .....	14
2.2.1	Méthode GREEN .....	14
2.2.2	Méthode ICA .....	14
2.2.3	Méthode PCA .....	14
2.2.4	Méthode CHROME.....	15
2.2.5	Méthode SSR .....	15
2.2.6	Méthode LGI.....	16
2.2.7	Méthode PBV .....	16
2.2.8	Méthode POS.....	17
3	Les nouvelles méthodes .....	17
3.1	Avant-propos .....	17
3.2	Méthodes basées sur un réseau RNN.....	18
3.3	Méthodes basées sur un réseau 3D CNN .....	18
3.4	Méthodes hybrides basées sur 3DCNN-RNN .....	18
4	Récapitulatif.....	19
<b>5</b>	<b>Analyse et conception</b> .....	<b>20</b>
1	Analyse .....	20
1.1	La méthode à développer .....	20
1.1.1	Hypothèses.....	20
1.1.2	Description du modèle 3D CNN.....	20
1.1.3	Entraînement du modèle 3D CNN .....	21
1.1.4	Validation du modèle 3D CNN .....	22
1.1.5	Implémentation d'un modèle dans pyVHR.....	22
1.2	Les notebooks à développer .....	22
1.2.1	Notebook 1 : Prise en main .....	22
1.2.2	Notebook 2 : Ajouter une méthode .....	23
2	Modélisation proposée.....	23

<b>6</b>	<b>Bilan et conclusion</b>	<b>25</b>
1	Bilan du semestre 9 .....	25
2	Planning du semestre 10 .....	25
	<b>Annexes</b>	<b>27</b>
<b>A</b>	<b>Planification, gestion de projet</b>	<b>28</b>
1	Evolution du projet .....	28
1.1	Diagramme de GANTT planifié .....	28
1.1.1	Planification pour le semestre 9 .....	28
1.1.2	Planification pour le semestre 10 .....	28
2	Description des tâches.....	29
2.1	Semestre 9.....	29
	Tâche 1 : Découverte du projet.....	29
	Tâche 2 : Étude de l'existant .....	30
	Tâche 3 : Etat de l'art / Veille .....	30
	Tâche 4 : Cahier des spécifications.....	30
	Tâche 5 : Planification.....	30
	Tâche 6 : Analyse et conception .....	30
	Tâche 7 : Rédaction du rapport S9 .....	30
	Tâche 8 : Préparation de la soutenance du S9.....	30
2.2	Semestre 10 .....	31
	Tâche 9 : Développement et entraînement du modèle de la nouvelle méthode.....	31
	Tâche 10 : Implémentation de la nouvelle méthode dans le Framework	31
	Tâche 11 : Elaboration du guide d'implémentation d'une nouvelle méthode.....	31
	Tâche 12 : Elaboration de la démonstration générale .....	31
	Tâche 13 : Rapport final.....	31
	Tâche 14 : Soutenance finale.....	31
<b>B</b>	<b>Cahier de Specification</b>	<b>33</b>
1	spécifications Fonctionnelles.....	33
1.1	Fonctionnalités à développer .....	33
	Fonction 1 : Conception et Implémentation d'une nouvelle méthode rPPG .	33
	Présentation de la fonction 1 : .....	33
	Description de la fonction 1 : .....	33
	Description précisée de la fonction 1 : .....	33
	Fonction 2 : Intégration d'une nouvelle méthode rPPG dans le Framework.	35
	Présentation de la fonction 2 : .....	35

	Description de la fonction 2 : .....	35
	Description précisée de la fonction 2 : .....	35
1.2	Tâches à réaliser .....	36
	Tâche 1 : Implémentation d'un notebook de démonstration pour une prise en main générale du Framework.....	36
	Présentation de la tâche 1 : .....	36
	Description de la tâche 1 : .....	36
	Description précisée de la tâche 1 : .....	36
	Tâche 2 : Implémentation et rédaction d'un guide ("notebook") pour l'in- tégration d'une nouvelle méthode rPPG par les futurs utilisateurs. ....	37
	Présentation de la tâche 2 : .....	37
	Description de la tâche 2 : .....	37
	Description précisée de la tâche 2 : .....	37
	Tâche 3 : Améliorations et Documentation générale. ....	38
	Présentation de la tâche 3 : .....	38
	Description de la tâche 3 : .....	38
2	Spécifications non fonctionnelles .....	39
2.1	Contraintes de développement et conception .....	39
2.2	Contraintes de fonctionnement et d'exploitation.....	39
	Performances .....	39
	Modes de fonctionnement .....	39
	Capacités.....	40
	Contrôlabilité .....	40
	Sécurité .....	40
<b>C</b>	<b>Cahier du développeur</b> .....	<b>41</b>
1	Rappel du diagramme de classe .....	41
2	Description détaillée du diagramme de classe .....	42
	Package Dataset.....	42
	Package Method.....	42
	Package Analysis.....	42
	Package Signals.....	42
	Package Utils .....	42
	Package Stats.....	43
3	Description de la structure du projet.....	43
	<b>Bibliographie</b> .....	<b>44</b>



# Table des figures

## 2 Description générale

2.1 UML - Cas d'utilisation du Framework pyVHR.....	4
2.2 UML - Architecture Existante.....	4
2.3 Processus général du traitement et de l'analyse vidéo par pyVHR .....	5

## 3 Etude de l'existant

3.1 Vidéo avant extraction du visage.....	7
3.2 Vidéo après extraction du visage.....	7
3.3 ROI - Région rectangulaire .....	8
3.4 ROI - Région Standart (front, joues, nez).....	8
3.5 ROI - Région avec de la peau (avec seuil adaptatif) .....	8
3.6 ROI - Région avec de la peau (avec seuil fixe) .....	8
3.7 Prise en main de l'objet "Dataset".....	9
3.8 Visualisation et traitement de la vérité terrain d'une video du dataset.....	9
3.9 Calcul et visualisation du signal BPM à partir de la vérité terrain.....	10
3.10 Comparaison des résultats de 2 méthodes rPPGs sur la même vidéo .....	11
3.11 Exemple de représentation statistique d'une comparaison de plusieurs méthodes ...	11

## 4 Etat de l'art / Veille technologique

4.1 Phénomènes d'absorption et de réflexion par la peau humaine [5].....	12
4.2 Les méthodes traditionnelles de remote-PPG .....	13

## 5 Analyse et conception

5.1 Architecture du modèle 3D CNN .....	21
---	----

5.2	Processus de génération de vidéo artificielle .....	21
5.3	Intégration du notebook dans un cas d'utilisation de pyVHR .....	23
5.4	Diagramme de classe.....	24
<b>A Planification, gestion de projet</b>		
A.1	Diagramme de GANTT estimé pour le semestre 9.....	28
A.2	Diagramme de GANTT estimé pour le semestre 10.....	28
A.3	Panel des prévisions des dates limites par GANTT Project .....	29
<b>B Cahier de Specification</b>		
B.1	Diagramme de la fonction 1 : Conception et Implémentation d'une nouvelle méthode rPPG .....	34
B.2	Diagramme de la fonction 2 : Intégration d'une nouvelle méthode rPPG dans le Framework.....	35
B.3	Diagramme de la tâche 1 : Implémentation d'un notebook de démonstration pour une prise en main générale du Framework.....	36
B.4	Diagramme de la tâche 2 : Implémentation et rédaction d'un guide ("notebook") pour l'intégration d'une nouvelle méthode rPPG par les futurs utilisateurs.....	37
<b>C Cahier du développeur</b>		
C.1	Diagramme de classe.....	41

# 1

## Introduction

### 1 Acteurs, enjeux et contexte

Lors d'une collaboration entre des chercheurs de l'Université de Milan et un enseignant chercheur de Polytech'Tours, un nouveau Framework python nommé pyVHR a été développé. Ce framework a pour but d'être utilisé comme une plateforme de test pour les méthodes d'estimation du rythme cardiaque à distance via l'analyse de séquences vidéo (méthodes rPPGs). Ainsi, ce Framework permet de tester, de comparer et d'analyser des méthodes rPPGs déjà existantes mais permet également aux utilisateurs d'intégrer leurs propres méthodes au Framework. Par conséquent, un utilisateur peut développer une nouvelle méthode rPPG et facilement faire des comparaisons avec des méthodes déjà existantes puis tester et analyser ses résultats. A noter que rPPG est l'acronyme de "remote photoplethysmography".

De plus, pyVHR est décrit dans la publication scientifique intitulée "An Open Framework for Remote-PPG Methods and their Assessment" écrit par Giuseppe Boccignone, Donatello Conte, Vittorio Cuculo, Alessandro D'Amelio, Giuliano Grossi et Raffaella Lanzarotti en 2020. [1]

Dans le cadre de mon projet de recherche et développement, je prendrai en main pyVHR, je développerai une nouvelle méthode rPPG et je l'intégrerai dans ce dernier. Ce projet de recherche et développement est encadré par Monsieur Conte Donatello, qui est un des membres du projet de Framework pyVHR et enseignant chercheur à Polytech'Tours. Pour rappel, le projet de recherche et développement est un projet en monôme qui se déroule tout au long de la cinquième année d'étude et qui a pour objectif de mettre en place tous les acquis de notre formation d'ingénieur en Informatique.

Ainsi, dans ce rapport, je présenterai le Framework pyVHR en détail et je présenterai mes travaux réalisés sur ce dernier. Cette description s'effectuera via les chapitres suivants : une description générale, un état de l'Art / Veille, une étude de l'existant, une analyse et conception ainsi qu'un bilan.

### 2 Objectifs

Ce projet de recherche et développement comporte deux grands objectifs. Le premier est de développer une méthode rPPG au vue de l'intégrer dans le Framework pyVHR. De ce premier objectif, un retour d'expérience ainsi qu'un guide pour reproduire cette opération sont

attendus. Le deuxième objectif est d'améliorer le Framework. Ce deuxième objectif prendra la forme d'amélioration de codes existants et l'ajout de documentation comme par exemple un nouveau guide de prise en main du Framework venant compléter les guides déjà existants.

### 3 Hypothèses

Lors de ce projet, je travaillerai sur le code déjà existant du Framework pyVHR. Le projet sera donc développé en python 3.6 et je conserverai toutes les librairies utilisées par le Framework (listée dans le fichier requirement.txt dans le projet). Concevoir une nouvelle méthode rPPG est relativement complexe, ainsi la nouvelle méthode développée sera une méthode qui n'est pas déjà implémentée dans pyVHR mais déjà décrite dans une publication scientifique.

### 4 Bases méthodologiques

Tout d'abord, plusieurs outils vont être utilisés lors de ce projet. En effet, en ce qui concerne la gestion de projet, l'outil Ganttproject sera utilisé pour gérer la planification du projet et indiquer les dates limites de chaque tâche. En ce qui concerne, la modélisation logiciel j'utiliserai l'outil diagrams.net proposé par google dans google drive. La rédaction de rapports et comptes rendus sera effectué sur Google Doc, sauf le rapport final qui est rédigé sous LaTeX. Enfin, la gestion des sauvegardes et des versions du code est réalisée via GitHub.

Pour poursuivre, la méthodologie de gestion de projet choisie est le modèle en cascade. Dans cette méthodologie, nous évoluons de phase en phase, et nous considérons qu'une phase est dépendante de sa phase précédente. Ainsi, cette méthode de gestion de projet est particulièrement adaptée à notre projet car il y a une dépendance forte et l'ordre a une grande importance entre toutes nos tâches. Au premier semestre, nous nous intéresserons plus particulièrement à l'aspect recherche de notre projet et au second semestre nous nous intéresserons plus à l'aspect développement de notre projet. Par ailleurs, tout au long du projet un compte rendu sur le travail effectué et à réaliser est transmis de façon hebdomadaire à l'encadrant du projet.

Enfin, pour plus de détails sur la planification, une partie planification est disponible en annexe.

# 2

## Description générale

### 1 Environnement du projet

Le projet est développé sous Python 3.6. Du fait de l'utilisation de python, il n'y a pas de contrainte de compatibilité entre les différents systèmes d'exploitation. Par ailleurs, il n'y a pas de contrainte sur le matériel utilisé, cependant, une machine comprenant une carte graphique NVIDIA peut permettre d'exécuter plus rapidement certains processus grâce à CUDA.

### 2 Caractéristiques des utilisateurs

Ce projet n'a pas encore d'utilisateur type. Cependant, de par sa nature, nous pouvons spéculer qu'il sera utilisé dans le cadre de projet de recherches. Ainsi, essentiellement les équipes de recherches travaillant dans le domaine du traitement du rythme cardiaque par ordinateur et les étudiants en informatique sont susceptibles d'utiliser ce Framework. Donc, des personnes maîtrisant déjà l'informatique. Cependant, le projet est open-source, n'importe qui peut l'utiliser à condition d'avoir un minimum de connaissance en langage python.

### 3 Fonctionnalités du système

Tout d'abord, l'utilisateur peut s'approprier le Framework via un large choix de personnalisation. En effet, le Framework a été créé de manière à facilement pouvoir ajouter ses propres ensembles de données et ses propres méthodes de rPPGs. De plus, l'utilisateur peut personnaliser ses tests de méthodes via des fichiers de paramétrages.

Par ailleurs, l'utilisateur dispose d'un large choix de méthodes et de jeux de données couramment utilisés dans le domaine, déjà implémentés dans le Framework. Ainsi, l'utilisateur peut réaliser une analyse scientifique d'une ou plusieurs méthodes, en respectant des conditions expérimentales similaires entre les différentes méthodes. Le Framework propose également des outils pour l'analyse.

A noter que l'utilisateur peut également utiliser certaines fonctions de manière indépendante, comme par exemple appliquer une extraction de visages sur une des vidéos d'un des jeux de données. Ceci peut servir de base pour d'autres implémentations.

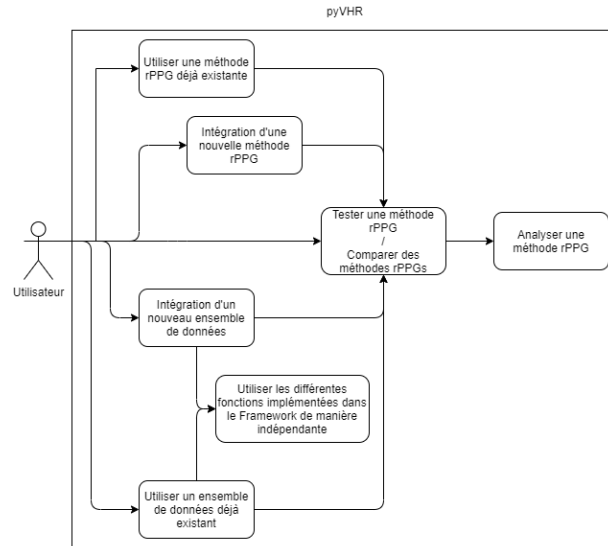


Figure 2.1 – UML - Cas d'utilisation du Framework pyVHR

Pour finir, le Framework propose également une documentation sous la forme de plusieurs jupyter notebooks pour permettre à l'utilisateur de visualiser et prendre en main ce dernier.

## 4 Structure générale du système

### Architecture du code

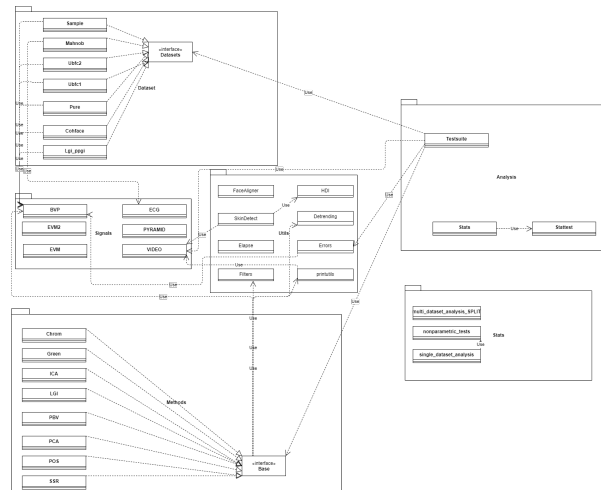


Figure 2.2 – UML - Architecture Existante

- **Package dataset** : Ensemble des classes pour récupérer les données des différents ensembles de données. L'objectif est d'obtenir un objet dataset uniforme quel que soit la source donnée pour faciliter la manipulation de donnée dans le Framework.
- **Package Methods** : Ensemble des classes qui définissent des méthodes rPPG. Une base en commun est définie dans l'interface Base.py.
- **Package Analysis** : Ensemble des classes permettant de lancer les processus, que ce soit les processus d'analyse statistique et aussi bien les processus pour tester les méthodes.
- **Package Signals** : Ensemble des classes permettant le traitement des signaux.
- **Package Utils** : Ensemble des classes "boîte à outils" du framework.

A noter qu'un fichier de configuration vient compléter l'architecture. Ce fichier configuration permet à l'utilisateur de personnaliser ses paramètres pour l'exécution de méthodes, avec un large panel de choix allant de l'ensemble de données utilisé au choix de l'extracteur de visage. Une configuration par défaut est également proposée.

### Description générale du processus de traitement du Framework

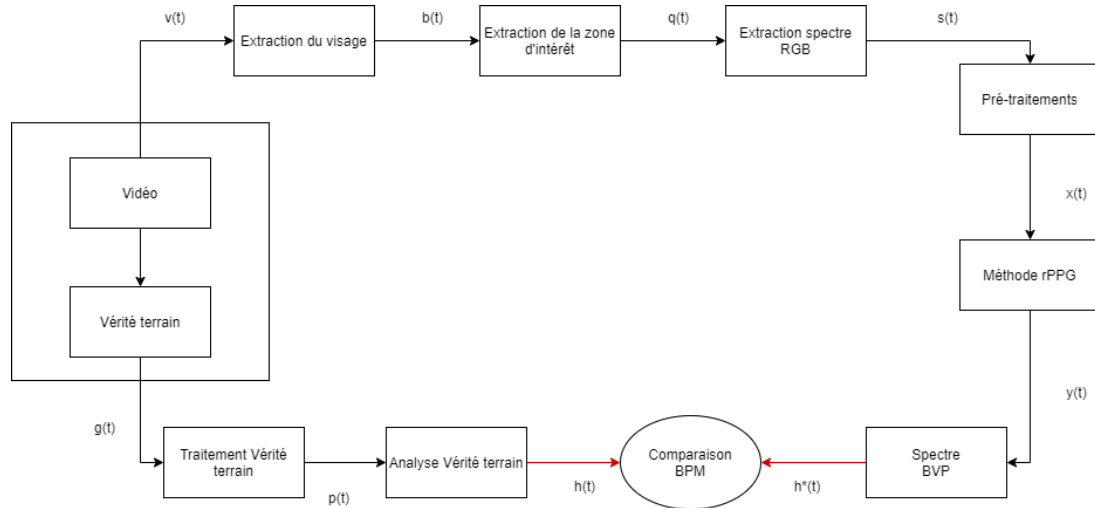


Figure 2.3 – Processus général du traitement et de l'analyse vidéo par pyVHR

Nous avons à disposition des bases données contenant à la fois des vidéos d'individus et également les mesures de rythme cardiaque associées aux vidéos. Ces mesures vont servir de références dans l'analyse des résultats de nos algorithmes, nous nommerons ces mesures "la vérité terrain". Ainsi, plus l'algorithme obtient des résultats proches de la vérité terrain, plus la méthode est pertinente, précise et donc efficace.

Pour appliquer nos algorithmes, autrement nommés "méthodes", nous devons effectuer un certain nombre de traitements. Ces traitements vont nous permettre de soumettre des données d'entrées adaptées à l'estimation du rythme cardiaque par photopléthysmographie à distance aux algorithmes. Le premier traitement nommé "Face Extraction" consiste à traquer le visage de l'individu, afin d'obtenir une vidéo avec uniquement la tête de l'individu. Dans un second temps, un traitement appelé "ROI Processing" est appliqué. Ce dernier permet de cibler une zone du visage comme le front, les joues ou bien la peau, afin de se concentrer uniquement sur les parties du visage les plus prometteuses pour nous fournir des informations exploitables. Dans un troisième temps, une phase de "RGB computation" est exécutée. Cette phase permet de calculer les intensités de couleur moyenne (ou médiane), afin de créer un signal RGB correspondant aux zones ciblées. Enfin, une phase de "Preprocessing" est appliquée. Dans cette phase, le signal est filtré afin de créer un signal insectes pour nos méthodes. Ensuite, les méthodes vont transformer ce signal en un spectre exprimant le rythme des variations du volume sanguin (Blood Volume Pulse / BVP). Traditionnellement, l'analyse de ce spectre permet de mesurer le rythme cardiaque d'un individu. Ainsi, via ce processus nous sommes en mesure de déterminer le rythme cardiaque d'un être humain à partir d'une séquence vidéo.

Pour valider une nouvelle méthode ou comparer plusieurs méthodes entre-elles, la vérité terrain est utilisée comme énoncé précédant. La vérité terrain est également sous forme d'un spectre BVP. Ainsi, un processus de conversion d'un spectre BVP en un signal de rythme cardiaque via une analyse doit être exécuté pour réaliser une comparaison du rythme cardiaque.

Enfin, pour comparer les résultats entre la vérité terrain et le résultat estimé par nos méthodes. Nous appliquons des calculs d'erreurs et des calculs d'analyses statistiques.

# 3

## Etude de l'existant

### 1 Prise en main du code existant

#### 1.1 Installation de pyVHR

Pour installer la bibliothèque déjà existante, il faut exécuter la commande “pip install pyvhr” dans un terminal. Cependant, pour que l’installation se déroule bien, il faut que son environnement de travail dispose de quelques prérequis. Tout d’abord, il faut avoir installé le langage de programmation python, une version 3.6 est requise. Ensuite, il faut s’assurer d’avoir un kit de développement C++ et également un kit de développement Fortran pour compiler certaines bibliothèques dont dépend pyVHR. A noter que pour le bon fonctionnement de cette dernière, l’installation du logiciel ffmpeg est également nécessaire. Ce dernier servira pour lors du traitement des flux vidéo.

Pour une utilisation optimale de la bibliothèque, il est conseillé de disposer d’une carte graphique NVIDIA et d’installer CUDA. CUDA permettra d’exécuter les différents calculs sur la carte graphique au lieu que sur le processeur, en vue de réduire le temps d’exécution du programme.

#### 1.2 Démonstrations

Actuellement, quelques démonstrations sont proposées pour prendre en main le code et mieux visualiser l’ensemble des possibilités proposées par le Framework.

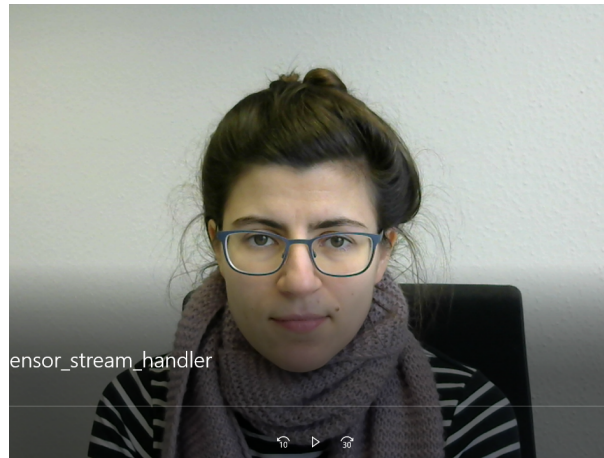
##### 1.2.1 Détection et Extraction d’un visage

La première étape du processus pour déterminer le rythme cardiaque d’un individu à partir d’une séquence vidéo est d’identifier clairement où se trouve la tête de ce dernier. Et ainsi identifier les pixels des images de la vidéo qui correspondent au visage du sujet à étudier. Pour ce faire, il faut réaliser une détection du visage et extraire ce dernier. Le Framework propose de faire ceci très simplement via la fonction “getCroppedFaces”, qui s’applique à une vidéo et prend en paramètre un détecteur et un extracteur de notre choix. Plusieurs détecteurs et extracteurs ont

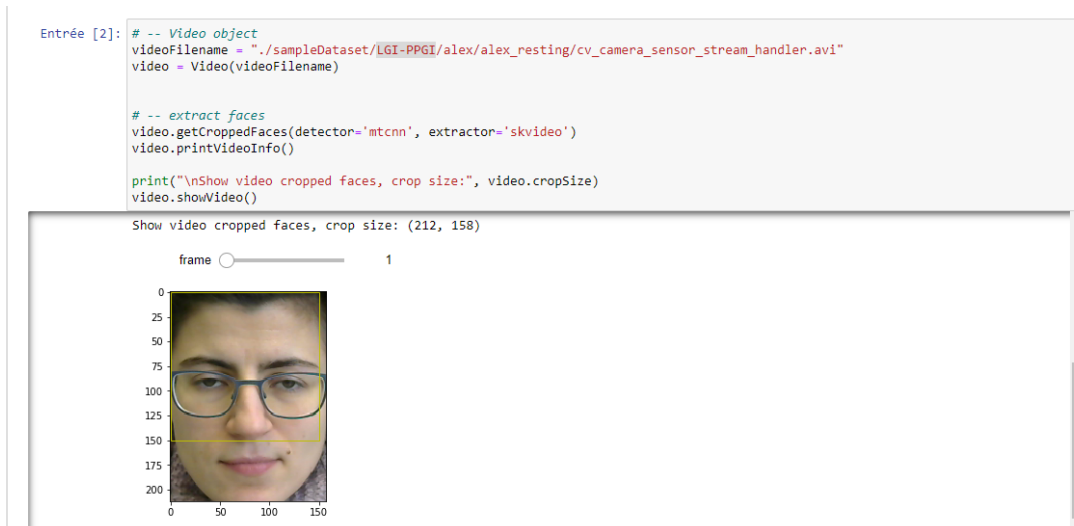


été implémentés. La liste des détecteurs proposés est la suivante : mtcnn, lib et mtcnn\_kalman. Par ailleurs, la liste des extracteurs proposés est la suivante : opencv et skvideo.

Voici un aperçu du résultat obtenu après l'application de la fonction `.getCroppedFaces` sur une vidéo avec l'utilisation du détecteur "mtcnn" et l'extracteur "skvideo" :



**Figure 3.1** – Vidéo avant extraction du visage



**Figure 3.2** – Vidéo après extraction du visage

### 1.2.2 Régions d'intérêt (Process ROIs)

Dans un second temps, la bibliothèque offre la possibilité de cibler plus précisément certaines zones du visage à partir d'une vidéo ayant subi une extraction de visage. Ceci permet de clairement identifier la peau ou le front ou les joues ou le nez du sujet de la vidéo et ainsi faire travailler les algorithmes sur une partie du corps pertinente pour l'identification du rythme cardiaque. En effet, par exemple les cheveux d'un individu ne sont pas très utiles pour déterminer le rythme cardiaque ainsi pour ne pas fausser nos estimations on souhaite ignorer cette zone de l'image. Ce processus d'identifier des zones est appelé ROIs processing ("regions of interest processing"). Voici quelques aperçus de résultats de ROIs processing :

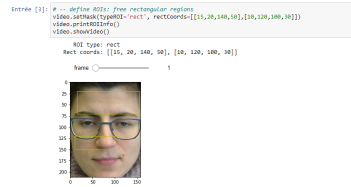


Figure 3.3 – ROI - Région rectangulaire



Figure 3.4 – ROI - Région Standart (front, joues, nez)

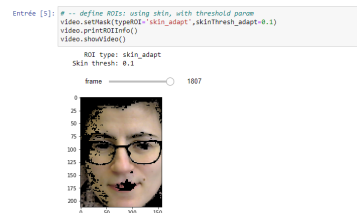


Figure 3.5 – ROI - Région avec de la peau (avec seuil adaptatif)

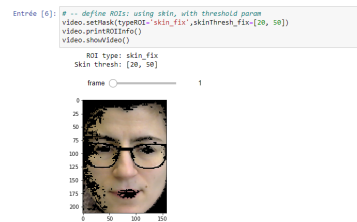


Figure 3.6 – ROI - Région avec de la peau (avec seuil fixe)

### 1.2.3 Gestion des ensembles de données

Des méthodes patrons ont été intégrés à pyVHR pour récupérer les données des bases suivantes : cohface, LGI\_PPGI, mahnob, pure, ubfc1 et ubfc2. Ces patrons permettent de créer une interface commune pour tous les jeux de données, afin de faciliter et généraliser l'analyse de ces derniers. L'interface se nomme "dataset".

Dans un premier temps, l'interface permet de récupérer l'ensemble des chemins de fichier de chaque vidéo, ainsi que l'ensemble des chemins de fichier de chaque signal GT (vérité terrain) associé à chaque vidéo. A noter, que le positionnement du dossier de l'ensemble de données est inscrit en dur dans chaque patron de base de données.

Dans un second temps, depuis l'interface "dataset", il est possible de visualiser et de traiter un signal de vérité terrain à partir des données terrains. A noter que la vérité terrain offre les données du signal BVP (le pouls), il sera donc nécessaire de le traiter pour le convertir en signal BPM (rythme cardiaque - nombre de battement par minute). L'un des traitements le plus typique est de repérer les piques sur le signal BVP pour ensuite pour déterminer le rythme cardiaque d'un individu.

Dans un troisième temps, le Framework propose également la possibilité de calculer et visualiser le rythme cardiaque d'un individu d'une vidéo à partir du signal BVP fourni par la

```

Entrée [2]: # -- dataset object
dataset = datasetFactory("LGI_PPGI")

# -- videos filenames of the dataset
print("List of video filenames:")
print(*dataset.videoFilenames, sep = "\n")

# -- GT signal filenames (ECG or BVP) of the dataset
print("\nList of GT signal filenames:")
print(*dataset.sigFilenames, sep = "\n")

List of video filenames:
./var/data/VHR/LGI-PPGI/alex\alex_gym\cv_camera_sensor_stream_handler.avi
./var/data/VHR/LGI-PPGI/alex\alex_resting\cv_camera_sensor_stream_handler.avi
./var/data/VHR/LGI-PPGI/alex\alex_rotation\cv_camera_sensor_stream_handler.avi
./var/data/VHR/LGI-PPGI/alex\alex_talk\cv_camera_sensor_stream_handler.avi

List of GT signal filenames:
./var/data/VHR/LGI-PPGI/alex\alex_gym\cms50_stream_handler.xml
./var/data/VHR/LGI-PPGI/alex\alex_resting\cms50_stream_handler.xml
./var/data/VHR/LGI-PPGI/alex\alex_rotation\cms50_stream_handler.xml
./var/data/VHR/LGI-PPGI/alex\alex_talk\cms50_stream_handler.xml

```

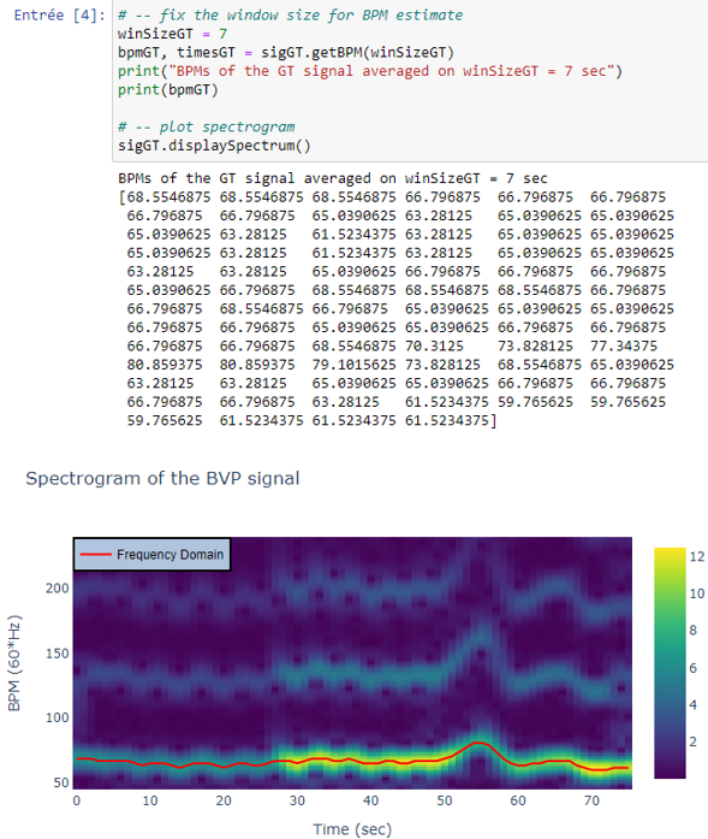
Figure 3.7 – Prise en main de l'objet "Dataset"



Figure 3.8 – Visualisation et traitement de la vérité terrain d'une video du dataset

vérité terrain. Le calcul se fait sur un lapse de temps qui est paramétrable. Par exemple, le calcul du rythme cardiaque (signal BPM) sur un intervalle de temps de 7 secondes.

Enfin, cette interface permet de facilement appliquer et faire des tests de méthodes rPPG via la classe TestSuite. Nous explorons cet aspect dans la partie suivante.



**Figure 3.9** – Calcul et visualisation du signal BPM à partir de la vérité terrain

#### 1.2.4 Application de méthodes rPPG

La Framework permet également de tester des méthodes de photopléthysmographie à distance (rPPG) en vue de comparer et étudier les résultats de ces dernières. Pour ce faire, il suffit de remplir un fichier de configuration “sample.cfg” en indiquant les méthodes à appliquer et la ou les vidéos à analyser. La vidéo doit être accompagnée par les résultats du terrain (vérité terrain), afin de pouvoir mesurer la véracité de la méthode. En plus de cela, dans ce fichier on peut également choisir des paramètres plus avancés comme le choix de l’extracteur ou du détecteur. Après cela il suffit de lancer une analyse via la classe TestSuite et sa fonction “start”. A noter que le temps d’exécution est assez long. Par ailleurs, les résultats sont stockés dans un fichier pandas. Ainsi, le résultat est donc facilement visualisable, pandas étant une bibliothèque permettant la manipulation et l’analyse des données.

```

** Run the test with the following config:
dataset: LGI_PPGI
methods: ['POS', 'CHROM']

**** Using Method: POS on videoID: 1

* Video filename: ./var/data/VHR/LGI-PPGI/alex\alex_resting\cv_camera_sensor_stream_handler.avi
  Total frames: 1836
  Duration: 73.44 (sec)
  Frame rate: 25 (fps)
  Codec: rawvideo
  Num frames: 1836
  Height: 480
  Width: 480
  Detector: mtcnn
  Extractor: skvideo
  Extracted faces: found! Loading...
  Skipping 1.90 seconds...

* POS params: start time = 3.0, end time = 73.4, winsize = 5.0 (sec)

* Errors: RMSE = 4.00, MAE = 1.70, MAX = 27.54, PCC = 0.58

**** Using Method: CHROM on videoID: 1

* Video filename: ./var/data/VHR/LGI-PPGI/alex\alex_resting\cv_camera_sensor_stream_handler.avi
  Total frames: 1836
  Duration: 73.44 (sec)
  Frame rate: 25 (fps)
  Codec: rawvideo
  Num frames: 1836
  Height: 480
  Width: 480
  Detector: mtcnn
  Extractor: skvideo
  Extracted faces: found! Loading...
  Skipping 1.90 seconds...

* CHROM params: start time = 3.0, end time = 73.4, winsize = 5.0 (sec)

* Errors: RMSE = 2.89, MAE = 1.64, MAX = 16.55, PCC = 0.77

```

Figure 3.10 – Comparaison des résultats de 2 méthodes rPPGs sur la même vidéo

### 1.2.5 Analyse Statistique

PyVHR propose également des visualisations statistiques des comparaisons entre les résultats calculés par les méthodes et les résultats terrains. Afin de pouvoir évaluer plus facilement l'efficacité d'une méthode. Ces visualisations sont proposées via la classe StatAnalysis.

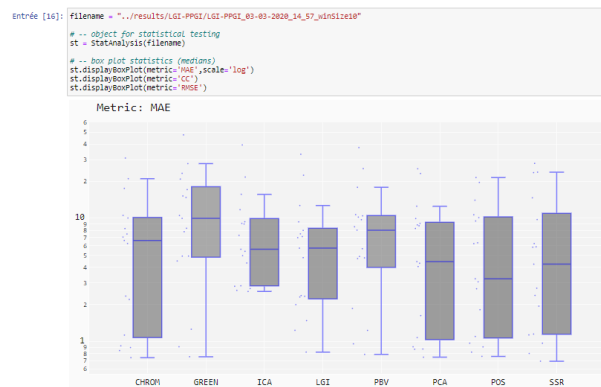


Figure 3.11 – Exemple de représentation statistique d'une comparaison de plusieurs méthodes

# 4

## Etat de l'art / Veille technologique

### 1 Les méthodes rPPGs

Lors de cet état de l'art nous allons nous intéresser aux principaux algorithmes de photopléthysmographie à distance (méthodes rPPGs). Pour rappel, une méthode de photopléthysmographie permet de convertir le spectre RGB (Rouge/Vert/Bleu) d'une zone d'intérêt (par exemple la peau d'un visage) en un signal BVP (Blood Volume Pulse) représentant les variations de volume sanguin en circulation au fil du temps puis en déduit le tempo (signal BPM). Autrement dit, une méthode de photopléthysmographie transforme une séquence d'images d'un individu en une estimation de son rythme cardiaque.

Les méthodes rPPGs peuvent exister grâce à un phénomène scientifique [5]. En effet, le battement du cœur provoque des variations de volume de sang dans les vaisseaux sanguins. Or, les tissus biologiques (plus vulgairement la peau) ne font pas que réfléchir de la lumière, elle en absorbe également. Ainsi, cette lumière absorbée vient se réfléchir sur les vaisseaux sanguins, mais de manière différente selon le volume de sang dans les vaisseaux. Par ailleurs, la lumière renvoyée par la peau et celle renvoyée par les vaisseaux sanguins sont différentes, elles ont chacune une longueur d'onde particulière. Ainsi, avec l'étude de l'évolution de la lumière émise par les vaisseaux, nous pouvons déterminer le rythme cardiaque d'un sujet. Néanmoins, la réflexion par la peau peut créer une source d'erreur.

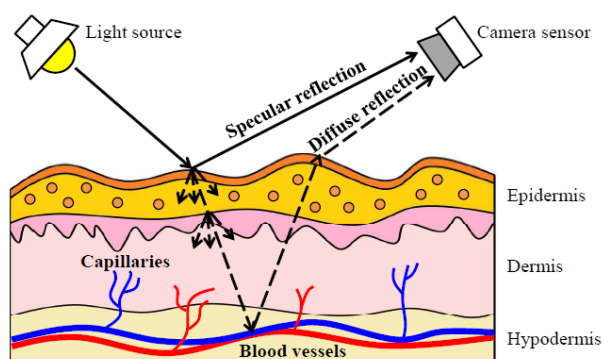


Figure 4.1 – Phénomènes d'absorption et de réflexion par la peau humaine [5]

Dans la vie de tous les jours, les méthodes de photopléthysmographie sont utilisées dans les appareils mesurant le rythme cardiaque comme l'électrocardiographe et l'oxymètre de pouls.

Ces derniers ont la réputation d'estimer le rythme cardiaque avec une grande précision mais ont le grand défaut de nécessiter un contact physique avec le sujet étudié via des capteurs et/ou des électrodes. Outre le fait qu'un contact direct avec le sujet peut être très contraignant dans certaines applications, l'utilisation d'électrodes et de capteurs peuvent également altérer légèrement le rythme cardiaque réel d'un sujet. En effet, ces derniers provoquent un effet d'inconfort ou de stress sur certains sujets.

Ainsi, l'idée de la photopléthysmographie à distance est née, avec la promesse d'une technologie sûre, peu coûteuse, fiable et efficace. En effet, à l'aide d'une simple caméra, une acquisition du rythme cardiaque humain sera effectuée. En d'autres mots, l'estimation d'un indicateur vital d'un sujet sera calculée sans les inconvénients du processus d'acquisition intrusif.

Dans l'étude suivante, nous nous intéresserons uniquement aux méthodes de photopléthysmographie à distance et ne nécessitant pas de caméra spécifique. Cette étude servira à la compréhension de manière plus approfondie du Framework pyVHR et également servira de base pour l'implémentation d'une nouvelle méthode dans ce dernier.

## 2 Les méthodes traditionnelles

### 2.1 Avant-propos

Dans un premier temps, nous allons étudier les méthodes rPPGs traditionnelles. Les méthodes rPPGs sont basées sur le domaine du traitement du signal. C'est-à-dire qu'elles sont construites sur une étude mathématique des 3 canaux d'un spectre RGB. Ainsi, la différence entre chaque méthode est comment les 3 canaux (rouge, vert et bleu) sont combinés et analysés pour extraire le signal BVP. Pour rappel, à partir d'un signal BVP, nous pouvons facilement déterminer le rythme cardiaque d'un sujet [4].



Figure 4.2 – Les méthodes traditionnelles de remote-PPG

Par ailleurs, il existe plusieurs grandes familles (d'approches) de méthodes rPPGs [4]. A partir de chaque famille, il peut exister de nombreuses méthodes qui dérivent d'une même approche. Les principales familles sont introduites ci-dessus.

La première famille est basée sur la séparation aveugle de source (Blind Sources Separation - BSS) [6], autrement dit, une étude des trois canaux sans émettre de relation entre eux. Ainsi dans cette approche, on déduit le signal BVP à partir d'une étude indépendante des canaux issus d'un spectre RGB.

La seconde famille est basée sur la chrominance du spectre RGB (CHROM).[3] La chrominance correspondant à l'information de couleur d'un flux vidéo. Ainsi, dans cette approche,

on détermine le signal BVP à partir de l'addition des canaux vidéo et d'un équilibrage des blancs de l'image.

La troisième famille est basée sur la signature BVP. En effet, les variations du volume sanguin dans la zone d'intérêt émettent des longueurs d'ondes caractéristique appelées signature. Ainsi, dans cette approche, on isole ces longueurs d'ondes pour déterminer le signal BVP.

La quatrième famille est basée sur la rotation d'un sous-espace spatial (Spatial Subspace Rotation - SSR). Le sous-espace spatial étant une zone de peau. Ainsi dans cette approche, on ne s'intéresse pas à la variation de couleur. Mais plutôt, à la mesure de la vitesse de rotation d'un sous-espace spatial en fonction du temps. Afin d'extraire le signal BVP en fonction des impulsions créées par le battement cardiaque.

## 2.2 Description des méthodes

### 2.2.1 Méthode GREEN

La méthode GREEN a été proposée en 2007 [9], néanmoins cette dernière est encore populaire dans le milieu des méthodes rPPGs. En effet, cette méthode est la plus simple à implémenter. GREEN, s'intéresse uniquement comme son nom l'indique au canal vert du spectre RGB. Autrement dit, dans cette méthode la moyenne de l'intensité de la couleur vert émise par une zone d'intérêt représente le spectre du signal BVP. Le choix de s'intéresser au canal vert n'a pas été fait au hasard. En effet, le vert correspond à un pic d'absorption par l'oxyhémoglobine, une protéine présente dans le sang permettant le transport de l'oxygène.

Mathématiquement nous pouvons transcrire cette méthode par  $y(t) = x_g(t)$ , où  $y(t)$  représente le spectre du signal BVP et  $x(t)$  le spectre RGB.[1]

A noter qu'en 2008 une variante de la méthode GREEN a été proposée [10], cette variante est nommée GREEN-RED. Comme son nom l'indique, cette méthode combine à la fois le canal vert et le rouge.

### 2.2.2 Méthode ICA

En 2011, une méthode de la famille BSS a été proposée [12] se nommant "Independent Component Analysis"(ICA). Cette méthode est une méthode statistique visant à séparer/décomposer des sources mélangées, en considérant que les sources sont indépendantes et non gaussiennes. Ainsi, dans cette méthode le spectre RGB ( $x(t)$ ) sera interprété sous la forme de mesures multivariées, correspondant aux sources mélangées. Plusieurs implémentations existent pour cette méthode, dans le Framework pyVHR la mise en œuvre suivante a été retenue.[1]

$$y(t) = \hat{z}_j(t), \text{ afin que } j = \arg \max_{k \in \{1,2,3\}} \{\text{SNR}(S(\hat{z}_k))\}$$

$$\text{où } \hat{z}(t) = \hat{W}x(t) \approx z(t) \text{ où } W \approx A^{-1}$$

avec  $W$  la matrice de décomposition et  $A$  la matrice de mélange, en considérant le processus de mélange par

$$x(t) = Az(t)$$

### 2.2.3 Méthode PCA

En 2011, une deuxième méthode de la famille BSS a été également proposée [11]. Cette dernière s'intitule "Principal Component Analysis" (PCA). PCA est une application de la



transformation de karhunen-Loève (KTL) visant à transformer des variables corrélées (liées entre elles) en variables décorrélées. Cette opération permet de réduire la redondance des données et réduire le nombre de variables. Ainsi, à partir du spectre RGB, la méthode PCA est capable d'extraire un sous-ensemble de composantes non corrélées nécessaire à la détermination du signal BVP. A noter que la méthode PCA n'est pas née de la recherche sur l'estimation du rythme cardiaque et est une méthode bien connue par le domaine des statistiques multivariées et du machine learning. Par ailleurs, cette méthode est une méthode statistique qui se base sur les objectifs suivants : maximiser les variances, minimiser les covariances et réduire la voluminosité des données.

Pour estimer le signal BVP avec PCA :[1]

$$y(t) = \hat{z}_j(t), \text{ afin que } j = \arg \max_{k \in \{1,2,3\}} \{\mathbf{SNR}(S(\hat{z}_k))\}$$

$$\text{où } \hat{z}(t) = \hat{W}^T(x(t) - \hat{\mu})$$

où  $\hat{\mu}$  la moyenne de l'échantillon,  $\hat{W}$  la matrice des vecteurs propres de la covariance de l'échantillon  $\hat{\Sigma}$  et  $x(t)$  le spectre RGB.

#### 2.2.4 Méthode CHROME

Comme indiqué précédemment, la lumière perçue par une caméra est en réalité la combinaison de deux réflexions, celle de la peau et celle des vaisseaux sanguins. Or, les erreurs résultantes de la réflexion de la peau dépendent de la position de la caméra en fonction du sujet et de la source de lumière. Elles-même dépendent du mouvement du sujet et donc sont imprévisible. Ainsi, cette composante de lumière diffusée par la peau, aussi appelée réflexion spectaculaire, peut-être une réelle faiblesse pour toutes les méthodes qui n'éliminent pas cette dernière. Ainsi, la méthode CHROME [3] propose de pallier cet handicap en utilisant la différence de couleur, aussi bien appelé signaux de chrominance. Ainsi, la méthode CHROME définit 2 vecteurs de chrominance  $X_{CHROM}$  et  $Y_{CHROM}$  composés des valeurs du spectre RGB normalisées comme ci-dessous.

$$X_{CHROM}(t) = 3x_r(t) - 2x_g(t)$$

$$Y_{CHROM}(t) = 1.5x_r(t) - x_g(t) - 1.5x_b(t)$$

Le signal BVP est calculé à partir de la formule suivante : [1]

$$y(t) = X_{CHROM}(t) - \alpha Y_{CHROM}(t)$$

$$\text{avec } \alpha = \frac{\sigma(X_{CHROM}(t))}{\sigma(Y_{CHROM}(t))} \text{ et } \sigma(\cdot) \text{ l'écart-type.}$$

#### 2.2.5 Méthode SSR

En 2016, la méthode SSR ou aussi nommée 2SR a été proposée [14]. Contrairement à de nombreuses méthodes, la méthode SSR ne repose pas sur des variations d'intensités lumineuses, ni sur l'évolution de la teinte de la peau. En effet, dans cette méthode nous nous intéressons à une zone de peau précise et nous suivons l'évolution de la position de cette dernière dans le temps, afin d'estimer le pouls. De manière plus technique, SSR consiste à construire un sous-espace de pixel représentant la peau puis de calculer l'angle de rotation du sous-espace sur une séquence d'image. En pratique, cette méthode peut avoir des problèmes de performance si la zone de peau est bruitée ou mal choisie, mais peut également surmonter d'autres problèmes comme la connaissance de la couleur de peau du sujet au préalable.

Le calcul du signal BVP s'effectue via la formule suivante. [1]

$$y(t) = \bar{P}(t)$$

Sachant que  $\bar{P}^{(t-l)} = \bar{P}^{(t-l)} - (\bar{p} - \mu(\bar{p}))$  avec  $\mu$  l'opérateur de moyenne

$$\text{où } \bar{p} = \overline{EV_1} - \frac{\sigma(\overline{EV_1})}{\sigma(\overline{EV_2})} \overline{EV_2}$$

$$\text{où } EV' = \sqrt{\frac{\lambda_1^t}{\lambda_2^t}} \cdot u_1^{tT} \cdot u_2^t \cdot u_2^{tT} + \sqrt{\frac{\lambda_1^t}{\lambda_3^t}} \cdot u_1^{tT} \cdot u_3^t \cdot u_3^{tT}$$

$$\text{avec } E = \left( \sqrt{\frac{\lambda_1^t}{\lambda_2^t}}, \sqrt{\frac{\lambda_1^t}{\lambda_3^t}} \right)^T$$

Sachant que  $C = \frac{X^T \cdot X}{N}$  avec  $N$  le nombre de pixel et  $X$  le spectre RGB alors  
 $\Lambda = \text{diag}(\lambda_1, \lambda_2, \lambda_3)$  vecteur propre de  $C$

et  $V' = (u_1^t)^T \cdot (u_2^t, u_3^t)$  la rotation entre le vecteur  $u_1$  et le plan orthonormé  $u_2, u_3$

### 2.2.6 Méthode LGI

L'une des problématiques lors de l'étude du visage est le mouvement interne à ce dernier. En effet, un visage se modèle en fonction des expressions faciales et de l'utilisation des articulations spécifiques au visage. Or, selon comment le visage est modelé, la lumière se réfléchit de manière différente. De plus, la modélisation de l'impact des mouvements du visage sur la réflexion de la lumière est complexe. Ceci, peut être un facteur de confusion pour certaines méthodes de rPPG. Ainsi, pour faire face à ce problème, la méthode LGI (Local Group Invariance) est née.

L'implémentation de cette méthode comporte plusieurs similarités avec la méthode SSR. En effet, elle se base sur la matrice de corrélation  $C$  et ses vecteurs propres ( $U$ ). [1]

$$C = \frac{X^T \cdot X}{N}$$

où  $X$  est le spectre RGB et  $N$  est le nombre de pixels du sous-espace.

L'opérateur de projection  $O$  pour le nouvel espace est calculé par  $O = I - UU^T$  avec matrice identité  $I$ . Finalement, la formule pour obtenir le spectre du signal BVP est la suivante :

$$y(t) = Ox(t)$$

### 2.2.7 Méthode PBV

Dans la méthode PBV (Pulse Blood Volume), proposée en 2014 [13], nous nous intéressons plus particulièrement à la signature de la lumière réfléchiée par changement de volume sanguin dans les vaisseaux. Ainsi, nous pouvons distinguer clairement les longueurs d'ondes associées aux variations de sang dans le corps sur notre spectre RGB. Les erreurs dues à la réflexion de la lumière sur peau n'est donc plus prise en compte, donc cette dernière ne parasite plus notre spectre RGB.

Pour calculer la signature des variations de volume de sang dans les vaisseaux, la formule est la suivante : [1]

$$P_{bv}^{c=r,g,b}(t) = \frac{\sigma(X_c)}{\sqrt{\sigma^2(X_r) + \sigma^2(X_g) + \sigma^2(X_b)}}$$

où  $X = \{X_r, X_g, X_b\}$  représente le spectre RGB pour une fenêtre de temps donné et  $\sigma(\cdot)$  l'écart-type.

L'utilisation de la signature se fait via la formule suivante :

$$y(t) = Mx(t)$$

où  $M = kP_{bv}(XX^T)^{-1}$  avec  $k$  le facteur de normalisation

### 2.2.8 Méthode POS

La méthode Plane-Orthogonal-to-Skin (POS), tout comme CHROME et PBV est basée sur des principes d'optique et de physiologie. D'ailleurs, POS partagent un certain nombre de propriétés avec ces dernières. En effet, la méthode POS cherche à éliminer la réflexion de lumière due à la peau, en construisant un plan orthogonal par rapport à la teinte de la peau en fonction du temps. Plus précisément, dans un premier temps POS fait une normalisation temporelle sur le spectre RGB représentant la teinte de la peau, puis une projection orthogonale du spectre RGB, et enfin le calcul du signal BVP ( $y(t)$ ). A noter que contrairement aux autres méthodes, la méthode POS estime le pouls à partir d'une multitude d'intervalle de temps issues de la division de la séquence vidéo d'entrée en petit morceau. Ceci a pour objectif de réduire le bruit du signal. Par ailleurs, il est intéressant de noter que les deux signaux projetés de POS sont en phase, contrairement aux signaux projetés de CHROME qui sont en opposition de phase. [1]

$$y(t) = X_{POS}(t) + \alpha Y_{POS}(t)$$

où

$$X_{POS}(t) = x_g(t) - x_b(t);$$

$$Y_{POS}(t) = x_g(t) + x_b(t) - 2x_r(t)$$

et

$$\alpha = \frac{\sigma(X_{POS}(t))}{\sigma(Y_{POS}(t))} \text{ avec } \sigma(\cdot) \text{ qui est l'écart-type.}$$

## 3 Les nouvelles méthodes

### 3.1 Avant-propos

Plus récemment, une nouvelle famille de méthodes est née. Cette famille se nomme la famille des Méta-Learners, certains chercheurs la surnomment également "meta-RPPG" [8]. L'approche de cette nouvelle famille se base sur un réseau de neurones qui apprend à déterminer le rythme cardiaque d'un sujet à partir d'une séquence vidéo. Contrairement, aux familles traditionnelles qui se basent sur un enchaînement prédéfini de traitement du signal. De plus, les Méta-Learners ont une capacité plus accrue à s'adapter. Or, nous savons que dans le domaine des rPPGs de nombreux facteurs limitants peuvent être un frein pour certaines méthodes traditionnelles comme la variation de couleur de peau des différents sujets, l'éclairage ambiant, la qualité de la vidéo ou bien l'évolution de l'articulation de visage au fil du temps. Ainsi, à condition que l'entraînement de notre méthode de méta-rPPG soit suffisamment riche en diversité, en qualité et en volume, les performances potentielles de ces nouvelles méthodes sont grandes. Par ailleurs, les méthodes de méta-rPPG profitent également de l'essor actuel de l'Intelligence Artificielle pour évoluer et s'améliorer rapidement, notamment car aujourd'hui nous comprenons de mieux en mieux les techniques de système apprenant.

Aujourd'hui, les choix d'implémentations des "Meta-rPPGs" sont très diverses. En effet, il existe plusieurs types et architectures de réseau de neurones. Par ailleurs, nous pouvons également choisir où intégrer notre réseau de neurones dans le processus traditionnel. Autrement dit, directement sur la séquence vidéo brute ou uniquement sur un visage déjà extrait de la vidéo. A noter que quel que soient nos choix, l'application de notre algorithme apprenant ne sera pas triviale. En effet, nous devons toujours prendre en compte à la fois la notion spatiale et temporelle dans l'élaboration de notre méta-Learner.

### 3.2 Méthodes basées sur un réseau RNN

L'acronyme RNN de "Recurrent neural network" signifie "réseau de neurones récurrents" en français. Le réseau de neurones récurrents est une méthode d'apprentissage basée sur un ensemble d'unités (aussi appelé neurones) interconnecté entre elles. De plus, les réseaux de neurones récurrents permettent de gérer les cas prenant la forme d'une séquence en fonction du temps. Ainsi, ce type de réseau permet d'introduire la notion de mémoire. C'est-à-dire que les entrées passées du réseau laissent une trace et ainsi peuvent impacter les prédictions futures du réseau. Cependant, les RNN simples rencontrent certaines limites, notamment à cause du problème de "vanishing-gradient" (gradient qui tend vers 0). Ainsi des variantes pour pallier ce problème ont vu le jour, notamment l'architecture Long Short Term Memory (LSTM) dont nous allons nous intéresser [7].

Les LSTMs sont bien connues pour être extrêmement efficaces lorsque l'on traite une "série temporelle". Dans le cas des méthodes rPPGs, les séries temporelles sont représentées par les séquences vidéos, qui est ni plus ni moins qu'une série d'images évoluant dans le temps. Par ailleurs, en 2018, il a été montré dans le domaine des méthodes PPG avec contact, qu'il était possible d'entraîner des LSTMs afin de prédire la pression artérielle à partir des signaux d'un électrocardiogramme, avec des résultats tout aussi précis (voir plus précis) que par les méthodes traditionnelles.

### 3.3 Méthodes basées sur un réseau 3D CNN

Une approche très populaire dans le domaine des méta-rPPGs est l'apprentissage automatique par réseau de neurones à convolution (CNN) [2]. Les CNNs ont été inspirés et construits sur les principes de fonctionnement de la vision des êtres vivants et sont dédiés à l'extraction de caractéristiques (réduction des données brutes en des groupes plus facile à traiter) pour les entrées 2D comme les images. Par ailleurs, ce type de réseau offre d'excellents résultats, notamment dans le domaine de la segmentation et de la reconnaissance d'objets.

Pour l'estimation du rythme cardiaque, la notion de temps est tout aussi importante que la notion d'espace. Or dans une image la notion de temps n'est pas induite. Ainsi, une évolution du 2D CNN a été développée, se nommant le 3D CNN. Le 3D CNN est particulièrement adapté au flux vidéo. En effet, il a été conçu pour extraire à la fois les caractéristiques spatiales et temporelles d'une entrée vidéo comme une séquence vidéo. Ce qui offre l'opportunité de pouvoir étudier l'évolution entre plusieurs images adjacentes.

### 3.4 Méthodes hybrides basées sur 3DCNN-RNN

Récemment, dans le domaine d'étude et d'analyse d'ensemble de données spatio-temporelles, une approche plus hybride a été proposée, fusionnant les réseaux de neurones RNN et CNN. Cette approche se traduit pour nous par la combinaison d'un réseau de neurones 3D CNN avec un réseau de neurones RNN d'architecture LSTM. Dans une telle approche et dans

le cas, le réseau 3D CNN a pour rôle de rendre l'ensemble de données initial plus facilement traitable (extraction des caractéristique), afin de le fournir au réseau RNN pour l'estimation du rythme cardiaque. L'objectif de cette approche étant de profiter à la fois des points forts des réseaux de neurones CNN et RNN, en vue d'obtenir de meilleurs résultats.

## 4 Récapitulatif

Nom de la méthode	Description brève	Approche / Famille
GREEN	Méthode basée sur le canal vert du spectre RGB	CHROM
ICA	Méthode basée sur la décomposition du spectre RGB	BSS
PCA	Méthode basée sur la transformation de karhunen-Loève (KTL) visant à obtenir un ensemble de variables non liées entre elles à partir du spectre RGB.	BSS
CHROME	Méthode basée sur les signaux de chrominance .	CHROM
SSR	Méthode basée sur l'évolution spatiale d'un sous-espace de pixel de peau en fonction du temps.	SSR
LGI	Méthode basée sur des transformations locales en fonction du temps.	SSR
PBV	Méthode basée sur la signature lumineuse des variations de volume de sang.	basée sur la signature BVP
POS	Méthode basée sur l'évolution de la teinte de peau dans l'espace RGB temporellement normalisé.	CHROM
RNN	Méthode basée sur un réseau de neurones de type RNN avec une architecture LSTM.	Meta-rPPG
3D CNN	Méthode basée sur un réseau de neurones de type 3D CNN	Meta-rPPG
Fusion CNN-RNN	Méthode basée sur la combinaison de neurones de type 3D CNN et RNN d'architecture LSTM.	Meta-rPPG

# 5

## Analyse et conception

### 1 Analyse

#### 1.1 La méthode à développer

##### 1.1.1 Hypothèses

L'un des premiers enjeux est de concevoir un modèle 3D CNN pouvant faire une prédiction du rythme cardiaque à partir d'une séquence vidéo provenant d'un des ensembles de données proposés par le Framework. Pour ce faire, l'analyse et la conception du modèle 3D CNN sera inspirée par le projet open source décrite dans la publication scientifique "3D convolutional neural networks for remote pulse rate measurement and mapping from facial video, Applied Sciences, vol. 9, n° 20, 4364 (2019)" [2] écrite par Frédéric Bousefsaf, Alain Pruski et Choubeila Maaoui. Quelques modifications seront apportées au projet, afin de le rendre compatible avec le reste du Framework. Ainsi, ce modèle sera considéré comme une nouvelle méthode à implémenter par la suite.

##### 1.1.2 Description du modèle 3D CNN

Notre modèle 3D CNN assurera à la fois l'extraction des caractéristiques ainsi que la classification. Pour ce faire, une première partie du réseau sera une couche de convolution 3D qui aura pour rôle d'extraire les caractéristiques. Puis, une seconde partie prendra le relais avec 2 couches denses afin d'assurer la classification en  $n$  classes. Les classes représentent un intervalle de BPM, par exemple un BPM égale entre 90 à 92.5. La taille de l'intervalle et donc le nombre de classes est modulable. Ainsi, en bout de chaîne nous pourrions proposer une estimation du rythme cardiaque du sujet, en partant du principe que la classe la plus probable correspond au rythme cardiaque du sujet.

Concernant l'implémentation, le modèle sera bien évidemment développé en python. En ce qui concerne la bibliothèque choisie, nous utiliserons TensorFlow via sa surcouche Keras.

Par ailleurs, le paramétrage de l'architecture du réseau sera repris en partie du projet cité ci-dessus. C'est-à-dire, que pour la partie du réseau de type CNN. La couche de convolution

3D est composée de 32 filtres 3D de taille de 58x20x20 (pour une vidéo de taille 20x20x60), suivis d'une couche 3D max pooling de taille 2x2x2. Une fonction d'activation de type ReLu est alors appliquée. Puis une régularisation de type Dropout est utilisée pour ignorer de manière aléatoire 20% du nombre total des unités de la couche de convolution pour éviter une surcharge de travail. Ensuite, pour la phase de classification, une première couche de type "Dense" composée de 512 neurones. Une fonction d'activation de type ReLu est également appliquée. Pour les mêmes raisons que la partie CNN, une régularisation de type Dropout avec un taux de 20% est alors utilisée. Pour finir, une deuxième couche de type "Dense" composée de 76 neurones (76 classes) est connectée avec la couche précédente. 75 neurones représentent les classes de rythmes cardiaques entre 55 et 240 bpm répartie avec un intervalle régulier de 2.5 bpm. Un neurone est également présent pour représenter la classe des rythmes cardiaques non compris entre 55 et 240 bpm. Sur cette dernière couche, une fonction d'activation softmax est bien évidemment utilisée pour déterminer la classe la plus probable. A noter que des personnalisations seront probablement nécessaires, notamment si l'on choisit de travailler sur des séquences plus grandes. En effet, par défaut dans pyVHR nous travaillons sur 5 secondes avec en moyenne 30 images par seconde soit 150 images et non 60 images. Une différence de précision est très probablement à prévoir.

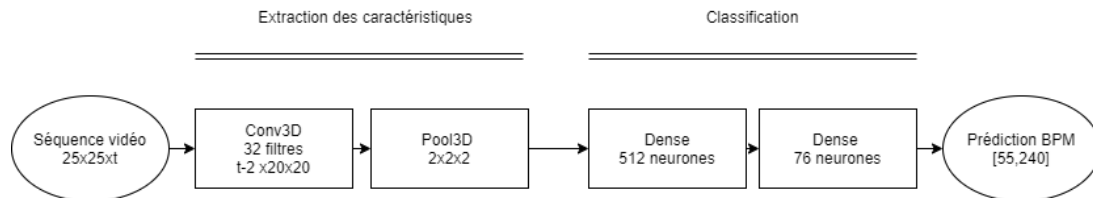


Figure 5.1 – Architecture du modèle 3D CNN

### 1.1.3 Entraînement du modèle 3D CNN

Une des phases essentielles pour notre modèle est son apprentissage. Ce dernier doit être suffisamment riche et diversifié, pour permettre au modèle d'apprendre à estimer le rythme cardiaque. Pour cette partie, il est intéressant de générer des vidéos artificielles. Ainsi, le modèle sera entraîné sur un plus large panel d'exemple de rythme cardiaque différents. De plus, nous ne serons pas limités à quelques vidéos pour l'apprentissage car nous pourrions en générer autant que nous en souhaitons. Pour générer ces données synthétiques, nous reprendrons les travaux de Frédéric Bousefsaf, Alain Pruski et Choubeila Maaoui [2], avec quelques adaptations, notamment sur la longueur de vidéo utilisée par séquence, seront effectuées.

Le processus de création de données est le suivant. Dans un premier temps, la forme de l'onde est définie en fonction de l'impulsion que l'on souhaite. A partir de cette forme d'onde, un signal est créé. A ce signal est ajouté une tendance linéaire, quadratique et cubique, afin d'obtenir un signal sur une durée voulue. Ensuite, le signal est converti en vidéo. Le procédé pour réaliser ceci est de simplement répéter le signal pour chaque pixel de la vidéo. Autrement dit, les pixels de l'image à l'instant  $t$  auront une valeur induite par l'état du signal à l'instant  $t$ . Enfin, pour rendre la vidéo réaliste et ne pas perturber l'apprentissage du modèle, du bruit est ajouté de manière aléatoire sur chaque image de la vidéo. Autrement dit, le bruit normalement produit par la capture vidéo de la caméra est reproduit. A noter que la taille et la durée de la vidéo (en termes de nombre d'images) générée peut être modifiée.

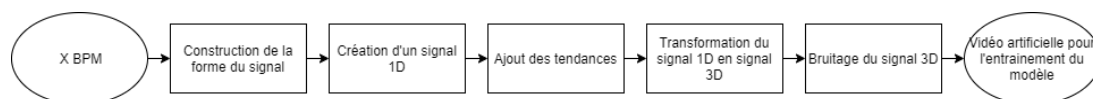


Figure 5.2 – Processus de génération de vidéo artificielle

A noter que les paramètres choisis pour l'entraînement sont les suivants. Un optimiseur de type Adam avec une fonction d'optimisation (optimisation function) initialisée à un taux d'apprentissage de  $10^{-3}$ , une fonction de perte (loss function) de type cross-entropy et un nombre de période (epoch) de 5000.

#### 1.1.4 Validation du modèle 3D CNN

Pour vérifier que le modèle fait de bonnes prédictions et qu'il réponde bien à nos attentes. Le modèle sera testé sur des échantillons des ensembles publiques de données, dans les mêmes conditions. Nous nous attendons à voir des résultats au moins similaires à la publication scientifique [2]. Une fois validée, le modèle sera prêt pour être incorporé dans le Framework.

#### 1.1.5 Implémentation d'un modèle dans pyVHR

Plusieurs défis vont intervenir lors de l'intégration du modèle entraîné dans le Framework.

En effet, nous souhaitons que le modèle puisse être utilisé avec le système de gestion d'ensemble de données déjà existant. Pour ce faire, la méthode devra manipuler des objets "dataset". De plus, une adaptation devra être faite pour travailler avec le fichier de la vidéo comportant extraction de visage, et non avec des fichiers .png comme dans le projet décrit dans la publication scientifique de Frédéric Bousefsaf, Alain Pruski et Choubeila Maaoui [2].

Un travail sur les résultats de sortie devra être également effectué. En effet, dans le projet de Frédéric Bousefsaf, Alain Pruski et Choubeila Maaoui [2], les résultats sont exportés au format matlab pour être ensuite visualisés et analysés. Ainsi, pour une intégration parfaite dans le Framework, les sorties du modèle seront redirigées vers les fonctions d'analyse et de visualisation déjà existantes dans pyVHR.

Par ailleurs, l'utilisation des fichiers de configurations interagissant avec la classe base.py devra être fonctionnelle pour cette nouvelle méthode. Ainsi, l'introduction d'une nouvelle configuration par défaut pour ce modèle sera définie. De plus, des adaptations dans la classe base.py seront effectués pour permettre d'utiliser le modèle dans le framework. Un maximum d'options de configuration doit être conservées. Afin de garantir le niveau de personnalisation des choix paramètres proposés par pyVHR et pouvoir faire des comparaisons de performance avec d'autres méthodes.

### 1.2 Les notebooks à développer

#### 1.2.1 Notebook 1 : Prise en main

Pour respecter et harmoniser notre production, nous allons réaliser ce tutoriel sous la forme d'un jupyter notebook. Ce jupyter notebook viendra compléter les fonctions déjà démontrées dans les autres tutoriels déjà existant. Ainsi, dans ce dernier sera expliqué comment fonctionnent les fichiers de configurations et comment modifier les paramètres de ces derniers. Pour rappel, les fichiers de configurations permettent de personnaliser son exécution, notamment pour la fonction TestSuite. Fonction qui sert à comparer 2 méthodes dans les mêmes conditions d'exécution. Ce qui est utile pour faire de réelles conclusions sur l'efficacité d'une méthode par rapport à une autre. A noter que le fichier configurations contient de nombreux paramètres, ce qui offre un large panel de choix de personnalisation pour l'utilisateur.

Par ailleurs, des détails sur le lancement du projet seront également proposés. Notamment, sur le positionnement des ensembles de données dans l'architecture du Framework. Cette



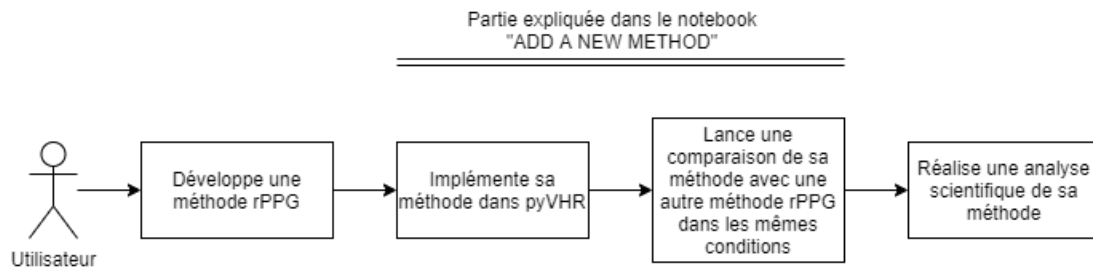
étape n'est pas expliquée dans les notebooks déjà existant, alors qu'elle est pourtant essentielle pour utiliser le framework. De plus, une procédure d'installation plus précise peut éventuellement être ajoutée.

Pendant ce processus, il sera intéressant d'ajouter de la documentation ou de clarifier le code existant, si le besoin se fait ressentir.

### 1.2.2 Notebook 2 : Ajouter une méthode

Ce document sera également sous la forme d'un jupyter notebook, toujours pour respecter et s'harmoniser avec l'existant. Dans ce notebook, je vais retracer toutes les étapes qui m'ont permis d'ajouter une méthode dans le framework, de sorte à conserver un maximum d'options proposées par le Framework. Ce document devra être suffisamment précis pour que les utilisateurs arrivent à implémenter leur nouvelle méthode sans trop de difficultés. Ainsi, ce document fournira une aide essentielle pour toute personne ayant besoin d'ajouter une nouvelle méthode dans pyVHR. Le framework ayant été imaginé aussi pour ça à l'origine.

Pendant ce processus, il sera intéressant d'ajouter de la documentation ou de clarifier le code existant, si le besoin se fait ressentir.

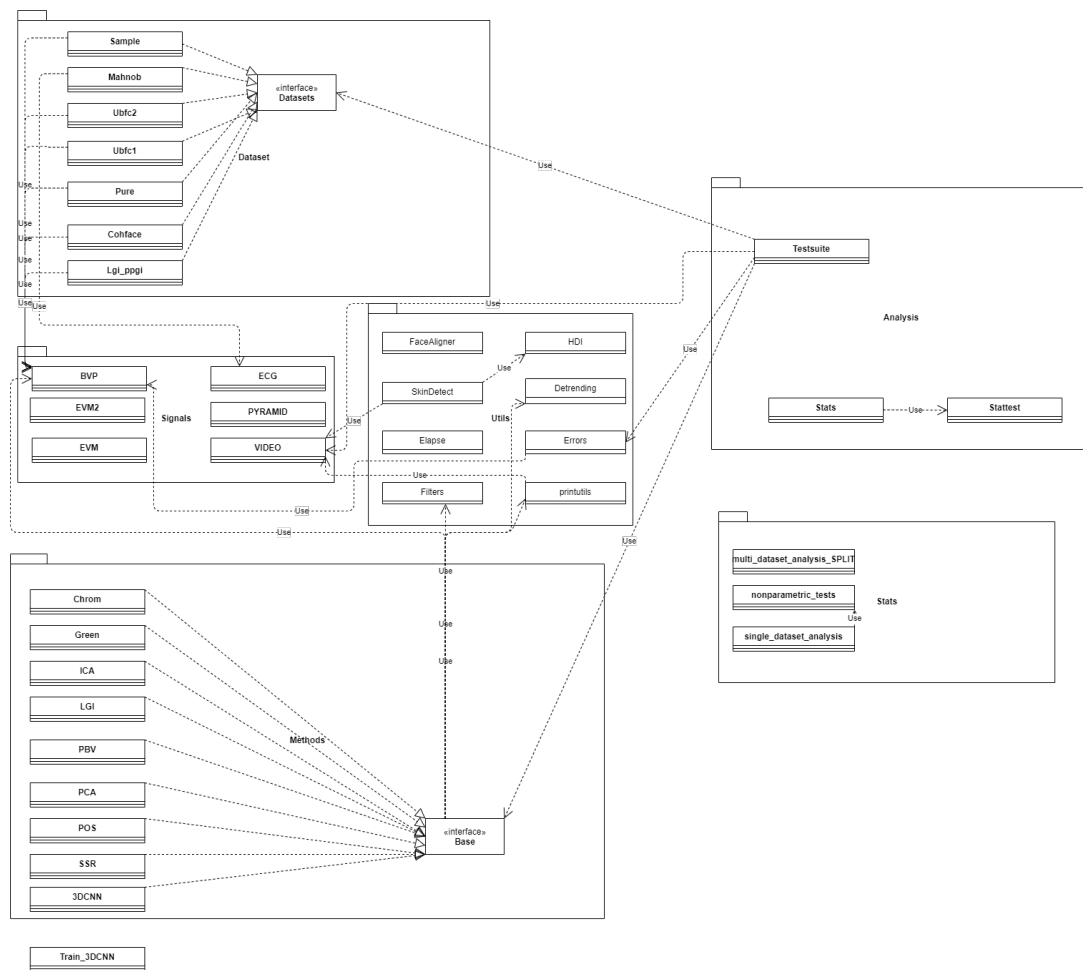


**Figure 5.3** – Intégration du notebook dans un cas d'utilisation de pyVHR

## 2 Modélisation proposée

Le Framework a été conçu pour pouvoir recevoir facilement de nouvelles méthodes. Ainsi, architecturalement, il n'y a pas de différence notable entre l'ancienne et le nouveau. A noter que l'on peut observer que dans le nouveau une classe 3DCNN a été ajoutée. Cette classe représente la classe de notre nouvelle méthode, ça sera donc ici que l'appel de notre modèle entraîné sera implémenté et effectué. L'ajout de cette classe entraînera également quelques modifications dans le code de la classe Base. Afin d'intégrer au mieux notre nouvelle classe dans le pyVHR. A noter que l'ajout d'une nouvelle méthode rime également avec des ajouts et des modifications dans les fichiers de configurations Sample.cfg et Default\_test.cfg directement liés avec Base.py.

Nous pouvons également observer l'ajout d'une classe "Train\_3DCNN". Cette classe n'est pas directement liée avec le reste du Framework. Sa fonction sera d'entraîner notre modèle, afin d'obtenir un modèle fonctionnel en vue de l'intégrer dans le Framework. Il sera peut-être intéressant de proposer aux utilisateurs cette classe, notamment pour leur offrir plus haut un niveau de personnalisation dans les paramètres



**Figure 5.4** – *Diagramme de classe*

# 6

## Bilan et conclusion

### 1 Bilan du semestre 9

Le semestre 9 était orienté vers la partie recherche de mon projet de recherche et développement. Ainsi, ce semestre a été consacré à la familiarisation avec le sujet, le Framework pyVHR et les méthodes rPPGs. Cette étape était importante pour pouvoir planifier la partie développement, mais également pour faire les bons choix afin que le développement se passe pour le mieux.

Ainsi lors de ce semestre 9, j'ai fait l'ensemble des tâches planifiées pour le semestre 9. C'est-à-dire,

- Une découverte du sujet et du projet
- Une étude de l'existant à propos du Framework pyVHR
- Un état de l'art et de Veille au sujet des méthodes rPPGs
- La rédaction d'un cahier de spécification
- La planification du projet
- Une analyse et conception pour préparer la phase de développement
- La rédaction du rapport du semestre 9 et la réalisation d'un PowerPoint pour la soutenance

Par ailleurs, l'ensemble des tâches planifiées au semestre 9 ont été finies lors de ce semestre. A noter, qu'une tâche a pris un plus de temps que prévu, l'état de l'art. Cependant, d'autres tâches comme la planification et la rédaction du rapport ont été plus rapide que planifiées. Ainsi, les temps de ces tâches se sont compensés et donc je n'ai pas accumulé de retard. La phase de recherche s'est finie en temps et en heure.

Enfin, les tâches qui restent à faire sont les tâches qui sont planifiées au semestre 10. C'est-à-dire essentiellement du développement. Cependant, si je ressens le besoin d'approfondir mes recherches lors de la phase de développement, je le ferai.

### 2 Planning du semestre 10

Lors du semestre 10, les tâches qui seront réalisées sont les suivantes :

- Développement et entraînement du modèle de la nouvelle méthode
- Implémentation de la nouvelle méthode dans le Framework

- Elaboration du guide d'implémentation d'une nouvelle méthode
- Elaboration de la démonstration générale
- Rédaction du rapport du semestre 9 et la réalisation d'un PowerPoint pour la soutenance

L'ensemble de la planification de ces tâches est décrit en annexe ([Planification](#)).

## Annexes

# A

## Planification, gestion de projet

### 1 Evolution du projet

#### 1.1 Diagramme de GANTT planifié

##### 1.1.1 Planification pour le semestre 9

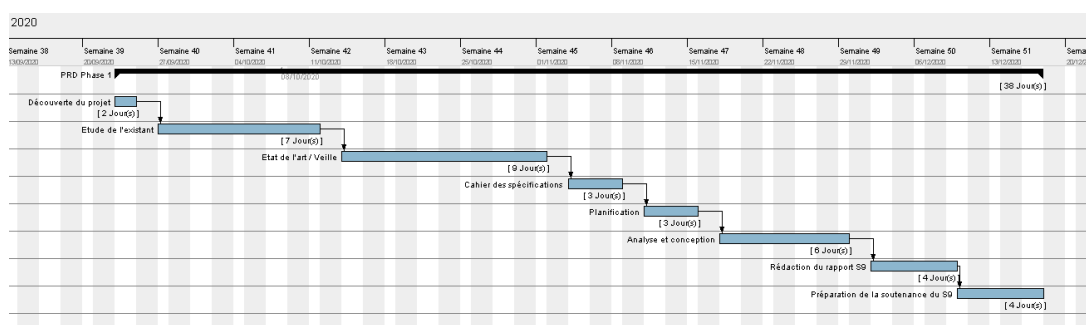


Figure A.1 – Diagramme de GANTT estimé pour le semestre 9

##### 1.1.2 Planification pour le semestre 10

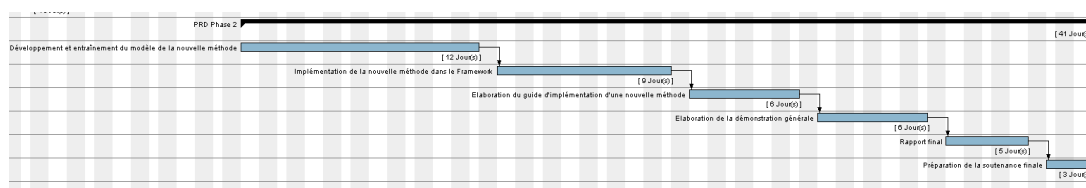



Figure A.2 – Diagramme de GANTT estimé pour le semestre 10



Nom	Date de début	Date de fin
☐ • PRD Phase 1	23/09/2020	17/12/2020
• Découverte du projet	23/09/2020	24/09/2020
• Etude de l'existant	27/09/2020	11/10/2020
• Etat de l'art / Veille	14/10/2020	01/11/2020
• Cahier des spécifications	04/11/2020	08/11/2020
• Planification	11/11/2020	15/11/2020
• Analyse et conception	18/11/2020	29/11/2020
• Rédaction du rapport S9	02/12/2020	09/12/2020
• Préparation de la soutenance du ...	10/12/2020	17/12/2020
☐ • PRD Phase 2	06/01/2021	08/04/2021
• Développement et entraînement ...	06/01/2021	31/01/2021
• Implémentation de la nouvelle ...	03/02/2021	21/02/2021
• Elaboration du guide d'impléme...	24/02/2021	07/03/2021
• Elaboration de la démonstration ...	10/03/2021	21/03/2021
• Rapport final	24/03/2021	01/04/2021
• Préparation de la soutenance fin...	04/04/2021	08/04/2021

Figure A.3 – Panel des prévisions des dates limites par GANTT Project

## 2 Description des tâches

### 2.1 Semestre 9

#### Tâche 1 : Découverte du projet

- Date de début : 23 septembre
- Date de fin : 24 septembre
- Duree : 2 jours
- Description : Lecture de la publication scientifique présentant le Framework.
- Discussion avec l'encadrant

### Tâche 2 : Étude de l'existant

- Date de début : 24 septembre
- Date de fin : 11 octobre
- Duree : 3 semaines
- Description : Prise en main du code existant en suivant les tutoriels / démonstrations proposés par le Framework. Revue de code.
- Document : Rapport Etude de l'existant

### Tâche 3 : Etat de l'art / Veille

- Date de début : 11 octobre
- Date de fin : 1 novembre
- Duree : 3 semaines
- Description : Description des différentes méthodes rPPGs existantes.
- Document : Rapport Etat de l'art

### Tâche 4 : Cahier des spécifications

- Date de début : 1 novembre
- Date de fin : 8 novembre
- Duree : 1 semaine
- Description : Définir les besoins fonctionnels et non fonctionnels du projet.
- Livrable : Cahier des spécifications fonctionnelles et cahier des spécifications non fonctionnelles.

### Tâche 5 : Planification

- Date de début : 8 novembre
- Date de fin : 15 novembre
- Duree : 1 semaine
- Description : Planification de l'organisation et de l'ordonnancement des tâches.
- Document : Diagramme de Gantt

### Tâche 6 : Analyse et conception

- Date de début : 15 novembre
- Date de fin : 30 novembre
- Duree : 2 semaines
- Description : Etudes et recherches pour le développement des différentes tâches du projet décrites dans le cahier des spécifications fonctionnelles.
- Livrable : Rapport d'analyse et de conception

### Tâche 7 : Rédaction du rapport S9

- Date de début : 30 novembre
- Date de fin : 9 décembre
- Duree : 1 semaine et demie
- Description : Rédaction du rapport.
- Livrable : Rapport S9

### Tâche 8 : Préparation de la soutenance du S9

- Date de début : 9 décembre
- Date de fin : 17 décembre
- Duree : 1 semaine



- Description : Rédaction du rapport.
- Document : PowerPoint de soutenance
- Soutenance 9

## 2.2 Semestre 10

### Tâche 9 : Développement et entraînement du modèle de la nouvelle méthode

- Date de début : 4 janvier
- Date de fin : 31 janvier
- Duree : 4 semaines
- Description : Mise en place d'un modèle 3D CNN pour la détermination du rythme cardiaque en analysant une séquence vidéo puis entraînement de ce dernier, en vue d'une implémentation dans le Framework. (inclut tests).
- Livrable : Le modèle entraîné

### Tâche 10 : Implémentation de la nouvelle méthode dans le Framework

- Date de début : 31 janvier
- Date de fin : 21 février
- Duree : 3 semaines
- Description : Implémentation du modèle entraîné dans le Framework, en respectant les conventions du Framework. (inclut tests).
- Livrable : Framework avec une nouvelle méthode

### Tâche 11 : Elaboration du guide d'implémentation d'une nouvelle méthode

- Date de début : 21 février
- Date de fin : 7 mars
- Duree : 2 semaines
- Description : Rédaction du guide permettant à n'importe quel utilisateur d'implémenter ses propres méthodes dans le Framework.
- Livrable : Un guide (Format Jupyter Notebook)

### Tâche 12 : Elaboration de la démonstration générale

- Date de début : 7 mars
- Date de fin : 21 mars
- Duree : 2 semaines
- Description : Développement du notebook de démonstration du Framework.
- Livrable : Un guide (Format Jupyter Notebook)

### Tâche 13 : Rapport final

- Date de début : 21 mars
- Date de fin : 1 avril
- Duree : 2 semaines
- Description : Rédaction du rapport final.
- Livrable : Rapport Final S10

### Tâche 14 : Soutenance finale

- Date de début : 1 avril
- Date de fin : 8 avril
- Duree : 1 semaine

- Description : Préparation de la soutenance finale.
- Livrable : PowerPoint
- Soutenance S10

# B

# Cahier de Specification

## 1 spécifications Fonctionnelles

Dans ce projet, 2 fonctions et 3 tâches majeures seront développées, afin de proposer une version améliorée du Framework pyVHR.

### 1.1 Fonctionnalités à développer

#### Fonction 1 : Conception et Implémentation d'une nouvelle méthode rPPG

##### Présentation de la fonction 1 :

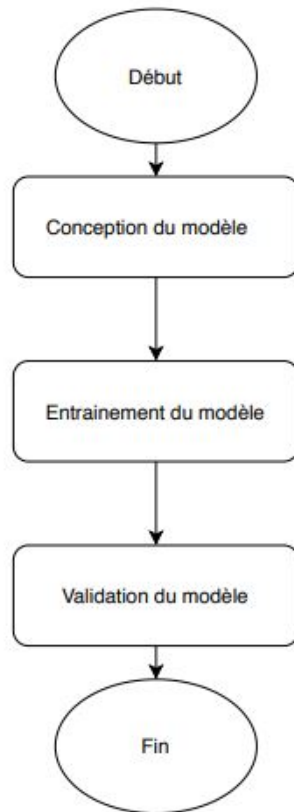
- Nom de la fonction : 3dcnn
- Role : Conception et entraînement d'un modèle 3D CNN permettant de convertir une séquence d'images contenant l'extraction d'un visage en un signal représentant le rythme cardiaque du sujet.
- Priorité : Primordiale

##### Description de la fonction 1 :

A l'aide d'une fonction préexistante dans le Framework pyVHR, nous pouvons extraire le visage d'un sujet sur la vidéo à analyser. La méthode devra proposer à partir de cette extraction, une analyse complexe permettant de transformer ce visage en l'information suivante : quel est le rythme cardiaque du sujet ? Une conception et un entraînement d'un modèle de Deep Learning (3D CNN) sera effectué avec des bases de données publiques et leurs vérités terrains associées. Le modèle subira également une phase de validation des résultats en vue d'une intégration dans le Framework.

##### Description précisée de la fonction 1 :

- Entrées : Un ensemble de séquence vidéo associé à des vérités terrains.
- Sorties : Un modèle entraîné donnant une estimation du rythme cardiaque correspondant aux variations de couleur du visage du sujet.



**Figure B.1** – *Diagramme de la fonction 1 : Conception et Implémentation d'une nouvelle méthode rPPG*

## Fonction 2 : Intégration d'une nouvelle méthode rPPG dans le Framework

**Présentation de la fonction 2 :**

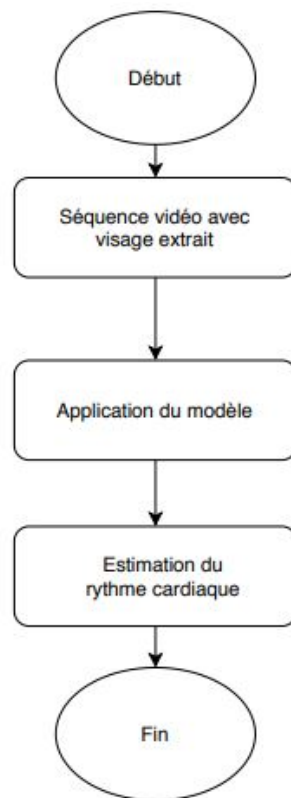
- Nom de la fonction : 3dcnn
- Role : Intégration d'une nouvelle méthode dans le Framework.
- Priorité : Primordiale

**Description de la fonction 2 :**

Pour une intégration parfaite dans le Framework, les choix de paramètres doivent pouvoir être choisis via le fichier de configuration déjà existant. Une configuration par défaut doit être également proposée dans le fichier de configuration. Le modèle entraîné sera intégré dans le Framework, en respectant le formalisme et les conventions imposés par le fichier base.py (l'interface générale des méthodes rPPGs du Framework).

**Description précisée de la fonction 2 :**

- Entrées : Une séquence vidéo avec une extraction du visage.
- Sorties : Une estimation du rythme cardiaque correspondant aux variations de couleur du visage du sujet, le tout, utilisable et compatible avec le reste du Framework.



**Figure B.2** – *Diagramme de la fonction 2 : Intégration d'une nouvelle méthode rPPG dans le Framework*

## 1.2 Tâches à réaliser

Tâche 1 : Implémentation d'un notebook de démonstration pour une prise en main générale du Framework.

### Présentation de la tâche 1 :

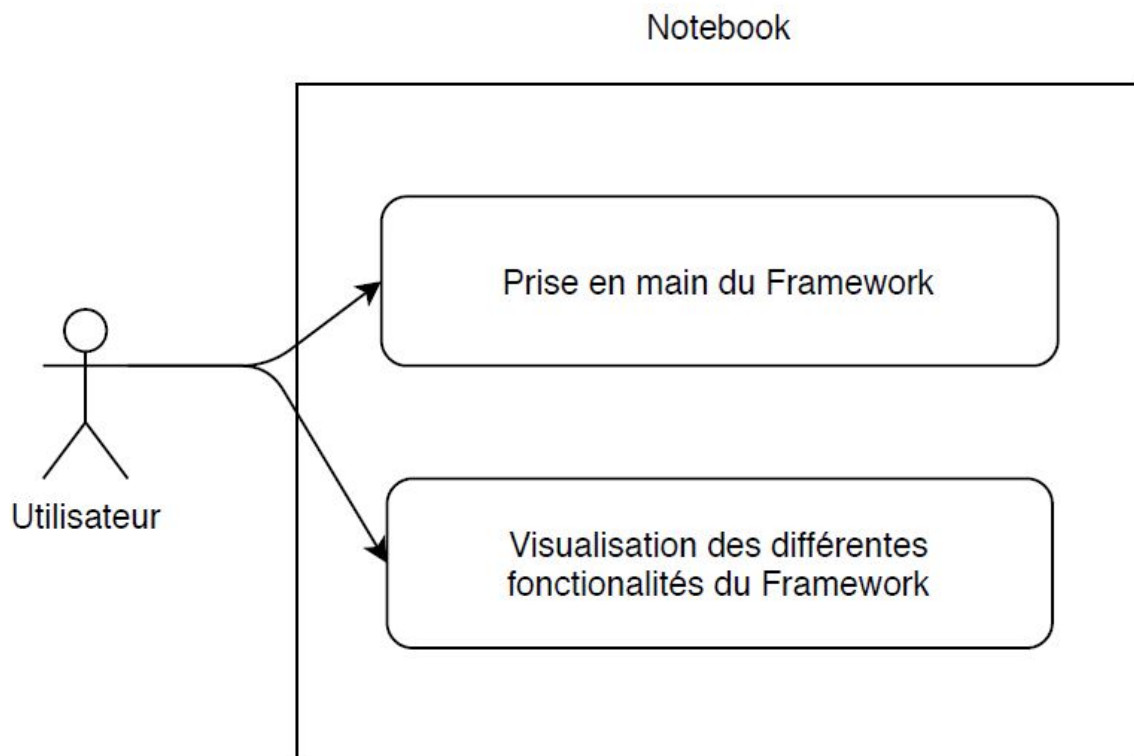
- Nom de la fonction : Configurations
- Role : Proposer aux utilisateurs du Framework pyVHR, une démonstration complémentaire aux démonstrations déjà existante. Afin, de pouvoir facilement prendre en main le Framework et l'utiliser sereinement.
- Priorité : Primordiale

### Description de la tâche 1 :

Le Framework propose actuellement plusieurs démonstrations sous la forme de “notebook” python, afin d'aider les utilisateurs du Framework à visualiser les différentes fonctionnalités de ce dernier et également pour proposer un mini-guide de prise en main de pyVHR. Cependant, ces derniers ne montrent pas encore l'utilisation de certaines fonctions, pourtant essentielles comme le paramétrage du fichier de configuration. Ainsi, l'objectif de notre notebook sera de venir implémenter un nouveau notebook, complémentaire et enrichissant les notebooks déjà existants. L'accent sera donc mis sur les fichiers de configurations en priorité.

### Description précisée de la tâche 1 :

- Entrées : Néant
- Sorties : Une aide à la prise en main du Framework.



**Figure B.3** – Diagramme de la tâche 1 : Implémentation d'un notebook de démonstration pour une prise en main générale du Framework

Tâche 2 : Implémentation et rédaction d'un guide ("notebook") pour l'intégration d'une nouvelle méthode rPPG par les futurs utilisateurs.

#### Présentation de la tâche 2 :

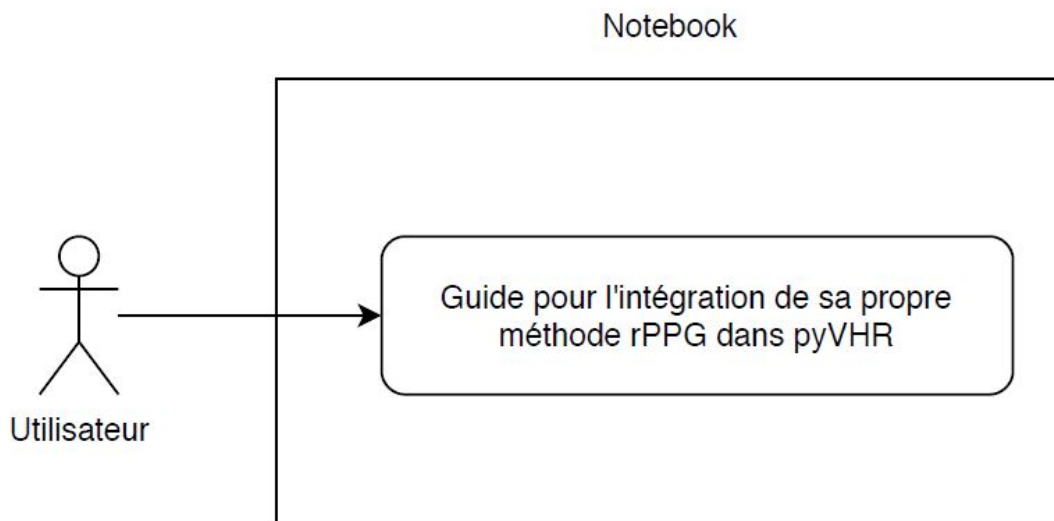
- Nom de la fonction : Add a New Method
- Role : Proposer aux utilisateurs un guide pour intégrer facilement leur propre méthode dans le Framework pyVHR.
- Priorité : Secondaire

#### Description de la tâche 2 :

L'objectif du Framework est que n'importe puisse ajouter sa méthode de rPPG dans ce dernier, par exemple pour comparer facilement sa méthode avec d'autres méthodes, dans les mêmes conditions expérimentales. Ainsi, avec mon retour d'expérience suite à l'implémentation de la fonction. Un guide sera créé afin de servir de tutoriel aux futurs utilisateurs pour l'implémentation de leur propre méthode rPPG.

#### Description précisée de la tâche 2 :

- Entrées : Connaissances acquises lors de l'implémentation de la fonction.
- Sorties : Un tutoriel pour l'ajout de méthode rPPG dans le Framework.



**Figure B.4** – Diagramme de la tâche 2 : Implémentation et rédaction d'un guide ("notebook") pour l'intégration d'une nouvelle méthode rPPG par les futurs utilisateurs

### Tâche 3 : Améliorations et Documentation générale.

#### Présentation de la tâche 3 :

- Nom de la fonction : Néant
- Role : Proposer une documentation supplémentaire et des améliorations si nécessaire au Framework pyVHR.
- Priorité : Primordiale et facultatif

#### Description de la tâche 3 :

Le Framework est encore jeune. Ainsi, proposer une documentation au Framework en ajoutant des explications dans le fichier de configuration et en améliorant les commentaires de certaines parties de code est intéressant. Par ailleurs, des améliorations de code existant peuvent être apportées si c'est nécessaire et/ou pertinent pour le Framework.



## 2 Spécifications non fonctionnelles

### 2.1 Contraintes de développement et conception

Les contraintes de développement sont les suivantes :

- Matériels : pas de nécessité, mais une carte graphique NVIDIA permet d'utiliser CUDA permettant d'exécuter plus rapidement certaines tâches.
- Langage de programmation : Python 3.6
- Logiciels et bibliothèques : Les dépendances du Framework pyVHR sont à conserver. Le projet s'inscrit comme la suite de ce dernier.

### 2.2 Contraintes de fonctionnement et d'exploitation

Les contraintes de fonctionnement et d'exploitation sont assez flexible. En effet, l'utilisateur peut gérer les paramètres d'exécution via un fichier de configuration "sample.cfg". Par ailleurs, il est nécessaire de disposer d'au moins un des ensembles de données suivant : 'PURE', 'UBFC1', 'UBFC2', 'LGI-PPGI', 'COHFACE', 'MAHNOB' pour utiliser le Framework. Ces ensembles de données sont publique. Cependant, cette nécessité peut être levée si l'utilisateur intègre sa propre de base de données dans le Framework.

#### Performances

Du point de vue de l'utilisateur, les performances temporelles sont très variables. En effet, cela dépend de la taille, la durée et de la qualité de la vidéo à analyser. Par ailleurs, le processus de détection et de l'extraction du visage du sujet de la vidéo peut prendre un peu de temps (de plusieurs secondes (sur gpu) à plusieurs minutes (sur cpu)). Cependant, une optimisation existe déjà dans le Framework permettant de sauvegarder ce processus à la première lecture de la vidéo. Ainsi, aux lectures suivantes ce processus peut être passé et donc faire gagner du temps à l'utilisateur.

Du point de vue de l'environnement, le projet étant codé en Python, il n'y a pas de dépendance à un système d'exploitation spécifique. En effet, python est un langage multi-plateforme. Comme dit précédemment, le Framework est à la fois compatible pour fonctionner complètement sur le processeur et également peut fonctionner en hybride processeur / carte graphique NVIDIA.

#### Modes de fonctionnement

Pour utiliser le Framework, il suffit de l'installer et de l'importer dans son projet python (comme une bibliothèque classique). Pour installer le projet, deux options se proposent à l'utilisateur. La première option consiste à l'installer directement sur son environnement via la commande "pip install pyvhr". La deuxième option consiste à télécharger le code source du projet sur GitHub (<https://github.com/phuselab/pyVHR>) puis de faire l'installation via la commande suivante : "python setup.py install". La mise sous-tension se fait via l'appel aux fonctions du Framework et l'arrêt à la fin de l'exécution de la fonction.

### Capacités

Il n’y a pas de capacités prédéfinies dans le Framework. Cependant, l’utilisateur peut être contraint par les capacités de sa machine. En effet, le Framework manipule des fichiers volumineux à stocker et analyser. Ainsi, la mémoire de stockage et la mémoire vive de la machine de l’utilisateur peuvent définir une limite pour l’utilisateur.

### Contrôlabilité

Les utilisateurs peuvent suivre l’exécution via des affichages de message dans la console. En effet, certaines fonctions possèdent des “print()” pour suivre leur évolution dans le traitement, quand ce dernier est assez long.

### Sécurité

Le Framework est utilisé en local par l’utilisateur, il n’y a donc aucun contrôle d’accès. Par ailleurs, l’utilisateur est libre de faire évoluer le projet comme il le souhaite. Ce projet open-source n’est donc pas soumis à des conditions de sécurité précises, en tout cas dans sa version actuelle.

## 1 Rappel du diagramme de classe

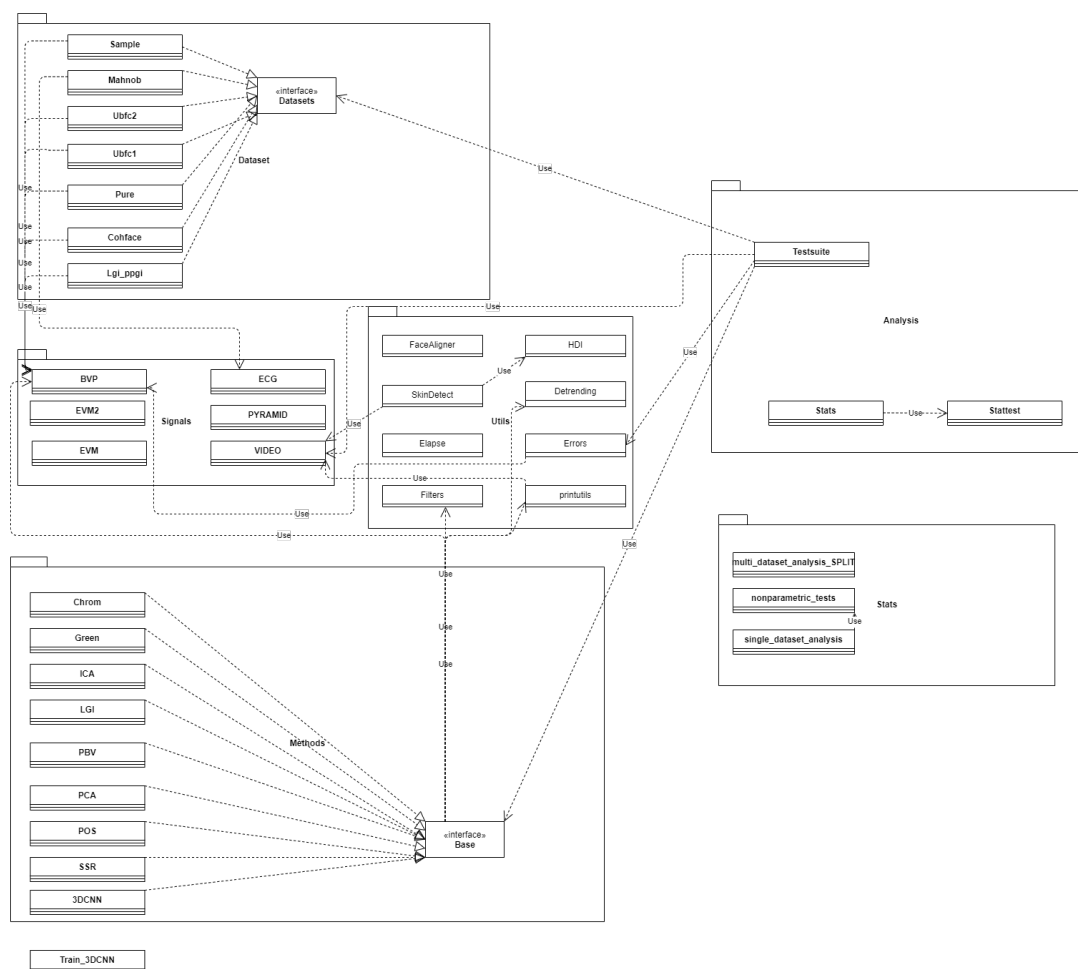


Figure C.1 – Diagramme de classe

## 2 Description détaillée du diagramme de classe

### Package Dataset

- Datasets : Définies l'objet datasets. Permet de mutualiser les traitements avec des ensembles de données d'origines diverses.
- Sample : Classe permettant l'analyse et l'extraction des données de l'ensemble Sample.
- Mahnob : Classe permettant l'analyse et l'extraction des données de l'ensemble Mahnob.
- Ubfc1 : Classe permettant l'analyse et l'extraction des données de l'ensemble Ubfc1.
- Ubfc2 : Classe permettant l'analyse et l'extraction des données de l'ensemble Ubfc2.
- Pure : Classe permettant l'analyse et l'extraction des données de l'ensemble Pure.
- Cohface : Classe permettant l'analyse et l'extraction des données de l'ensemble Cohface.
- Lgi\_ppgi : Classe permettant l'analyse et l'extraction des données de l'ensemble Lgi\_ppgi.

### Package Method

- Base : Définit l'objet méthode (Base). Permet de mutualiser les traitements des différentes méthodes
- Chrom : Classe permettant de définir la méthode Chrom.
- Green : Classe permettant de définir la méthode Green.
- ICA : Classe permettant de définir la méthode ICA.
- LGI : Classe permettant de définir la méthode LGI.
- PBV : Classe permettant de définir la méthode PBV.
- PCA : Classe permettant de définir la méthode PCA.
- POS : Classe permettant de définir la méthode POS.
- SSR : Classe permettant de définir la méthode SSR.
- 3DCNN : Classe permettant de définir la méthode 3DCNN.

### Package Analysis

- Testsuite : Définition de la classe permettant de définir une comparaison de plusieurs méthodes sur un même jeu de données.
- Stats : Classe définissant l'objet statistique.
- Stattest : Classe permettant de faire une analyse statistique des résultats.

### Package Signals

- BVP : Classe pour le traitement du signal BVP.
- EVM2 : Classe définissant des indicateurs de performances.
- EVM : Classe définissant des indicateurs de performances.
- ECG : Classe pour la réalisation d'électrocardiogramme.
- PYRAMID : Classe permettant la représentation gaussienne d'une vidéo.
- VIDEO : Classe définissant l'objet Vidéo, pour le traitement vidéo au sein du Framework

### Package Utils

- FaceAligner : Classe permettant l'alignement d'un visage d'une vidéo.
- SkinDetect : Classe permettant de détecter la peau d'une vidéo.
- Elapse : Définition d'un chronomètre
- Filters : Définitions de filtres.
- HDI : Calculs de densités.
- Detrending : Calcul de la variabilité de la fréquence cardiaque.
- Errors : Définitions des taux d'erreurs des méthodes.
- Printutils : Définition des outils d'affichages.

### Package Stats

- `Multi_dataset_analysis_split` : Création de graphiques impliquant plusieurs ensembles de données.
- `Nonparametric_tests` : Définitions de méthodes statistiques.
- `Single_dataset_analysis` : Création de graphiques impliquant un seul ensemble de données.

## 3 Description de la structure du projet

Le Framework suit la même organisation que décrite par le diagramme de classe. Le Framework est donc organisé en 6 packages : « analysis », « datasets », « methods », « resources », « signals » ; « stats » et « utils ». Ces 6 packages se partagent donc les 38 classes décrites dans la partie précédente.

# Bibliographie

- [1] Giuseppe Boccignone and Donatello Conte and Vittorio Cuculo and Alessandro D'Amelio and Giuliano Grossi and Raffaella Lanzarotti. *An Open Framework for Remote-PPG Methods and their Assessment*. IEEE Access, <https://doi.org/10.1109/access.2020.3040936>, 2020.
- [2] Frédéric Bousefsaf, Alain Pruski, Choubeila Maaoui. *3D convolutional neural networks for remote pulse rate measurement and mapping from facial video*. Applied Sciences, vol. 9, n° 20, 4364 , 2019.
- [3] Gerard de Haan and Vincent Jeanne. *Robust pulse-rate from chrominance-based rPPG*. IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, MONTH 2013.
- [4] R. M. FOUAD, OSAMA A. OMER, AND MOUSTAFA H. ALY. *Optimizing Remote Photoplethysmography Using Adaptive Skin Segmentation for Real-Time Heart Rate Monitoring*. IEEE Access, Received April 27, 2019, accepted June 8, 2019, date of publication June 12, 2019, date of current version June 26, 2019.
- [5] Wenjin Wang, Bert den Brinker, Sander Stuijk, and Gerard de Haan. *Algorithmic Principles of Remote-PPG*. IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING, VOL. PP, NO. 99, MONTH 2016.
- [6] Rencheng Songa, Senle Zhanga, Juan Chenga, Chang Lia, Xun Chenb. *New insights on super-high resolution for video-based heart rate estimation with a semi-blind source separation method*. Computers in Biology and Medicine, November 2019.
- [7] D. Botina-Monsalve, Y. Benezeth, R. Macwan, P. Pierrart, F. Parra, K. Nakamura, R. Gomez, J. Miteran. *Long Short-Term Memory Deep-Filter in Remote Photoplethysmography*. IEEE Xplore, 2020.
- [8] Eugene Lee Evan Chen Chen-Yi Lee. *Meta-rPPG : Remote Heart Rate Estimation Using a Transductive Meta-Learner*. Institute of Electronics, National Chiao Tung University, 2020.
- [9] W. Verkruyse et al. *Remote plethysmographic imaging using ambient light*. Opt. Exp., vol. 16, no. 26, pp. 21 434–21 445, Dec. 2008.
- [10] M. Hulsbusch. *An image-based functional method for opto-electronic detection of skin perfusion*. Ph.D. dissertation (in German), Dept. Elect. Eng., RWTH Aachen Univ., Aachen, Germany, 2008.
- [11] M. Lewandowska et al. *Measuring pulse rate with a webcam - a noncontact method for evaluating cardiac activity*. Proc. Federated Conf. Comput. Sci. Inform. Syst. (FedCSIS), Szczecin, Poland, Sept. 2011.

- [12] M.-Z. Poh et al. *Advancements in noncontact, multiparameter physiological measurements using a webcam*. IEEE Trans. Biomed. Eng., vol. 58, no. 1, pp. 7–11, Jan. 2011.
- [13] G. de Haan and A. van Leest. *Improved motion robustness of remote-PPG by using the blood volume pulse signature*. Physiol. Meas., vol. 35, no. 9, pp. 1913–1922, Oct. 2014.
- [14] W. Wang et al. *A novel algorithm for remote photoplethysmography : Spatial subspace rotation*. IEEE Trans. Biomed. Eng., vol. 63, no. 9, pp. 1974–1984, Sept. 2016.

# Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos

Florian GIGOT

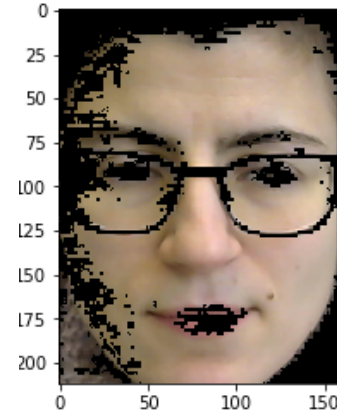
Encadrement : Donatello CONTE



En collaboration avec Polytech

## Objectifs

- Découvrir et implémenter des méthodes rPPGs
- Prise en main et amélioration du Framework pyVHR
- Rédaction de plusieurs guides (Jupyter Notebook)



Extraction d'une zone d'intérêt

## Mise en œuvre

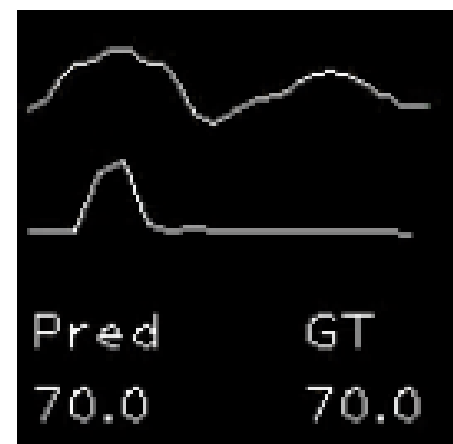
Au cours du premier semestre de mon projet de Recherche et Développement, j'ai mené essentiellement des recherches pour mieux appréhender le sujet et préparer le second semestre. Le second semestre est lui dédié au développement de la solution.

CHROME PBV  
POS PCA  
GREEN-RED  
RNN ICA  
LGI GREEN  
SSR  
CNN

Méthodes rPPGs

## Résultats attendus

- Implémentation d'une nouvelle méthode rPPG
- Réalisation de guides pour les futurs utilisateurs
- Axes d'améliorations du Framework pyVHR



Estimation du rythme cardiaque d'un individu





# Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos

Florian GIGOT

Encadrement : Donatello CONTE

## Objectifs

- Découvrir et implémenter des méthodes rPPGs
- Prise en main et amélioration du Framework pyVHR
- Rédaction de plusieurs guides (Jupyter Notebook)

## Mise en œuvre

Au cours du premier semestre de mon projet de Recherche et Développement, j'ai mené essentiellement des recherches pour mieux appréhender le sujet et préparer le second semestre. Le second semestre est lui dédié au développement de la solution.

## Résultats attendus

- Implémentation d'une nouvelle méthode rPPG
- Réalisation de guides pour les futurs utilisateurs
- Axes d'améliorations du Framework pyVHR



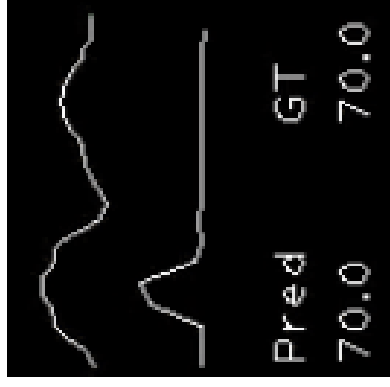
En collaboration avec Polytech



Extraction d'une zone d'intérêt

CHROME PBV  
POS PCA  
GREEN-RED  
RNN  
LGI ICA  
GREEN  
SSR  
CNN

Méthodes rPPGs



Estimation du rythme cardiaque d'un individu

# Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos

## Résumé

Le sujet de Projet de Recherche et Développement (PRD) est « Développement d'algorithmes pour l'estimation du rythme cardiaque par analyse de vidéos ». Ce sujet se base sur le Framework python pyVHR traitant des méthodes rPPGs. Ainsi, lors de ce projet j'ai étudié les méthodes rPPGs et pris en main pyVHR. Afin, d'implémenter une nouvelle méthode rPPG dans ce dernier et proposer un retour d'expérience via des guides pour les futurs utilisateurs.

## Mots-clés

Estimation du rythme cardiaque à distance, pyVHR, Méthodes rPPGs, Traitement vidéo, Traitement du signal, Deep Learning, Analyse statistique

## Abstract

The subject of Research and Development Project (RDP) is “Development of algorithms for remote heart rate estimation, by video analysis”. This subject is based on the pyVHR python framework dealing with rPPGs methods. Thus, during this project I studied rPPGs methods and took in hand pyVHR. In order to implement a new rPPG method in the framework and to propose a feedback via guides for the future users.

## Keywords

Remote heart rate estimation, pyVHR, rPPGs methods, Video processing, Signal processing, statistical analysis

**Entreprise**

Polytech



**Tuteur entreprise**

Donatello CONTE

**Étudiant**

Florian GIGOT (DI5)

**Tuteur académique**

Donatello CONTE