**Département électrique**

**Projet de fin d’année II**

**Implémentation d’une technique d’identification faciale sur une carte Raspberry PI 3**

Présenté par

**Iheb Helali**

Classe

**2AGE3**

**2AGE2**

Réalisé à

**ENIT (Ecole National d’Ingénieur de Tunis)**

**Soutenu le 24/04/2018**

**Devant le Jury :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Président** | **:** | **M** | **Faouzi BOUANI** |
| **Rapporteur** | **:** | **M** | **Samir SAKARANI** |
| **Encadrant ENIT** | **:** | **Mme** | **Monia Turki** |

Année Universitaire : 2018/2019

Remerciements

C'est avec grand honneur que nous vous proposons cette page en signe de remerciement et de gratitude à toutes les personnes qui nous ont soutenus, de près ou de loin, dans la réalisation de ce modeste projet.

Merci à mon enseignante responsable Monia Turki pour le temps consacré à ce travail. Vos observations étaient d'une grande importance pour conduire mon travail jusqu'au bout. Vous avez tout le temps à mes côtés pour me supporter et me motiver. Votre encadrement a été l'occasion de découvrir le domaine de la recherche.

Table des matières

**Tables des figures iv**

**Liste des tableaux v**

**Introduction générale 1**

**1 présentation du notre projet et les limites d’un projet réalisé précédemment**

1.1 Introduction du chapitre 1 2

1.2 Une description du projet déjà réalisé 2

1.2.1 Description du projet 2

1.2.2 Principaux modules constituant le système d’accès 3

1.2.3 Limite du système 4

1.3 Description du notre projet et solutions adoptées 4

1.3.1 Les solutions adoptées 4

1.3.2 Cahier de charge 4

1.3.3 Matériel utilisé 5

1.3.4 Les techniques et les logiciels utilisés 6

1.3.4.1 python 3 6

1.3.4.2.Opencv 3 6

1.3.4.3 base de donnée (gérer par la langage SQL) 7

1.3.5 Conclusion 8

**2 Algorithmes de traitement pour l’identification faciale**

2.1 Introduction 9

2.2 Description de l’algorithme de Viola Jones 9

2.2.1 Le descripteur Pseudo-Haar 10

2.2.2 Le calcul des descripteurs Pseudo-Haar 10

2.2.2.1 Calcul préliminaire 11

2.2.2.2 Image intégrale 12

2.2.2.3 Classification de l’image 12

2.2.2.4 Conclusion 14

2.2.3 Expérience et résultats 14

2.3 Description de l’algorithme LBP (Local Binary Pattern) 15

2.3.1 Le principe de l’algorithme LBP 16

2.3.2 Expérience et résultats 18

2.4 La comparaison entre le LBP et Haar Cascade 18

2.5 Conclusion 19

**3 Réalisation du système d’identification faciale**

3.1 Introduction 20

3.2 L’installation du Raspbian et la configuration de la raspberry 20

3.2.1 L’installation du système d’exploitation (Le raspbian) 20

3.2.2 Configuration du réseau sur la raspberry pi 3 22

3.2.3 L’installation de l’open cv et python 3 23

3.2.4 L’installation du SQL 24

3.3 L’implémentation d’algorithme 26

3.3.1 la base de donnée 26

3.3.1.1 création de la base de donnée pour le LBP 26

3.3.1.2 création de la base de donnée SQL pour le Haar\_Cascade 27

3.3.2 Test et validation 27

3.3.2.1 Mise en œuvre de l’identification par Haar\_cascade 27

3.3.2.2 Mise en œuvre de l’identification par le LBP 29

3.3.2.3 Les tests pour le Haar\_ cascade (le streaming) 30

3.3.2.4 Problèmes rencontrées 30

3.4 Conclusion 30

**Conclusion générale 31**

**Références bibliographiques 32**

Table des figures

Figure 1.1 : Le système de contrôle d’accès et de surveillance 3

Figure 2.1 : Exemple de fonctions rectangulaires 10

Figure 2.2 : Un exemples réelle et idéale des fonctions rectangulaires 11

Figure 2.3 : La méthode de calcul de l’image intégrale 12

Figure 2.4 : Représentation schématique de la cascade pour la détection de visage 14

Figure 2.5 : La schématisation de processus du LBP 15

Figure 2.6 : Etapes de calcul de l'opérateur LBP. 17

Figure 3.1 : La logiciel  « SD card formater»  20

Figure 3.2 : Le Flashing du Raspbian sur la carte SD 21

Figure 3.3 : Le Desktop de la raspberry 21

Figure 3.4 : Les étapes de la connexion du raspberry à distance 22

Figure 3.5 : L’interface phpmyadmin 25

Figure 3.6 : Dataset trainée manuellement 26

Figure 3.7 : La base de donnée test 27

Figure 3.8 : Base de données sur la Raspberry 27

Figure 3.9 : Les tests faites en streaming 27

Liste des tableaux

Tableau 1.1 : les différentes composantes utilisées pour programmer 5

Tableau 2.1 : Analyse des performances de l'algorithme « haar cascade » 12

Tableau 2.2 : Analyse des performances de l'algorithme « LBP » 18

Introduction générale

La détection faciale est une technique qui utilise plusieurs méthodes pour détecter un visage à travers soit une vidéo, soit une photo. De nos jours il y a de nombreux appareils et sites Web qui utilisent la détection de visage pour identifier une personne. A titre d’exemple, nous citons l’application Facebook qui propose de taguer un ami chaque fois que son visage a été reconnu dans une photo téléchargée. Cette technique est devenue accessible aux utilisateurs de l’informatique et ne nécessite qu’une machine équipée d’un appareil photo de haute définition et d’un logiciel de traitement de l’image.

La détection faciale est une technologie informatique qui détermine l’emplacements et tailles de visages humains d'une façons arbitraire (numérique). On peut classer cette détection comma Elle peut être considérée comme une dérivation de la détection de classe d'objet. Dans la détection de classe d'objet, la tâche est de trouver les emplacements et les tailles de tous les objets dans une image qui appartient à une classe donnée.

Pour ce projet de fin d’année, nous sommes demandés d’implémenter une technique d’identification faciale sur une carte Raspberry PI. Nous visons ce fait un objectif sécuritaire. Il s’agit par exemple de limiter l’accès à un endroit à un nombre fini de personnes. L’accès à un endroit à un nombre fini de personnes. L’accès est autorisé par une reconnaissance faciale de la personne demandant l’accès. Les caractéristiques faciales de la personne doivent être enregistrés auparavant dans une base de données.

Ce Rapport est composé de trois chapitres. Dans le premier chapitre, nous présentons l'unité de traitement et les outils nécessaires utilisées telles que l’opencv et python. Dans le deuxième chapitre, nous présentons les algorithmes utilisés pour la détection faciale. Le troisième chapitre sera consacré à la configuration de la carte et aux tests des fonctionnalités du système de surveillance global implémenté. Une conclusion générale clôturera le rapport.

1. Chapitre 1 : Présentation du notre projet et les limites du projet réalisé auparavant

1.1 Introduction du chapitre 1

De nombreuses études récentes prennent en compte le problème de la sécurité et présentent le développement de nouvelles technologies en faveur des solutions qu'elles proposent. Le visage d’un être humain contient beaucoup d’informations sur l’identité et l'état émotionnel de la personne. La reconnaissance des visages est un problème intéressant, et il a un grand impact sur des applications importantes dans de nombreux domaines tels que l'identification pour les forces de l'ordre, authentification bancaire et l'accès au système de sécurité et l'identification personnelle. Tout au long de ce chapitre, nous décrirons en détail la méthode de contrôle d'accès et de surveillance de tous les habitats ayant fait l'objet d'un PFA [1], puis présenterons le contexte de notre projet pour aborder le projet déjà concrétisé en mettant en évidence les caractéristiques que notre système doit garantir

**1.2 Une description du projet déjà réalisé**

**1.2.1 Description du projet**

Le visage d’un être humain contient beaucoup d’informations sur l’identité et l'état émotionnel de la personne. La reconnaissance des visages est souvent une opération très demandée dans diverses applications dans un nombre de domaines croissant. Nous citons à titre d’exemple l’identification des forces de l'ordre, l’identification des personnes pour l’accès aux smartphones ou aux différentes applications permettant la gestion des comptes courants bancaires ou la messagerie électronique.

Notre travail dans le cadre de projet de Fin d’Année II, les recherches articule autour de la reconnaissance du visage d’une personne. La reconnaissance faciale est en général réalisée en trois étapes : la maitrise de la représentation du visage, l’extraction de caractéristiques du visage et reconnaissance de la personne par comparaison de ces caractéristiques à une base de données contenant un ensemble de visage annotés.

Le système de contrôle d’accès et de surveillance des habitats réalisé au sein de l’Unité Signaux et Systèmes de U2S l’ENIT [2] se décomposent en deux sous-systèmes :

* Un sous-système interne
* Un sous-système externe

La figure 1.1 illustre le système complet et résume les différentes parties du système

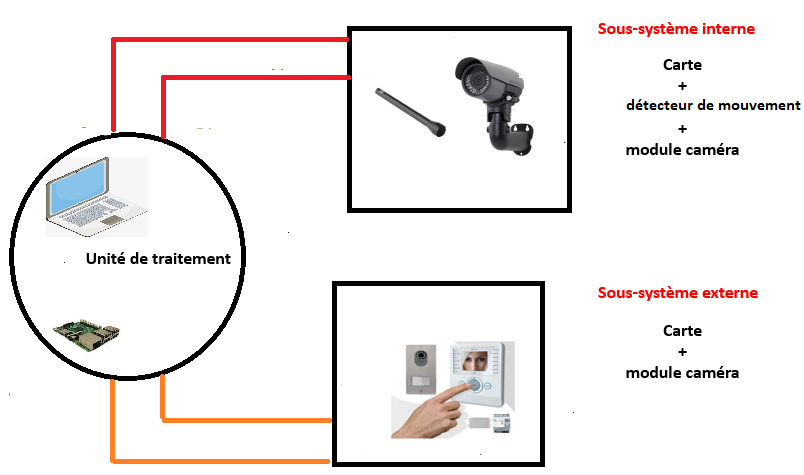


Figure 1.1 Le système de contrôle d’accès et de surveillance [2]

1.2.2 Modules principaux qui forment le système d'accès

* ISS (Internas System) est un sous-système interne qui assure la surveillance vidéo à l'intérieur de la maison. Ce système envoie automatiquement un sms dés qu’il détecte un intrus à l’intérieur de la maison. Ce système présente 5 composantes essentiels qui sont cité par la suite :
* Un microprocesseur 8 bits,
* Un capteur PIR de déplacement,
* Un modem GSM, un commutateur d'entrée,
* Commutateurs de sortie
* Un terminal de surveillance vidéo sur réseau intégré (ENVMT).).
* ESS est un sous-système externe qui offre un moyen de contrôle d'accès fondé sur la reconnaissance faciale. Ce sous-système externe capture une image de la personne qui demande l'accès à l'habitat et la transmet à une cellule de contrôle pour vérifier si la personne est bien autorisée à le pénétrer. Il se constitue de 4 éléments principaux :
* Une caméra,
* Un capteur de déplacement,
* Un microcontrôleur 8 bits et
* Un module de reconnaissance faciale

1.2.3 Limite du système

Les limites constatées du système réalisées sont essentiellement les suivantes :

* Utilisation d’un ordinateur pour le traitement. Une solution n’est pas souvent pratique et couteuse et limite le nombre des applications.
* Utilisation du logiciel de traitement MATLAB qui est souvent confrontés à plusieurs obstacles : le cout d’accès à la licence, n’est pas portable, etc…

1.3 Description de notre projet et solutions adoptées

1.3.1 Les solutions adoptées

Choix de l’unité de traitement : nous avons opté dans notre projet le remplacement du PC par une carte Raspberry.Pi3 qui est capable de réaliser toutes les tâches que vous voulez qu'un ordinateur puisse accomplir, beaucoup moins chers et offrant les mêmes opportunités de développement et d’exécution.

Nous avons opté dans notre projet le remplacement du MATLAB par PYTON3 qui est un logiciel open source. Parmi les intérêts de l’utilisation de python 3 est sa bonne conception. Il présente une facilité par rapport à d’autres langages pour transformer vos idées en codes. De plus, Python est livré avec des bibliothèques standard étendues et dispose de puissants types de données tels que des listes, des ensembles et des dictionnaires qui facilitent l’organisation de vos données. Contrairement au MATLAB, Les algorithmes sont propriétaires, ce qui signifie que vous ne pouvez pas voir le code de la plupart des algorithmes que vous utilisez et devez-vous assurer qu'ils ont été implémentés correctement

1.3.2 Cahier des charges

Dans le cadre du projet de fin d’année II, notre travail est essentiellement d’implémenter et tester l’algorithme de reconnaissance faciale des images extraites des vidéos en temps réels. Le système de contrôle d’accès doit être en mesure d’ajouter une nouvelle personne à la base de données. Le SQL sera adopté pour la gestion de la base de donnée.

1.3.3 Matériel utilisé

Tableaux 1.1 Les différentes composantes utilisées pour programmer [4]

|  |  |
| --- | --- |
| Une carte Raspberry pi3  RÃ©sultat de recherche d'images pour "raspberry pi 3" | Lancé en 2013, le Raspberry Pi est un nano ordinateur détaillé à environ 35 $ USD. La dernière version, la Raspberry Pi 3, possède un processeur ARM 1200 MHz, 4 ports USB, 1 port pour la connexion internet, un boitier Bluetooth, un port HDMI, un lecteur de carte Micro S.A. représente la carte sur laquelle on va implémenter notre système. |
| Un ordinateur | Nous devons utiliser un ordinateur pour préparer une carte micro avec une image de système d'exploitation Raspbian pour le Pi. |
| I/O Devices | Un clavier USB standard et une souris USB sont requis. |
| MicroSD Card | Une carte microSD avec au moins 32 Go de stockage est nécessaire. Elle est utilisée pour le stockage secondaire pour le Pi. Une carte de classe 10 est recommandée pour le transfert de données.La vitesse avec la classe 10 est intéressante. |
| Power Supply | Pour tous les modèles Raspberry Pi, une unité d'alimentation micro USB de 5 V est requise. La capacité actuelle recommandée du bloc d'alimentation pour Raspberry Pi 3 modèle B est de 2,5 ampères. Pour tous les autres modèles, une alimentation de 2 ampères est amplement suffisante. |
| Monitor | Un moniteur HDMI ou VGA. Pour un moniteur HDMI, vous avez besoin d'un câble HDMI mâle à mâle (voir la figure 1-6). Il est généralement emballé avec le moniteur HDMI. |
| Camera | Pour capturer des images à l’aide d’une webcam USB standard. Raspberry Pi 3 a quatre Ports USB. Vous pouvez utiliser l'un de ceux-ci pour connecter une webcam au Raspberry Pi Une webcam Star. |

1.3.4 les techniques et les logiciels utilisés

1.3.4.1 Python3

Van Rossum est l'auteur principal du langage de programmation Python. Il joue un rôle central dans l'orientation du développement, de l'amélioration et de la promotion évolution du langage de programmation Python. Il détient le titre de dictateur bienveillant pour la vie pour Python. Il travaille actuellement (à compter de février 2017) pour Dropbox et consacre presque la moitié de son temps à poursuivre le développement du langage de programmation Python. Parmi les intérêts de l’utilisation de de python 3 sa bonne conception, il est plus facile que d’autres langues de transformer vos idées en code. De plus, Python est livré avec des bibliothèques standard étendues et dispose de puissants types de données tels que des listes, des ensembles et des dictionnaires. Ceux-ci aident vraiment à organiser vos données. Il est gratuit, notre code peut être exécuté partout. De plus, cela fonctionne sous Windows, Linux et OS. Définitions de classe et de fonction. Les fonctions et les classes peuvent être définies n'importe où.

1.3.4.2 Opencv 3

OpenCV (Open Source Computer Vision Library) est une bibliothèque de logiciels open source de vision informatique et d’apprentissage automatique. OpenCV a été conçu pour fournir une infrastructure commune aux applications de vision par ordinateur et pour accélérer l'utilisation de la perception de la machine dans les produits commerciaux. OpenCV étant un produit sous licence BSD, il est facile pour les entreprises d’utiliser et de modifier le code. La bibliothèque contient plus de 2500 algorithmes optimisés, qui incluent un ensemble complet d'algorithmes classiques et avancés de vision par ordinateur et d'apprentissage automatique. Ces algorithmes peuvent être utilisés pour détecter et reconnaître des visages, identifier des objets, classer les actions humaines dans des vidéos, suivre les mouvements d’une caméra, suivre des objets en mouvement, extraire des modèles 3D d’objets, produire des nuages de points 3D à partir de caméras stéréo, assembler des images pour obtenir une haute résolution. L’image d'une scène entière, recherchez des images similaires dans une base de données d'images, supprimez les yeux rouges des images prises au flash, suivez les mouvements des yeux, reconnaissez les paysages et créez des repères pour les superposer à la réalité augmentée, etc.

. Grâce à OpenCV, coder la reconnaissance faciale est maintenant plus facile que jamais. La reconnaissance faciale par codage informatique comporte trois étapes simples, similaires à celles que notre cerveau utilise pour reconnaître les visages. Ces étapes sont:

* La Collecte de données : Recueillir les données de visage des personnes que vous souhaitez identifier. Former le dispositif de reconnaissance: insérez les données de visage et les noms respectifs de chaque visage dans le dispositif de reconnaissance afin qu'il puisse apprendre.
* Reconnaissance : Nourrissez les nouveaux visages de ces personnes et voyez si le programme de reconnaissance de visage que vous venez de former les reconnaît.

OpenCV possède trois systèmes de reconnaissance de visage intégrés et, grâce à son code propre, vous pouvez les utiliser simplement en modifiant une seule ligne de code. Voici les noms de ces identificateurs de visage et leurs appels OpenCV:

-EigenFaces - cv2.face. createEigenFaceRecognizer ()

-FisherFaces - cv2.face. createFisherFaceRecognizer ()

-Histogrammes de modèles binaires locaux (LBPH) - cv2.face. CreateLBPHFaceRecognizer

1.3.4.3 Présentation Base de données (gérées par la langage SQL)

**Introduisons la notion de la base de données :**

Une base de données est dotée d'une vaste fonctionnalité de recherche. Par exemple, un département des ventes avait la possibilité de trouver rapidement tout le personnel de vente qui avait réalisé un certain nombre de ventes sur une période de temps prédéfinie. Une base de données peut actualiser des enregistrements en grand nombre, voire des millions ou plus de données. Cela peut être utile, par exemple, si vous souhaitez ajouter de nouvelles colonnes ou appliquer un patch de données d'une sorte ou d'une autre.

Si la base de données est relationnelle, ce qui est le cas de la majorité des bases de données, elle peut faire des références croisées d'enregistrements dans différentes tables. Cela implique que vous pouvez créer des relations entre les tables. Par exemple, si vous avez lié une table Clients à une table Commandes, vous pouvez trouver toutes les commandes de la table Commandes qu'un seul client de la table Clients a traitées, ou les affiner davantage pour ne retourner que les commandes traitées dans une période donnée - ou presque tout type de combinaison que vous pouvez imaginer

Une base de données peut effectuer des calculs d'agrégats complexes sur plusieurs tables. Par exemple, vous pourriez dresser la liste des dépenses de plusieurs points de vente au détail, y compris tous les sous-totaux possibles, puis un total final. Une base de données peut assurer la cohérence et l'intégrité des données, ce qui signifie qu'elle peut éviter la duplication et assurer l'exactitude des données grâce à sa conception et à une série de contraintes.

**Présentons ensuite la langage SQL:**

SQL (**S**tructured **Q**uery **L**anguage) traduit Langage de requêtes structuré ou langage d’interrogation structuré) est un langage de quatrième génération (L4G), non procédural, conçu par IBM dans les années 70. SQL est basée sur l’algèbre relationnelle (opérations ensemblistes et relationnelles). SQL a été normalisé dès 1986 mais les premières normes, trop incomplètes, ont été ignorées par les éditeurs de SGBD. La norme actuelle SQL-2 (appelée aussi SQL-92) date de 1992. Elle est acceptée par tous les SGBD relationnels. Ce langage permet l’accès aux données et se compose de quatre sous-ensembles :

* LID est Le Langage d’Interrogation de Données : Le langage de requête de données est utilisé pour récupérer des données à partir de tables basées sur des conditions que nous pouvons facilement applique
* LMD (Data Manipulation Language DML) est Le Langage de Manipulation de Données : Les commandes DML sont utilisées pour gérer les données sauvegardées dans la table et non la table elle-même. Les commandes DML ne sont pas auto-commandées. Cela fait que les changements ne sont pas définitifs dans la base de données, ils peuvent être retirés.
* LDD (Data Definition Language DDL), Le Langage de Définition de Données : Ceci inclut les changements à la structure de la table comme la création de la table, la modification de la table, la suppression d'une table, etc. Toutes les commandes DDL sont auto-commandées. Cela signifie qu'il sauvegarde en permanence toutes les modifications dans la base de données.
* LCD (Data Control Language DCL), Le Langage de Contrôle de Données : sont les commandes qui permettent de donner et de reprendre le pouvoir à tout utilisateur de la base de données. Intérêt de l’utilisation d’une base de donné SQL.

**Intérêt de l’utilisation d’une base de donnée SQL :**

Au lieu d’utiliser une base de donnée statique, qui est accessible seulement par l’utilisateur de raspberry. On peut utiliser une base de donnée SQL ou elle peut être efficace et rapide pour ajouter ou supprimer une personne de la base de données du système sans pouvoir ouvrir la carte elle-même. De plus, cela peut être bénéfique lorsque on a par exemple deux carte raspberry pi 3, ils appartiennent au même système de sécurité et chacune a une tâche différente de l’autre, et on veut que les deux utilisent la même base de données, l’une par exemple pour la streaming et l’autre pour l’intérieur de la maison .

1.3.5 Conclusion

Nous présentons dans ce chapitre le système de surveillance mis en place l'année précédente un système basé sur la reconnaissance faciale et axé sur la valeur ajoutée par notre projet cette année. Dans le chapitre suivant, nous montrons les différents algorithmes d'identification faciale tels que Haar Cascade et LBP.

Chapitre 2 : Algorithmes de développement pour la reconnaissance faciale et la détection des intrus

2.1 Introduction du chapitre 2

Afin de bien faire fonctionner notre système de contrôle d'accès et de garantir son bon fonctionnement, nous avons soigneusement choisi tout au long de ce chapitre les algorithmes de reconnaissance facial appropriés. Cette reconnaissance peut être définie comme l'identification ou la vérification d'une personne à la suite d'une image digitale ou d'une image filmée par une source vidéo. Une façon d'y parvenir est de faire une comparaison entre les caractéristiques faciales sélectionnés de l'image et une base de données. Le choix de chaque algorithme se fait à la suite d'une étude comparative entre différents algorithmes rassurant les objectifs visés.

**2.2 Description de l’algorithme de Viola Jones**

Une des techniques classiques de détection des visages était présentée par Paul Viola et Michael Jones (2001) [5] qui utilisent classification en cascade pour la détection des visages. Il se traduit par la numérisation d'une image par un fenêtrage tout en calculant un ensemble de caractéristiques pour vérifier la présence ou pas d’un visage ou d’un corps humain. Ils sont quatre composants dans la technique proposée par Viola et Jones. La phase d’apprentissage de l’algorithme Violas et Jones consiste à préparer une banque d’images positives et négatives. Les images positives sont les images qui contiennent les zones d’intérêt (le visage et le corps humain) et les images négatives sont des images des objets.

Nous présentons les étapes du fonctionnement de l’algorithme intitulé « Viola Jones » :

* Il va générer des fonctionnalités numériques de type Haar caractéristiques de l'image utilisées dans la reconnaissance d'objets
* L'introduction d'une nouvelle représentation de l'image appelée Intégral Image qui permet de calculer rapidement les caractéristiques utilisées par le détecteur.
* Un classificateur efficace et simple construit à l'aide de l'apprentissage AdaBoost (algorithme pour sélectionner un petit nombre de caractéristiques visuelles critiques à partir d'un très grand ensemble de fonctionnalités probables).
* Combinaison des classificateurs dans une cascade qui permet aux zones de fond de l'image d'être rapidement négligées tout en dépensant plus de calcul sur les régions les plus importantes qui construit le visage.

2.2.1 Le descripteurs Pseudo-Haar

Les descripteurs utilisés par cet algorithme s’appellent descripteurs pseudo-Haar. Ce sont des descripteurs de structure rectangulaires suffit à exprimer les contours et les changements de texture. On préféra utiliser des descripteurs plutôt que les pixels directement pour la détection de visage puisque parmi les grands intérêts d'utiliser les descripteurs est qu'ils peuvent agir pour coder ad-hoc du domaine de connaissance qui est difficile à apprendre en utilisant un nombre fini de quantité de données à trainer. Pour ce système, il existe également un deuxième intérêt important pour les descripteurs. C’est que Le système fonctionne beaucoup plus rapidement qu'un système basé sur les pixels. Les descripteurs simples utilisées rappellent la base de Haar qui utilisent principalement les fonctions Papageorgiou et al. Plus spécifiquement, nous utilisons trois types de descripteurs. La valeur d'une fonction à deux rectangles est la différence entre la somme des pixels dans deux régions rectangulaires. Les régions ont la même taille et forme et sont horizontalement ou verticalement adjacente (voir la figure 2.1). Une fonction trois-rectangle calcule la somme dans deux rectangles extérieurs soustraits de la somme dans un rectangle central. Enfin un quatre rectangle fonctionnalité calcule la différence entre les paires diagonales de des rectangles. Nous distinguons plusieurs types de descripteurs pseudo-Haar qui se différencient par leurs formes et leurs orientations.

* Descripteurs de frontières indiquant les limites d’un objet.
* Descripteurs lignes indiquant toute forme linéaire dans un objet.
* Descripteurs de centre indiquant toute forme centrale d’un objet (un nez dans un visage…).

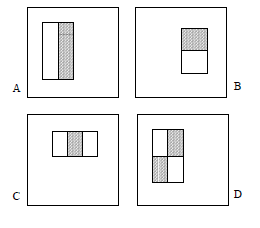


Figure 2.1 Exemple de fonctions rectangulaires [3]

.

2.2.2 Le calcul des descripteurs Pseudo-Haar

2.2.2.1 Calcul préliminaire

La somme des pixels qui se situent dans les rectangles blancs sont soustraits de la somme de pixels dans les rectangles gris. Les descripteurs à deux rectangles (Descripteurs de frontières) sont montrés dans (A) et (B). La figure 1.1 (C) montre un rectangle à trois fonction (Descripteurs lignes), et (D) une fonction à quatre rectangles. On va prendre l’exemple d’un descripteur réel 16\*16 pixels normalisée (Les valeurs des pixels vont par exemple de 0 à 255 s’ils sont codées sur 8 bits d’où on divise par la valeur entière 256 pour qu’elle soit entre 0 et 1). La figure 1.2 illustre précisément l’exemple :

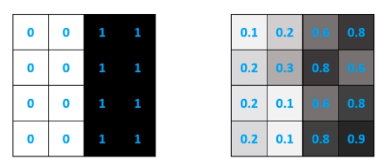


Figure 2.2 Un exemples réelle et idéale de fonctions rectangulaires (Descripteur de frontière)

La première image à gauche représente le descripteur idéal qui contient deux valeurs d’intensités lumineuses 0 ou 1 d’où la couleur blanc est affectée au pixel de valeur 0 et le noir au pixel de valeur égale à 1. Contrairement au deuxième image à droite qui représente un descripteur réel qui contient des valeurs réellement détectées sur une image. L’algorithme de Viola Jones va comparer à quel point le scenario réel est proche de l’exemple idéale.

* Nous calculerons le delta (fonction rectangle) qui est la différence entre les deux régions, la première est appelée la région "sombre" et la seconde "blanche".



* ∆=1 si le descripteur est idéal, sinon plus que on ait proche de 1 c’est que on est proche des descripteurs Haar. Pour notre cas ∆=0.56.

2.2.2.2 Image intégrale

Le descripteur pseudo-Haar correspond au calcul la différence entre les sommes des pixels de deux régions rectangulaires voisines ou plus. Pour calculer la somme d’une région rectangulaire, il nous faut autant d’accès à l’image originale que le nombre de pixels de la région, ce qui nécessite un temps de calcul énorme. Pour améliorer le processus de calcul des Violas et Jones ont présenté le concept de l’image intégrale qui facilitent le calcul des descripteurs. Les fonctions rectangles peuvent être calculées très rapidement en utilisant une représentation intermédiaire de l'image. D’où les champs de cette image contiennent la somme des pixels au-dessus et à gauche [3].



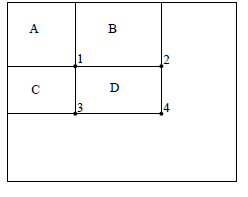
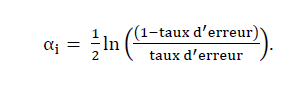


Figure 2.3 La méthode de calcul de l’image intégrale [3]

Le total des pixels faite dans le rectangle D pour la figure 2.3 peut être calculée avec quatre références de tableau. La valeur de l'intégrale l'image à l'emplacement 1 est la somme des pixels en rectangle A. La valeur à l'emplacement 2 est A + B, à l'emplacement 3 est A + C, et à l'emplacement 4 est A+B + C + D. La somme dans D peut être calculé comme 4 + 1- (2 + 3)

2.2.2.3 Classification de l’image

Pour calculer les descripteurs pseudo-haar. L’algorithme de viola Jones va balayer l'image utilisant fenêtre 24\*24. Nous trouvons 180000 descripteurs pour cette fenêtre qui représente un nombre très grand. Pour classifier les descripteurs selon leurs pertinences nous allons utiliser Adaboost qui est l'une des technique les plus populaires de boosting. Elle fait une combinaison des descripteurs afin de former un classificateur fort. D’où la complexité de système va être réduite à l'aide de cette technique. Adaboost fait plusieurs itérations sur l’ensemble des descripteurs, il effectue la classification des images à chaque itération de la base de données et fait le calcul du taux d’erreur correspondant. Le taux d’erreur est calculé en faisant le rapport du nombre d’images qui sont mal classées par rapport à la totalité de la base. Un classifieur faible est définie par un descripteur et un taux d’erreur calculé en balayant toute l’image. Un poids αi à va être attribuée à chaque classifieur faible, ce poids désigne un critère très important pour un classificateur le poids est exprimé comme suit :



On va attribuer une valeur de poids importante lorsque l'erreur est faible pour les classifieurs ayant un taux d’erreur égal à 50% serons négligées. On souhaite avoir une bonne détection, et cela dépend du nombre de classifieur faible qui forment le classifieur fort.



H : le classifieur fort

T : nombre de classifieur faibles formant ce classifieur fort

hi : les poids associés à chaque classifieurs faible

Cette section décrit un algorithme pour construire une cascade des classificateurs qui permettent une performance de détection plus efficaces tout en réduisant radicalement le temps de calcul. L'idée clé est que les classificateurs boostés peuvent être construits dans le but de rejeter les sous-fenêtres négatives tout en détectant presque tous les cas positifs (c'est-à-dire le seuil d'un classificateur renforcé peut être ajusté pour que le taux de faux négatifs soit proche de zéro). Les classificateurs les plus simples sont utilisés pour rejeter la majorité des sous-fenêtres avant que les classificateurs les plus complexes soient appelés. La forme générale du processus de détection est celle d’un arbre de décision, ce que nous appelons une “cascade“ (voir Figure 2.4) Un résultat positif du premier classificateur déclenche l’évaluation d'un deuxième classificateur qui a été ajusté pour atteindre des taux de détection très élevés. Un résultat positif à partir du deuxième classificateur déclenche un troisième classificateur, etc. Un résultat négatif à tout moment conduit à l'immédiat rejet de la sous-fenêtre. Les étages de la cascade sont fabriqués par des classificateurs en formation. en utilisant AdaBoost puis en ajustant le seuil à minimiser les faux négatifs. Notez que le défaut AdaBoost seuil est conçu pour produire un faible taux d'erreur sur la formation. Les données. En général, un seuil inférieur donne une détection plus élevée.

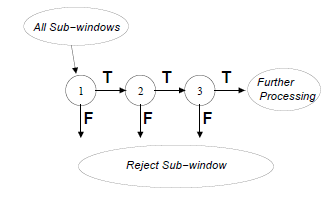


Figure 2.4 Représentation schématique de la cascade pour la détection de visage[5]

Dans le cas où une fenêtre contient un visage, le classifieur suivant va la recevoir sinon elle sera rejetée et nous passons à la fenêtre suivante.

2.2.2.3 conclusion

Nous avons présenté une approche de détection d'objet qui minimise le temps de calcul tout en obtenant une détection élevée et précis. L’approche a été utilisée pour construire une détection de visage système qui est environ 15 plus rapide que tout approche précédente.

**2.2.3 Expérience et résultats**

L'implémentation est utilisée pour tester la performance de la méthode de « Haar\_cascade » sur différents types de visage images. Plusieurs paramètres sont variés pour voir leurs influences sur la performance. Pour cette expérience, nous avons recueilli beaucoup d'images de visage, certains d'entre eux sont collectés à partir de photographies prises avec un Star webcam et certains sont pris depuis la webcam de pc. Dans l'algorithme proposé, différents types d'images de visage ont été reconnus. D’où l'image du visage d'un inconnu est comparée à des images de visage connues stockée dans une grande base de données.

Le tableau suivant indique le taux global de reconnaissance faciale:

Tableaux 2.1 Analyse des performances de l'algorithme « haar cascade »

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre d’image stockée dans la base de données | Nombre d’image testée à l’entrée | Les visages reconnues | Les visages non reconnues | Le taux de réussite |
| 156 | 50 | 41 | 9 | 81% |

Ce taux de réussite est considéré très petit vue l’environnement de travail et le matériel utilisées

**2.3 Description de l’algorithme LBP (Local Binary Pattern)**

Les modèles binaires locaux (LBP) sont des modèles simples et admettent un opérateur de texture performant qui agit sur les pixels d'une image en faisant un seuil à proximité de chaque pixel (pixel central) et en considérant le résultat comme un nombre binaire. En raison de sa grande puissance et de sa simplicité en informatique, le LBP est désormais une approche appréciée dans de nombreuses applications. Il a fourni d'excellents résultats dans divers domaines. En effet, parmi ses atouts, sa solidité aux modifications monotones des niveaux de gris, par exemple par des modifications de luminosité, est l'un de ses intérêts. Une autre propriété importante est sa simplicité de calcul, qui permet d'analyser des images dans des conditions difficiles en temps réel. L'algorithme de la reconnaissance faciale concernant comportent trois étapes :

* Créer une bonne base de données avec de multiples images de visages personnalisées pour chaque individu.
* Un descripteur est ainsi calculé chaque fois pour chaque image de la base de données.
* Contrôler le descripteur sur les images de la base de données.

Cet algorithme est élaboré sur la base du descripteur LBP et décrit à la figure 2.5 :

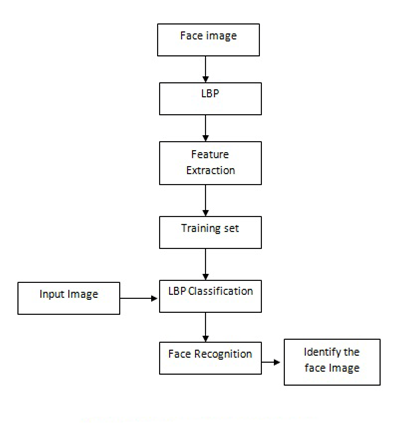


Figure 2.5 : la schématisation de processus [7]

2.3.1 Le principe de l’algorithme LBP

La principale fonction de l'algorithme LBP est la détermination d'un histogramme qui dénombre le nombre d'occurrence de chaque motif binaire. Ainsi, chaque image de la base de donnée va avoir un histogramme. Et ce dernier se déterminent par un processus bien défini illustrant ou in va utiliser la version standard de LBP [2]. Les différentes étapes :

Etape 1 : Extraire les pixels voisines de chaque pixel à traitée

Etape 2 : on va accorder un seuillage définie pour les pixels voisines, si le niveau en gris du pixel voisine est supérieur à cella de la centrale, elle reçoit automatiquement la valeur de 1 sinon elle reçoit la valeur de 0.

Etape 3 : pour un voisinage de 8 pixels, on détermine la matrice poids. En effet, le motif binaire 8 bits est transformé en décimales et celle-ci est ensuite enregistrée à la position du pixel correspondant dans le masque LBP.

Etape 4 : Faire une multiplication les éléments de la matrice de seuillage avec ceux de la matrice poids.

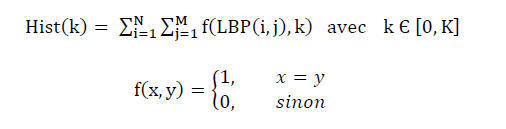
Etape 5 : La détermination du motif binaire local LBP est faite en faisant la somme des valeurs de la matrice obtenue. Le motif binaire local sera calculé comme suit :



gc : la valeur d’intensité du pixel du centre

gp : la valeur d’intensité du pixel voisine avec p=0…7

Par suite, on va calculer un histogramme qui représente le nombre d’occurrence des motifs binaires. Prenant le cas d’une matrice de taille M\*N et la taille maximale de l’histogramme est K, cette formule nous permet de calculer l’histogramme :



La Figure 2.6 présente les différentes étapes de calcul de la forme binaire LBP des pixels locaux:



Figure 2.6 Les différentes étapes de calcul de la fonction LBP [7]

La taille d'un histogramme égale à 256 pour un voisinage de 8 pixels, ce nombre est considérée assez grand pour un calcul de détection faciale. Une étude est faite par des chercheurs de Bengladesh montre que parmi les 256 motifs binaires il y a 58 motifs uniforme (un motif qui contient une transition ou deux transitions) sinon le motif est non uniforme. Dans notre cas, le motif binaire est 01101010 contient 6 transitions donc c'est in motif non uniforme Cette étude montre que 90% des motifs d’une image sont uniformes, ce qui justifie le choix de 59 motifs parmi 256 (58 valeurs représentant les motifs binaires uniformes et une valeur exprimant le nombre des motifs binaires non uniformes). Ensuite, on va parcourir chaque image dans la base de données et calculer les histogrammes correspondants. Par suite, la phase de reconnaissance est déterminée par le calcul de la distance Ӽ2 entre l’image de test et la totalité des images stockées dans la base de donnée. Si nous considérons S comme l’histogramme de l’image test, M comme l’histogramme de l’une des images de la base d’apprentissage, la distance entre ces deux histogrammes s’écrit :



Si : le nombre d’occurrence du motif i dans l’histogramme S

Mi : le nombre d’occurrence du motif i dans l’histogramme M.

En Calculant les distances Ӽ2 entre une image test et toutes les images de la base d’apprentissage, la personne sera identifiée par celle qui donne le minimum de distance.

**2.3.2 Expérience et résultats**

Après avoir compris le principe de fonctionnement de l’algorithme local de reconnaissance faciale « LBP », nous allons tester et évaluer ses performances.

La base des visages de test est la même utilisée pour l’algorithme « Haar-Cascade ». Les résultats d’identification faciale et de la distinction entre un visage connu et autre inconnu. Nous nous référencions des résultats faites par une PFE en 2016 [2] :

Tableaux 2.2 Analyse des performances de l'algorithme « LBP »[2]

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Nombre d’image stockée dans la base de données | Nombre d’image testée à l’entrée | Les visages reconnues | Les visages non reconnues | Le taux de réussite |
| 1127 | 1227 | 1100 | 124 | 89% |

**2.4 La comparaison entre le LBP et Haar Cascade**

Les résultats des deux expériences précédentes montrent que deux algorithmes de détection faciale peuvent détecter le visage avec précision. En d'autres termes, l’efficacité des deux algorithmes est presque identique. Bien que LBP sur le plan statistique il détecte selon des calculs faits est de 7,3% de visages de plus que Haar, les résultats obtenus pour les deux algorithmes sont presque identiques. Les deux algorithmes n'ont pas de problèmes de détection avec visages raciaux, visages poilus, visages différents avec des lunettes. Sur la base des résultats faite auparavant, l'algorithme LBP est l'algorithme le plus fiable pour trouver des visages puisqu’il est plus efficace coté calcul et rapidité, Cela pourrait être un énorme avantage dans la mise en œuvre sur le système embarqué pour la reconnaissance des visages instantanément. [3]

**2.5 Conclusion**

Dans ce chapitre, nous décrivons en détail les deux algorithmes LBP et Haar Cascade qui constituent les deux techniques utilisées pour la détection faciale. Nous essaierons de mettre en œuvre ces deux algorithmes sur la carte Raspberry pi 3 et nous nous contenterons de faire le point sur les différentes étapes de réalisation de notre système et de le mettre à l'épreuve au chapitre ci-après (chapitre 3)

1. Chapitre 3 : Mise en place du système de reconnaissance faciale
2. 3.1 Introduction du chapitre 3

Nous détaillons dans ce chapitre les étapes de réalisation de notre système. Nous nous concentrerons sur la configuration framboise en commençant par le paramétrage d'Opencv, puis en implémentant SQL. Ensuite, nous procédons à la validation du système en effectuant les tests nécessaires à l'aide d'une vidéo en continu qui filme une personne à reconnaître.

**3.2 L’installation du Raspbian et la configuration de la raspberry**

**3.2.1 l’installation du système d’exploitation (Le raspbian)**

Ce procédé peut être fait en suivant les étapes suivantes :

Etape1 : nous allons formater la carte mémoire à travers le logiciel « SD card Formater » dont l’interface est montrée dans la figure 3.1.

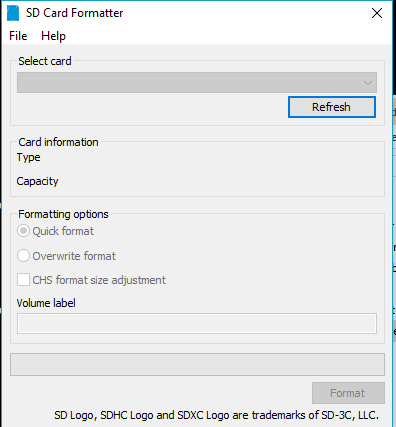


Figure 3.1 Le logiciel « SD card formater »

Etape 2 : Le Raspberry Pi alimente son système d'exploitation à partir du lecteur de carte micro SD. Il est par conséquent essentiel de bien les préparer afin que le nano ordinateur se mettre en marche correctement. Nous allons utiliser le logiciel BalenaEtcher pour le booting du Raspbian sur la carte SD. Comme il est indiqué sur la figure 3.2, nous sélectionnons le système d’exploitation « Raspbian » pour le copier sur le la carte mémoire.

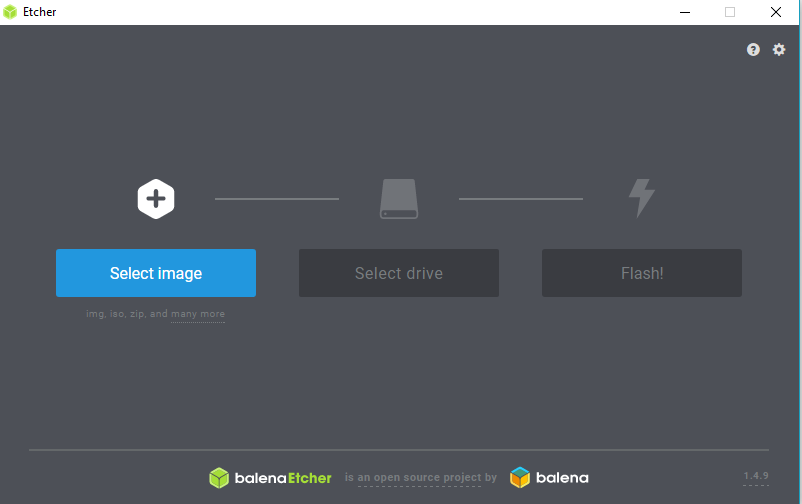


Figure 3.2 Le Flashing du Raspbian sur la carte SD

Etape 3 : une fois le booting est fini, on ouvre la Desktop du Raspbian en utilisant un écran, un clavier et une souris. (Figure 3.3)



Figure 3.3 Le Desktop de la raspberry

Étape 4: On fait un reboot pour actualiser le système d’exploitation, de plus, on peut ouvrir le terminal pour commencer à programmer

**3.2.2 Configuration du réseau sur la raspberry pi 3**

Le client DHCP sera par défaut activé sur Raspbian. De fait, à chaque fois que la carte est raccordée à un câble Ethernet encastré dans la pièce (ou via un commutateur), nous en profitons dans la mesure où le réseau est alimenté par un service DHCP (Dynamic Host Configuration Protocol) dont la fonction essentielle est de faire la distribution des adresses IP aux serveurs connectés. Une fois que l'on a récupéré une adresse IP dynamique à travers la commande ifconfig du terminal du Raspberry, on peut se connecter avec cette adresse depuis un poste informatique sur le même réseau que notre carte, juste pour assurer quelques tâches à distance en se servant/distant à travers deux logiciels Putty connus et en connexion avec un poste distant sans devoir relier la carte sur un écran.

Dans notre cas, on a utilisé le téléphone portable comme un point d’accès. La connexion entre le téléphone est le portable est une opération classique par contre pour connecter la carte raspberry au point d’accès nécessite le raccordement de la carte à un écran, clavier et souris.

A partir du PC et à travers le logiciel IPSCANNER, nous pouvons reconnaitre l’adresse IP de la carte.

L’accès à la carte raspberry à partir de l’ordinateur est donc rendu possible à travers la connexion bureau à distance (voir figure 3.4) ou le logiciel PUTTY. L’utilisation de l’un ou l’autre des logiciels cités dépends si nous voulons visualiser à tout le contenu de la carte ou uniquement accéder au terminal des commandes.

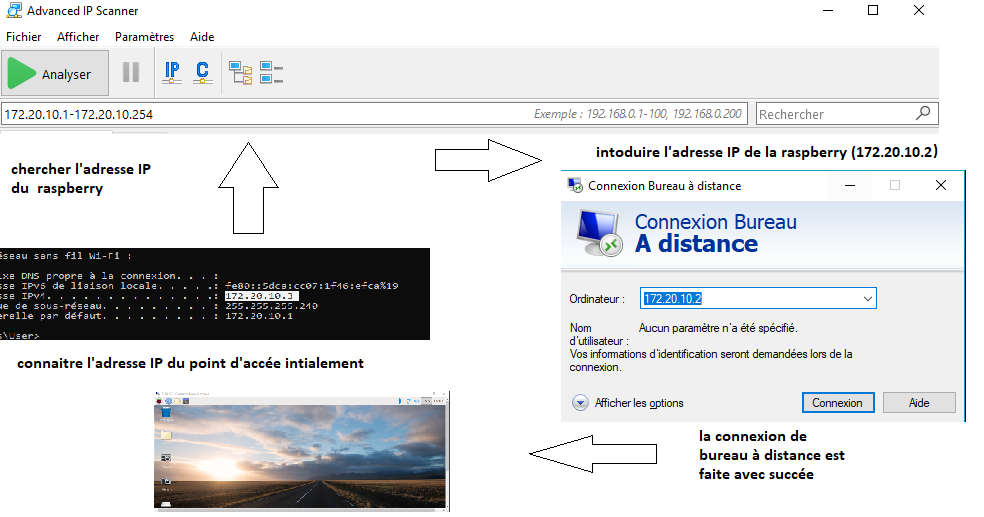


Figure 3.4 Les étapes de la connexion du raspberry à distance

**3.2.3 L’installation de l’open cv et python 3**

**Etape 1**: Etendre le système de fichier. On introduit la commande : sudo raspi-config. On choisit Advanced Options puis Expand file system. Après le redémarrage, votre système de fichiers aurait dû être étendu pour inclure tout l'espace disponible sur votre carte micro-SD.

**Etape 2** : La première étape consiste à mettre à jour et mettre à niveau tous les packages existants par la commande : $ sudo apt-get update && sudo apt-get upgrade. Nous devons ensuite installer des outils de développement, dont CMake, qui nous aide à configurer le processus de construction OpenCV par la commande : $ sudo apt-get install build-essential cmake pkg-config. Ensuite, nous devons installer des packages d’entrées / sorties d’image parceque ils nous permettent de charger divers formats de fichiers d’image à partir du disque. JPEG, PNG, TIFF, etc. sont des exemples de formats de fichier de ce type et cela est fait par la commande suivante : $ sudo apt-get install libjpeg-dev libtiff5-dev libjasper-dev libpng12-dev. Enfin, installons les fichiers d’en-tête Python 2.7 et Python 3 pour pouvoir compiler OpenCV avec des liaisons Python Avec la commande : sudo apt-get install python2.7-dev python3-dev.

**Etape 3** : Télécharger le code source OpenCV par la commande : wget -O opencv\_contrib.zip <https://github.com/Itseez/opencv_contrib/archive/3.3.0.zip>.

**Etape 4** : Installation du python3 avec la commande : sudo python3 get-pip.py .Ensuite , on va utiliser **l’environnement virtuel** de python pour plusieurs causes. Premièrement, il est important de comprendre qu’un environnement virtuel est un outil spécial utilisé pour conserver les dépendances requises par différents projets à des emplacements distincts en créant des environnements Python isolés et indépendants pour chacun d’entre eux. En bref, cela résout le dilemme « Le projet X dépend de la version 1.x, mais le projet Y nécessite 4.x». Il garde également vos paquets de sites globaux propres, rangés et exempts de tout encombrement. Installons maintenant l’environnement virtuel de python et cela est fait par les commandes suivantes : sudo pip install virtualenv virtualenvwrapper/ sudo rm -rf ~/. cache/pip . Nous devrons simplement utiliser la redirection de sortie de cat et de sortie pour gérer la mise à jour ~ / .profile par les commandes :

-echo -e "\n# virtualenv and virtualenvwrapper" >> ~/.profile

-echo "export WORKON\_HOME=$HOME/. virtualenvs" >> ~/.profile

- echo "export VIRTUALENVWRAPPER\_PYTHON=/usr/bin/python3" >> ~/.profile

- echo "source /usr/local/bin/virtualenvwrapper.sh" >> ~/.profile

**Etape 5** : Installation du NumPy sur notre Raspberry Pi avec la commande : pip install numpy

**Etape 6**: Compilation and Installation du OpenCV a l’aide de la commande : make -j4

**Etape 7** : Tester le opencv3 install comme il est indiqué dans les la figure 3.5 :

**3.2.4 L’installation du SQL**

Etape 1 : Installer un serveur web sur votre Raspberry (Apache) :

* Mettre à jour Raspberry Pi3 : -sudo apt update

- sudo apt upgrade

-sudo apt update

* Installer serveur apache: sudo apt install apache2
* Donner des droits au dossier d’apache qui vous permettra de facilement administrer ces sites : -sudo chown -R pi:www-data /var/www/html/

-sudo chmod -R 770 /var/www/html/

* Vérifier qu’Apache fonctionne : ouvrir le navigateur web de la Raspberry (par défaut Midori sous Raspbian), et d’aller à l’adresse « http://127.0.0.1 ». Vous devriez alors obtenir une page avec un message du genre « It Works! » et plein d’autre texte.
* Installer PHP sur Raspberry Pi3: sudo apt install php php-mbstring
* Vérifier que PHP fonctionne : Supprimer le fichier « index.html » dans le répertoire « /var/www/html ». sudo rm /var/www/html/index.html \*Créez un fichier « index.php » dans ce répertoire avec cette ligne de commande : echo "<?php phpinfo(); ?>" > /var/www/html/index.php

Etape 2 : Installation du serveur MySQL sur Raspberry Pi3

* sudo apt-get install mysql-server
* Entrer un mot de passe pour l'utilisateur root. Il faut bien noter cela, car nous devrons l'utiliser pour accéder au serveur MYSQL et y connecter PHPMyAdmin : utilisateur : root/password : password
* Si vous n'êtes pas invité à entrer un mot de passe, vous devrez entrer la commande suivante. Répondez à toutes les questions afin qu'il soit configuré pour répondre à vos exigences de sécurité. sudo mysql\_secure\_installation
* Accéder aux modifications de la base de données et commencer à les modifier, entrez la commande suivante : sudo mysql -u root –p
* Saisir le mot de passe que nous venons de créer : password
* Vous pouvez quitter la ligne de commande MYSQL en entrant simplement « quit ».

Etape 3 : Installation de Raspberry pi PHPMyAdmin

* Installation du paquet PHPMyAdmin : sudo apt-get install phpmyadmin
* Un écran nous demandant quel type de serveur Web vous souhaitez utiliser s’affiche. Sélectionnez apache2.Sélectionnez oui à la prochaine invit L'outil d'installation va maintenant demander un mot de passe, entrez password. Il nous sera demandé de définir un mot de passe pour PHPMyAdmin : password : Sudo ln-s/share.phpmyadmin/var/html
* Configurer Apache pour PHPMyAdmin : sudo nano /etc/apache2/apache2.conf
* Maintenant, au bas de ce fichier, entrons la ligne suivante : Include /etc/phpmyadmin/apache.conf .Une fois terminé, sauvegardez et quittez en appuyant sur CTRL + X puis sur o
* Maintenant, redémarrez le service Apache en entrant la commande suivante : sudo /etc/init.d/apache2 restart
* Vous devriez maintenant pouvoir accéder à PHPMyAdmin à partir d’un navigateur : http://172.20.10.2/phpmyadmin
* Si tout fonctionne, alors vous êtes prêt à partir. Vous devrez vous connecter en utilisant user : root/password : password

La figure 3.7 affiche la page phpmyadmin qui nous permet d’ouvrir le mysql

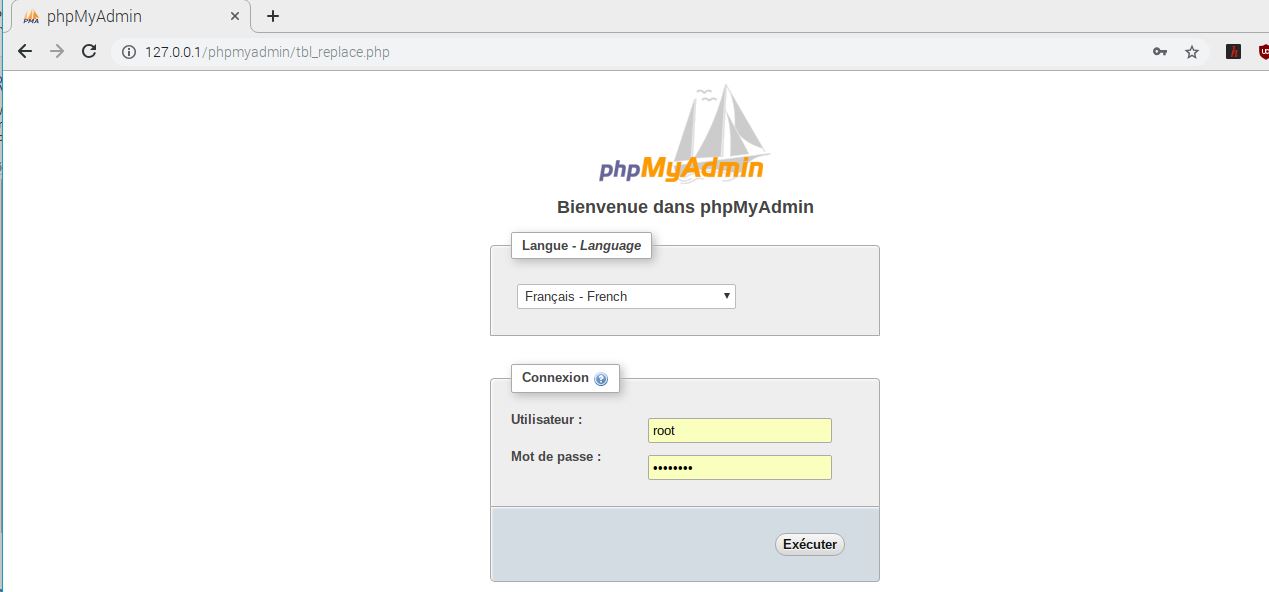


Figure 3.5 L’interface phpmyadmin

**3.3 L’implémentation d’algorithme**

Pour déterminer l'identité de la personne qui a présenté une demande d'accès, nous procéderons en trois étapes principales :

* La création d'une base de données comprenant une douzaine de photos. On a associé pour chaque personne plusieurs images avec des grimaces diverses.
* La détection des visages dans la base de données pour une utilisation ultérieure dans le cadre d'une formation d’un reconnaisseur.
* La reconnaissance faciale pour tester le reconnaisseur à reconnaître les visages pour lesquels elle a été formée.

**3.3.1 La base de donnée**

3.3.1.1 Création de la base de données manuellement pour le LBP

Pour la plupart d'entre nous, nous voudrons reconnaître les visages qui ne font pas partie d'un ensemble de données actuelles et reconnaître les visages de nous-mêmes, d'amis, de membres de la famille, de collègues, de collègues, etc. Pour ce faire, nous devons recueillir des exemples de visages que nous voulons reconnaître et les quantifier d'une manière ou d'une autre. L’algorithme introduit dans **l’annexe 1** permet de faire des centaines de la photo dans la base de donnée après avoir introduits son Id qui sera coller au mot « user », En effet, tous les photos de la dataset ont la structure suivante « user. id. Numéro de la photo »

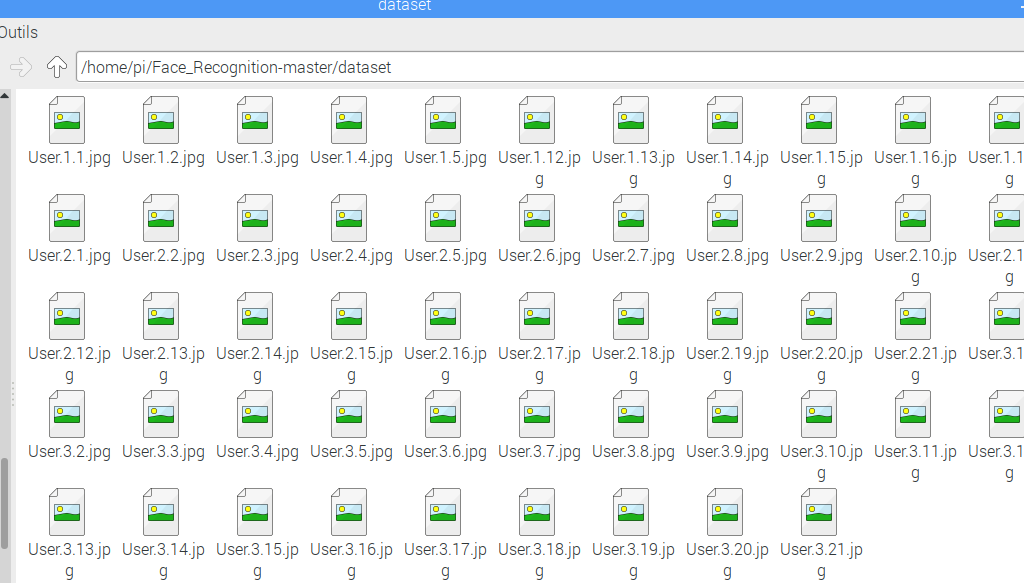


Figure 3.6 Dataset trainée manuellement

***3.3.1.2 Création de la base de donnée SQL pour le Haar\_Cascade***

On a créé une table qui comporte 4 attributs, une clé primaire qui est l’Id et 3 autres attributs qui sont le nom, et 2 photos (2 biodata). Une base test qui contient 4 personnes va être insérée aux personnelles, comme il est indiqué dans la figure 3.6 :

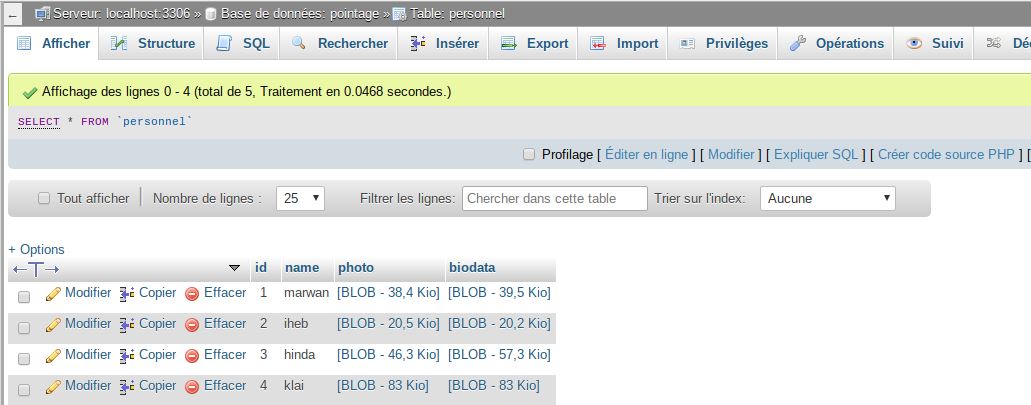


Figure 3.7 La base de donnée test

Suite à la création de la base de donnée, des dossiers portant les noms des personnels vont être crées automatiquement dans un dossier intitulé dataset comme il est indiqué dans la figure 3.7 ou les algorithmes vont les parcourir pour faire l’identification faciale en streaming. Les dossiers vont être supprimer automatiquement après l’achèvement du processus.

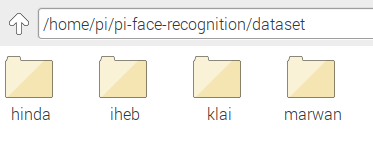


Figure 3.8 Base de données sur la Raspberry

**3.3.2 Test et validation**

3.3.2.1 Mise en œuvre de l’identification par Haar\_cascade

A la suite de la présentation de notre base de données, nous attaquons maintenant le développement de l'algorithme de reconnaissance faciale en suivant ces démarches : Importer les modules nécessaires :

* cv2 : OpenCV est le module qui comprend les fonctions de détection et de reconnaissance faciale..
* os : nous employons ce module afin d'extraire les désignations des photos dans le directoire de la base de données, puis on va récupérer le numéro individuel qui sera utilisé comme étiquette pour le visage dans cette image.
* Image : Puisqu' OpenCv ne peut pas supporter le mode GIF des photos qui forment la base, il est possible d'utiliser le module PIL pour la lecture des nuances de gris de la photo en format.
* Numpy : Nous stockons nos images dans des matrices numpy.
* Imutils : série de fonctions pratiques pour faciliter les opérations de traitement d'image de base telles que la conversion, la rotation, le redimensionnement, la squelettisation et l'affichage d'images Matplotlib avec OpenCV et Python
* Apgrase : les arguments de la ligne de commande fournissent des informations supplémentaires à un programme lors de l'exécution. Cela nous permet de donner à notre programme différentes entrées à la volée sans changer le code. Vous pouvez établir l'analogie selon laquelle un argument de ligne de commande est similaire à un paramètre de fonction
* Pickle : Le module pickle implémente un algorithme fondamental mais puissant pour la sérialisation et la désérialisation d'une structure d'objet Python. « Pickling » est le processus par lequel une hiérarchie d'objets Python est convertie en un flux d'octets et « unpickling » est l'opération inverse, par laquelle un flux d'octets est reconverti en une hiérarchie d'objets
* Face recognition et dlib : deux bibliothèques déjà prête, faite par Davis King (créateur of dlib) and Adam Geitgey (créateur de face\_recognition)
* VideoStream et FPS : importer ces deux modules à partir de imutils afin d’assurer le streaming vidéo
* mysql. connector / pymysq : pour utiliser la langage SQL afin de gérer la base de donnée.
* Chargez les données d'encodage du visage
* Installer notre détecteur de visage en utilisant la méthode en cascade Haar.
* Lancer une vidéo en temps réels
* Initialisez notre VideoStream - nous allons utiliser une caméra USB.
* Démarrez nos images par seconde, fps, compteur.
* La Vérification s'il y a des correspondances
* Si des correspondances sont trouvées, nous utiliserons un système de vote pour déterminer de quel visage il s'agit le plus probablement
* Cette méthode fonctionne en vérifiant quelle personne de l'ensemble de données à le plus de correspondances (en cas d'égalité, la première entrée du dictionnaire est sélectionnée).
* De là, nous dessinons simplement des rectangles entourant chaque visage avec le nom prédit de la personne

3.3.2.2 Mise en œuvre de l’identification par le LBP

Après avoir présenté notre base de données, on attaque maintenant le développement de l’algorithme de reconnaissance de visage en suivant ces étapes :

* Importer les modules nécessaires :
* cv2 : OpenCV est le module qui comprend les fonctions de détection et de reconnaissance faciale.
* os : nous employons ce module afin d'extraire les désignations des photos dans le directoire de la base de données, puis on va récupérer le numéro individuel qui sera utilisé comme étiquette pour le visage dans cette image.
* Image : Puisqu' OpenCv ne peut pas supporter le mode GIF des photos qui forment la base, il est possible d'utiliser le module PIL pour la lecture des nuances de gris de la photo en format.
* Numpy : Nous stockons nos images dans des matrices numpy.
* Chargement de la cascade de reconnaissance du visage :
* Détection du visage de chacune des photos.
* Entraîner le Reconnaisseur. Créer l’objet reconnaisseur facial :
* Nous utiliserons le reconnaisseur LBP
* Essai du détecteur de visage:
* Nous montrons aussi le nom de la personne en établissant un indice de confiance de chaque reconnaissance. Plus la variable de confiance est élevée, moins le reconnaissant a de confiance en la reconnaissance. Une valeur de confiance de 0.0 est une parfaite reconnaissance (algorithme LBP).

3.3.2.3 Les tests pour le Haar\_ cascade (le streaming)

Nous allons faire les tests en utilisant la technique de Haar cascade, les deux personnes détectées à l’entrée s’affichent dans la figure ci-dessous sont deux frères qui portent les noms de Iheb Helali et Oussema Helali ont des descripteurs en commun, avec ça il a réussi à les identifier et les distinguer l’un de l’autre.

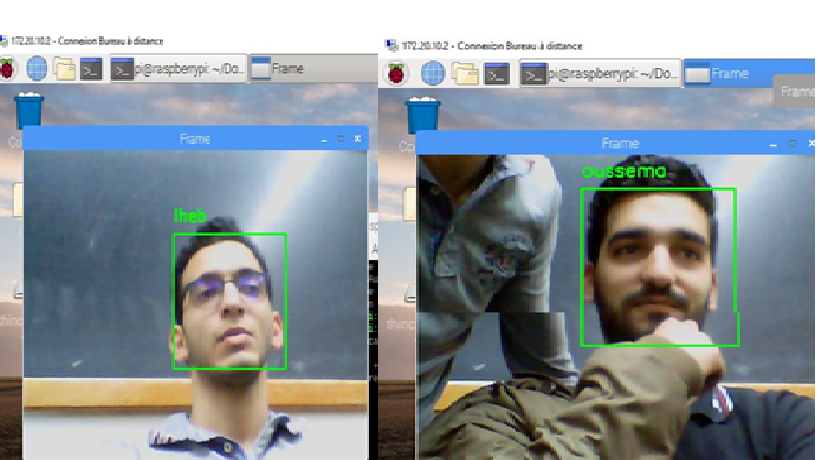


Figure 3.9 Les tests faites en streaming

3.3.2.4 Problèmes rencontrées

Le contenu de notre PFA 2 est très intéressant. La réalisation d'un système embarqué et la connaissance de la carte Raspberry Pi3 nous parait une étape bénéfique dans notre parcours. Et malgré toutes les difficultés et quelquefois les défaillances, nous avons beaucoup appris. Mais il faut dire que ce sont les échecs, plus que les succès, qui nous font apprendre le mieux. Premièrement, pour démarrer les implémentations, nous avons passé des jours à connaître ce que signifie une carte Raspberry et ses fonctionnalités. Nous nous sommes heurtés à des problèmes de réalisation de l'algorithme LBP au niveau des modules de l'opencv ou de la fonction qui nous fournit la distance de confiance et l'identifiant de la personne (recognizer. Predict). Non reconnue par l'algorithme.

**3.4 Conclusion**

La bonne utilisation et la bonne exploitation des technologies de surveillance et des techniques d’identification nous ont amenés à mettre en œuvre un système de reconnaissance faciale. Nous avons donc présenté dans ce dernier chapitre, les diverses étapes de mise en œuvre de ce dispositif tout en le validant par la prise en charge des tests nécessaires.

Conclusion générale

Contrôle d'accès et surveillance des lieux d'habitation et des institutions publiques, tout le monde recherche cette préoccupation. Pour cela, en ce travail de fin d'année, nous sommes parvenus à mettre en place un système de surveillance en temps réelle avec une démarche réfléchie pour traiter les questions de sécurité et de surveillance de l'habitat.

Pour ce faire, nous avons choisi d'utiliser une carte Raspberry Pi3 uniquement pour résoudre le dilemme de la transportabilité du système. Nous avons effectué une analyse comparative des algorithmes les plus couramment utilisés dans le domaine : on a comparé les deux algorithmes de la reconnaissance faciale "Haar\_casacde" et "LBP" en choisissant finalement "LBP" grâce à sa Fiabilité et son efficacité

D’une part, il lui manque la partie de l’alerte après la reconnaissance faciale de la personne. D'autre part, nous pourrons raccorder le module caméra à Internet pour avoir la possibilité de contrôler notre maison en un temps réel, et en particulier que la carte Raspberry Pi3 a un module WiFi qui vous permet de le gérer sans avoir de problème de transférabilité de la chaine.

Références bibliographiques

[1] Mekki Arij, Khedhira Ameny. *Implémentation d’un système de contrôle d’accès sur une carte Raspberry Pi*. Mai 2016$a

[2] SAWSSEN BEJAOUI *Conception et réalisation d’un système de contrôle d’accès et de surveillance intelligente*. Soutenu le 24 Septembre 2016.

[3] Kushsairy Kadir, Mohd Khairi Kamaruddin, Haidawati Nasir, Sairul I Safie, Zulkifli Abdul Kadir Bakti. *A Comparative Study between LBP and Haar-like features for Face Detection Using OpenCV*. In : url : <https://www.semanticscholar.org/paper/A-comparative-study-between-LBP-and-Haar-like-for-Kadi>

[4] Lentin Joseph (India), *Raspberry Pi Image Processing Programming*. Systéme embarqué,2017

[5] Paul Violar, Michael Jones. *Rapid Object Detection using a Boosted Cascade of Simple Features*. In : url : https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf

[6] Raspberry Pi. Url : https://fr.wikipedia.org/wiki/Raspberry\_Pi.

[7] Brian O’Connor et Kaushik Roy. *“Facial Recognition using Modified Local Binary Pattern and Random Forest*, Département of Computer Science, North Carolina ATState University, Greensboro”. In : November 2013.