الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي و البحث العلمي

Université Abou Bekr Belkaid
Tlemcen Algérie

جامعة أبي بكر بلقايد
تلمسان الجزائر

République Algérienne Démocratique et Populaire

Université Abou Bakr Belkaid– Tlemcen

Faculté des Sciences

Département d'Informatique

Mémoire de fin d'études

Pour obtenir le diplôme de

MASTER EN INFORMATIQUE

Spécialité : Modèle Intelligent et Décision(M.I.D)

Sur le thème

Reconnaissance de la langue des signes

Réalisé par :

BOUTALEB Mohamed Yasser

BOUCOIRRA Mohamed Seddik

Soutenue publiquement le 24 / 05 /2017, devant le Jury composé de :

Président	Mr Benazzouz Mourtada	U. Tlemcen
Encadreur	Mr Fethallah Hadjila	U. Tlemcen
Examinateur	Mr Smahi Ismail	U. Tlemcen

Année Universitaire 2016 – 2017

Dédicaces

On dédie ce modeste travail:

À toutes nos familles
À tous nos amis

À tous ceux qui nous ont aidés

✍ BOUTALEB & BOUCOIRRA.

Remerciements

Louange à Dieu qui nous a donné la force, le courage, la patience, et l'espoir nécessaire pour accomplir ce travail et surmonter l'ensemble des difficultés.

On exprime notre gratitude, et nos remercîments à nos parents qui ont fait de leur mieux pour nous aider.

On tient à remercier vivement :

- Notre encadreur Mr. Fethallah Hadjila pour ses conseilles, et son suivi durant la réalisation de ce projet.
- Mr. Mortada Benazzouz qui nous a permis d'intégrer cette formation.
- L'ensemble des enseignants du département des Mathématiques et Informatique ayant consacré leurs vies sur la voie noble de l'enseignement et le partage.

On remercie aussi les personnes qui nous ont aidés et encouragés le long de ce travail.

✍ BOUTALEB & BOUCOIRRA.

Table des matières

Liste des figures

Liste des tableaux

Liste des Acronymes

RGB: Red Green Blue

SVM: Support Vector Machine.

CMJN: Cyan Magenta Jaune Noir.

CMYK: Cyan Magenta Yellow Black.

TLS: Teinte Luminance Saturation.

HLS: Hue Luminance Saturation.

NDK: Native Development Kit.

SMO: Sequential Minimal Optimization.

SVC: Support Vector Classication

RBF: Radial Basis Function

Introduction générale

Comme nous le savons, la technologie basée sur la vision par ordinateur pour la reconnaissance des gestes de la main est une partie très importante de l'interaction homme-machine, Cependant, en raison du développement rapide au niveau matériel, et logiciel, la mise en place de nouvelles méthodes plus efficaces d'interaction homme-machine, devient une nécessité primordiale. En particulier, des technologies telles que la reconnaissance vocale, et la reconnaissance des gestes, reçoivent une très grande attention ces derniers temps.

De nombreux chercheurs se sont efforcés d'améliorer la technologie de la reconnaissance des gestes de la main qui à une grande valeur dans de nombreuses applications telles que la reconnaissance de la langue des signes, la réalité augmentée (réalité virtuelle), l'interprétation gestuelle des personnes handicapées, et le contrôle des robots.

A cet effet, nous allons présenter à travers ce mémoire la conception et l'implémentation d'une application mobile pour la reconnaissance de la langue des signes, et des gestes de la main.

Dans le premier chapitre, nous évoquerons brièvement les différentes définitions, concernant l'image numérique, la vision par ordinateur et son interaction avec la gestualité humaine, et un état de l'art sur une nouvelle méthode pour la reconnaissance des signes de la main.

À travers le deuxième chapitre nous allons présenter, et expliquer en détail, les principes de fonctionnement de notre approche utilisée pour la reconnaissance en temps réel des gestes de la main, et de son application. Une conclusion générale fera le point de ce travail, et donnera quelques perspectives envisagées.

Chapitre I

Introduction aux images numériques et la vision par ordinateur

I.1 Introduction

De nos jours, les nouveaux supports et formes de création et de narration, de production, de diffusion, de stockage, inscrivent aujourd'hui l'image numérique en haut de l'échelle de la révolution numérique, ciblant une grande variété des domaines.

Sur cet aspect nous allons présenter dans ce chapitre quelques définitions générales sur l'image numérique, le traitement d'images, ainsi que la vision par ordinateur et son interaction avec la reconnaissance de la gestualité humaine.

I.2 Définition de l'image

C'est un ensemble structuré d'informations qui peut être décrit mathématiquement sous la forme d'une fonction $f(x, y)$, définie dans un domaine borné, tel que x et y sont les coordonnées spatiales d'un point de l'image et f est une fonction d'intensité lumineuse et de couleur.

Sous cet aspect, la numérisation d'image c'était l'un des sujets primordiaux des recherches scientifiques qui visent l'exploitation d'image par la machine et les technologies modernes existantes [**1]**.

I.3 L'image numérique

Le terme d'image numérique désigne, dans son sens le plus général, toute image qui a été acquise, traitée et sauvegardée sous une forme codée représentable par des nombres (valeurs numériques).

La numérisation est le processus qui permet de passer de l'état d'image physique (image optique par exemple) qui est caractérisée par l'aspect continu du signal qu'elle représente (une infinité de valeurs dans l'intensité lumineuse par exemple), à l'état d'image numérique qui est caractérisée par l'aspect discret (l'intensité lumineuse ne peut prendre que des valeurs quantifiées

en un nombre fini de points distincts). C'est cette forme numérique qui permet une exploitation ultérieure par des outils logiciels sur ordinateur [**1].**

I.3.1 Types d'images numériques

On distingue deux types d'images à la composition, et au comportement différent : images matricielles et les images vectorielles.

a. Images matricielles

Une image matricielle, ou « carte de points (*de l'anglais* bitmap » est composée comme son nom l'indique d'une matrice (tableau) de points à plusieurs dimensions, chaque dimension représente une dimension spatiale (hauteur, largeur), ou autres (niveau de résolution). Dans le cas des images à deux dimensions, les points sont appelés pixels [**1]**.

b. Images vectorielles

Elle est composée de différents objets repérés par leurs coordonnées et comportant différents attributs (bordure, fond, forme, coordonnées). Leur avantage c'est qu'elles peuvent être facilement redimensionnées. Leur codage dépend directement du logiciel qui a permis de les créer [**1]**.

I.3.2 Formalisation de l'image numérique

Du point de vue mathématique une image optique est généralement représentée par une fonction bidimensionnelle représentant des caractéristiques particulières du signal lumineux de l'image en chaque point de son espace (intensité, couleur,…).Le passage à une représentation numérique se fait en réalisant une discrétisation des coordonnées spatiale de ce signal dans les deux dimensions de l'image (donnant la définition de l'image), et une discrétisation du signal par un échantillonnage (quantification) codé numériquement avec une certaine précision (nombres codés sur un certain nombre de bits). L'image est donc constituée par un ensemble régulier d'éléments appelés « pixels » et elle est généralement appelée image « bitmap » **Figure I.1**.

Le nombre de bits accordé à l'échantillonnage du pixel détermine la précision numérique de la représentation. Un seul bit ne permet de ne représenter que des valeurs purement monochromes (Noir ou Blanc), tandis qu'un nombre plus élevé permet de coder un nombre plus important de niveaux de gris ou de couleurs distincts [**1]**.

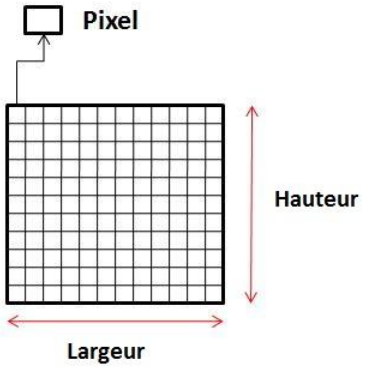


Figure I.1 *Représentation bidimensionnelle d'une image numérique* **[1]**.

Concernant les modes colorimétriques d'une image numérique, un système de codage de la couleur, couplé à celui de la représentation des niveaux de luminosité, doit être employé pour représenter les informations concernant la colorimétrie. De nombreux systèmes de codage ayant leurs particularités propres existent, comme par exemple:

- **RGB (ou RVB)**: basé sur un mélange additif (combinaison de rayons lumineux) de trois couleurs primaires (Rouge, Vert, Bleu) **Figure I.2** [**1**].
- **CMYK ou (CMJN)**: basé sur un mélange soustractif (combinaison de pigments colorés) de trois couleurs primaires (Cyan, Magenta, Jaune) et du noir. **Figure I.2** [**1**].

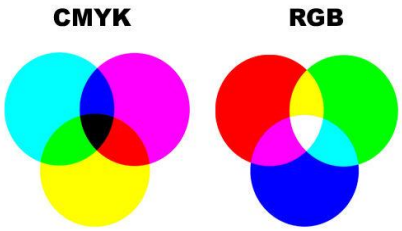


Figure I.2 *Codage RGB (Red-Green-Blue), et CMYK (Cyan-Magenta-Yellow-Black)* **[1]**.

- **HLS (ou TLS)**: basé sur la perception physiologique de la couleur par l'œil humain (Teinte, Luminance, Saturation). **Figure I.3 [1].**

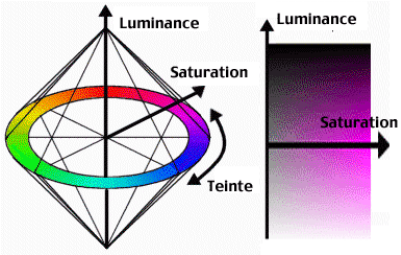


Figure I.3 *Codage* ***HLS*** *(Hue, Luminance, Saturation)* [**1**].

I.4 La vision par ordinateur

La vision par ordinateur (ou aussi appelée vision artificielle ou vision numérique) est une branche de l'intelligence artificielle dont le principal but est de permettre à une machine d'analyser, traiter et comprendre une ou plusieurs images prises par un système d'acquisition (par exemple : les caméras, etc.) [**2**]. Comme elle s'articule sur deux types de traitement d'images :

I.4.1 Traitements de bas niveau

Il existe plusieurs techniques de traitement d'image numérique de bas niveau, dont les plus utilisées sont les suivantes :

a. Le filtrage

Pour améliorer la qualité visuelle de l'image, on doit éliminer les effets des bruits (parasites) en lui faisant subir un traitement appelé filtrage. Il consiste en effet à modifier la distribution fréquentielle des composantes d'un signal selon des spécifications données. Le système linéaire utilisé est appelé filtre numérique. En pratique, il s'agit simplement de créer une nouvelle image en se servant des pixels de l'image à traiter (originale). Il existe deux grandes familles de filtrage [**1**] :

- Filtre passe-bas : atténue le bruit et les détails.
- Filtre passe-haut : accentue les détails et les contours.

Dont chacune comporte un grand panel de méthodes par exemple le filtre médian, le filtre moyenne et le filtre gaussien. **Figure I.4** [**1**].

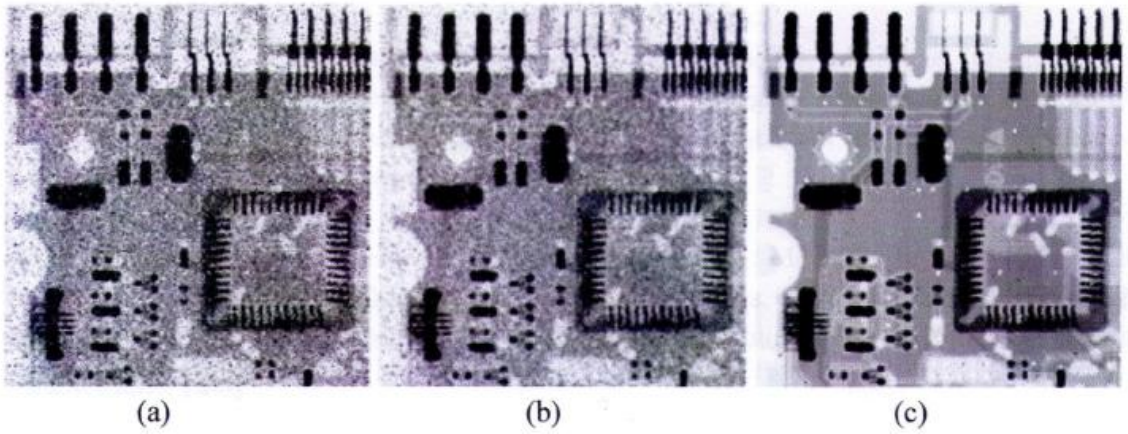


Figure I.4 *Illustration d'une image filtrée, (a) filtre médian, (b) le filtre moyenne, (c) le filtre gaussien.*

b. La restauration

La restauration a pour but d'inverser l'effet du phénomène dégradant. Il s'agit donc de produire une image la plus proche de la réalité physique de la scène observée. Le plus souvent, cette étape est la première dans la chaine de traitements constituant un système de vision [**1]**.

c. L'amélioration

L'amélioration a pour but de satisfaire l'œil de l'observateur humain. C'est pourquoi l'image produite peut être différente de la réalité. Cette amélioration peut servir dans un premier temps à faciliter la visualisation de l'image sur un écran d'ordinateur [**1]**.

d. La compression

On classe les techniques de compression par extension du fichier informatique. Il s'agit là de faciliter le traitement et surtout le stockage des images par une réduction adéquate de leur volume d'information. Il existe deux majeurs types de compression [**1]** :

- Compression réversible : permet de retrouver exactement l'image après décompression.
- Compression irréversible : entraine parfois une perte considérable d'informations.

e. La segmentation

La segmentation des images consiste à créer une partition de l'image en région. Il s'agit simplement d'utiliser la complémentarité des sources d'informations pour mieux identifier les limites des composantes homogènes de l'image [**1]**.

Il existe de nombreuses méthodes de segmentation, que l'on peut regrouper en quatre principales classes :

- La segmentation fondée sur les régions.
- La segmentation fondée sur les contours.
- La segmentation fondée sur la classification ou le seuillage des pixels en fonction de leur intensité.
- La segmentation fondée sur la coopération entre les trois premières segmentations.

I.4.2 Traitements de haut niveau

Le traitement de haut niveau consiste à extraire des informations à partir d'une image numérique, parmi les techniques utilisées dans ce domaine, en trouve :

a. Extraction des caractéristiques

L'extraction de caractéristiques visuelles (ou visual features extraction en anglais) consiste en des transformations mathématiques calculées sur les pixels d'une image numérique. Les caractéristiques visuelles permettent généralement de mieux rendre compte de certaines propriétés visuelles de l'image, utilisées pour des traitements ultérieurs entrant dans le cadre d'applications telles que la détection d'objets ou la recherche d'images par le contenu [**3]**.

b. La reconnaissance de formes

La reconnaissance de formes est une branche de la vision artificielle, elle consiste à identifier des formes pré-décrites dans une image numérique, et par extension dans un flux vidéo numérique. Les formes recherchées sont souvent des formes géométriques, descriptibles par une formule mathématique, telles que : cercle ou ellipse, droite etc. Elles peuvent aussi être de nature plus complexe : lettre, chiffre, empreinte digitale, etc. Les algorithmes de reconnaissance peuvent

travailler sur des images en noir et blanc, avec en blanc les contours des objets se trouvant dans l'image. Ils peuvent aussi travailler sur des zones de l'image prédéfinies issues de la segmentation de l'image [**4]**.

I.5 Gestualité humaine et langues des signes

Les gestes sont un des moyens de communication les plus riches que l'être humain possède. Ils permettent d'agir sur le monde physique, et servent aussi à communiquer. De plus, le geste permet à la fois d'émettre des informations, mais aussi d'en recevoir.

Dans notre travail, on s'intéresse plus sur les gestes de la main, ou la main joue le rôle d'organe d'expression pour l'émission d'informations visuelles. Cela comprend la langue des signes, le geste Co-verbal, qui accompagne la parole, ou les gestes permettant une communication basique lorsqu'on ne peut pas utiliser la parole, comme dans un environnement bruité ou en plongée sous-marine [**5]**.

Les langues des signes désignent les langues gestuelles (produites par les mouvements des mains, du visage et du corps dans son ensemble) que les personnes atteintes de surdité ont développées pour communiquer. Elles assurent toutes les fonctions remplies par les langues orales [**5]**.

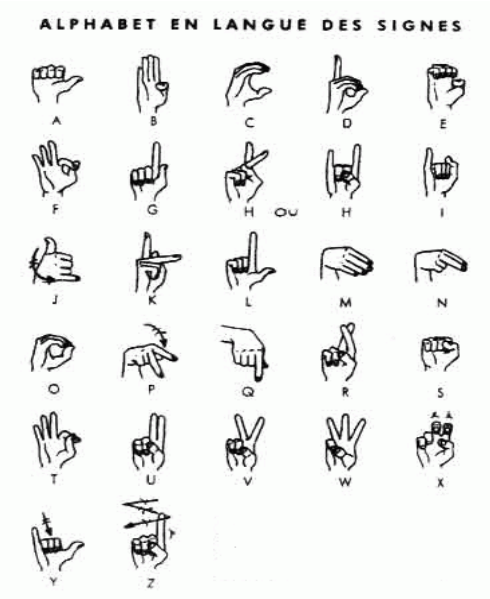


Figure I.5 *La représentation des 26 lettres alphabétiques en langue des signes Français* [**5]**.

I.6 Etat de l'art

La détection, et la reconnaissance des gestes de la main à fait l'objet de nombreux travaux de recherche ces dernières années, à travers cet état de l'art, nous allons présenter brièvement une méthode basée sur les paramètres de la forme **[6**, **14]**, dont les étapes principales sont les suivantes:

I.6.1 Traitement d'image et segmentation

La séquence d'entrée des images RGB doit d'abord être convertie en images YCbCr **[6]**, afin de surmonter les problèmes causés par la luminosité.

L'algorithme utilisé pour la segmentation est le K-means, l'image est segmentée en K Clusters, L'algorithme K-means minimise de façon itérative la somme des distances de chaque objet à son Clusters central, jusqu'à ce que la somme ne puisse pas être diminuée plus. Le résultat du regroupement K-means est un ensemble de clusters bien séparés des autres clusters. Dans ce cas, deux Clusters sont formés, le premier représente la région de la main, et l'autre représente l'arrière-plan. Les valeurs des pixels de chaque région vont être mises à 1, et 0, respectivement, afin d'avoir, à la fin, une représentation binaire de la main.

Après la segmentation, la seconde étape est la délimitation de la zone de la main, cela se fait en fixant les premiers pixels blancs rencontrés, en tant que points d'extrémités de la zone de la main, via un balayage en quatre sens gauche-droit, droit- gauche, haut-bas, bas-haut.

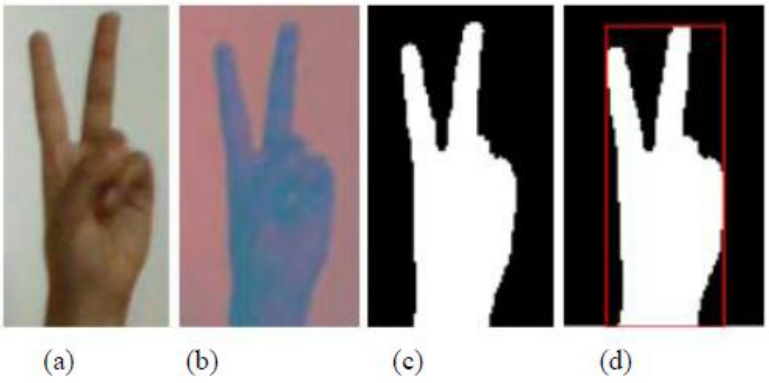


Figure I.6 *(a)Image RGB d'entrée (b) image YCbCr (c) représentation binaire de la main (d) représentation de la zone de la main en rouge* **[6]**.

I.6.2 Extraction des caractéristiques

Les éléments clefs utilisés pour l'extraction des points caractéristiques sont :

a. Le centre de la masse

Ce centre est calculé, afin de partitionner la main en deux régions celle des doigts, et sans doigts.

b. Détection du pouce

La détection du pouce est considérée comme l'une des parties principales de l'extraction des points caractéristiques, vu que le pouce représente une forme de caractéristique très importante dans la gestualité de la main.

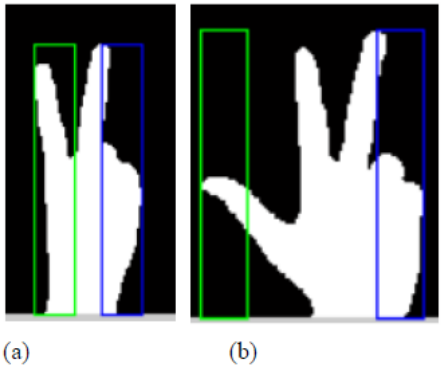


Figure I.7*(a)Le pouce n'est pas détecté (b) Le pouce est détecté* **[6]**.

c. Détection de la région des doigts

La région des doigts est obtenue à partir du centre de la masse. Dans cette étape, le nombre et les points d'extrémités des doigts levés, et les doigts pliés, spécifiques à chaque geste de la main, seront calculés en utilisant la distance euclidienne.

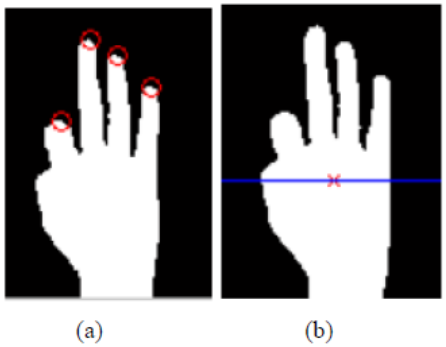


Figure I.8 *(a)Représentation des extrémités des doigts en rouge (b) Représentation du centre de la masse, et la ligne de séparation en bleu* **[6].**

I.6.3 Interprétation

Les gestes de la main sont classés en fonction des caractéristiques calculés dans la section précédente. Une séquence binaire de 5 bits est générée dans laquelle chaque position d'un bit correspond à un doigt. La **Figure I.9** illustre un exemple de trois doigts levés, sans la présence du pouce, avec la séquence binaire [01110] comme une interprétation:

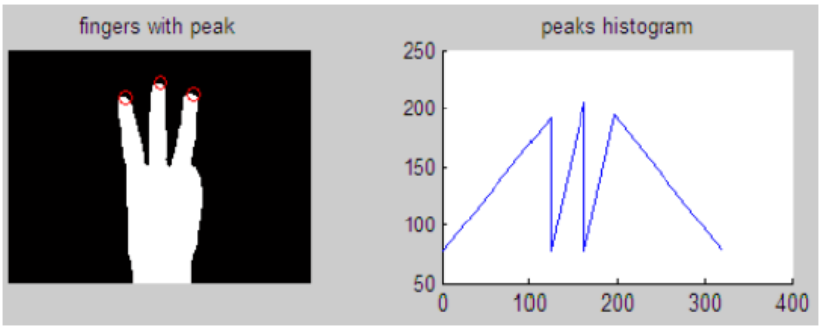


Figure I.9 *exemple d'interprétation d'un geste de la main* **[6].**

I.7 Conclusion

Nous avons présenté dans ce chapitre quelques définitions de bases sur l'image numérique, et le traitement d'image, comme nous avons projeté la lumière sur la vision par ordinateur et son interaction avec la gestualité humaine à travers un état de l'art.

De ce fait, dans le prochain chapitre, nous allons présenter notre approche utilisée pour la reconnaissance en temps réel des signes de la main.

Chapitre II

Approche utilisée et résultats expérimentaux

II.1 Introduction

De nos jours, la technologie basée sur la vision de la reconnaissance des gestes de la main, joue un rôle important dans notre vie quotidienne, passant par nos appareils téléphoniques intelligents (aussi appelés Smartphone), jusqu'à notre empreinte biométrique.

À cet effet, le but de ce dernier chapitre est de lever le voile sur cette technologie, en expliquant, et en détaillant les moindres principes de fonctionnement de notre approche utilisée pour la reconnaissance en temps réel des signes de la main, et de son application. En clôturant le chapitre par des tests expérimentaux, et une discussion exhaustive des résultats.

II.2 l'Approche utilisée

Notre approche utilisée consiste à faire une reconnaissant vidéo en temps réel des signes de la main, en passant par quatre phases majeures, la détection de la main, l'extraction des points caractéristiques de la main spécifiques à chaque signe, puis l'apprentissage et la construction du modèle de classification, et à la fin en passe au test et la prédiction.

Le diagramme suivant sur la **Figure II.1** donne une explication généralisée du mode de fonctionnement de l'approche utilisée :

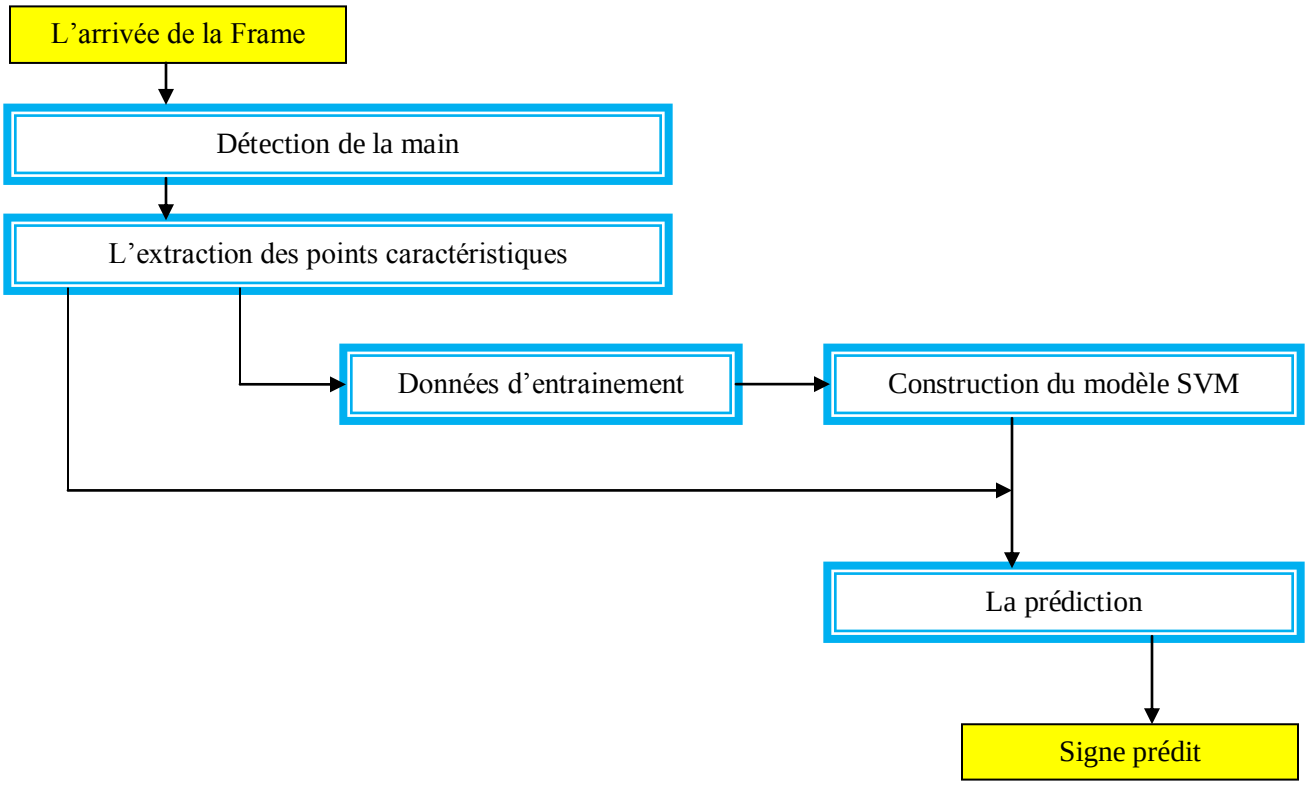


Figure II.1 *Diagramme explicatif de l'approche utilisée.*

II.2.1 La phase de détection

La détection de la main se fait en quatre phases secondaires illustrées dans la **Figure II.2** :

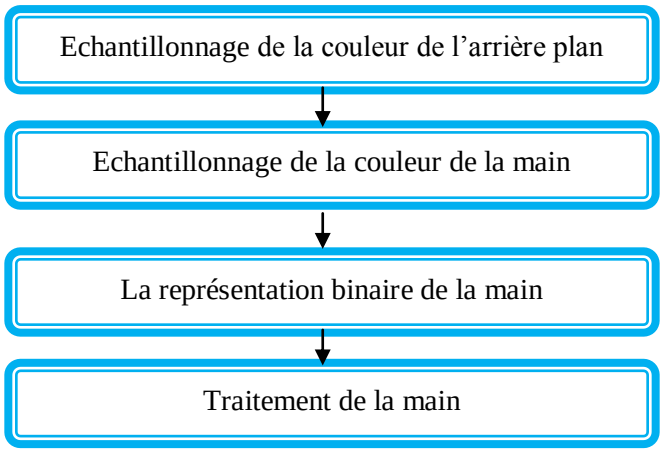


Figure II.2 *Illustration des phases de la détection de la main.*

a. Echantillonnage de la couleur de l'arrière plan

L'échantillonnage se fait à l'aide des carrés pour la prise des différents couleurs de l'arrière plan, dans ce cas, plus le nombre des carrés est élevé, plus l'échantillonnage est bon.

b. Echantillonnage de la couleur de la main

L'échantillonnage de la main se fait de la même façon expliquée précédemment, sauf que les carrés sont placés d'une façon différente, très rapproché l'un de l'autre, afin que la prise de la couleur cible juste la couleur de la peau de la main, sans se confondre avec la couleur de l'arrière plan.

c. La représentation binaire de la main

Une fois l'échantillonnage des couleurs est terminé, des profils de couleur seront crées selon le nombre des carrés utilisés.

Chaque couleur du profil produit une image binaire, qui vont être toutes additionnées à l'aide de l'opérateur « OU ». La même chose se fait pour l'arrière plan. À la fin, une opération logique « ET » sera faite entre les deux images binaires résultantes, afin d'obtenir une représentation binaire de la main lisse et sans bruit, la **Figure II.3** illustre la phase de la représentation:

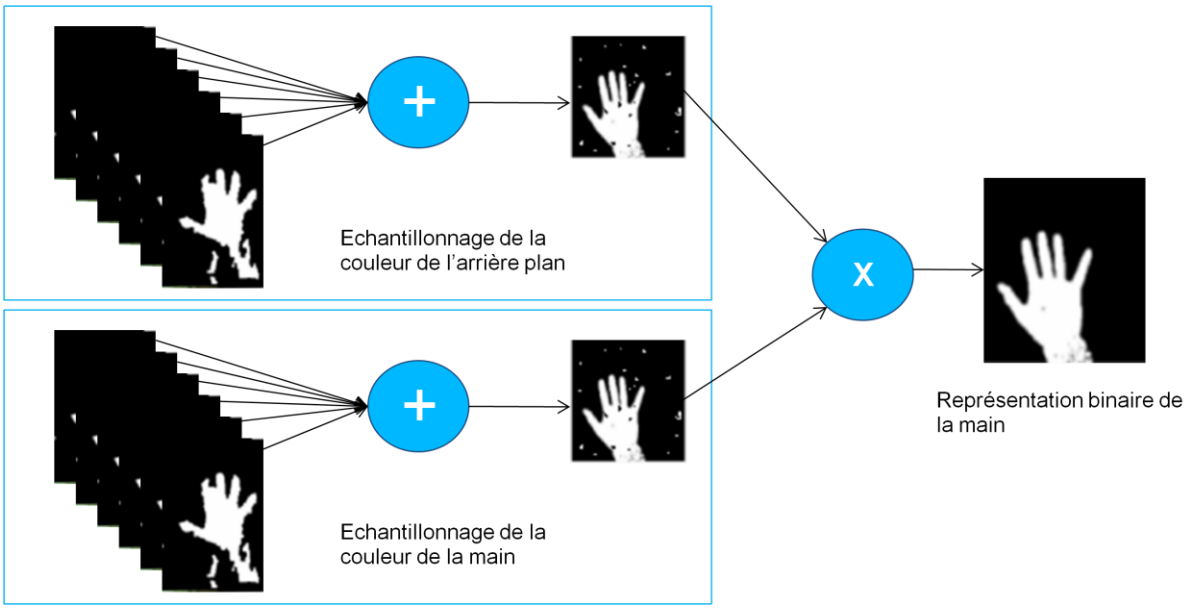


Figure II.3 *Illustration de la phase de la représentation binaire*

d. Traitement de la main

Une fois la représentation finale de la main est faite, la main sera traitée suivant plusieurs étapes. Le but de ce traitement est la préparation des éléments nécessaires pour le calcul des points caractéristiques spécifiques à chaque signe.

L'originalité de la technique du traitement de la main revient à Zhi-hua Chen **[7**, **15]** :

1) **Le contour de la main :** Afin de trouver le contour de la main dans l'image finale résultante, il suffit juste de chercher le plus grand contour, la **Figure II.4** illustre le contour de la main en vert.

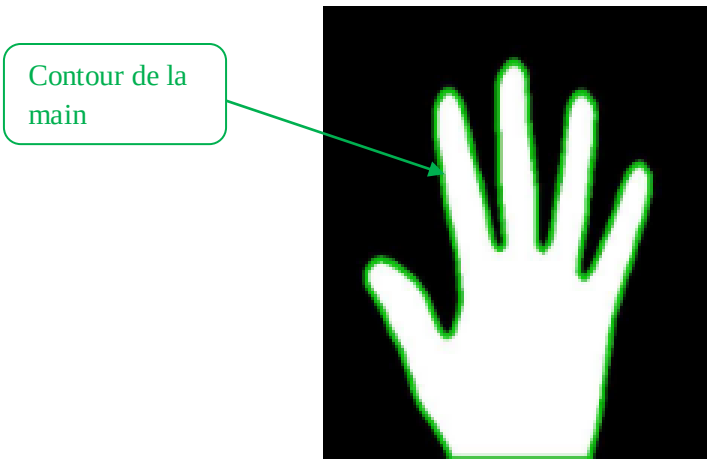


Figure II.4 *Représentation du contour de la main en vert.*

2) **L'enveloppe convexe du contour (ou Convex Hull) :** Le but de l'utilisation de l'enveloppe convexe, c'est la détection des points convexes, et les points défauts de convexité par la suite. La **Figure II.5** illustre l'enveloppe convexe de la main en rouge.

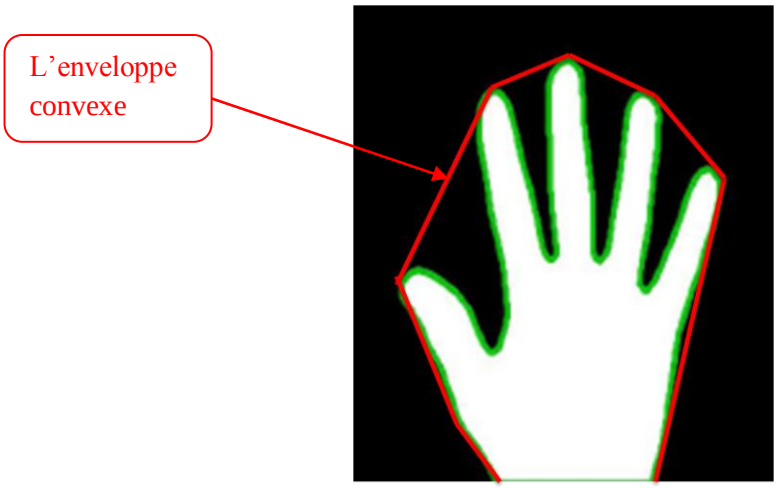


Figure II.5 *Représentation de l'enveloppe convexe du contour de la main en rouge.*

3) **Les points convexes :** Les points convexes jouent un rôle primordial dans cette technique pour l'énumération des doigts, et la construction des vecteurs des doigts, afin d'extraire les points caractéristiques par la suite, la **Figure II.6** présente les points convexes en couleur jaune.

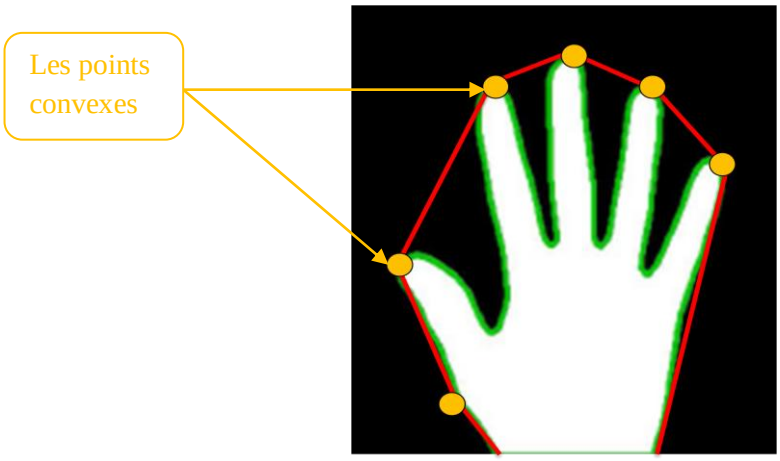


Figure II.6 *Représentation des points convexes de l'enveloppe convexe du contour de la main en jaune.*

4) **Les points des défauts de convexité (ou convexity defects) :** Ces points vont êtres utilisés pour le calcul des deux clefs nécessaires pour l'extraction des caractéristiques, la profondeur et l'angle illustrés dans la **Figure II.10**. La **Figure II.7** illustre ces points en bleu.

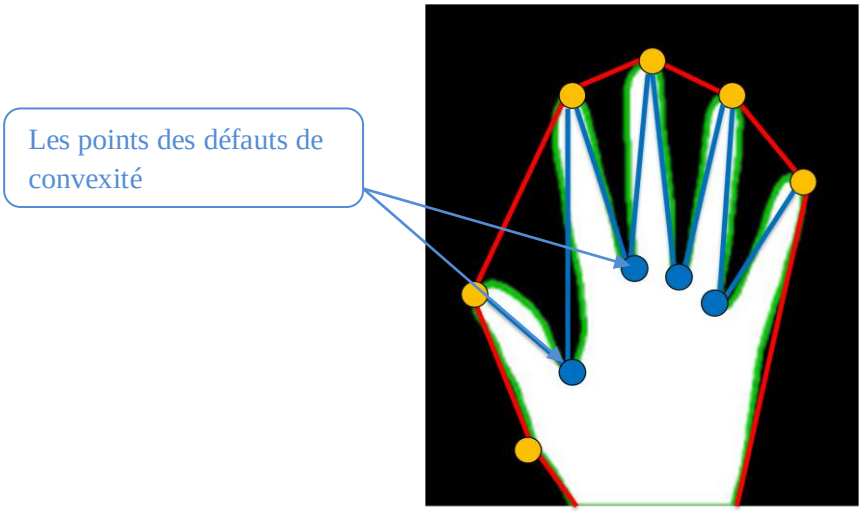


Figure II.7 *Représentation des points défauts de convexité.*

5) **La délimitation de la zone de la main :** Cette étape permet de limiter, et déterminer l'espace des points coordonnées qui vont êtres utilisés pour la prise des points caractéristiques. La méthode utilisée est la même que celle déjà vue dans la section (**I.6.1 Traitement d'image et segmentation**) La **Figure II.8** présente la zone de la main, en orange.

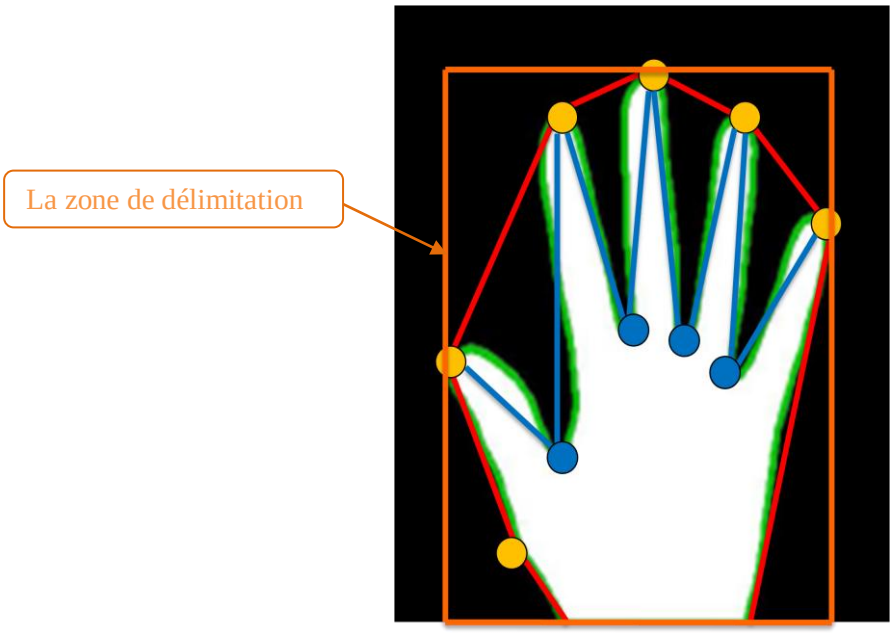


Figure II.8 *Représentation la zone de la main en orange.*

6) **Le centre de la paume de la main et le cercle central :** Pour calculer ce centre il suffit de parcourir tout les pixels blancs (à l'intérieure du contour de la main) tout en conservant la distance entre chaque pixel et le point le plus proche du contour. La plus grande distance trouvée représentera le rayon du cercle central.

   Ce cercle est utilisé pour rassurer l'unicité des caractéristiques spécifiques à chaque signe **Figure II.9** :

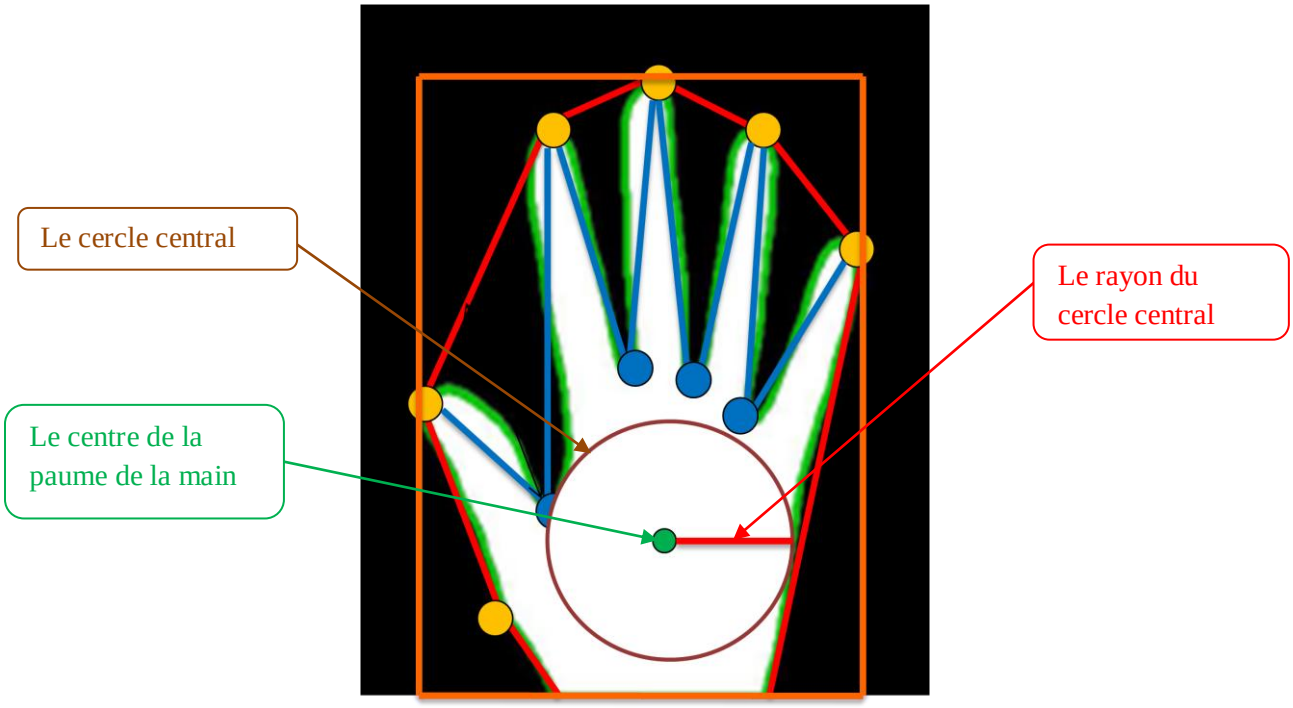


Figure II.9 *Représentation du cercle central et le centre de la paume de la main.*

1) **Filtrage des points défauts de convexité :** C'est l'étape la plus importante qui va nous servirait beaucoup par la suite à définir le nombre, la localisation, et les vecteurs des doigts, pour extraire les points caractéristiques des signes de la main. Le point sera rejeté s'il ne vérifier pas les contraintes suivantes (**II.1**), (**II.2**), et (**II.3**) proposées par Zhi-hua Chen **[7, 15]** :

$$\boldsymbol{Profondeur} > \textit{le } \boldsymbol{rayon} \textit{ du cercle} * 0.7 \quad \textbf{(II.1)}.$$

$$Cos(\boldsymbol{Angle}) \geq -0.7 \ \textbf{(II.2)}.$$

$$\boldsymbol{L'index\ du\ point\ défaut} \leq 4 \ \textbf{(II.3)}.$$

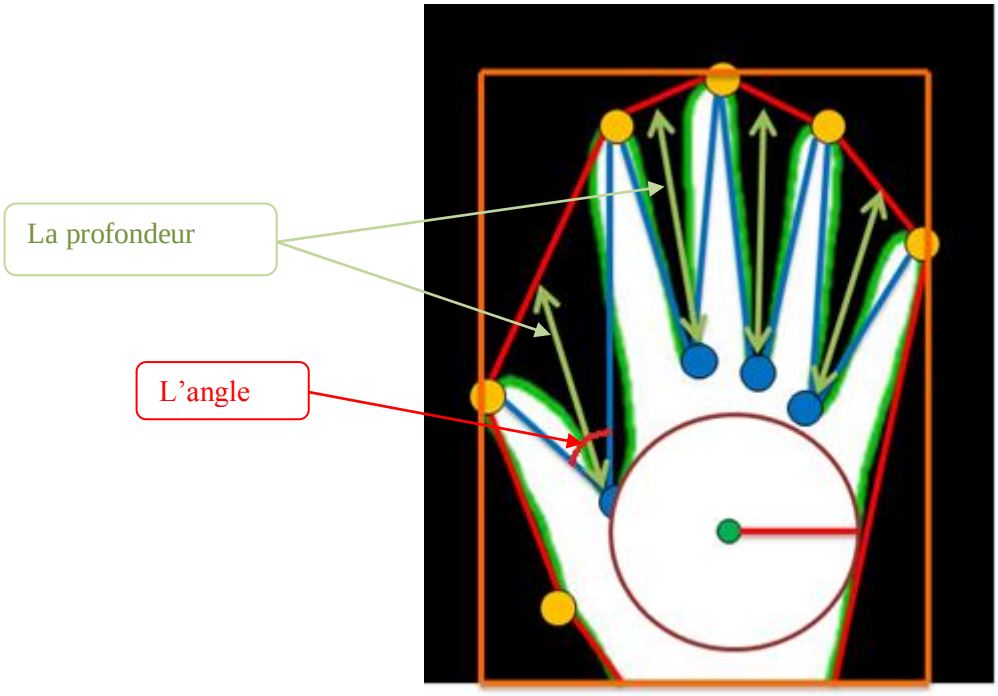


Figure II.10 *Représentation des contraints d'élimination des points défaut de convexité non-importants.*

II.2.2 L'extraction des points caractéristiques

Les points caractéristiques des signes de la main sont représentés par les vecteurs des doigts, calculés essentiellement en fonction du rayon du cercle central, et le reste des éléments vu dans la section précédente, chaque vecteur contiens les cordonnées des points des extrémités des doigts divisés par le rayon du cercle central, comme le montre l'équation (**II.4)**. Le centre du cercle central représente l'origine du repaire des cordonnées :

$$V_0 = [\frac{x_0}{r}, \frac{y_0}{r}, \frac{x_1}{r}, \frac{y_1}{r}, \frac{x_2}{r}, \frac{y_2}{r}, \frac{x_3}{r}, \frac{y_3}{r}, \frac{x_4}{r}, \frac{y_4}{r}]$$ **(II.4)**.

Tel que :

r *est le rayon du cercle central.*

x_i ,y_i *ce sont les cordonnées des points d'extrémités des doigts, avec* $i \in [0, \dots, 4]$.

Ces points caractéristiques vont être reformulés en forme de chaines de caractères avec le numéro d'identification du geste correspondant, puis stockés dans un fichier Texte en tant que donnés d'apprentissage. Cette reformulation repend aux besoins de notre classificateur utilisé. L'équation (**II.5**) représente un exemple d'échantillon d'apprentissage :

$$
\begin{array}{l}
Signe_1 \cdot Nb\_trame \left\{ \begin{array}{l} Signe_1 \;\; 1{:}\frac{x_0}{r} \;\; 2{:}\frac{y_0}{r} \;\; 3{:}\frac{x_1}{r} \;\; 4{:}\frac{y_1}{r} \ldots 10{:}\frac{y_4}{r} \\ \ldots. \\ Signe_1 \;\; 1{:}\frac{x_0}{r} \;\; 2{:}\frac{y_0}{r} \;\; 3{:}\frac{x_1}{r} \;\; 4{:}\frac{y_1}{r} \ldots 10{:}\frac{y_4}{r} \end{array} \right. \\
\ldots\ldots \\
Signe_n \cdot Nb\_trame \left\{ \begin{array}{l} Signe_n \;\; 1{:}\frac{x_0}{r} \;\; 2{:}\frac{y_0}{r} \;\; 3{:}\frac{x_1}{r} \;\; 4{:}\frac{y_1}{r} \ldots 10{:}\frac{y_4}{r} \\ \ldots. \\ Signe_n \;\; 1{:}\frac{x_0}{r} \;\; 2{:}\frac{y_0}{r} \;\; 3{:}\frac{x_1}{r} \;\; 4{:}\frac{y_1}{r} \ldots 10{:}\frac{y_4}{r} \end{array} \right.
\end{array}
\qquad \textbf{(II.5)}.
$$

Tel que :

Nb_trame *est le nombre de trames.*

$Signe_i$ *est le numéro d'identification du signe (geste) avec* $i \in [1, \ldots, n]$.

$\boldsymbol{n}$ *est le nombre total des signes dans la base.*

II.2.3 Classification et prédiction

L'Algorithme de classification que nous utilisé est le Multi-class SVM (Machines à vecteurs de support multi classes), amélioré avec les algorithmes SMO (Sequential minimal optimization) **[16, 8, 9]** pour la résolution des problèmes quadratiques, afin d'avoir une meilleure optimisation. Implémenté par Chih-Chung Chang et Chih-Jen Lin **[10**, **17]** voir la section (**II.3.3 Bibliothèque et logiciels utilisés**).

Notre choix s'explique par la réputation, la facilité de mise en place, l'efficacité, et la rapidité de cet algorithme.

a. Modèle SVM utilisé

Le modèle SVM que nous avons utilisé est le C-SVC (C-Support Vector Classication) pour l'utilisation multi-classe. Il permet à l'algorithme de séparer clairement des échantillons qui sont séparés par une marge très étroite. L'entrainement consiste à minimiser la fonction d'erreur **[11, 12]**:

$$\min_{\omega, b, \xi} \quad \frac{1}{2}\omega^T \omega + C \sum_{i=1}^{l} \xi_i \quad (\mathbf{II.6}).$$

Tel que : $Y_i(\omega^T \phi(x_i) + b) \geq 1 - \xi_i ; \xi_i \geq 0, \ i = 1 \dots l$ (**II.7**).

ω *est le vecteur normal de l'hyperplan à l'origine.*

C *est une constante permettant de contrôler le compromis entre nombre d'erreurs de classement, et la largeur de la marge.*

ξ *est la variable d'écart.*

Si l'algorithme SVM n'arrive pas à définir une marge claire, il utilise le paramètre de coût C pour permettre certaines erreurs d'entrainement, et produire une marge souple. Si la valeur du coût est trop élevée, elle interdit les erreurs d'entrainement, produisant une marge étroite, et une classification rigide **[11, 12]**.

L'algorithme détermine la classe d'un échantillon inconnue, en le comparant avec les vecteurs du support des échantillons d'entrainement, la fonction utilisée pour classifier l'échantillon x est (**II.8**) **[11, 12]**:

$$sgn\left(\sum_{i=1}^{l} y_i \propto_i K(x_i, x) + b\right) \quad (\mathbf{II.8}).$$

Avec :

K *est la fonction du noyau.*

y_i *est la classe d'association (-1 ou +1).*

$\propto_i$ *Le coefficient du poids.*

x_i *Les vecteurs du support.*

b est la distance entre l'hyperplan et l'origine.

b. L'approche de classification Multi-classe utilisée

L'approche utilisée pour la Multi-classe classification est la « Un contre un » **[12]**, par exemple, si le nombre de classe est k alors $k(k-1)/2$ classificateurs seront construit, et chacun entraine les données à partir de deux classes, Pour l'entrainement de la ième classe et jème classe, une simple Deux-classe classification sera utilisé **[11, 12]**:

$$min \quad \frac{1}{2}(\omega^{ij})^T\omega^{ij} + C\sum_t(\xi^{ij})_t \text{ (II.9)}.$$
$$w^{ij}, b^{ij}, \xi^{ij}$$

Tel que $(\omega^{ij})^T\phi(x_t) + b^{ij} \geq 1 - \xi_t^{ij}$ si x_t est dans la ième classe (**II.10**).

$(\omega^{ij})^T\phi(x_t) + b^{ij} \geq -1 + \xi_t^{ij}$ si x_t est dans la jème classe (**II.11**).

$\xi_t^{ij} \geq 0$ (**II.12**).

Pour la classification, la stratégie du vote est utilisée, en cas ou deux classes possèdent un vote identique, tout simplement, la classe qui apparait la première dans le tableau de stock des noms des classes, sera choisie **[11, 12]**.

c. Choix du noyau

Nous avons utilisé le noyau linéaire (**II.13**) et le noyau Radial Gaussien RBF (**II.14**) pour sa grande réputation, et son efficacité:

$$k(x_i, x_j) = x_i \cdot x_j \text{ (II.13)}.$$

$$k(x_i, x_j) = e^{-\gamma\left\|x_i - x_j\right\|^2} \text{ (II.14)}.$$

II.3 Implémentation

Nous avons choisi d'implémenter notre approche sous Android, notre choix s'explique par une utilisation accrue de ce dernier partout à travers la planète, comme il à réussi quelque chose d'historique en surpassant le système d'exploitation Windows dans une période très courte, et en devenant l'OS le plus populaire pour surfer sur le net en 2016 **[13, 18]**, **Figure II.11**.

De plus la portabilité des appareils mobiles, l'intégrité de leurs cameras puissantes, et la souplesse de leur manipulation, nous ont beaucoup aidé, de mener à bien la mise en œuvre, et le test de notre application.

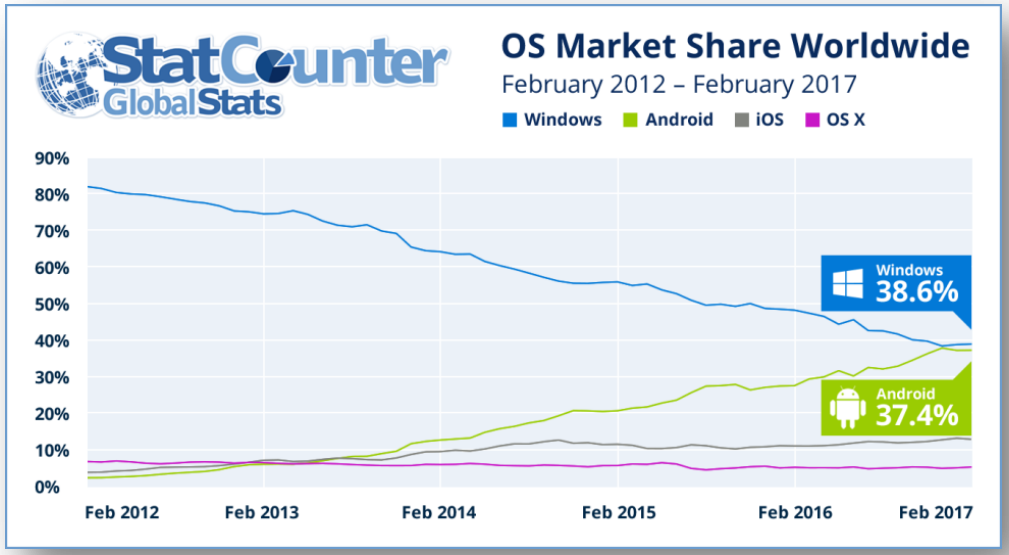


Figure II.11 *L'analyse de StatCounter sur les parts de marché des systèmes d'exploitation mobiles pour la navigation sur internet.* **[18]**

II.3.1 Organigramme de fonctionnement général de notre application

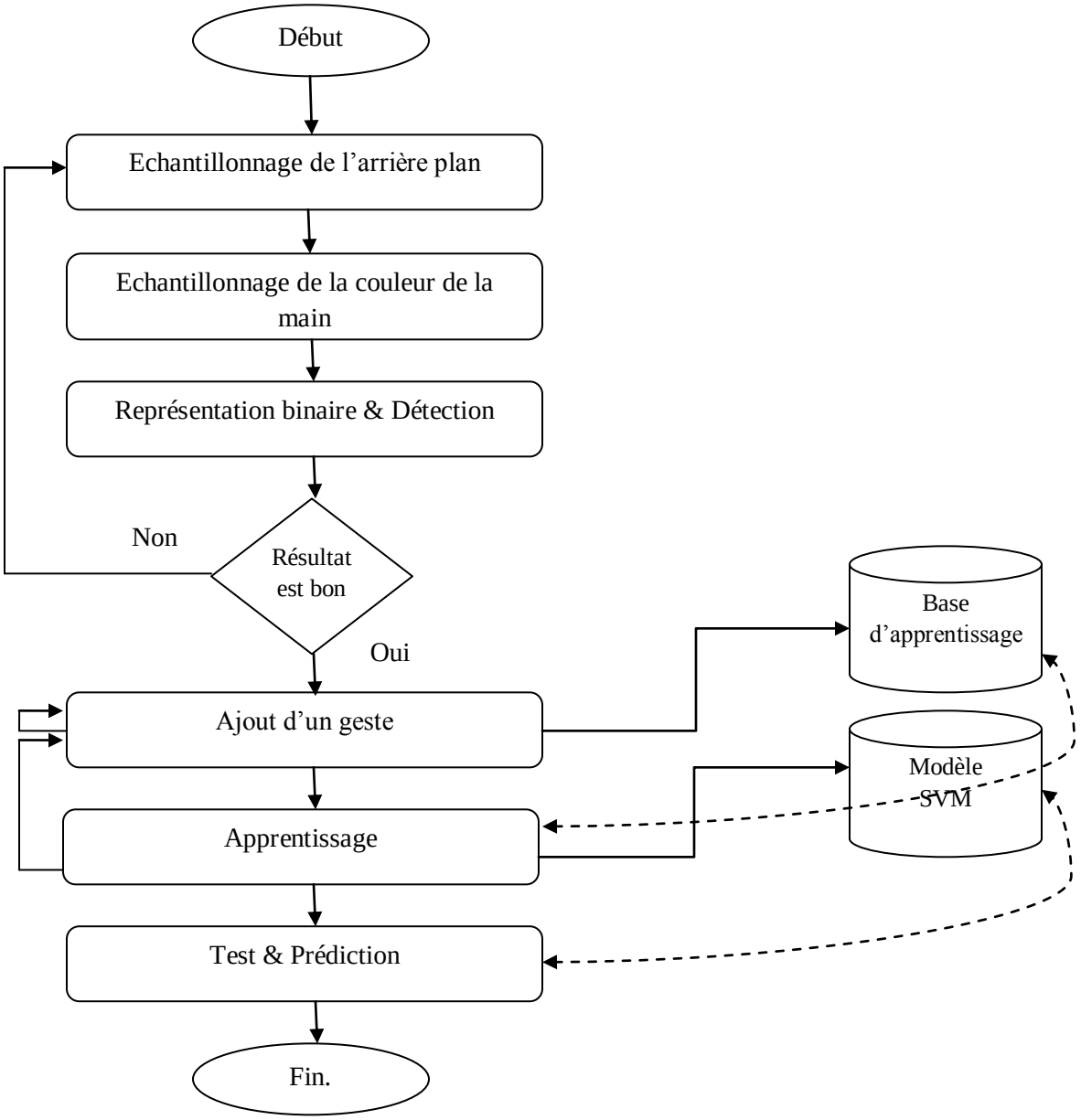


Figure II.12 *Organigramme de fonctionnement général de l'application.*

II.3.2 Présentation de l'application

a. Interface principale

L'interface principale de notre application, démarre sous le mode d'échantillonnage de la couleur de l'arrière plan (mode 1) comme le montre la **Figure II.13** :

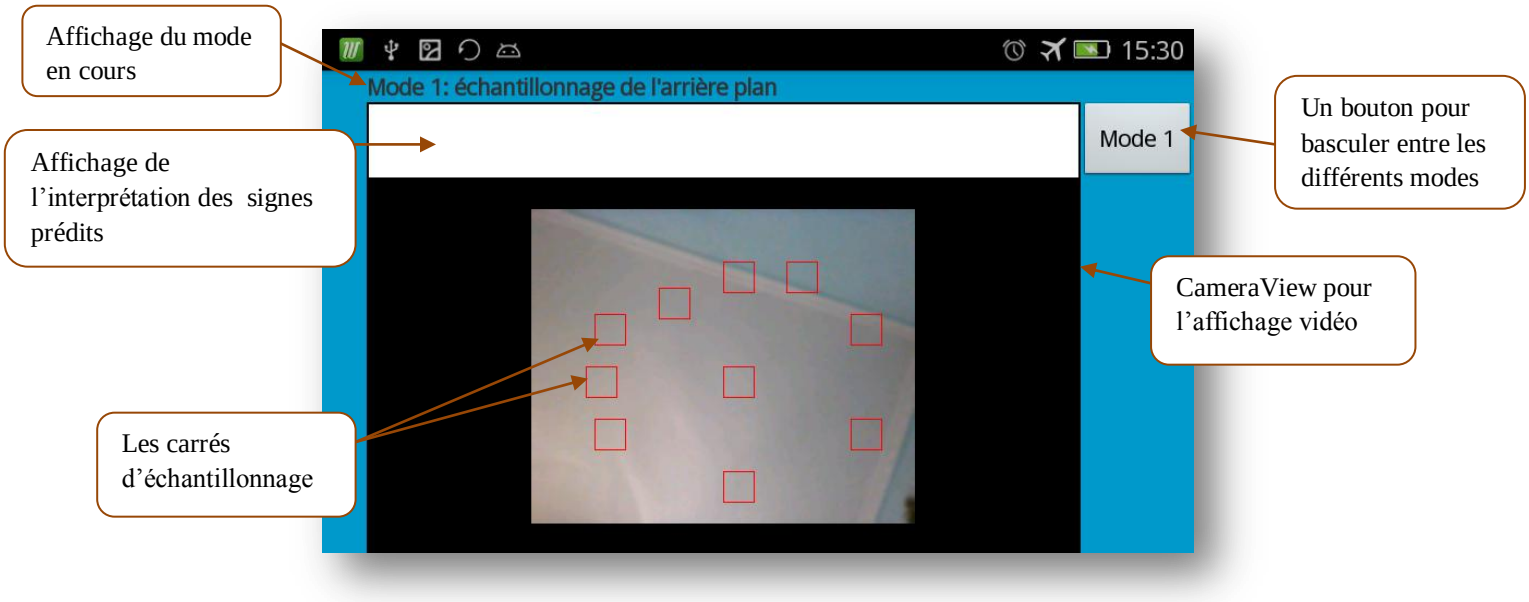


Figure II.13 *Prise d'écran de l'interface graphique principale de notre application (Main Activity).*

Pour un meilleur résultat il est préférable que la couleur du fond soit unie.

b. Mode d'échantillonnage de la main

L'échantillonnage de la couleur de la main se fait de la même façon que celle de l'arrière plan. Afin de réussir un bon échantillonnage, et des bons résultats, la main doit couvrir tout les carrés d'échantillonnage, comme il est montré sur la **Figure II.14**:

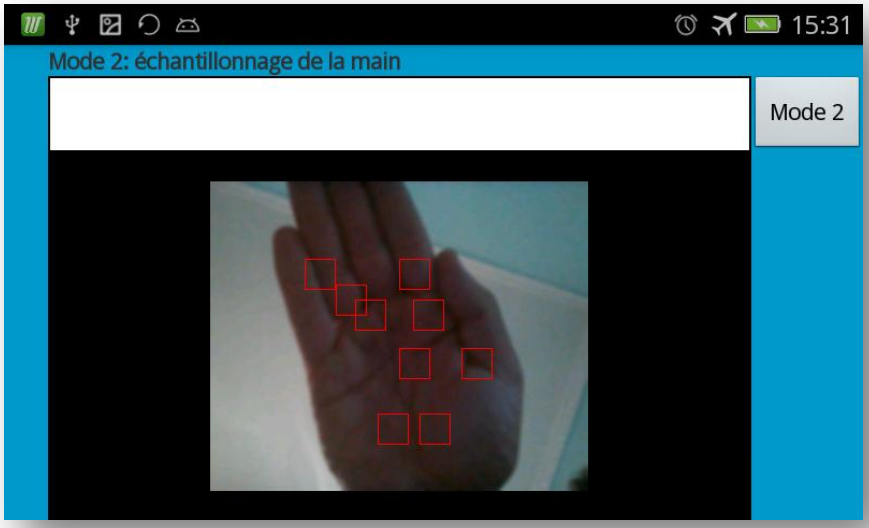


Figure II.14 *Prise d'écran lors du mode d'échantillonnage de la main.*

c. Mode de la représentation binaire

Une fois l'échantillonnage est terminé, on passe au troisième mode, pour la représentation binaire de la main en Noir et Blanc comme le montre la **Figure II.15** :

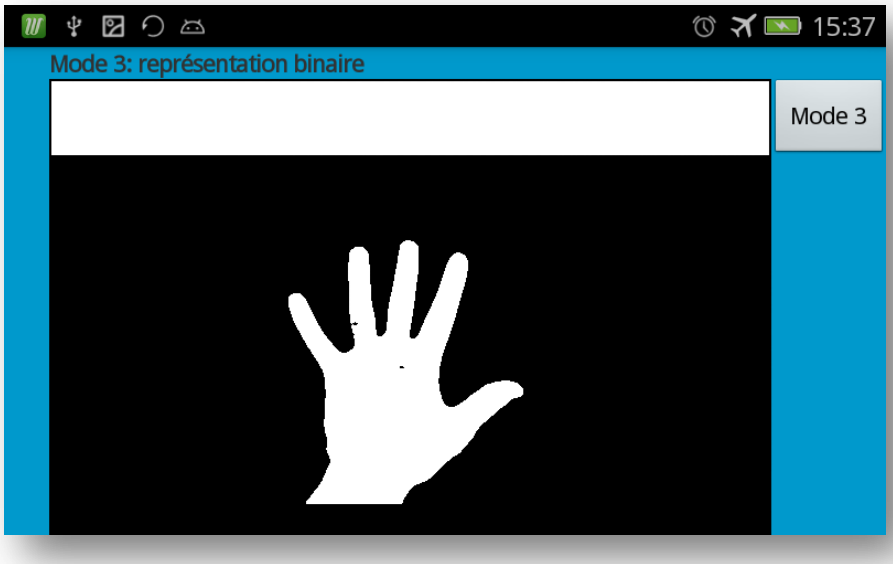


Figure II.15 *Prise d'écran lors de la représentation binaire de la main.*

d. Mode de détection et la classification en temps réel

Une fois dans le mode de la détection de la main (Mode 4), trois boutons supplémentaires vont apparaitre, le premier est pour l'ajout d'un nouveau signe désiré dans la base d'entrainement, le deuxième est pour faire l'apprentissage et la génération du modèle SVM, et le troisième c'est pour passer au test et la prédiction.

Les doigts détectés vont être numérotés, et les points d'extrémités de ces derniers vont être présentés en rouge, les points défauts, et le centre du cercle en vert, le cercle central en noire, et les vecteur des doigts en bleu. Comme le montre la **Figure II.16** :

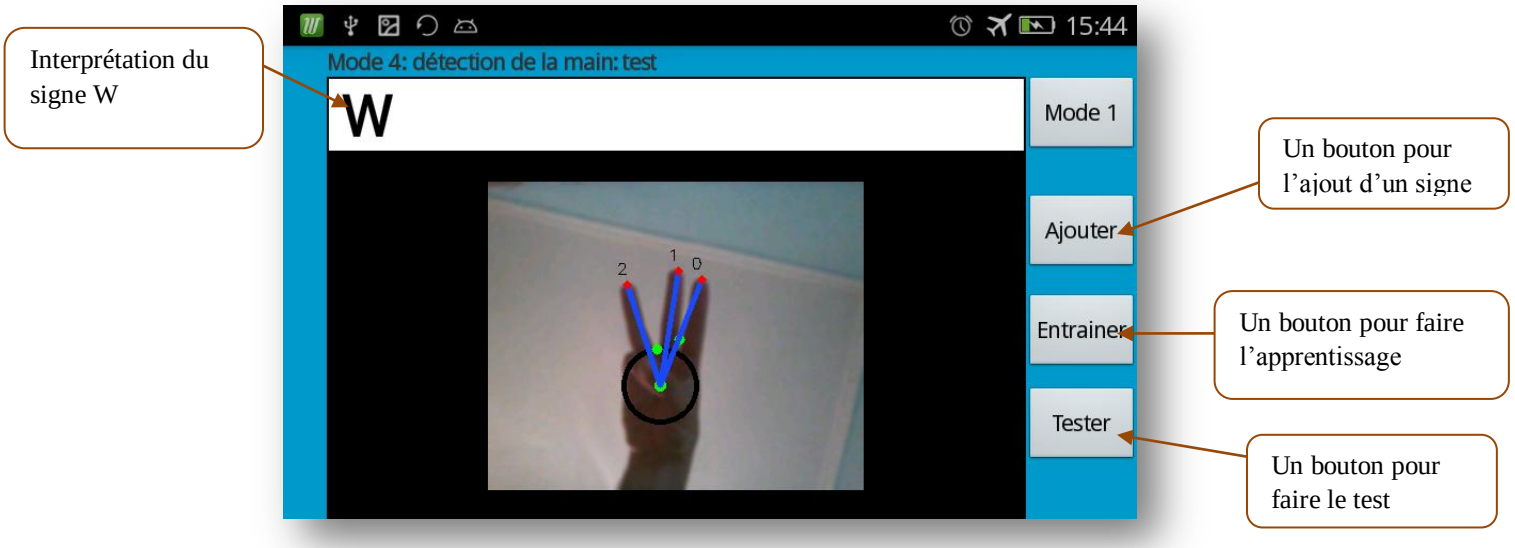


Figure II.16 *Prise d'écran lors du test (Interprétation de la lettre W dans la langue des signes Français.*

e. Menu

Notre application est dotée d'un menu qui offre à l'utilisateur les possibilités suivantes :

- Changer la source vidéo, en basculant entre les différentes cameras qui existent sur l'appareil utilisé.
- Configurer manuellement les différents paramètres utilisés pour la détection, et l'apprentissage.

- Changer la résolution vidéo.
- La gestion des données, cette option permet l'affichage et la suppression des données de la base d'entrainement, **Figure II.18**.
- Réinitialisation et mise à zéro, cette option permet la suppression de toutes les données, et le modèle SVM, comme elle remet tout les paramètres par défaut.

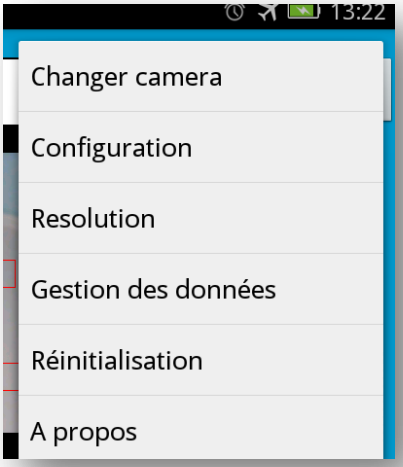


Figure II.17 *Prise d'écran du Menu de l'application.*

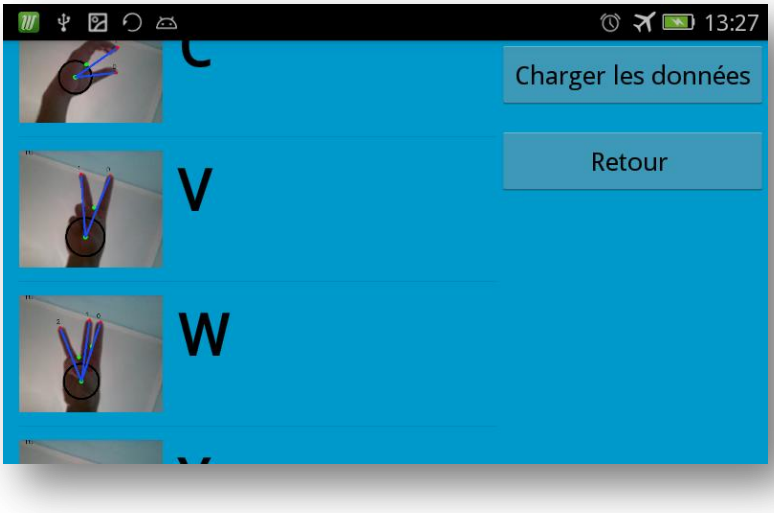


Figure II.18 *Prise d'écran de la section de la gestion des données.*

II.3.3 Bibliothèques et logiciels utilisés

Pour assurer la bonne réalisation, et le bon déroulement de notre projet, nous avons utilisé un ensemble de logiciels, dont les plus importants sont les suivants :

a. Niveau Android ciblés :

Nous avons ciblé le niveau 19 d'Android 4.4, KitKat (2013), puis qu'il est encore largement utilisé, comme il contient tout les packages nécessaires pour le développement, et l'utilisation de notre application.

b. Le compilateur NDK (Native Development Kit) pour Android :

Le NDK d'Android il permet de compiler des codes source écrits en C ou C++ pour Android. Dans notre cas, il nous a permet d'utiliser la Bibliothèque LIBSVM programmée en C++.

c. IDE utilisé :

Nous avons utilisé Android Studio 2.3 pour le développement de notre application, c'est un nouvel environnement pour développement et programmation entièrement intégré qui a été récemment lancé par Google pour les systèmes Android. Il a été conçu pour fournir un environnement de développement et une alternative à Eclipse qui est l'IDE le plus utilisé.

d. Bibliothèque SVM utilisée :

La bibliothèque SVM utilisée est celle de Chih-Chung Chang et Chih-Jen Lin de l'université national de Taiwan, LIBSVM version 3.22 **[10**, **17].**

C'est une implémentation de l'Algorithme SVM amélioré avec les algorithmes SMO, largement réputée pour son efficacité, et sa rapidité. Et puise qu'elle s'exécute sous l'Android JNI framework (C++), ça nous a beaucoup facilité les taches au niveau de la classification et la prédiction en temps réel.

e. OpenCV :

Une grande partie de notre application repose sur l'utilisation de la bibliothèque spécialisée dans le Computer Vision, OpenCV version 2.4.9 pour Android.

L'utilisation de notre application nécessite aussi l'installation de OpenCV Manager, c'est un service Android pour la gestion des bibliothèques binaires de l'OpenCV, il permet aussi de partager les bibliothèques dynamiques entre les différentes applications dans le même appareil, en même temps. Comme il offre les avantages suivants **[19, 20]** :

- Moins d'utilisation de la mémoire.
- Optimisations spécifiques aux ressources matérielles pour toutes les plates-formes prises en charge.
- Mises à jour régulières et corrections de bugs.

Dans notre projet, nous avons utilisé la version 3.0 de OpenCV Manager, **Figure II.19**:

Figure II.19 *Une prise d'écran de l'interface OpenCV Manager que nous avons utilisé.*

II.3.4 Environnement du travail

Afin de mener à bien le développement et les tests de notre application, nous avons utilisé un ensemble de matériels, dont les caractéristiques principales sont :

a. Machine utilisée pour le développement :

Ordinateur Portable HP ProoBook, avec la configuration suivante :

- Processeur : Intel® Core™ I5-2430M 2.40 GHz.
- RAM : 04 Gb.
- Système d'exploitation : Microsoft Windows 7 Professional x64-based V6.1.7600.

b. Appareil Android utilisé pour le test:

Téléphone portable HUAWEI Y336-U02, avec la configuration suivante :

- Processeur : Quad Core à 1.2 Ghz.
- RAM : 512M
- Camera : arrière 5MPx, et l'avant 2MPx
- Résolution : 2592x1944px
- Système d'exploitation : Android 4.4.2.

II.4 Tests et résultats

Notre test s'articule sur deux scénarios, avec des configurations différentes au niveau du noyau SVM utilisé, et ses paramètres. Les deux scénarios se basent sur la détection, et la reconnaissance des signes alphabétiques illustrés précédemment dans le **Chapitre I** sur la **Figure I.3**.

II.4.1 Configuration

Le tableau suivant **Table II.1** présent en détaille les paramètres utilisés dans chaque scénario :

	Scénario 1	Scénario 2
Noyau	Gaussien	Linéaire
Nombre de trames	10	10
Nombre de carrés d'échantillonnage	10	10
Le paramètre Gamma	0.001	-
La valeur du coût C	4	4
Nombre de classe	26	26

Table II.1 *Configuration spécifique à chaque scénario du test.*

II.4.2 Extraction des points caractéristiques et préparation de la base d'apprentissage

Le tableau suivant **Table II.2** présente un échantillon sur les points caractéristiques extraits de la première trame arrivée, correspondants à chaque signe alphabétique ajouté à la base d'apprentissage. Ces valeurs sont calculées en utilisant la méthode expliquée précédemment dans la section (**II.2.2 L'extraction des points caractéristiques)** :

		Doigt 1	Doigt 2	Doigt 3	Doigt 4	Doigt 5
1	**A**	x/r = 2.2712 y/r = -0.1936	x/r = 0.7218 y/r = -1.2500	Non détecté	Non détecté	Non détecté
2	**B**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
3	**C**	x/r = 2.3237 y/r = -0.6699	x/r = 1.9259 y/r = -2.0725	Non détecté	Non détecté	Non détecté
4	**D**	x/r = 1.4102 y/r = -0.4230	x/r = -0.3424 y/r = -2.4377	Non détecté	Non détecté	Non détecté
5	**E**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
6	**F**	x/r = 1.8422 y/r = -0.6663	x/r = 0.3919 y/r = -2.5869	Non détecté	Non détecté	Non détecté
7	**G**	x/r = 0.7367 y/r = -2.7574	x/r = -0.8209 y/r = -0.8221	Non détecté	Non détecté	Non détecté
8	**H**	x/r = 1.5648 y/r = -2.5863	x/r = -0.6954 y/r = -2.3255	Non détecté	Non détecté	Non détecté
9	**I**	x/r = 0.7995 y/r = -1.1993	x/r = -1.0661 y/r = -2.1322	Non détecté	Non détecté	Non détecté
10	**J**	x/r = 1.4339 y/r = 1.7714	x/r = 1.4761 y/r = -0.2530	Non détecté	Non détecté	Non détecté
11	**K**	x/r = 2.4693 y/r = -1.4770	x/r = 0.7615 y/r = -3.6463	Non détecté	Non détecté	Non détecté
12	**L**	x/r = 2.0629 y/r = -1.1064	x/r = 0.6282 y/r = -2.8224	x/r = -0.7314 y/r = -0.8626	Non détecté	Non détecté
13	**M**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
14	**N**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
15	**O**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
16	**P**	x/r = 2.0054 y/r = 1.1193	x/r = 2.2619 y/r = -0.6995	x/r = 2.1453 y/r = -2.5883	Non détecté	Non détecté
17	**Q**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
18	**R**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
19	**S**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
20	**T**	x/r = 1.5913 y/r = -0.5304	x/r = 0.5508 y/r = -2.7134	Non détecté	Non détecté	Non détecté
21	**U**	Non détecté	Non détecté	Non détecté	Non détecté	Non détecté
22	**V**	x/r = 1.5281 y/r = -3.3734	x/r = 0.0576 y/r = -3.7194	Non détecté	Non détecté	Non détecté
23	**W**	x/r = 1.2393 y/r = -3.7179	x/r = 0.3913 y/r = -3.9136	x/r = -0.7827 y/r = - 3.5874	Non détecté	Non détecté

24	**X**	x/r = 1.8700 y/r = -0.8228	x/r = 1.1687 y/r = -1.9541	Non détecté	Non détecté	Non détecté
25	**Y**	x/r = 2.0860 y/r = -0.4366	x/r = -0.6953 y/r = -1.9566	Non détecté	Non détecté	Non détecté
26	**Z**	x/r = -1.4509 y/r = -2.5875	x/r = -1.0398 y/r = -0.9752	Non détecté	Non détecté	Non détecté

Table II.2 *Echantillon de la base d'apprentissage, limité à la première trame.*

II.4.3 Apprentissage

Une fois la base d'apprentissage est prête, on passera à l'entrainement. Un modèle SVM sera généré à la fin, par le (**LIBSVM**) en tant que Fichier de format Texte avec l'ensemble des données, et des paramètres suivants :

- **svm_type** : Le type de modèle SVM utilisé, dans notre cas c'est le C-SVM, expliqué précédemment dans la section (**II.2.3 Classification et prédiction**).
- **kernel_type** : Le type de noyau utilisé, dans notre cas Linéaire (**II.13**) / Radial Gaussien (**II.14**).
- **nr_class** : Nombre de classes utilisées, 26 classes dans notre cas.
- **total_sv** : Nombre total des vecteurs du support.
- **rho** : C'est la distance *b* entre l'hyperplan et l'origine de fonction de décision (**II.8**) vue dans la section (**II.2.3 Classification et prédiction**).
- **label** : Le numéro (un entier) d'identification de chaque classe.
- **SVs** : Les vecteurs du support.

II.4.4 Tests

Le tableau suivant **Table II.3** présente les résultats du test des deux scénarios, les lettres alphabétiques prédites en vert ce sont les exemples bien classés, et en rouge ce sont les lettres mal classées. Cette classification est faite à l'aide de la fonction de décision (**II.8**) vue dans la section (**II.2.3 Classification et prédiction**), la rapidité de la discision dépend de nombre d'échantillons d'entrainement, et le type de noyau utilisé :

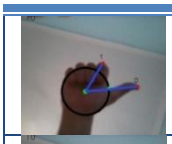
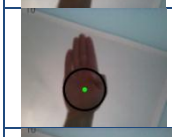
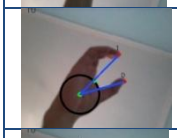
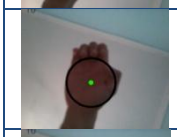
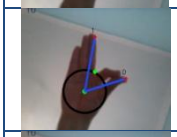
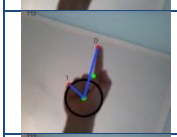
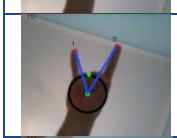
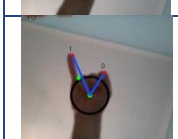
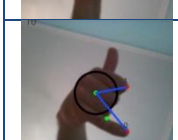
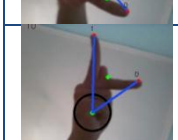
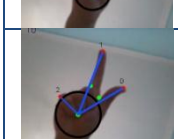
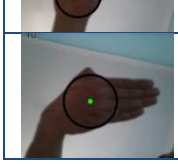
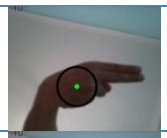
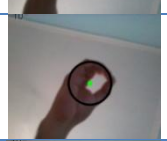
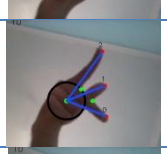
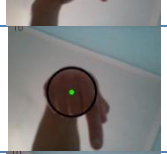
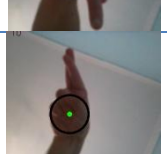
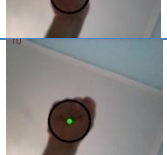
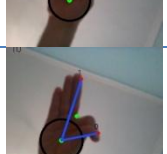
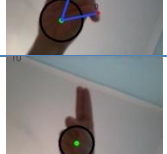
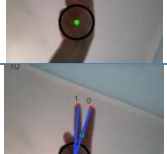
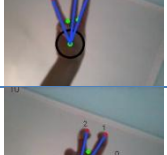
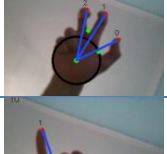
Image	Lettre Originale	Prédiction scénario 1	Prédiction scénario 2
	A	A	A
	B	U	U
	C	C	C
	D	D	D
	E	U	U
	F	F	F
	G	G	G
	H	H	H
	I	I	I
	J	J	J
	K	K	K
	L	L	L
	M	U	U
	N	U	U
	O	U	U
	P	P	D
	Q	U	U
	R	U	U
	S	U	U
	T	T	F
	U	U	U
	V	V	V
	W	W	W
	X	X	X
	Y	Y	Y
	Z	Z	Z

Table II.3 *Résultats des deux scénarios.*

II.4.5 Discussion des résultats

D'après le tableau **Table II.3** vu dans la section précédente nous avons 9 lettres alphabétiques mal classées sur 26 pour le premier scénario, avec un pourcentage de 65.38% de prédiction réussite, et 11 sur 26 dans le deuxième scénario, avec un pourcentage de 57.69% de prédiction réussite, ce qui nous explique l'avantage du noyau gaussien. Comme on peut facilement distinguer que dans tout les signes mal reconnus, dans le scénario 1 par exemple, aucuns doigts n'est détecté, comme le montre le tableau **Table II.2** ce qui implique l'impossibilité d'extraction des points caractéristiques et par conséquence les mauvaises prédictions.

II.4.6 Avantages

Le test exhaustif que nous avons mené à montré l'efficacité de notre approche utilisée avec un taux de réussite 65.38% dans le premier scénario. Comme on peut citer d'autres avantages de plus :

- La rapidité de la prédiction en temps réel, grâce à l'algorithme de classification SVM, et aux techniques de traitement utilisées.
- Classification très efficace des signes de la main avec les doigts levés et séparés.
- Une très bonne optimisation qui ne requit pas des ressources matérielles très importantes.

II.4.7 Limitations

La technique utilisée pour l'extraction des points caractéristiques, expliquée dans la section (**II.2.2 L'extraction des points caractéristiques**) à limité notre approche à la reconnaissance des signes de la main avec le pouce levé, ou au moins deux doigts levés et séparés, afin de pouvoir trouver les points convexes, les points défauts de convexité, et tout le reste des éléments nécessaires pour l'extraction des caractéristiques spécifiques à chaque signe. Ce qui explique les mauvaises prédictions, comme par exemple le signe correspondant à la lettre « **S »** (voir la **Table II.3**) ou tous les doigts sont pliés.

II.5 Conclusion

Nous avons présenté à travers ce dernier chapitre la conception et l'implémentation de notre approche utilisée pour la reconnaissance de la langue des signes de la main en temps réel, commençant par les méthodes, et les techniques utilisées pour la détection de la main et l'extraction des points caractéristiques, jusqu'à l'apprentissage et la prédiction.

A la fin, nous avons conclu le chapitre avec un test expérimental exhaustif qui à montré l'efficacité, et les limitations de notre approche.

Conclusion générale et perspectives

De nous jours, la vision par ordinateur joue un rôle vital dans notre vie quotidienne, ce qui nous a motivés à présenter à travers ce mémoire la conception, et l'implémentation d'une approche utilisée pour la reconnaissance de la langue des signes de la main en temps réel. Dans le premier chapitre nous avons présenté quelques définitions de bases sur l'image numérique, et le traitement d'image, comme nous avons projeté la lumière sur la vision par ordinateur, et son interaction avec la gestualité humaine à travers un état de l'art.

Nous avons consacré le deuxième chapitre à donner une présentation détaillée sur les techniques, et les méthodes adoptées pour notre approche. Nous avons clôturé le chapitre par un test expérimental exhaustif. Les résultats obtenus sont montrés très satisfaisants, mais ça n'ignore pas la présence de certaines limitations qui peuvent êtres surmontées en améliorant cette approche avec une deuxième procédure qui prend en considération les signes auquel aucun doigt n'est détecté.

Bibliographie

[1] M.ANDRE, Introduction aux techniques de traitement d'images, Eyrolle 1987

[2] A.Rosenfeld, A. Kak (1982). Digital Picture Processing. Academic Press. ISBN 0-12-597301-2

[3] Th. Gevers and A.W.M. Smeulders, Content-based Image Retrieval: An Overview, from the book Emerging Topics in Computer Vision, G. Medioni and S. B. Kang (Eds.), Prentice Hall, 2004

[4] D.Paulus and J.Hornegger (1998), Applied Pattern Recognition (2e édition), Vieweg.

[5] F.Souza, I. (2005), Sémiogenèse des langues des signes. Étude de langues des signes primaires (LSP) pratiquées par des sourds brésiliens, thèse de doctorat d'État, Université de Paris VIII.

[6] Surabhi S. Gatagat, Devyani D. More, Kalpana D. Varat, Janhavi S. Toshkhani, "Shape Parameter-Based Recognition of Hand Gestures" International Journal of Innovative Research in Computer and Communication Engineering Vol. 3, Issue 10, October 2015.

[7] Z.Chen, J.Kim, Jianning Liang, Jing Zhang, and Yu-Bo Yuan, "Real-Time Hand Gesture Recognition Using Finger Segmentation," The Scientific World Journal, vol. 2014, Article ID 267872, 9 pages, 2014. doi:10.1155/2014/267872

[8] S. Keerthi, S. K. Shevade, C. Bhattacharyya, and K. R. K. Murthy. Improvements to Platt's SMO algorithm for SVM classier design. Neural Computation, 13:637 649, 2001.

[9] H. Chen, R.-E. Fan, and C.-J. Lin. A study on SMO-type decomposition methods for support vector machines. IEEE Transactions on Neural Networks, 17:893 908, July 2006.

[10] C.Chih-Chung and L.Chih-Jen (2011). "LIBSVM: A library for support vector machines". ACM Transactions on Intelligent Systems and Technology. 2 (3).

[11] E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifers. In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, pages 144/152. ACM Press, 1992.

[12] H.Kressel. Pairwise classifcation and support vector machines. In B. Sch□olkopf, C. J. C. Burges, and A. J. Smola, editors, Advances in Kernel Methods Support Vector Learning, pages 255/268, Cambridge, MA, 1998. MIT Press.

[13] Parts de marché selon StatCounter, sur gs.statcounter.com, mars 2017.

Webographie

[14] https://www.ijircce.com/upload/2015/october/99_32_Shape.pdf

[15] https://www.hindawi.com/journals/tswj/2014/267872/

[16] http://www.csie.ntu.edu.tw/~cjlin/papers/generalSMO.pdf

[17] https://www.csie.ntu.edu.tw/~cjlin/libsvm/

[18] http://gs.statcounter.com

[19] http://opencv.org/opencv-3-2.html

[20] http://opencv.willowgarage.com/wiki/AndroidExperimental

Annexe

Annexe I

OpenCV:

OpenCV (pour Open Computer Vision) est une bibliothèque graphique libre distribuée sous licence BSD, initialement développée par Intel, spécialisée dans le traitement d'images en temps réel. La société de robotique Willow Garage et la société Isteez se sont succédé au support de cette bibliothèque, Depuis 2016 et le rachat de ItSeez par Intel, le support est de nouveau assuré par Intel. NVidia a annoncé en septembre 2010 qu'il développerait des fonctions utilisant CUDA pour OpenCV.

Cette bibliothèque s'est imposée comme un standard dans le domaine de la recherche parce qu'elle propose un nombre important d'outils issus de l'état de l'art en vision des ordinateurs tels que :

- lecture, écriture et affichage d'une vidéo (depuis un fichier ou une caméra)
- détection de droites, de segment et de cercles par Transformée de Hough
- détection de visages par la méthode de Viola et Jones
- cascade de classificateurs boostés
- détection de mouvement, historique du mouvement
- poursuite d'objets par mean-shift ou Camshift
- détection de points d'intérêts
- estimation de flux optique (Méthode de Lucas–Kanade)
- triangulation de Delaunay
- enveloppe convexe
- ajustement d'une ellipse à un ensemble de points par la méthode des moindres carrés

Résumé

Les gestes de la main jouent un rôle très important dans notre vie quotidienne comme l'un des moyens de communication non-verbale les plus riches. Ainsi, dans le domaine d'Interaction Homme-Machine (IHM), la main peut servir à pointer (pour remplacer la souris), à manipuler des objets (pour la réalité augmentée ou virtuelle) ou à communiquer par gestes avec un ordinateur. Sur cet aspect nous allons présenter à travers ce mémoire la conception et l'implémentation d'une approche pour la détection et l'interprétation des signes de la main en temps réel, en utilisant l'échantillonnage de la couleur et la classification à base de machines à vecteurs de support.

Mots clés: Détection de la main, reconnaissance des formes, Interaction homme-machine, machine à vecteurs de support, traitement d'image, image numérique, langue des signes.

Abstract

The hand gestures play a very important role in our daily life as one of the richest non-verbal means of communication. Thus, in the field of Human-Machine Interaction (HMI), the hand can be used to point, to manipulate objects (for augmented or virtual reality), or to communicate by gestures with a computer. In this paper we will present the conception and the implementation of an approach for the detection and interpretation of hand signs in real time, using colors sampling and SVM-based classification.

Key words: Hand detection, shape recognition, Human-Machine Interaction, Support vector machine, image processing, signed language.

ملخص

إشارات اليد تلعب دورا هاما جدا في حياتنا اليومية باعتبارها واحدة من أغنى الوسائل غير اللفظية للاتصال. بالخصوص في مجال التفاعل بين الإنسان والآلة. يمكن لليد إن تقوم بمهام فأرة الحاسوب و التحكم في الواقع الافتراضي, أو التواصل عن طريق الإشارات مع جهاز كمبيوتر. في هذه المذكرة سوف نقوم بعرض تصميم وتنفيذ برمجي لنهج معلوماتي في التعرف على إشارات اليد في الوقت الحالي، باستعمال عينات من الألوان و شعاع الدعم الآلي في التصنيف للتعرف على اليد.

كلمات دلالية: التعرف على اليد، التعرف على الشكل، التفاعل بين الإنسان والآلة، شعاع الدعم الآلي، معالجة الصور، لغة الإشارة .