

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT ET DE LA RECHERCHE SCIENTIFIQUE

ÉCOLE NATIONALE POLYTECHNIQUE



DÉPARTEMENT D'ÉLECTRONIQUE

PROJET DE FIN D'ETUDE EN VUE DE L'OBTENTION DU DIPLOME
D'INGÉNIEUR D'ÉTAT EN ÉLECTRONIQUE

ÉTUDE, SIMULATION ET IMPLÉMENTATION D'UN MODULE DE SEGMENTATION DE L'IRIS SUR FPGA

Réalisé par :

M. Mohamed Amir BENLOUCIF

M. Housseem Eddine DEGHDACHE

Proposé et Encadré par :

M. Lahcen ABDELOUEL

M. Mohamed TAGHI



Projet de Fin d'Études réalisé au Laboratoire des Dispositifs de Communications
et de Conversions Photovoltaïques
École Nationale Polytechnique
10 avenue Hacène BADI – El Harrach
BP182–16200 Alger
Algérie

Tél : (+213) 21 52 53 01/03

Fax : (+213) 21 52 29 73

Web : <http://www.enp.edu.dz/>

ملخص

هذا العمل داخل في اطار تطوير التطبيقات البيومترية للتعرف عن طريق القزحية. وجهنا اهتمامنا بشكل خاص لدراسة وحدة التجزئة و طرق الكشف عن الاشكال الدائرية. بعد القيام بدراسة نظرية ومحاكاة نظام مرجعي ,تم تحليل النتائج واقتراح تقسيم العملية بين برامج أو أجهزة.

في هذا المشروع قمنا باقتراح تصميم جديد لوحدة التجزئة ثم بتثبيت البرنامج على بنية مزدوجة تجمع بين وحدة المعالجة المركزية و لوحة FPGA.

الكلمات المفتاحية : بيومتري , تجزئة القزحية , الكشف عن الاشكال الدائرية , تحويلة هوغ , FPGA.

Résumé

Ce travail s'inscrit dans le cadre de développement d'une application de reconnaissance biométrique basée sur l'iris. Nous nous intéressons tout particulièrement au module de segmentation et aux méthodes de détection des contours circulaires.

Après étude théorique et simulation d'un système de référence, les résultats sont analysés et un partitionnement logiciel/matériel est proposé. Nous avons élaboré un nouveau modèle de segmentation. Enfin la solution est implémentée autour d'une architecture mixte FPGA/CPU.

Mots clés : Biométrie, segmentation de l'iris, détection de contours circulaires, transformée de Hough, FPGA.

Abstract

This work is concerned with the development of a biometric application based on iris recognition. We are interested particularly in studying the segmentation module and the circular contour detection methods.

After a theoretical study and simulation of a reference system, the results are analyzed and a partitioning software/hardware solution is proposed. A new design for a segmentation module is suggested. Finally the solution is implemented on the bases of a mixed architecture FPGA/CPU.

Key words: Biometrics, iris segmentation, circular contours detection, Hough transform, FPGA.

Remerciements

L'aboutissement de notre travail a été possible grâce au concours de plusieurs personnes à qui nous voudrions témoigner toute notre reconnaissance.

Nous voudrions tout d'abord adresser toute notre gratitude à monsieur L. ABDELOUEL et M. Taghi, nos promoteurs, qui se sont montrés présents à nos côtés du début à la fin, et les remercier pour leurs judicieux conseils, qui ont contribué à alimenter notre motivation pour aller jusqu'au bout.

Nous tenons à exprimer nos sincères remerciements à tous les enseignants du département d'électronique, tous sans exception, pour leurs enseignements, et qui par leurs compétences nous ont soutenu dans la poursuite de nos études.

Nous réservons une pensée toute particulière à nos chers parents qui nous ont toujours soutenus par leurs encouragements et leurs prières. Qu'ils trouvent, en ce mémoire, le témoignage de notre profonde reconnaissance.

Nous désirons enfin remercier nos camarades de classe, plus particulièrement Abderrezak, Ridha, Hayet et Abdelatif d'avoir été à nos côtés dans les moments difficiles.

A nos parents, sans qui tout cela n'aurait été qu'un rêve

A nos frères, sœurs et chères familles

A nos amis

Nous dédions ce travail

Table des matières

Remerciements	ii
Dédicace	iii
Table des matières	iv
Table des figures	vii
Liste des tableaux	ix
Introduction générale	1
1 État de l'art	3
1.1 Introduction	3
1.2 Les systèmes biométriques	4
1.3 Les données d'un système biométrique	5
1.4 Les technologies des systèmes biométriques	5
1.4.1 Reconnaissance des empreintes digitales	6
1.4.2 Reconnaissance de la forme de la main	7
1.4.3 Reconnaissance des traits du visage	7
1.4.4 Reconnaissance de la Rétine	8
1.4.5 Reconnaissance de l'iris	9
1.4.6 Reconnaissance vocale	9
1.4.7 Reconnaissance de la dynamique de signature	10
1.4.8 Reconnaissance de la dynamique de la frappe au clavier	10
1.5 Critères d'évaluation des systèmes biométriques	11
1.6 Conclusion	13
2 Reconnaissance par l'iris	14
2.1 Introduction	14

TABLE DES MATIÈRES

2.2	Systèmes de reconnaissance par l'iris	14
2.2.1	Les bases de données publiques	15
2.2.2	Les difficultés de la reconnaissance par l'iris	17
2.3	Architecture d'un système d'iris	18
2.3.1	Étage d'acquisition	18
2.3.2	Étage de traitement	19
2.3.3	Étage de classification/décision	20
2.4	Quelques méthodes connues de la reconnaissance par l'iris	20
2.4.1	Méthode de BOLES	21
2.4.2	Méthode de DAUGMAN	21
2.4.3	Méthode de WILDES	22
2.4.4	Méthode SANCHEZ- REILLO et Al	22
2.4.5	Méthode de Y. WANG et Al	23
2.5	Les méthodes de détection des contours circulaires	24
2.5.1	Approche contours	24
2.5.2	Approche contours actifs	24
2.5.3	Approche globale : Transformée de HOUGH	25
2.6	Conclusion	29
3	Étude, simulation et profilage du système Masek	30
3.1	Introduction	30
3.2	Le flot de conception classique	30
3.3	Étude du système « MASEK » pour l'identification par biométrie de l'iris	33
3.3.1	Étude du module de segmentation de l'iris	34
3.3.1.1	Détection de contours par filtrage de CANNY	34
3.3.1.2	Localisation de la zone de l'iris	35
3.3.1.3	Détection des paupières et des cils	36
3.3.2	Module de normalisation	36
3.3.3	Modules d'extraction des caractéristiques et de décision	38
3.4	Simulation et vérification des spécifications fonctionnelles du système MA- SEK	39
3.4.1	Spécification fonctionnelles	39
3.4.2	Simulation du système de reconnaissance MASEK	41
3.5	Profilage et partitionnement logiciel/matériel	42
3.5.1	Profilage	42
3.5.2	Partitionnement	44
3.6	Conclusion	46

TABLE DES MATIÈRES

4	Simulation et implémentation du module de segmentation élaboré	47
4.1	Étude du module de segmentation élaboré	47
4.1.1	Introduction	47
4.1.2	Organisation de la segmentation en blocs de traitement	48
4.1.2.1	La partie logicielle	48
4.1.2.2	Bloc de la CHT	48
4.1.3	Fonctionnement du Bloc de la CHT	50
4.1.4	Spécifications fonctionnelles et optimisations des ressources dans l'algorithme de la CHT	52
4.2	Étude détaillée du fonctionnement de l'algorithme de segmentation	55
4.3	Simulation du module de segmentation par la transformée de HOUGH	56
4.3.1	Organigramme de l'algorithme	56
4.3.2	Résultats de la simulation du module de segmentation élaboré	58
4.4	Partie Implémentation et résultats	60
4.4.1	Présentation de la plateforme d'implémentation	60
4.4.2	Travaux réalisés sur la plateforme de développement	63
4.4.2.1	Codage HANDEL-C de la transformée de HOUGH circulaire	63
4.4.2.2	Implémentation de la CHT sur la plateforme virtuelle PALSIm	64
4.4.2.3	Implémentation et gestion de la communication FPGA/CPU	64
	Conclusion et perspectives	68
	Bibliographie	70
A		72
A.1	Les filtres de Canny	72
A.2	Transformée de HOUGH linéaire	74
A.3	HANDEL - C	75
A.4	Architecture et ressource du FPGA VIRTEX II	79
A.5	MATLAB	80
A.6	CHT Code C	83
A.7	CODE MATLAB pour la communication	85

Table des figures

1.1	Architecture d'un système biométrique	4
1.2	Les données des systèmes biométriques	5
1.3	Procédé de la reconnaissance des empreintes digitales	6
1.4	Biométrie par la forme de la main	7
1.5	Reconnaissance par le visage	8
1.6	Anatomie de l'oeil	8
1.7	Reconnaissance de la dynamique de signature	10
1.8	(a) Relation entre TFA et TFR (b) Seuil de décision dans les systèmes biométriques	11
1.9	Comparaison entre modalités biométriques	12
1.10	Les parts de marché par technologie	13
2.1	Image de l'iris	15
2.2	Les différentes bases de données	17
2.3	Système d'identification par biométrie de l'iris	18
2.4	(a) Réflexions lors de l'acquisition (b) Image de l'iris acquise sous lumière naturelle (c) Image de l'œil acquise sous lumière infrarouge (CASIA V1.0)	19
2.5	Exemple de détection de droites . Cellules d'accumulation Matrice (a,b)	26
2.6	Exemple de détection de droites. (a) Espace des variables, (b) Espace de HOUGH	27
2.7	Détection d'un cercle à partir de trois points avec un rayon fixé	28
2.8	Exemple de détection de deux cercles	28
3.1	Les niveaux du flot de conception classique	31
3.2	Les différentes étapes du flot de conception	32
3.3	Différents étages du système MASEK	33
3.4	Détection de contours par filtrage de CANNY (a) (d) Image de contour globale (e) Image de contour	35
3.5	Détection de l'iris et de la pupille en utilisant la transformée CHT	35
3.6	Détection des paupières et des cils en utilisant la transformée LHT	36

TABLE DES FIGURES

3.7	Principe de la normalisation	37
3.8	(a) Image normalisée de l'iris, (b) Masque de bruit (b) Image de l'iris : (c) Image normalisée, (d) Iris code	38
3.9	Vue générale du système de MASEK	41
3.10	Etapas de simulation du système MASEK pour l'image '047_2_4'. (a) acquisition (b) image de contour verticale (c) image de contour horizon- tale et verticale (d) iris détecté (e) élimination des occlusions (f) iris et masque normalisés (g) signature binaire	42
3.11	Représentation graphique du profiler de MATLAB	43
3.12	Module de calcul	45
3.13	Histogramme des temps d'exécution du module de segmentation	45
3.14	(a) Module de calcul logiciel, (b) accélérateur matériel	46
4.1	Processus de la segmentation de l'iris	48
4.2	Bloc de la CHT	50
4.3	Système de votes (Le cercle de centre (a,b) et de rayon R obtient 5 votes)	51
4.4	Passage de l'accumulateur 3D vers l'accumulateur 2D	53
4.5	Nuage de pixels formant le contour de l'iris	54
4.6	Déroutement de l'algorithme de segmentation	56
4.7	Organigramme de l'algorithme élaboré	57
4.8	Résultats de la détection de la pupille	58
4.9	Résultats de la détection de l'iris	59
4.10	Les résultats pour la mauvaise détection de l'iris	59
4.11	Le kit de développement RC203E de CELOXICA	60
4.12	Kit de développement DK5	61
4.13	Flot de conception dans DK5	62
4.14	Couche PAL	63
4.15	Extrait du code HANDE-C de la CHT	65
4.16	l'exécution sur la plateforme PALSIm	65
4.17	Communication MATLAB/FPGA	66
4.18	FPGA vers MATLAB	67
A.1	Représentation d'une droite et d'un point dans les espaces cartésien et de HOUGH	74
A.2	Détection d'une droite à partir de trois points	75
A.3	Détection d'une droite à partir de trois points dans l'espace(r, θ)	75
A.4	Séparation et regroupage des branches parallèles	77
A.5	Communication par canal entre branches parallèles	78
A.6	Domaine de validité des variables	78
A.7	Architecture de la VIRTEX II	79

Liste des tableaux

3.1	Tableau de correspondance des fonctions principales du système MASEK	40
3.2	Temps d'exécution des différents modules du processus	44
4.1	Résultats de la segmentation en utilisant l'algorithme élaboré	58

Introduction générale

Cadre général

Aujourd'hui, la nécessité d'identifier les utilisateurs des installations et des services est très importante pour déterminer les droits et les privilèges d'accès.

La biométrie apporte la réponse à ce problème. Elle représente une alternative efficace face aux techniques d'identification classiques très limitées. L'exploitation des traits physiologiques et comportementaux propres à chaque individu, a donné naissance à des systèmes d'identification/authentification dont la fiabilité est inégalée auparavant.

Plus précisément, la biométrie de l'iris est une technique assez jeune parmi les autres techniques biométriques, mais c'est une des technologies qui assurent un niveau de sécurité optimal dans de larges applications dans les domaines aussi variés que l'e-commerce, la médecine légale, le passeport et la carte d'identité biométriques.

Dans les systèmes de reconnaissance d'iris actuels, la personne qui cherche à se faire identifier doit simplement fixer l'objectif d'une caméra qui fait l'acquisition de son iris. Un processus de reconnaissance est lancé par la suite. Il enchaîne un nombre de traitements utilisant différentes méthodes mathématiques, pour extraire des données contenant l'essentiel de l'information qui permet de différencier deux individus.

Problématique

Du fait de ces nombreux traitements successifs, le processus d'identification est généralement effectué par des systèmes informatiques complexes et coûteux. Dans ce PFE nous nous intéressons au problème de la reconnaissance basée sur l'iris, auquel on veut

apporter une solution basée sur un système matériel/logiciel afin de réduire le coût et le temps de traitement.

Objectifs du projet

Ce travail s'inscrit dans le cadre de développement d'une application de reconnaissance biométrique basée sur l'iris. Nous nous intéressons tout particulièrement au module de segmentation. Plus spécifiquement, les objectifs suivants sont poursuivis au long de ce projet de fin d'étude :

- Étude d'un système de référence.
- Étude et simulation des méthodes de détection des contours circulaires,
- Profilage et partitionnement logiciel/matériel d'un système de reconnaissance biométrique basé sur l'iris,
- Implémentation du module de segmentation autour d'une architecture mixte CPU/FPGA,
- Implémentation et gestion de la communication FPGA/CPU.

Organisation du mémoire

Ce mémoire contient quatre chapitres. Le chapitre I fait un survol de la biométrie en générale, on y décrit les différentes modalités utilisées, et les critères de choix.

Dans le chapitre II nous passons en revue l'état de l'art sur les études menées sur le système de reconnaissance par l'iris en générale avec ses différents modules, avec une attention particulière au module de segmentation.

Dans le chapitre III, nous étudions le système de référence MASEK, de la simulation jusqu'au partitionnement matériel/logiciel.

Le chapitre IV, détaille le module de segmentation proposé, expose les résultats de la simulation, et aborde l'implémentation sur matériel de la solution proposée.

Chapitre 1

État de l'art

1.1 Introduction

La biométrie est un terme dont on entend de plus en plus parler dans la vie de tous les jours. Si de nombreuses applications utilisent aujourd'hui la biométrie, celle qui correspond au plus grand déploiement dans notre pays est la mise en place des passeports électroniques biométriques (Premier PEB en Janvier 2012) et de la carte nationale d'identité biométrique électronique (CNIBE, fin 2010) utilisant une puce électronique dans laquelle sont stockés les renseignements sur le titulaire, notamment sa photo, ses empreintes digitales, et sa signature numérisée.

Cependant, la biométrie n'est pas vraiment récente. Son apparition remonte au 19ème siècle, avec les premières études sur l'anthropométrie, s'inscrivant dans un sens large de la biométrie, qui s'intéressaient à la mesure des particularités dimensionnelles des êtres vivants. Ce n'est qu'au début du XXIème siècle que la « biométrie » a été utilisée pour la première fois dans un sens plus restrictif pour « l'identification des personnes en fonction de caractéristiques biologiques ».

Le procédé de reconnaissance le plus ancien est l'analyse des empreintes digitales, Cette technique est toujours utilisée de manière automatique avec les traitements informatiques pour l'identification criminelle.

Aujourd'hui la biométrie n'est plus limitée aux empreintes digitales, de nombreuses modalités sont aujourd'hui utilisées pour des applications de contrôle d'accès à des locaux ou à des objets personnels. On peut citer le visage, la voix, la signature, l'iris, la forme de la main, et d'autres encore sont à l'étude comme la démarche, la forme de l'oreille ou la dynamique de frappe au clavier.

1.2 Les systèmes biométriques

Le système « à reconnaissance biométrique », est un système automatique de mesures basé sur la reconnaissance des caractéristiques propres de l'individu (physiques ou comportementales).

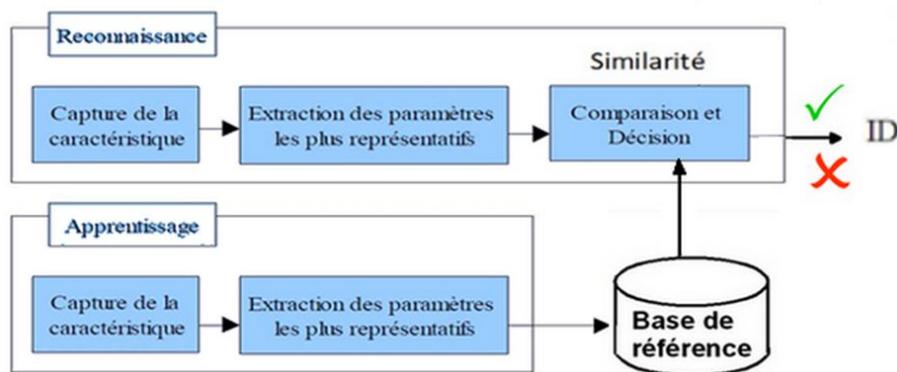


FIGURE 1.1: Architecture d'un système biométrique

La plupart des systèmes biométriques passent par deux phases. La première est celle d'inscription (ou apprentissage) pour permettre l'ajout de modèles à une base de données. La seconde phase est l'identification/vérification, où un modèle, créé pour un utilisateur, est comparé avec les modèles préinscrits dans la base de données en quête d'identification (ou vérification) réussite et détection d'imposteurs (voir Fig 1.1).

1.3 Les données d'un système biométrique

Les données analysées par les systèmes biométriques sont basées sur :

- **L'analyse morphologique** : comme l'identification par empreinte digitale, l'iris, la rétine, la forme de la main, de l'oreille, les traits du visage.
- **L'analyse comportementale** : comme l'identification par la dynamique du tracé de la signature, la dynamique de frappe sur clavier, la voix et la démarche de l'individu.
- **Analyse des traces biologiques** : comme l'odeur, la salive, l'ADN, le sang ... etc.

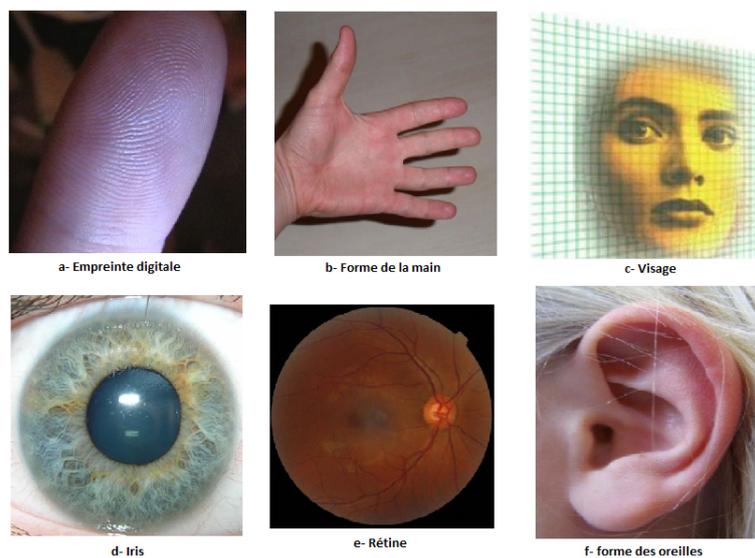


FIGURE 1.2: Les données des systèmes biométriques

1.4 Les technologies des systèmes biométriques

Les technologies les plus fréquemment utilisées sont au nombre de huit. Il s'agit de :

- la reconnaissance des empreintes digitales,
- la reconnaissance de la main,

- la reconnaissance de la rétine,
- la reconnaissance de l'iris,
- la reconnaissance de visages,
- la reconnaissance vocale,
- la reconnaissance de la dynamique de signature,
- la reconnaissance de la dynamique de la frappe au clavier.

Les six premières analysent des caractéristiques morphologiques ; les deux dernières, des caractéristiques comportementales. Les autres techniques sont encore au stade de l'étude.

1.4.1 Reconnaissance des empreintes digitales

Cette technologie de reconnaissance s'appuie sur les structures et l'arrangement particulier des lignes papillaires des empreintes digitales appelées « minuties ». Les minuties les plus fiables sont conservées lors de l'enrôlement. Dans la pratique judiciaire, il faut de 8 à 17 points (mais le plus souvent 12 suffisent) sans discordance pour qu'on estime établie l'identification.

L'acquisition des données est faite par un capteur électronique de type optique, thermique, capacitif ou à ultrasons. Cette dernière est considérée comme la plus fiable, mais aussi la plus coûteuse.

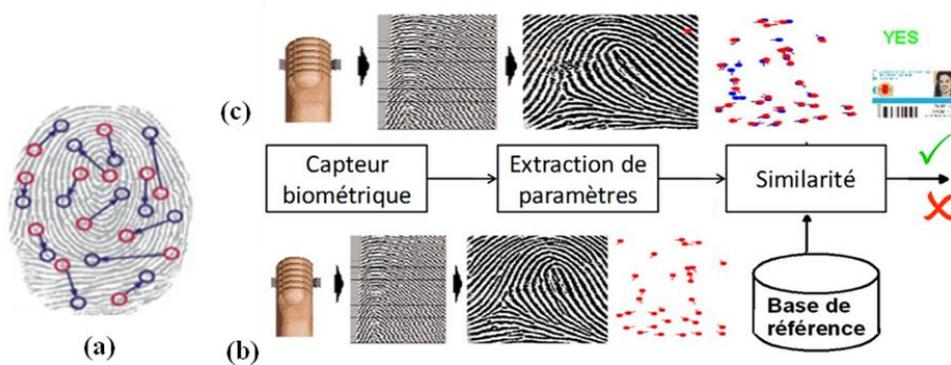


FIGURE 1.3: Procédé de la reconnaissance des empreintes digitales

Après l'acquisition des données, l'extraction de caractéristiques fait appel à plusieurs

méthodes (localisation des minuties, analyse spectrale à l'aide d'ondelettes, traitement de textures, etc.) et à la fin les informations extraites sont sauvegardées dans une base de données (comme indiqué dans la Fig 1.3).

1.4.2 Reconnaissance de la forme de la main

Cette technologie offre une solution très conviviale et facile à mettre en œuvre. Le système biométrique prend une photo de la main et examine 90 caractéristiques de la géométrie de celle-ci dans l'espace. Le système compare ainsi la forme tridimensionnelle de la main, de la longueur et de la largeur des doigts, la largeur et l'épaisseur des paumes, la forme des articulations, les dessins des lignes de la main ... etc.

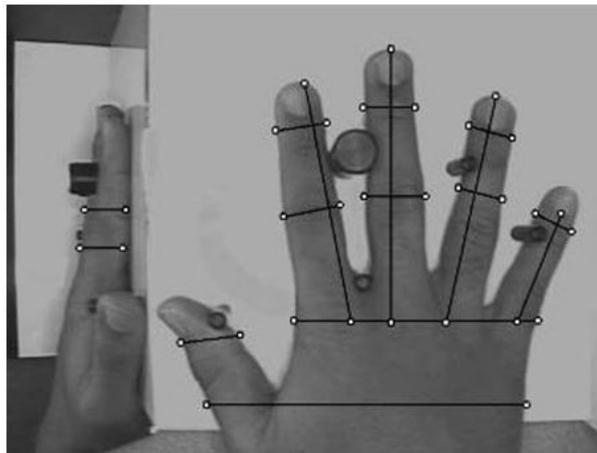


FIGURE 1.4: Biométrie par la forme de la main

Les lecteurs du contour de la main offrent un niveau très raisonnable d'exactitude, mais peuvent avoir des taux de fausses acceptations élevés pour des jumeaux ou d'autres membres de la même famille.

1.4.3 Reconnaissance des traits du visage

La biométrie par reconnaissance du visage consiste tout simplement à "reconnaître" quelqu'un par sa photo. Elle se base sur un relevé de différents points caractéristiques du visage tels que l'écart entre les yeux, la forme de la bouche et du menton, le tour du

visage, la position des oreilles,...etc. En tout, plus de 60 critères fondamentaux existent. Un tracé géométrique personnel est alors enregistré comme gabarit (chaque visage est codé sous forme de fichier de 84 octets).

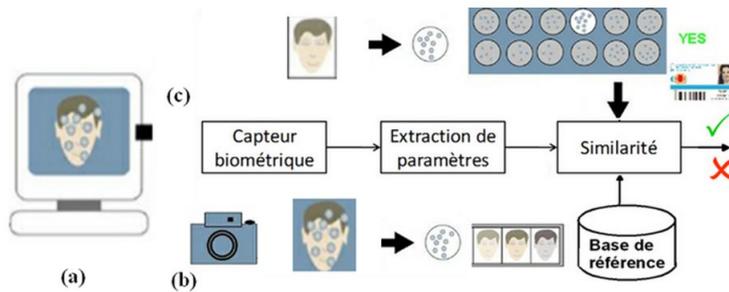


FIGURE 1.5: Reconnaissance par le visage

Cette technique est la plus simple et la moins contraignante. Mais elle a encore de gros progrès à faire, Car elle présente une fiabilité moyenne (incapacité de différencier des vraies jumeaux, utilisation d'un maquillage ou d'un masque en silicone, perturbation par des éléments tels que les lunettes, la barbe, la moustache, une blessure,...).

1.4.4 Reconnaissance de la Rétine

La rétine est une membrane nerveuse très sensible sur lequel viennent se former les images. Elle couvre environ 75 % du globe oculaire et on y recense plus de 130 millions de cellules nerveuses. La grande variété de configurations des vaisseaux sanguins présenterait la même diversité que les empreintes digitales.

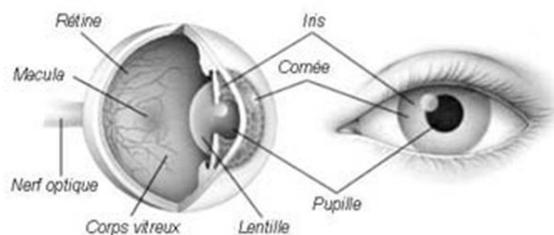


FIGURE 1.6: Anatomie de l'oeil

Après la capture d'une image de la rétine, le système repère l'emplacement et la forme d'un certain nombre de points de repères allant jusqu'à 92. Cette technique est réputée pour être fiable, mais aussi coûteuse. Elle est utilisée pour des applications de haute sécurité, notamment dans les domaines militaires et nucléaires. Malheureusement cette technique souffre encore d'une réticence psychologique de l'utilisateur. On accepte difficilement l'idée de recevoir un rayon lumineux dans l'œil, même s'il est inoffensif.

1.4.5 Reconnaissance de l'iris

L'identification ou l'authentification par l'iris est une des technologies qui (avec la rétine) assure un haut niveau de sécurité. Elle utilise plus de paramètres et caractéristiques que les autres méthodes d'identification avec une fiabilité impressionnante (1 sur 10 puissance 72). La probabilité de trouver 2 iris suffisamment identiques est pratiquement impossible et même des vrais jumeaux ne peuvent avoir les mêmes caractéristiques d'identification.

Pour reconnaître l'iris, il faut une caméra spéciale couplée à un système de gestion. Le sujet doit se positionner de façon précise pour être identifié, les dispositifs actuels fonctionnant avec environ 10 à 40 cm de distance entre l'œil et le capteur. Cette technique fera l'objet de l'étude du chapitre suivant.

1.4.6 Reconnaissance vocale

Chaque individu possède une voix propre à lui, et qui est quasiment impossible à imiter. Donc on peut extraire quelques caractéristiques essentielles d'un enregistrement vocal comme la fréquence, l'intensité et la tonalité pour être traitées dans le système biométrique afin d'obtenir la signature vocale de l'individu.

C'est une technologie non intrusive qui n'exige aucun contact physique avec le lecteur du système. Par contre, elle présente pleins d'inconvénients car elle dépend de l'état émotionnel et physique de l'utilisateur, ainsi que la qualité de l'enregistrement vocal surtout dans les appels téléphoniques qui sont une source très importante pour cette technologie.

Les fraudes sont possibles en enregistrant, à son insu, la voix d'une personne autorisée,

ou en l'obligeant à lire un texte aléatoire ce qui rend la reconnaissance vocale d'une très mauvaise fiabilité.

1.4.7 Reconnaissance de la dynamique de signature

Chaque personne a son style d'écriture. Les systèmes d'authentification ou d'identification de signature ou d'écriture incluent habituellement un stylo optique et une tablette d'écriture digitale. Le système analyse les mouvements du stylo, la vitesse d'écriture, son accélérations et la pression exercée lors de l'écriture.

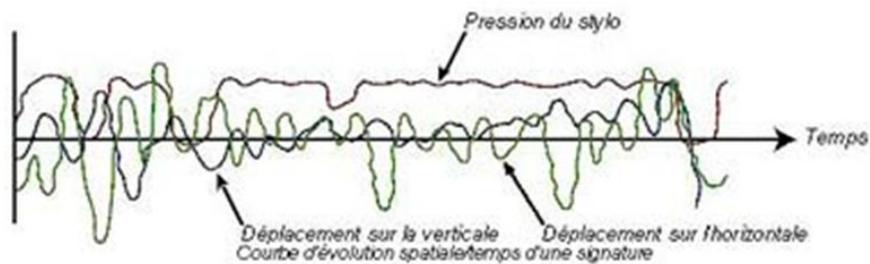


FIGURE 1.7: Reconnaissance de la dynamique de signature

Cette technologie n'est pas encore très fiable. Les difficultés liées à la capture d'une signature viennent du fait qu'une personne ne signe jamais deux fois de la même façon, elle présente néanmoins l'avantage d'être utilisable à distance, dans des applications de commerce électronique par exemple.

1.4.8 Reconnaissance de la dynamique de la frappe au clavier

Cette technologie s'appuie sur les variations de la durée séparant la frappe de deux touches du clavier d'un ordinateur et la pression exercée sur les touches lors de la saisie d'un identifiant et d'un mot de passe. Cette authentification vient donc s'ajouter à celle reposant sur le secret (identifiant / mot de passe) connu de l'utilisateur.

La technologie ne nécessite pas de matériel particulier mais l'état de santé et la fatigue peuvent altérer la façon de frapper les touches.

1.5 Critères d'évaluation des systèmes biométriques

Les performances des systèmes d'authentications biométriques s'expriment par :

- **T.F.R.** - Taux de faux rejets (FRR : False Rejection Rate) : Pourcentage de personnes rejetées par erreur.
- **T.F.A.** - Taux de fausses acceptations (FAR : False acceptance Rate) : Pourcentage d'acceptations par erreur.
- **T.E.E.** - Taux d'égale erreur (EER : Equal Error Rate), donne un point sur lequel le T.F.A. est égal au T.F.R.

Les deux indices T.F.A. et TFR sont liés : une diminution du TFA entraîne systématiquement une augmentation du TFR (et inversement). Il s'agit donc d'adapter le système en fonction du niveau de sécurité souhaité. La Fig 1.8 [3][1] présente la façon de choisir le seuil de décision.

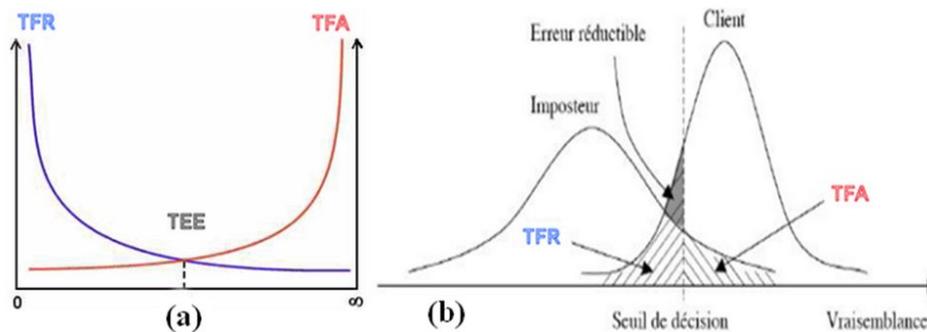


FIGURE 1.8: (a) Relation entre TFA et TFR (b) Seuil de décision dans les systèmes biométriques

Cependant, pour une meilleure évaluation du système de reconnaissance, et comparaison entre différentes modalités, il faut tenir compte des facteurs subjectifs.

Une étude comparative des principales technologies biométriques, réalisée par la Société de conseil et d'intégration International Biometric Group (présentée dans la Fig 1.9) [1] permet de définir quel est le système le mieux adapté à l'application à sécuriser. On note

que le système biométrique parfait n'existe pas, les modalités les plus performants sont intrusives et plutôt coûteuses.

En effet, les reconnaissances de l'iris, de la rétine ou des empreintes digitales sont généralement mal perçues par le public. Cependant, Il existe d'autres modalités, moins intrusives, comme la reconnaissance vocale et les biométries du visage qui présentent l'avantage d'être naturelles aux êtres humains, tout en apportant un niveau de sécurité suffisant pour un grand nombre d'applications.

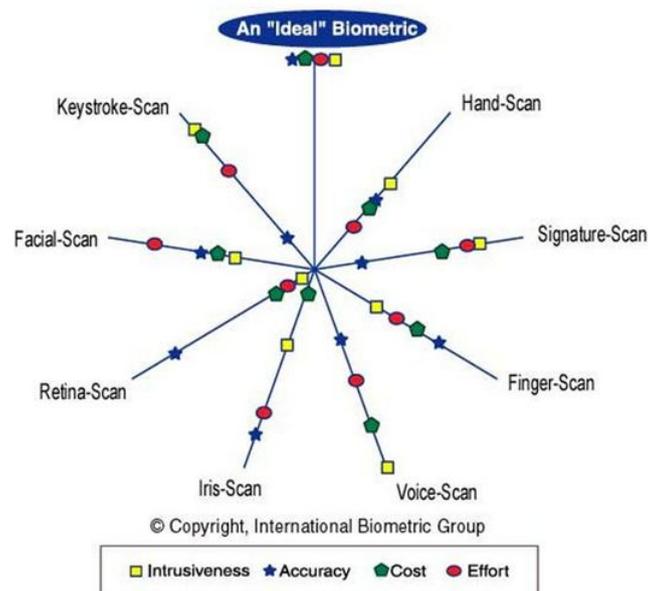


FIGURE 1.9: Comparaison entre modalités biométriques

Effort : effort requis par l'utilisateur

Intrusiveness : niveau de perception par l'utilisateur du test comme intrusif

Cost : coût de la technologie (lecteurs, capteurs, etc...)

Accuracy : efficacité de la méthode (capacité à identifier quelqu'un)

Enfin, pour réussir, un système biométrique doit présenter une logique de marché [1], c'est-à-dire qu'il doit exploiter le même sens que le périphérique auquel il est joint. Par exemple, la reconnaissance vocale est plus justifiée dans le cadre de l'utilisation du téléphone cellulaire. De même, l'authentification d'une personne à l'aide de sa rétine ou de son iris est plus naturelle lorsque celle-ci désire accéder à son compte bancaire via un guichet automatique, la plupart étant déjà muni d'une caméra. Finalement, un système

biométrique qui analyse l'empreinte digitale est plus normalement incorporé à un clavier ou une souris reliant l'ordinateur.

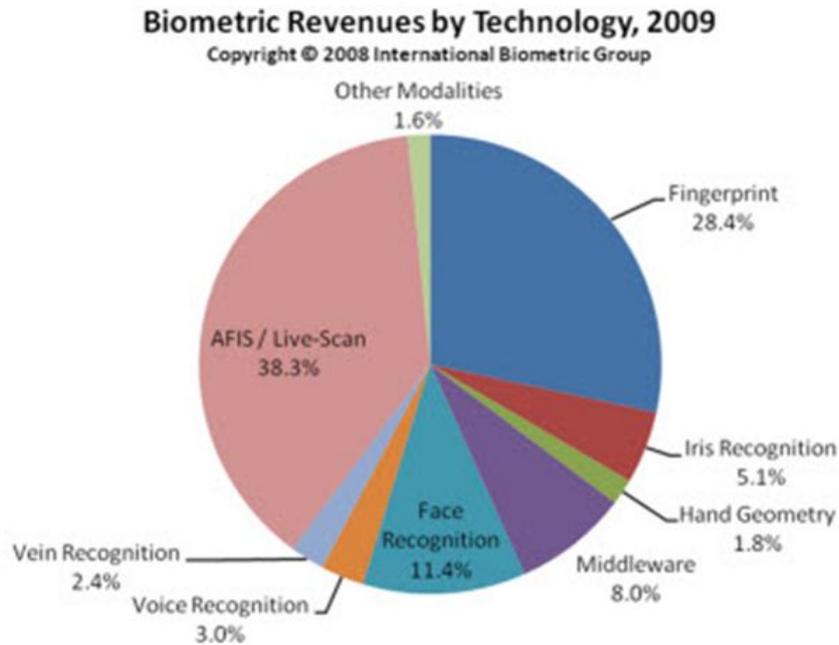


FIGURE 1.10: Les parts de marché par technologie

1.6 Conclusion

Dans ce chapitre nous avons présenté une vue générale de la biométrie et un survol sur quelques techniques biométriques et caractéristiques des systèmes biométriques. Dans la suite on présentera de façon plus détaillée le système de reconnaissance par l'iris.

Chapitre 2

Reconnaissance par l'iris

2.1 Introduction

Bien que chacune des techniques biométriques présente un intérêt particulier suivant l'application visée, nous constatons que les systèmes de reconnaissance basés sur l'iris présentent bien des avantages et qu'ils sont parmi les plus fiables.

La forme et l'apparence générale de l'iris est déterminée génétiquement, sa texture détaillée est propre à chaque individu. L'identification par l'iris utilise plus de paramètres que la plupart des autres méthodes. Les deux principales problématiques des systèmes biométriques basés sur cette technique sont la localisation de l'iris et l'analyse de sa texture pour en tirer les caractéristiques.

Nous exposerons dans ce chapitre les différentes approches utilisées pour traiter les problématiques en question, ainsi que l'architecture et les différents modules d'un système de reconnaissance par l'iris.

2.2 Systèmes de reconnaissance par l'iris

L'iris est le diaphragme de l'œil qui entoure l'ouverture circulaire, appelée la pupille. La texture de l'iris est l'une des textures distinctives les plus riches du corps humain. Elle comporte des arcs de ligaments, des cryptes, des arêtes, des sillons et des collerettes.

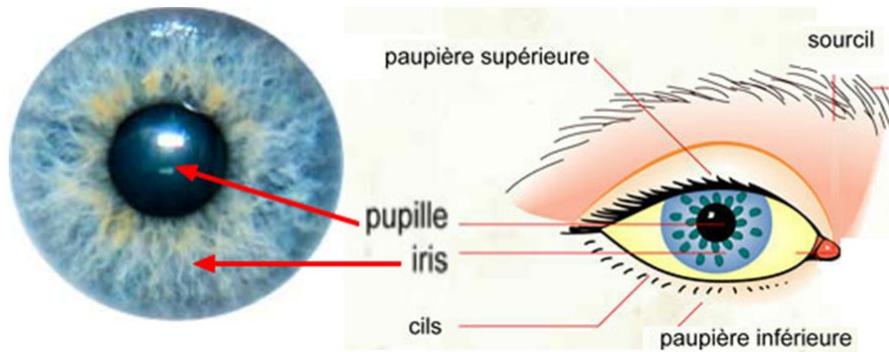


FIGURE 2.1: Image de l'iris

Les deux cercles que constituent les frontières de l'iris avec le blanc de l'œil, ainsi que les frontières de la pupille avec l'iris ne sont pas parfaitement concentriques. De plus, avec les contractions et les dilatations de l'iris ainsi que la variation des distances d'acquisition entre les personnes et l'objectif, la taille du disque de l'iris n'est pas toujours constante. Toutes ces caractéristiques ont fait de l'iris un champ d'étude et de recherche pendant le dernier siècle (à partir de 1930). Et c'est JOHN DAUGMAN [12] qui a réussi à élaborer un système complet de reconnaissance par l'iris et a inspiré les autres chercheurs à fonder de nouveaux systèmes.

2.2.1 Les bases de données publiques

Cette technologie a attiré plein de chercheurs et scientifiques à cause de sa performance. Plusieurs bases de données existent principalement à des fins de test d'algorithmes de reconnaissance par l'iris. Les plus importantes bases de données sont : CASIA¹, UPOL, UBIRIS, ICE, et UBATH.

CASIA V1 est la première base de données partagée publiquement avec 756 images de 108 personnes. Son principal inconvénient est qu'elle présente des images pré-traitées avec un minimum de bruit. La dernière version de la base, CASIA V4 avec 54,601 images de plus de 1800 personnes, présente des données très variées. On trouve des images de l'iris distant, images floues, images avec des bruits oculaires et des réflexions, ainsi que

1. CHINESE ACADEMY OF SCIENCE INSTITUTE OF AUTOMATION

des images photographiées brutes (sans aucun traitement).

Une autre base de données concurrente est celle d'ICE créée par le NIST². La base contient 2953 images de 132 personnes acquises par une caméra dédiée.

UBIRIS est une autre base de données très utile pour tester la robustesse des algorithmes de reconnaissance d'iris aux différents types de dégradations de qualité d'images d'iris. Dans ce but, plusieurs variations des conditions d'acquisition et diverses dégradations des images (illumination, contraste, réflexion, dé focus et occlusion) ont été introduites dans cette base de données. Elle contient 1877 images de résolution 400*300 de 241 personnes capturées en deux sessions.

UPOL est une base de données d'iris qui contient 384 images de 64 sujets européens. La base est propre, la qualité des images est très bonne sans aucune occlusion des paupières et des cils. Les images sont acquises en couleur au format PNG avec la résolution 768*576. La Fig 2.2 présente quelques échantillons des différentes bases de données :

2. NATIONAL INSTITUTE of STANDARDS AND TECHNOLOGY

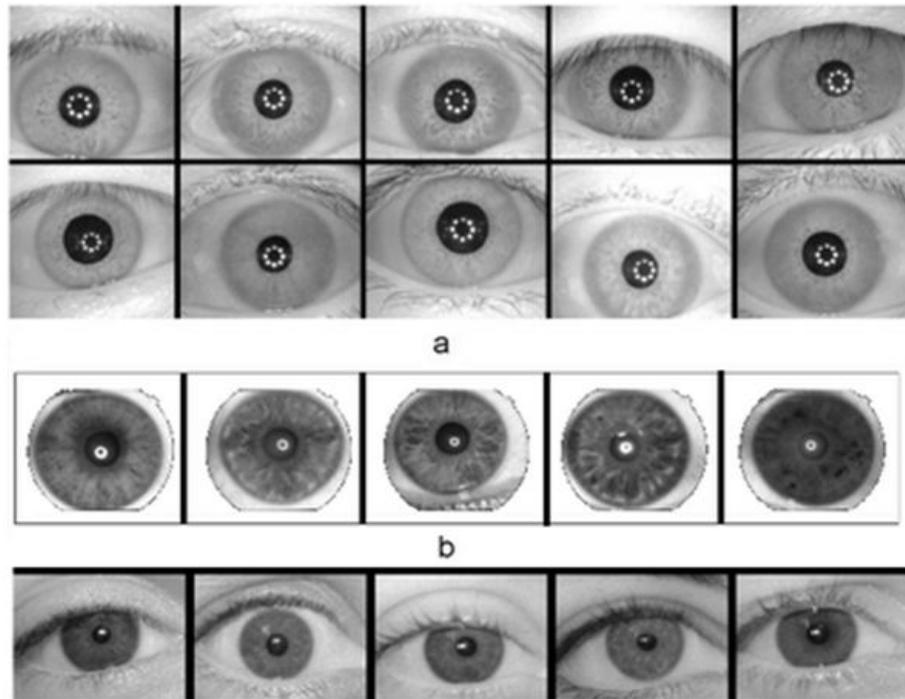


FIGURE 2.2: Les différentes bases de données
a) CASIA b) UPOL c) UBIRIS

2.2.2 Les difficultés de la reconnaissance par l'iris

La biométrie de l'iris présente plusieurs difficultés et défis que nous devons surmonter pour définir un système de reconnaissance fiable basé sur cette modalité.

La difficulté commence dès l'étape de l'acquisition de l'iris. En plus des réflexions dues à la cornée, la texture de l'iris peut être couverte de bruits oculaires. Les bruits causés par les paupières et les cils sont des bruits plus ou moins difficiles à éliminer. Sans oublier que la présence de lentilles ou lunettes peut aussi causer des réflexions indésirables.

D'autre part, la richesse de la texture de l'iris dépend elle-même des personnes. En effet certaines textures d'iris ne sont pas très riches en motifs particuliers et peuvent engendrer des erreurs de reconnaissance en tout genre.

Enfin, l'iris est un muscle qui se dilate et se contracte selon la quantité de lumière dans l'environnement d'acquisition. Ces mouvements du muscle de l'iris induisent une

déformation de la texture de l'iris. Malheureusement, cette déformation est totalement non linéaire et difficilement prédictible.

2.3 Architecture d'un système d'iris

Un système d'iris se compose essentiellement de cinq modules repartis sur trois étages :

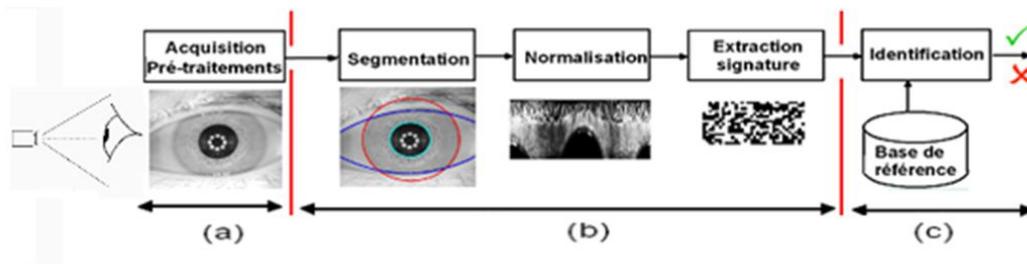


FIGURE 2.3: Système d'identification par biométrie de l'iris

- **Étage d'acquisition** : Capture des images de l'iris à l'aide d'un dispositif de vision.
- **Étage de traitement** : Localisation de l'iris, extraction des signaux utiles et codage des caractéristiques sous forme de vecteurs ou de coefficients (signature de la personne).
- **Étage de classification/décision** : La signature est utilisée pour la comparaison avec les références de la base de données.

2.3.1 Étage d'acquisition

L'acquisition d'une image d'iris est considérée comme l'une des plus difficiles en biométrie. En effet, l'iris est un objet de petite taille, sombre, localisé derrière la cornée qui est hautement réfléchissante. Les réflexions spéculaires peuvent donc compromettre l'échantillon de l'iris.

Aussi, les personnes dont l'iris est fortement pigmenté en noir, présentent un faible contraste entre la région de l'iris et la pupille dans des conditions de lumière naturelle, ce qui génère un fort taux d'erreurs dans les systèmes de référence. La qualité de l'image de l'œil joue donc un rôle déterminant dans le succès de l'identification.

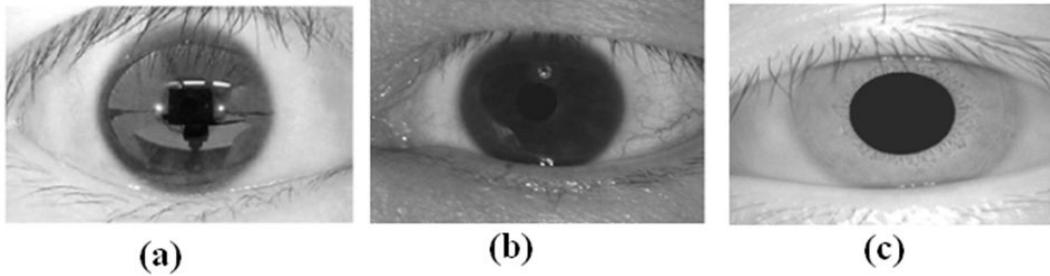


FIGURE 2.4: (a) Réflexions lors de l'acquisition (b) Image de l'iris acquise sous lumière naturelle (c) Image de l'œil acquise sous lumière infrarouge (CASIA V1.0)

Pour l'acquisition, la norme ISO impose l'utilisation d'une ou plusieurs sources infrarouge comme illuminateur puissant. D'une part, l'éclairage ne pose pas de gêne pour l'utilisateur vu l'invisibilité de la lumière infrarouge, d'autre part, on récupère des textures très riches surtout pour les iris sombres.

Les conditions aléatoires et instables de l'acquisition produisent des images de tailles variables, ce qui rend la comparaison impossible. Pour pouvoir générer des signatures comparables, il faut envisager un prétraitement afin de redimensionner les iris extraits, pour les rendre tous de dimensions égales.

2.3.2 Étage de traitement

Une fois l'acquisition faite, une série de traitements sont alors lancés afin de localiser l'iris dans l'image acquise et de calculer une signature appelée « Iriscode ». Cette étape est centrée autour de trois principaux modules :

Module de segmentation

Ce module isole la région de l'iris, en délimitant l'iris de la zone sclérotique et en détectant la pupille dans le disque de l'iris. Les cils et les paupières, qui occluent la zone supérieure et inférieure de l'iris sont isolés à leur tour, pour ne garder que les parties visibles de la texture de l'iris. Généralement, l'iris et la pupille sont approximés par des cercles et les paupières par des ellipses. Différentes approches et méthodes de détection de contours

circulaires ont été utilisées par les chercheurs, elles seront développées dans la section suivante de ce chapitre.

Module de normalisation

Le module de segmentation détecte la région de l'iris délimitée par deux cercles non concentriques. Le module de normalisation transforme cette zone annulaire en une image rectangulaire de dimension fixe, et normalise l'histogramme.

Module d'extraction des paramètres

Le module d'extraction de paramètres applique les filtres définis par la méthode d'extraction à des points prédéfinis de l'image. À l'issue de cette opération, on obtient un ensemble de coefficients représentatifs de l'iris. Chaque coefficient est codé en un bit selon son signe. Le résultat de ce module est un code binaire de longueur fixe appelé "iris code".

2.3.3 Étage de classification/décision

Ce module a pour rôle de quantifier le taux de similarité entre deux signatures d'iris distinctes. On obtient ce score en calculant la distance de HAMMING entre les deux iris codes. Ce score est compris entre 0 et 1. Il vaut 0 lorsque les deux iris codes sont strictement identiques, et 0.5 lorsqu'ils sont constitués de suites binaires décorrélées.

2.4 Quelques méthodes connues de la reconnaissance par l'iris

Depuis l'apparition du premier système de reconnaissance par l'iris élaboré par de J. DAUGMAN [7], les travaux de recherche sur le sujet ont connu une expansion de plus en plus importante. Nous allons citer quelques méthodes des plus connues.

2.4.1 Méthode de Boles

En 1996, W.W. BOLES [4] introduit une nouvelle technique basée sur une transformée en ondelettes monodimensionnelles.

Le traitement commence par localiser le centre de la pupille par détection des contours circulaires, puis de la même manière le diamètre extérieur de l'iris. En fonction du ratio entre le diamètre de l'iris de référence et celui de l'iris à identifier, on définit un ensemble de n cercles virtuels de centre celui de la pupille, sur lesquels on extrait n signaux caractéristiques du relief de l'iris (signaux normalisés à 256 points par sous-échantillonnage).

Une représentation zero-crossing est alors générée par une transformée en ondelettes (décomposition sur 8 niveaux, mais on conserve uniquement les 4ème, 5ème et 6ème). Il s'agit en fait d'un codage des points d'inflexion des n signatures de l'iris pour différents niveaux de résolution ; ce codage est obtenu en utilisant une onde mère spécifique du type dérivée seconde d'une fonction de lissage [5].

Enfin la comparaison est traitée par quatre fonctions de dissimilitude entre 2 représentations zero-crossing. Le score final est la moyenne des résultats fournis par ces fonctions de dissimilitude. L'efficacité de cette technique n'a toujours pas été démontrée sur plus de 2 iris différents[6].

2.4.2 Méthode de Daugman

En 1992, J . DAUGMAN [7] fut le premier à publier ses recherches sur la mise au point d'un procédé d'analyse de texture de l'iris.

Le module de segmentation repose sur la détection précise des centres et rayons respectifs de l'iris et de la pupille par des opérateurs intégro-différentiels sur des arcs de cercle, exploitant le fort contraste aux frontières de la pupille et de l'iris.

Pour le module de normalisation, une conversion de l'image circulaire de l'iris en un équivalent rectangulaire (conversion polaire à angle constant) connue sous le nom de « Rubber Sheet ».

Quant au module d'extraction des caractéristiques, il utilise le codage binaire de la phase locale extraite par un banc de filtres passe-bande en quadratures (filtres de GABOR),

dont certaines zones où se situent les reflets et les paupières sont supprimées.

Enfin pour la décision, le calcul du taux de similarité entre deux codes d'iris se fait par comptage de la proportion de bits qui diffèrent (opération du type XOR).

L'évaluation des algorithmes de J. DAUGMAN dans des conditions réelles d'utilisation lors de la commercialisation par la société IriScan, a donné un FRR de 10% [10].

2.4.3 Méthode de Wildes

Près de 2 ans plus tard, R.P. WILDES [17] propose un système concurrent. Les contours de l'iris sont extraits par transformée de HOUGH, appliquée à la détection de cercle sur les contours de l'image, et les paupières sont modélisées par des arcs paraboliques. La normalisation se fait par l'alignement spatial entre deux images d'iris à comparer (rotation et adaptation d'échelle) réalisé par une technique d'enregistrement d'image (résolution par une procédure de minimisation itérative).

Ensuite la texture de l'iris est représentée par une pyramide Laplacienne, comparable à une décomposition en ondelettes (4 bandes) par des filtres passe-bande isotropes (filtres gaussiens). L'étape de comparaison est basée sur le calcul d'un facteur normalisé de corrélation par sous-bloc de 8×8 pixels. Pour chaque bande fréquentielle on conserve une seule valeur via la médiane : le score est donc représenté par une série de 4 indices de similitudes. La décision finale est prise par un classificateur du type discriminant linéaire de FISHER.

Le test du système proposé par R.P. WILDES n'a produit aucune erreur de faux rejet ou de fausse acceptation sur une base de données comprenant 60 iris de 40 personnes [18].

2.4.4 Méthode Sanchez- Reillo et Al

En 1999, l'approche de SANCHEZ- REILLO et Al. [16] reprend dans l'ensemble celle de J. DAUGMAN. La texture de l'iris est partiellement transformée en un équivalent rectan-

gulaire, puis codée en rendant binaires les résultats de filtrage avec la partie imaginaire de filtres complexes de GABOR.

La comparaison consiste à calculer la distance de HAMMING entre deux codes. A partir d'une base de données de plus de 200 images (au moins 10 images de 20yeux), l'étude de SANCHEZ- REILLO et Al montre que leur système atteint un EER de 3.6% pour une taille de code d'iris de 1860 bits.

2.4.5 Méthode de Y. Wang et Al

La même année, Y. WANG et Al [19] brevètent une nouvelle méthode d'extraction de caractéristiques à partir de la représentation rectangulaire de l'iris (proposée par J. DAUGMAN). Ils ont commencé par filtrer l'image rectangulaire de l'iris, soit par filtrage de GABOR suivant 4 directions et pour 6 fréquences différentes, soit par transformée en ondelettes 2D (ondelettes de DAUBECHIES du 4ème ordre) sur 5 niveaux de faible résolution uniquement.

Ils obtiennent alors n images résultats ($n = 24$ pour le cas, et $n = 13$ pour le cas). La signature d'un iris est alors constituée de la série de n vecteurs [moyenne écart type], extraits de ces n images.

Pour effectuer la comparaison, Y. WANG et Al suggèrent la distance euclidienne pondérée. Sur 160 images (10 images de 16 yeux), le meilleur taux de classification (identification en groupe fermé) est de 93.8%.

En 2002, une amélioration au système est apportée [13] : le vecteur caractéristique comprend 384 valeurs, qui correspondent à l'écart absolu moyen (somme sur l'image des différences entre l'intensité des pixels et la moyenne de l'image) de 384 blocs de 8x8 pixels appartenant à la texture de l'iris (texture préalablement filtrée par un filtre passe-bande symétrique circulaire) ; un classificateur plus avancé (Nearest Feature Line) est utilisé pour la comparaison.

Lors de nouvelles expérimentations sur une population de 134 iris (de 7 à 25 images par œil), le taux de classification atteint 99,85% (ou FAR=0.1% et FRR=0.83% en

mode vérification).

2.5 Les méthodes de détection des contours circulaires

Différentes approches de traitement existent pour la détection des formes sur les images et les vidéos. Les systèmes biométriques de reconnaissance par l'iris font recours à ces techniques, Dans ce qui suit nous allons présenter quelques méthodes.

2.5.1 Approche contours

DAUGMAN a utilisé un opérateur intégrro-différentiel pour la localisation de l'iris, de la pupille et des paupières. L'opérateur intégrro-différentiel est défini par :

$$\max(r, x_p, y_o) \mid G_\sigma(r) * \frac{\delta}{\delta r} \oint_{r, x_0, y_0} \frac{I(x, y)}{2\pi r} ds \mid \quad (2.1)$$

Où $I(x, y)$ est l'image de l'œil, r est le rayon recherché, $G_\sigma(r)$ est une fonction de lissage gaussienne et s le contour du cercle paramétré par : r, x_0, y_0 L'opérateur parcourt le chemin circulaire où il y'a un grand changement dans la valeur des pixels tout en variant le rayon et la position du contour circulaire.

L'opérateur est appliqué de manière itérative, avec la réduction progressive du lissage jusqu'à atteindre la localisation exacte. De la même façon on procède à la détection des paupières juste en changeant le chemin d'intégration d'un contour circulaire à celui d'un arc.

2.5.2 Approche contours actifs

Un contour actif (ou snake contour [14]) est un ensemble de points d'une courbe (fermée ou non) qu'on va tenter de déplacer pour leur faire épouser une forme.

Il s'agit d'une technique d'extraction de données utilisée en traitement d'images. L'idée de cette méthode est de déplacer les points pour les rapprocher des zones de fort gradient tout en conservant des caractéristiques comme la courbure du contour ou la répartition des points sur le contour ou d'autres contraintes liées à la disposition des points.

La recherche d'objets par les contours actifs est une opération de minimisation d'énergie. De façon équivalente, cela revient à dire que son comportement est sous l'influence des forces dérivées de ces énergies, auxquelles on peut rajouter des forces de contraintes extérieures.

L'énergie d'un contour dépend de sa forme et de sa position dans l'image. Ces deux paramètres définissent les forces intérieures et extérieures du contour.

L'évolution des contours actifs est une opération d'équilibre entre des forces intérieures représentées par l'élasticité, la rigidité et éventuellement la viscosité du contour dont l'action vise à ce que le contour garde une certaine forme, et les forces extérieures qui tentent de diriger le contour vers les forts gradients de l'image.

Il existe plusieurs méthodes pour calculer les forces extérieures toutes basées sur le calcul du gradient. On cite le champ GVF (Gradient Vector Flow).

Cette approche permet de prendre en compte l'ensemble de l'image pour le calcul des forces extérieures et donc a l'avantage de ne pas dépendre du contour initial. Dans la pratique l'utilisation de ce champ donne des résultats intéressants.[12]

2.5.3 Approche globale : Transformée de Hough

Il existe des approches globales pour la détection de contours pour lesquelles on ne recherche pas seulement des pixels contours, mais on cherche le contour au complet comme pour la transformée de HOUGH (HT).

La HT permet de détecter n'importe quelle forme définie analytiquement. Elle est connue pour être une technique assez robuste avec une bonne tolérance aux bruits, ce qui permet de détecter des objets partiellement recouverts.

Principe

Le principe de la transformation de HOUGH (HT) est de transformer l'image vers l'espace de paramètres dit espace de HOUGH et d'identifier la courbe dans cet espace. Elle repose

sur l'élaboration d'un accumulateur qui discrétise l'espace de HOUGH. Il est incrémenté au fur et à mesure, et à la fin du traitement il indique les paramètres qui définissent la courbe recherchée.

Dans l'exemple de la Fig 2.5, pour la détection des paramètres a et b de la droite, on construit une image des votes (Matrice (a,b)), chaque point permet de voter pour une droite particulière. Les droites recevant le plus de votes sont conservées.

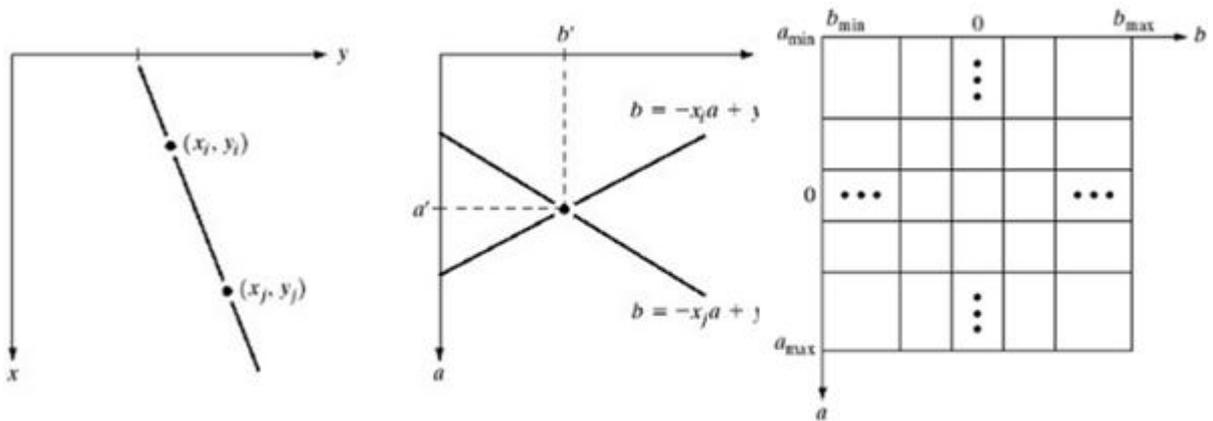


FIGURE 2.5: Exemple de détection de droites . Cellules d'accumulation Matrice (a,b)

Cependant, ceci ne se fait pas sans difficultés, car en fonction de la dimension de l'espace de HOUGH, l'accumulateur peut prendre des dimensions importantes, et requiert une quantité énorme de mémoire. Aussi, en vue du nombre inconnu de piques et celui de faux piques que peut contenir l'accumulateur, il est possible d'obtenir de fausses détections. On verra dans les chapitres qui suivent comment y remédier.

Dans le cas général, la courbe est paramétrée comme suit :

$$f(x, a) = 0 \quad \forall x \in C$$

C étant le contour de la courbe, et $a : a_0, \dots, a$ les paramètres de la courbe. L'espace de HOUGH sera de dimension : $\dim(a) = n$.

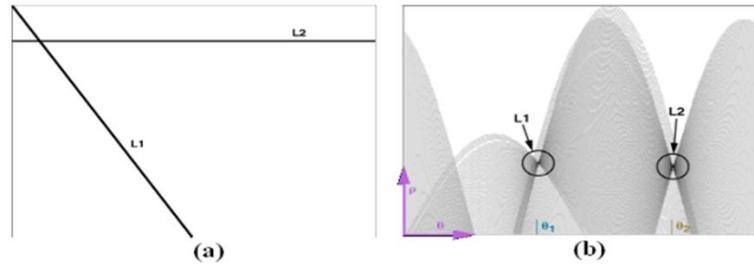


FIGURE 2.6: Exemple de détection de droites. (a) Espace des variables, (b) Espace de HOUGH

Algorithme

- Détection des points de contours : cette étape va permettre d’avoir les coordonnées des points de contour de l’image.
- Discrétisation des paramètres a_j et initialisation de l’accumulateur.
- Pour chaque point de contour, calculer : $f(x_i, a)$ et incrémenter l’accumulateur pour : $|f(x_i, a_j)| < \epsilon$.
- Recherche des maxima locaux de l’accumulateur pour faire ensuite la décision sur les paramètres a_j qui obtiennent le plus de votes (la case de l’accumulateur dont la valeur est maximale).

Transformée de Hough pour détecter les cercles

Tout cercle peut être représenté par son équation : $r^2 = (x - a)^2 + (y - b)^2$, r pour le rayon et (a, b) sont les coordonnées du centre. À partir de ces trois paramètres le cercle est entièrement défini.

La transformée de HOUGH circulaire est utilisée pour déterminer les paramètres d’un cercle à partir d’un certain nombre de points connus appartenant à ce dernier.

Comme montré sur la Fig 2.7, l’intersection de l’ensemble des cercles de même rayon R désigne le centre du cercle de même rayon R et qui passe par les centres des cercles précédents.

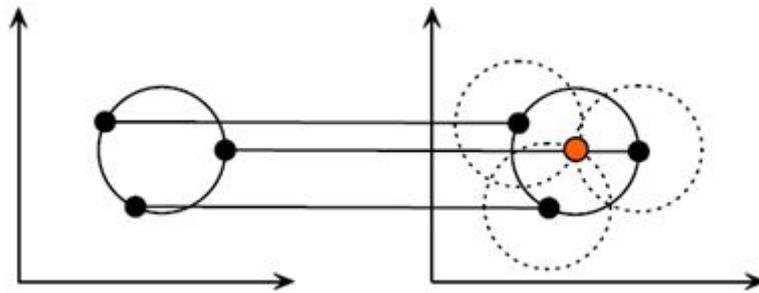


FIGURE 2.7: Détection d'un cercle à partir de trois points avec un rayon fixé

Donc sur une image contenant plusieurs points dont certains appartiennent au même cercle, la transformée de HOUGH peut être employée afin de trouver les paramètres des différents cercles présents sur l'image, dans notre cas la pupille et l'iris en vue leur forme circulaire .

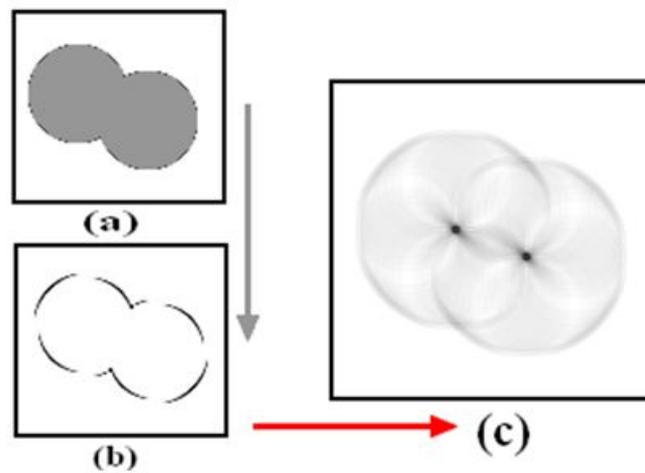


FIGURE 2.8: Exemple de détection de deux cercles

La transformée de HOUGH est donc entièrement adaptée au procédé de segmentation afin d'isoler la zone de l'iris et de la pupille qui nous intéresse.

2.6 Conclusion

Dans ce chapitre, nous avons exposé l'architecture d'un système de reconnaissance basé sur la biométrie de l'iris, et détaillé les différentes phases de traitement nécessaires à l'aboutissement de la reconnaissance.

On a vu que les méthodes classiques sur lesquels se sont basées les recherches, respectaient la même architecture commençant par l'isolation de la région de l'iris, pour ensuite extraire les caractéristiques après avoir effectué une normalisation qui rend les signatures comparables.

Seuls changent les méthodes employées lors des différentes phases de traitement. D'ailleurs, certains systèmes sont nés suite au mariage des différentes méthodes issues de systèmes différents tel le système MASEK, qui pour la segmentation utilise la transformée de HOUGH employée par WILDES avec la méthode de « Rubber Sheet » utilisée par DAUGMAN pour la normalisation.

Parmi les méthodes de détection de contours circulaires exposées, la transformée de HOUGH est devenu un outil incontournable qui est considéré comme un standard dans la détection de formes.

Elle reste une méthode puissante pour la détection de formes sur les images malgré son approche à force brute qui nécessite parfois beaucoup de mémoire et un temps de calcul important. Pour avoir des résultats de segmentation de meilleure qualité, il est possible d'appliquer plus d'une méthode de détection de formes, on pourrait utiliser la transformée de HOUGH circulaire pour localiser grossièrement l'iris et la pupille, et d'affiner la détection en utilisant la méthode des contours actifs par la suite.

Chapitre 3

Étude, simulation et profilage du système Masek

3.1 Introduction

Dans ce chapitre, nous allons décrire de façon non exhaustive les différents éléments et étapes qui interviennent dans une démarche d'implémentation d'un système de reconnaissance basé sur l'iris sur une plateforme matérielle. Nous décrivons d'abord le flot de conception standard permettant de transformer, compiler et implanter un algorithme. Nous étudions par la suite de façon détaillée les phases de simulation, profilage et partitionnement logiciel /Matériel du module de segmentation.

3.2 Le flot de conception classique

Un flot de conception d'un système sur un circuit reconfigurable regroupe plusieurs niveaux d'abstraction et englobe toutes les étapes de conception permettant d'élaborer le fichier binaire (bitstream) de configuration du FPGA.

Le flot de conception classique représenté par la Fig 3.1, admet une entrée sous la forme d'une spécification fonctionnelle/contraintes de l'application, et à chaque niveau, le concepteur s'intéresse à la résolution d'un problème lié à la spécification de l'application, la définition de l'architecture ou à l'implémentation du design sur circuit spécifique.

En ce qui concerne la partie implémentation, les outils commerciaux se chargent de la synthèse (logique /physique), du placement routage et de l'implémentation des traitements sur le circuit reconfigurable. Quant à la partie conception et définition de l'architecture, on distingue deux niveaux :

- La conception au niveau **comportemental** qui s'intéresse à l'étude des fonctionnalités du système, à la définition des tâches concurrentes et indépendantes de l'application et finalement au partitionnement matériel/logiciel.
- La conception au niveau **architecture** qui cherche à modéliser l'architecture de l'application en tenant compte des contraintes d'implémentation sur le matériel.

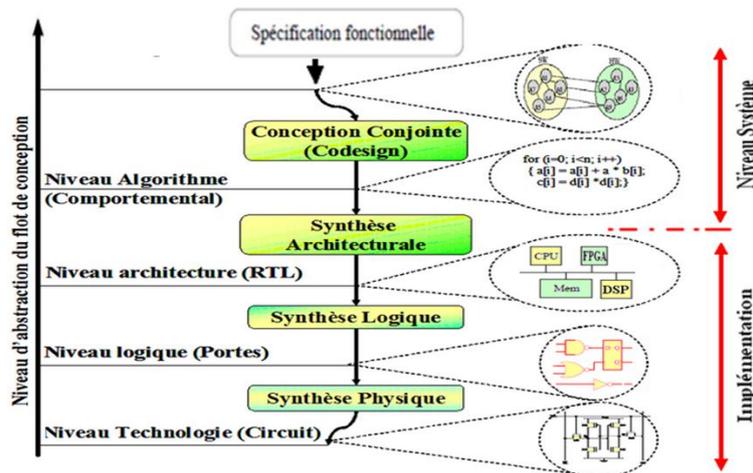


FIGURE 3.1: Les niveaux du flot de conception classique

La Fig 3.2 représente de façon détaillée les différentes opérations et étapes du niveau système du flot de conception. On distingue :

1. La simulation

2. **Le profilage** : Il est réalisé au cours de la simulation du système. Il génère les statistiques (taux d'utilisation du processeur par exemple) pour chaque opération (calcul et échange de données) sur un traitement élémentaire ou sur un sous-système de l'application. L'intérêt du profilage est de confirmer les estimations manuelles et d'anticiper sur une répartition des traitements à implémenter sur le matériel.

3. **Le partitionnement logiciel/matériel** : Il consiste à déterminer les tâches/fonctions qui vont être implémentées sur un ou plusieurs processeurs et les tâches de traitement effectuées par des accélérateurs matériels. C'est une étape déterminante du processus de conception, car elle fixe l'architecture finale du système ainsi que ses performances. Durant cette étape, le concepteur fixe également les interfaces entre la partie matérielle et le logiciel.

4. **La synthèse et la compilation** : Une fois le partitionnement effectué, il ne reste que les étapes de compilation de la partie logicielle (en fonction du processeur cible) et de la synthèse de la partie matérielle.

Si l'outil de simulation dispose d'une bibliothèque de composants (IPs) matériels et logiciels, il est possible de passer à la co-simulation du système complet.

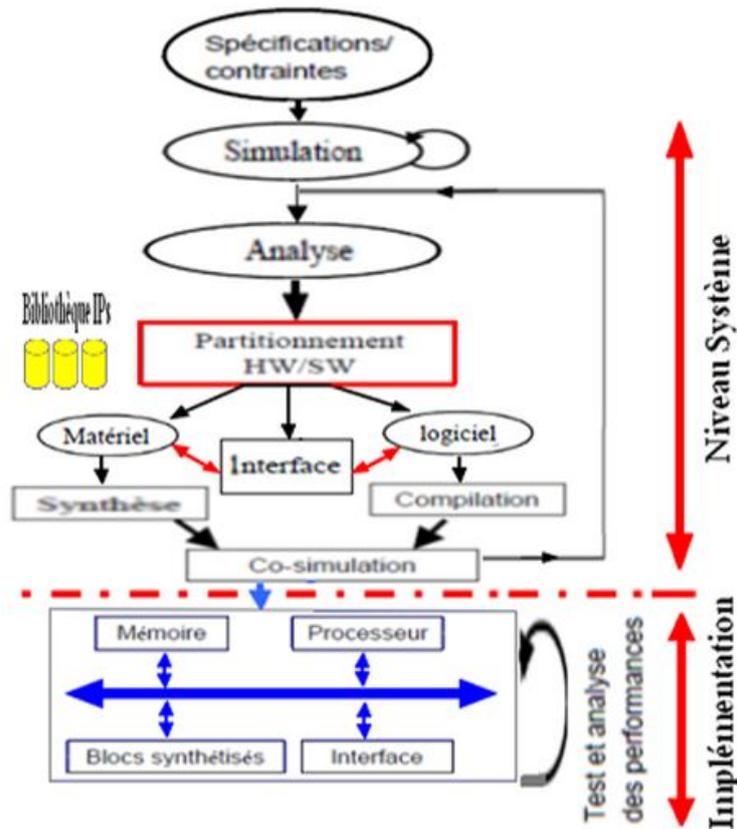


FIGURE 3.2: Les différentes étapes du flot de conception

Dans les sections qui suivent, nous allons détailler les différentes étapes du flot de concep-

tion présenté dans la Fig 3.1 pour un système de reconnaissance biométrique basée sur l'iris. Notre étude s'est basée sur le système de référence MASEK.

3.3 Étude du système « Masek » pour l'identification par biométrie de l'iris

Le système « MASEK » est un système de référence pour la reconnaissance par la biométrie de l'iris, développé par LIBOR MASEK de l'Université de Western AUSTRALIA. C'est un système Open Source écrit en MATLAB qui automatise le procédé de reconnaissance par l'iris et qui englobe toutes les phases de traitement détaillées dans le chapitre précédent (Fig 3.3).

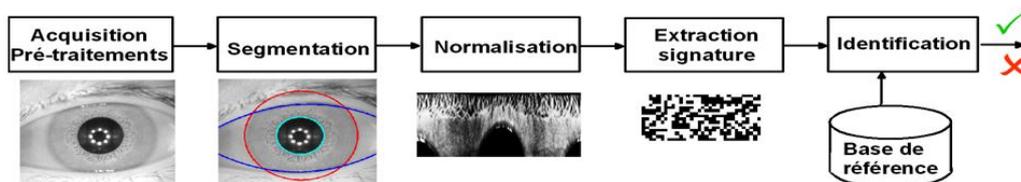


FIGURE 3.3: Différents étages du système MASEK

Le système comprend donc un module de segmentation basé sur la transformée de HOUGH qui permet de localiser la région de l'iris, et d'exclure les zones de bruits d'occlusion comme les paupières et les cils ainsi que les réflexions. Le système inclut aussi un module de normalisation basé sur la méthode de normalisation pseudo polaire. Il transforme la région de l'iris extraite précédemment, en un rectangle de dimensions constantes en prenant compte des différents bruits présents sur l'image en entrée. L'extraction des caractéristiques s'effectue par un filtrage unidimensionnel de GABOR (1D de LOG-GABOR) quantifié sur quatre niveaux avec le codage quatre quadrants. Le dernier module (identification) emploie la distance de HAMMING pour la prise de décision.

3.3.1 Étude du module de segmentation de l'iris

À partir d'une image de l'œil, la segmentation permet la détection des cercles qui approximent les contours de l'iris et de la pupille. Dans le système de MASEK la segmentation est basée sur les trois opérations suivantes :

- La détection de contours par filtrage de CANNY
- La transformée de HOUGH circulaire (CHT) pour la détection de la région de l'iris
- La transformée de HOUGH linéaire (LHT) pour la détection des zones de bruit.

Un filtrage passe-bas (filtre gaussien 2D) est parfois appliqué avant l'étape de détection de contours.

3.3.1.1 Détection de contours par filtrage de Canny

Dans une image, les contours sont des zones de fort contraste dû aux discontinuités du niveau de gris et donc à un saut d'intensité d'un pixel à un autre. Leur détection réduit de façon significative la quantité d'information en filtrant l'information inutile sur l'image, tout en préservant ses propriétés structurelles.

À partir d'une image en niveaux de gris lissée ou floutée, les contours verticaux et horizontaux sont extraits par filtrage de CANNY (voir Annexe A.1) dans deux directions perpendiculaires. L'image résultante est le module du gradient, image en niveaux de gris dont les pixels les plus clairs indiquent les contours (voir Fig 3.4). L'image du contour final s'obtient à partir du seuillage des maxima locaux du module du gradient résultant.

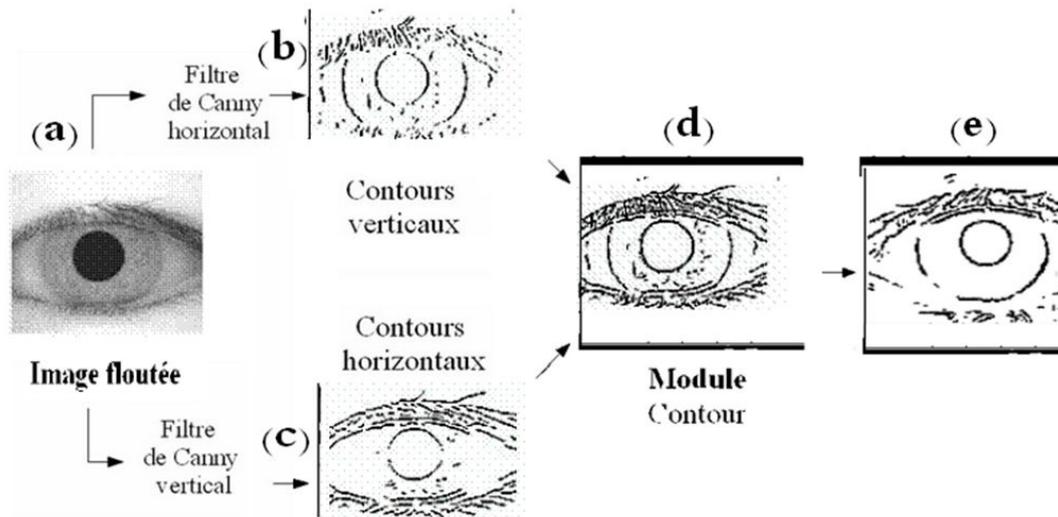


FIGURE 3.4: Détection de contours par filtrage de CANNY (a) (d) Image de contour globale (e) Image de contour

3.3.1.2 Localisation de la zone de l'iris

La localisation de la zone de l'iris est l'étape essentielle du système de reconnaissance. Les résultats et les performances du système entier sont tributaires de la précision de la détection de cette zone. En effet pour avoir la meilleure empreinte binaire à la fin, il faut veiller à avoir la meilleure détection de la région de l'iris (voir fig 3.5).

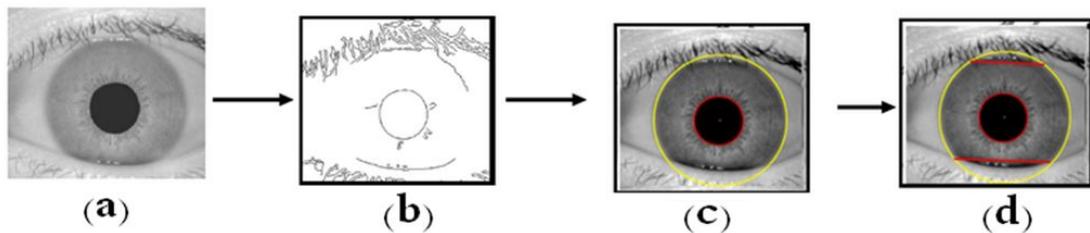


FIGURE 3.5: Détection de l'iris et de la pupille en utilisant la transformée CHT

La région de l'iris est délimitée de l'intérieur par le cercle de la pupille et de l'extérieur par la zone sclérotique. Dans le système de MASEK la transformée de HOUGH circulaire permet de déterminer tout d'abord le cercle correspondant à la frontière entre la sclé-

tique et l'iris puis, dans les limites de ce cercle, le cercle formé entre la pupille et l'iris. La transformée de HOUGH linéaire est employée par la suite pour détecter les zones des paupières et les isoler.

3.3.1.3 Détection des paupières et des cils

Les paupières sont isolées par les droite résultantes de la transformée de HOUGH linéaire en utilisant une détection de contours CANNY horizontale.

En raison de l'espace de HOUGH réduit pour la détection linéaire (2 dimensions), la transformée de HOUGH linéaire est moins lourde que la précédente.

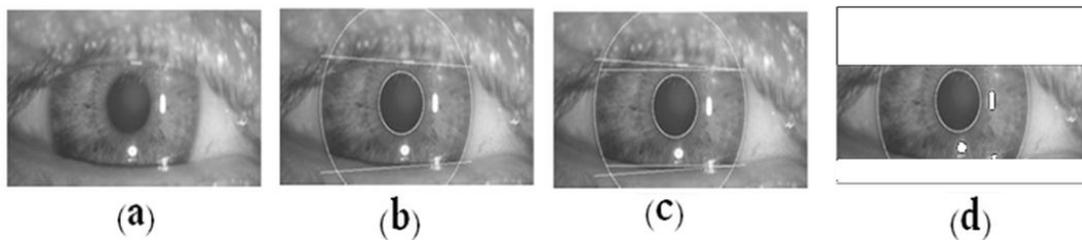


FIGURE 3.6: Détection des paupières et des cils en utilisant la transformée LHT

Le seul problème dans la détection des paupières utilisant la transformée de HOUGH linéaire est la perte éventuelle d'informations significatives et nécessaires dans les zones isolés des paupières, entraînant une baisse dans la précision de la détection. (voir Fig 3.6).

3.3.2 Module de normalisation

La normalisation permet d'avoir des images de la région d'iris de mêmes dimensions, et ce, hormis le fait qu'elles aient été prises sous différentes conditions.

Le module de normalisation du système MASEK emploie une technique basée sur le modèle « rubber sheet » de DAUGMAN. Ce modèle consiste à représenter chaque point de coordonnées cartésiennes (x, y) dans la région de l'iris, par une paire de coordonnées polaires (r, θ) .

Le centre de la pupille est pris pour référence et un vecteur radial parcourt la région de l'iris. Selon chaque direction, un certain nombre de points sont pris, ceci définit la résolution radiale. Quant à la résolution angulaire, elle est définie par le nombre de rayons parcourus.

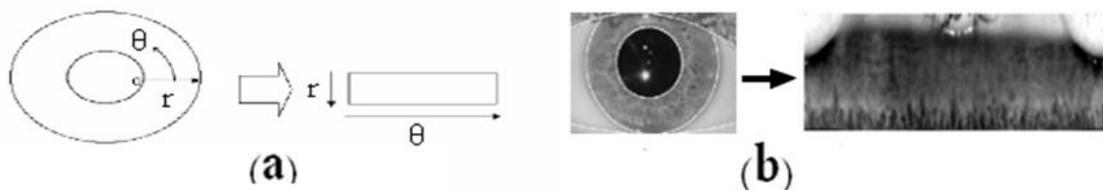


FIGURE 3.7: Principe de la normalisation

Étant donné la pupille, qui généralement est non concentrique avec l'iris, une technique est prévue pour réajuster les points suivant l'angle parcouru.

$$r' = \sqrt{\alpha\beta} \pm \sqrt{\alpha\beta^2 - \alpha - r_I^2} \quad (3.1)$$

avec

$$\alpha = \alpha_x^2 + \alpha_y^2 \quad (3.2)$$

et

$$\beta = \cos(\pi - \arctan \frac{\alpha_y}{\alpha_x} - \theta) \quad (3.3)$$

où α_x et α_y donnent les coordonnées du centre de la pupille relativement au centre de l'iris, r' est la distance entre le contour de la pupille et celui de l'iris à un angle θ et r_I est le rayon de l'iris. Cette donne donc la formule du rayon de la région de l'iris en fonction de l'angle θ .

Un nombre constant de points est pris selon chaque direction radiale, en faisant abstraction de la largeur de la région de l'iris à cet angle. De cette méthode résulte un tableau bidimensionnel avec comme largeur la résolution angulaire et dont le nombre de lignes correspond à la résolution radiale. Un autre tableau bidimensionnel qui représente le masque est créé indiquant les régions contenant les bruits d'occlusion détectés par la segmentation.

3.3.3 Modules d'extraction des caractéristiques et de décision

Le module d'extraction de caractéristiques du système MASEK, est basé sur la convolution du modèle bidimensionnel normalisé de l'iris, avec les ondelettes unidimensionnelles LOG-GABOR. Pour ce faire, l'image normalisée doit être décomposée en un nombre de signaux à une dimension, en considérant chaque ligne de l'image normalisée (qui correspond à un anneau dans la région de l'iris) comme un signal 1D.

Par la suite on calcule le produit de convolution avec les ondelettes de GABOR de même dimension.

L'intensité dans les zones de bruits détectés sur l'image normalisée, est remplacée par la moyenne de l'intensité des pixels voisins pour limiter l'influence du bruit sur le filtrage.

A la sortie des filtres, la phase est quantifiée sur quatre niveaux selon la méthode de DAUGMAN des quatre quadrants, et à chaque échantillon correspond deux bits selon le quadrant où il se situe. A la fin du procédé d'encodage, on aura une signature binaire et un masque qui indique les régions corrompues dans la signature de l'iris et qu'il ne faut pas considérer. Quant aux régions nulles, où la notion de phase est obsolète, elles sont ajoutées à leur tour au masque de bruit.

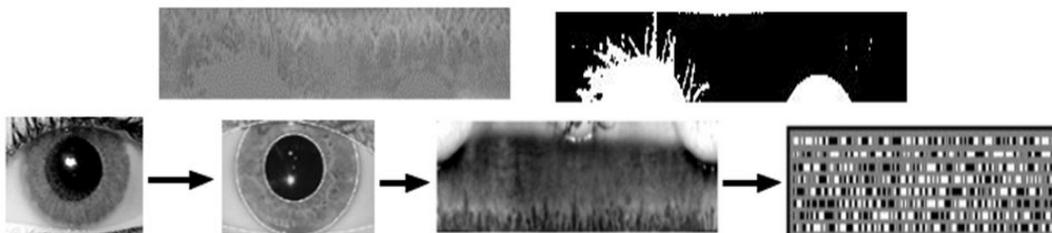


FIGURE 3.8: (a) Image normalisée de l'iris, (b) Masque de bruit (b) Image de l'iris : (c) Image normalisée, (d) Iris code

Pour le module de décision le system MASEK emploi la distance HAMMING comme métrique pour évaluer le taux de ressemblance de deux signatures.

3.4 Simulation et vérification des spécifications fonctionnelles du système Masek

3.4.1 Spécification fonctionnelles

Avant d'entamer l'élaboration et l'implémentation sur matériel du module de segmentation, on doit s'assurer du bon fonctionnement du système de reconnaissance par l'iris. Une première étape, serait donc d'analyser le code de l'application afin de connaître la liste des fonctions appelées. Le tableau 3.1 donne une description des fonctions principales du système MASEK :

Nom du module	Fonction associée	Traitement
Createiristemplate	Script principale	Crée l’empreinte de l’iris ainsi que le masque.
Segmentiris	Createiristemplate	Script générale de la segmentation. Détection des contours circulaires (iris pupille) et linéaires (paupières) dans l’image ainsi que la détection des réflexions et bruits
Findcircle	Segmentiris	Détection du centre et rayon d’un cercle caractérisé par les entrés de la fonction (rmin, rmax, horz, vert)
Houghcircle	Findcircle	Remplissage de l’accumulateur H utilisé pour la détection du cercle
Addcircle	Houghcircle	Teste de la condition
Adjgamma	Findcircle	Ajuster la luminosité de l’image
Canny	Findcircle	Application du filtre CANNY pour la détection de l’image de contours
Hysthresh	Findcircle	Application d’un seuillage par hystérésis
Nonmaxup	Findcircle	Suppression du non-maxima dans l’image de contours
Findline	Segmentiris	Détection des paupières
Linecoords	Segmentiris	Génération des points d’une ligne
Circlecoords	Createiristemplate Normaliseiris	Génération des points d’un cercle
Normaliseiris	Createiristemplate	Script de normalisation, Transforme la région circulaire en rectangulaire
Encode	Createiristemplate	Script générale d’extraction de caractéristiques génération du gabarit de l’iris ainsi que le mask (pour les bruits)
Gaborconvolve	Createiristemplate	Application du filtre de gabor afin d’extraire les caractéristiques
Gethammingdistance	-	Calcul la distance Hamming entre deux gabarits (avec les masks associés)

TABLE 3.1: Tableau de correspondance des fonctions principales du système MASEK

La Fig 3.9 donne une vue générale du système de MASEK, et montre les interactions entre les différentes fonctions.

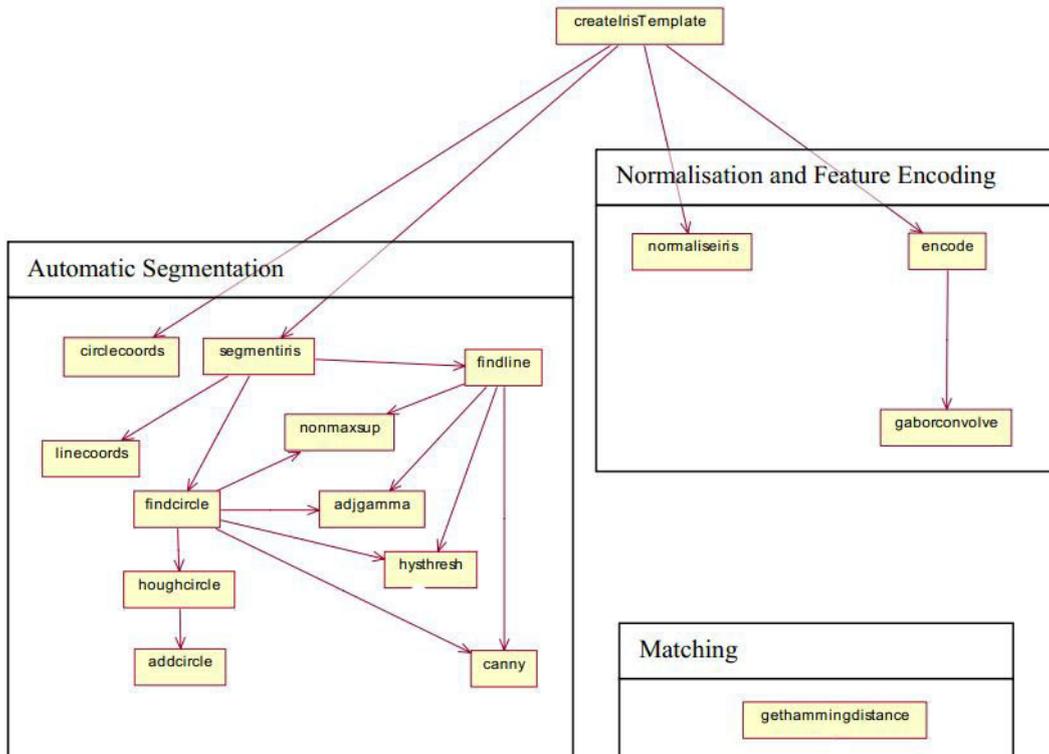


FIGURE 3.9: Vue générale du système de MASEK

3.4.2 Simulation du système de reconnaissance Masek

Le système de MASEK, part de l'acquisition d'une image de l'œil en niveaux de gris, enchaîne les traitements comme décrit sur la Fig 3.9 et prend fin lorsque la signature binaire et le masque sont prêts.

Le système a besoin de certaines données au début pour s'adapter à la base de données utilisée (dimensions de l'image, plage de variation du rayon ...).

Les différentes initialisations et constantes qui doivent être fixées sont prises pour la CASIA Database v1.0.

La Fig 3.10 illustre les résultats en suivant l'évolution à travers les différentes étapes de la simulation.

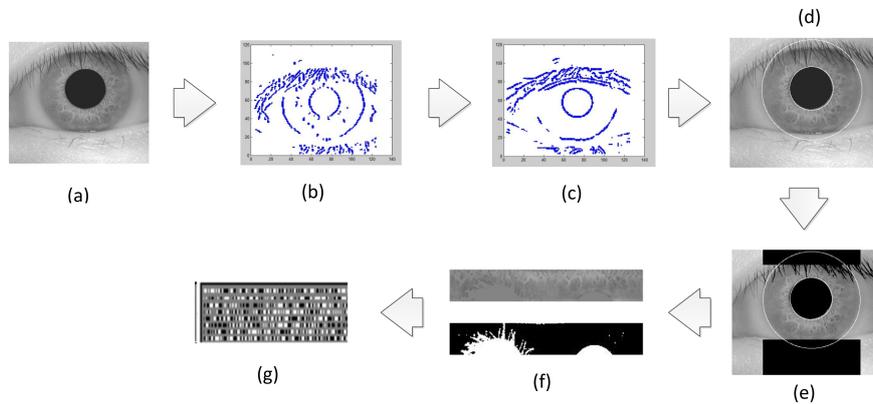


FIGURE 3.10: Etapes de simulation du système MASEK pour l'image '047_2_4'. (a) acquisition (b) image de contour verticale (c) image de contour horizontale et verticale (d) iris détecté (e) élimination des occlusions (f) iris et masque normalisés (g) signature binaire

3.5 Profilage et partitionnement logiciel/matériel

Après avoir simulé sur MATLAB le système de MASEK, et afin de pouvoir proposer une implémentation matérielle, une étude détaillée du code du système est nécessaire, même indispensable à l'estimation globale des performances et à l'exploration d'architecture.

3.5.1 Profilage

Dans une seconde étape on cherche à identifier les goulots d'étranglement affectant les performances des simulations. Pour cela il faut suivre l'arbre des appels de fonctions ainsi que leurs coûts. Le coût peut être un temps d'exécution, un nombre de cycles ou d'instructions, une consommation ou toute autre information pouvant être fort utile dans les phases d'exploration.

Pour notre application, on cherche à développer un accélérateur matériel pour le système de reconnaissance par l'iris, le temps d'exécution est une bonne mesure de performance. Pour déterminer le temps de processeur utilisé dans certaines fonctions, on peut manuellement ajouter dans le code des instructions de chronométrage du temps consommé. MATLAB offre en outre, un outil appelé "profiler" qui est en mesure de comptabiliser le temps consommé par chaque fonction lors de son exécution, puis de présenter les

résultats de cette analyse sous forme graphique comme indiqué dans la Fig 3.11.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
createiristemplate	1	14.703 s	0.058 s	
segmentiris	1	14.492 s	0.033 s	
houghcircle	2	13.870 s	13.870 s	
nonmaxsup	4	0.409 s	0.409 s	
findline	2	0.256 s	0.004 s	
canny	4	0.080 s	0.013 s	
normaliseiris	1	0.064 s	0.023 s	
hysthresh	4	0.050 s	0.050 s	
imwrite	5	0.043 s	0.009 s	
imresize	4	0.041 s	0.000 s	
radon	2	0.032 s	0.001 s	
interp2	1	0.032 s	0.010 s	

FIGURE 3.11: Représentation graphique du profiler de MATLAB

Le profilage a été exécuté pour le traitement de l'image « 047_2_4.bmp » de la base de donnée CASIA v1.0.

Le tableau 3.2, donne le temps d'exécution de chaque module ainsi que le pourcentage par rapport au temps total de l'exécution. Il décrit les résultats d'essais réalisés sur vingt images distinctes, prises aléatoirement dans la base CASIA v1.0.

Createiris (second)	Segment (second)	Normalise (second)	Encode (second)	Segment %	Normalise %	Encode %
11.369	11.180	0.057	0.020	98,34	0,50	0,18
14.687	14.459	0.081	0.024	98,45	0,55	0,16
12.425	12.235	0.056	0.018	98,47	0,45	0,14
10.750	10.535	0.058	0.024	98,00	0,54	0,22
12.932	12.699	0.080	0.022	98,20	0,62	0,17
12.639	12.422	0.064	0.021	98,28	0,51	0,17
12.869	12.660	0.066	0.024	98,38	0,51	0,19
13.637	13.419	0.074	0.020	98,40	0,54	0,15
11.664	11.454	0.067	0.023	98,20	0,57	0,20
11.181	11.002	0.056	0.021	98,40	0,50	0,19
8.476	8.317	0.050	0.017	98,12	0,59	0,20
6.685	6.534	0.041	0.010	97,74	0,61	0,15
11.605	11.425	0.050	0.023	98,45	0,43	0,20
6.735	6.543	0.056	0.015	97,15	0,83	0,22
7.123	6.984	0.046	0.014	98,05	0,65	0,20
6.637	6.454	0.042	0.014	97,24	0,63	0,21
8.646	8.466	0.047	0.015	97,92	0,54	0,17
6.324	6.180	0.042	0.013	97,72	0,66	0,21
5.240	5.104	0.039	0.014	97,40	0,74	0,27
11.666	11.521	0.048	0.014	98,76	0,41	0,12

TABLE 3.2: Temps d'exécution des différents modules du processus

Il est clair d'après le tableau, que la segmentation de l'iris occupe la majeure partie du temps du processus.

3.5.2 Partitionnement

En utilisant un langage haut niveau (algorithme décrit en C/MATLAB), le système est implémenté sur une architecture logicielle avec un processeur (module de calcul) qui effectue tous les traitements nécessaires et une RAM permettant de stocker les données (voir Fig 3.12).

D'après les résultats du profilage, la segmentation est la partie du traitement qui prend le plus de temps. Elle occupe la première position avec plus de 90 % du temps d'exécution. Les autres modules du procédé à l'instar du module de normalisation « normaliseiris »

ou de l'extraction de caractéristiques « encode » ont une durée d'exécution négligeable devant celle du module de segmentation.

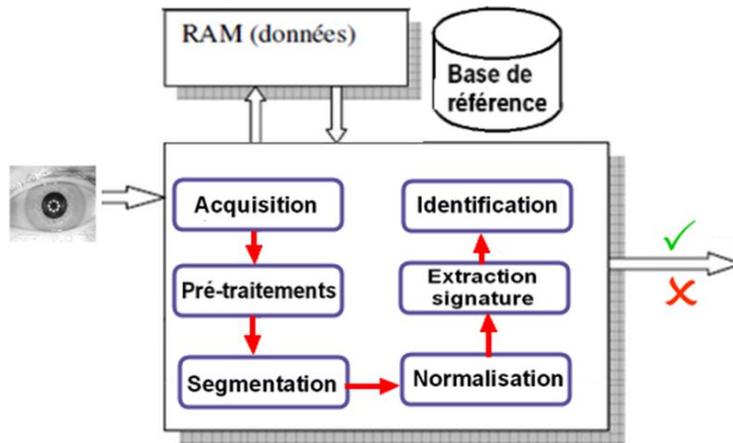


FIGURE 3.12: Module de calcul

Une analyse plus détaillée du module de segmentation qui fait appel au détecteur de CANNY, au calcul et au seuillage des maxima locaux et finalement à la transformée de HOUGH, montre clairement que la transformée de HOUGH circulaire est la plus importante en terme de temps d'exécution (voir Fig 3.13).

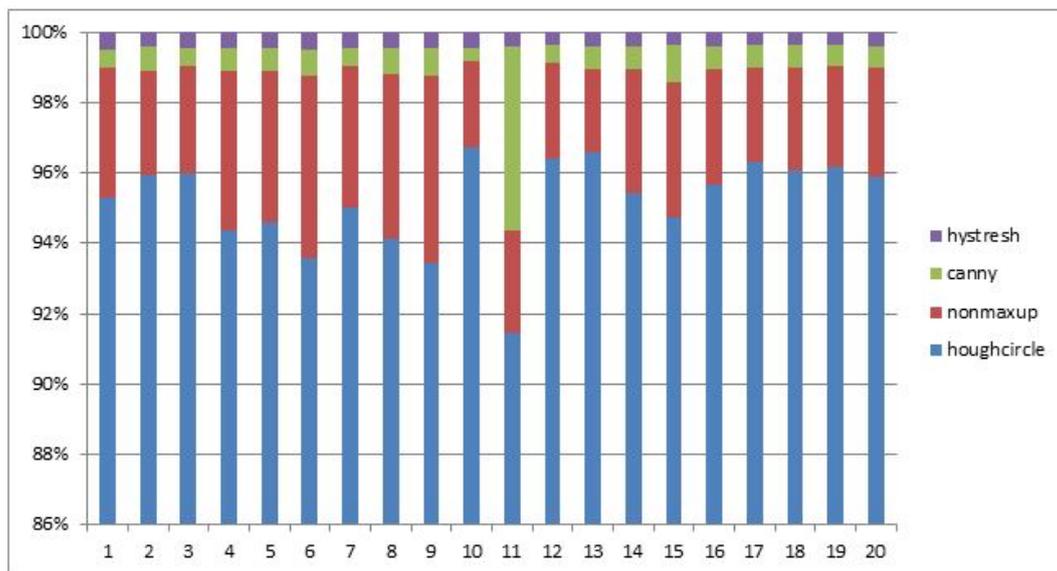


FIGURE 3.13: Histogramme des temps d'exécution du module de segmentation

L'implémentation sous forme matériel dans un FPGA, du module de segmentation, plus spécialement la transformée de HOUGH circulaire est plus que justifiée. Ce qui nous ramène à proposer une nouvelle architecture présentée à la fig 3.14.

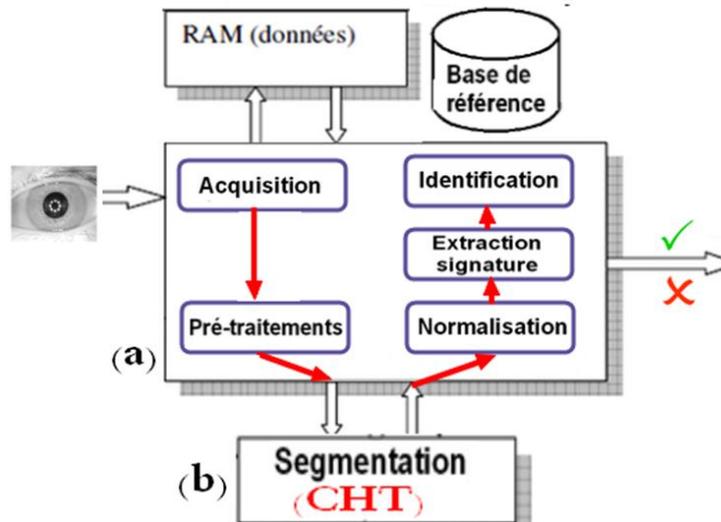


FIGURE 3.14: (a) Module de calcul logiciel, (b) accélérateur matériel

On s'attend à ce que cette architecture améliore le temps d'exécution de l'application et les performances du système.

3.6 Conclusion

En accord aux étapes du flot de conception, nous avons d'abord fait l'étude fonctionnelle du système MASEK. Nous nous sommes assurés par la simulation de son bon fonctionnement. À la suite de l'analyse des performances et en se référant aux résultats du profilage, nous avons pu mettre en évidence que la partie qui ralentissait le processus de la reconnaissance était principalement la transformée de HOUGH. Nous avons alors envisagé l'alternative de son implémentation matérielle et on a proposé une nouvelle architecture. Dans le chapitre suivant, on s'en intéressera aux détails de son implémentation sur FPGA.

Chapitre 4

Simulation et implémentation du module de segmentation élaboré

Dans ce chapitre, nous commençons par l'étude détaillée et la simulation du module de segmentation développé. La première partie du chapitre se terminera par l'établissement de statistiques pour évaluer l'efficacité de l'algorithme élaboré. Par la suite nous passerons à l'implémentation matérielle de la transformée de HOUGH. Nous aborderons donc les spécificités du matériel utilisé, et décrirons la démarche suivie lors de l'implémentation. Nous finirons par exposer les résultats de la partie pratique.

4.1 Étude du module de segmentation élaboré

4.1.1 Introduction

La segmentation de l'iris est principalement constituée de trois fonctions : La détection de contours, assurée par les filtres CANNY, la transformée de HOUGH circulaire (CHT) pour la détection de l'iris et de la pupille, et la transformée de HOUGH linéaire utilisée dans la détection des bruits d'occlusion.

Comme décidé suite au partitionnement logiciel matériel, la transformée de HOUGH circulaire sera implémentée sur FPGA. Le reste du module de segmentation, qui comporte la détection de contours est implémenté sur MATLAB.

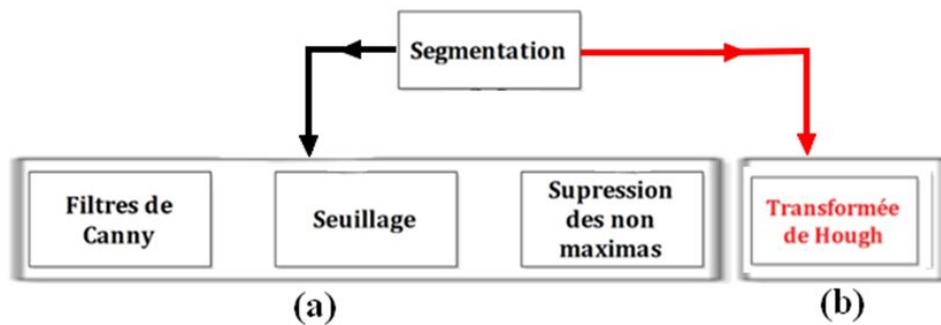


FIGURE 4.1: Processus de la segmentation de l'iris

4.1.2 Organisation de la segmentation en blocs de traitement

4.1.2.1 La partie logicielle

La partie logicielle de la segmentation sur MATLAB a été organisée en trois étages de traitement :

Etage 1 : Phase de traitement pour la détection de la pupille :

- Il fait l'acquisition de l'image de l'iris ;
- Il génère l'image de contour globale ;
- Il transmet les différents paramètres essentiels à la transformée de HOUGH.

Etage 2 : Phase de traitement pour la détection de l'iris :

- Il reçoit et stocke les données de détection de la pupille ;
- Il génère l'image de contour verticale ;
- Il transmet les différents paramètres essentiels à la transformée de HOUGH pour la détection de l'iris.

Etage 3 : Phase de calcul du masque :

- Reçoit et stocke les données de détection de l'iris ;
- Génère l'image de l'iris segmentée ;
- Calcul le masque.

4.1.2.2 Bloc de la CHT

La transformée de HOUGH circulaire est implémentée de façon indépendante, elle est appelée à deux reprises lors de la segmentation, ou elle reçoit à chaque fois les données

nécessaires, et fait le même traitement pour la détection soit de la pupille ou de l'iris, seuls changent les paramètres qu'on lui transmet. Ces paramètres sont :

L'image de contour

Le filtre CANNY génère suite à son traitement, une image de contours. La transformée de HOUGH a besoin des coordonnées de ces points. En effet, la transformée de HOUGH est un algorithme qui à partir de points appartenant à un cercle, trouve les paramètres de ce dernier. Toutefois parmi les difficultés rencontrées dans le cas pratique, il y'a :

- La présence d'une multitude de points qui n'appartiennent ni à l'iris ni à la pupille, et qui représentent donc du bruit. Lorsque l'orientation de l'image de contour est mal choisie, les points du bruit, prennent le dessus sur les points de contour significatifs, ce qui conduit à des détections aléatoires et complètement erronées.
- A cause du faible contraste entre l'iris et la zone sclérotique, les points de contour de l'iris sont peu nombreux par rapport aux points de contour de la pupille, ce qui rend la détection de la pupille plus précise.
- Dans le cas où l'iris est trop sombre, le faible contraste entre la région de l'iris et la pupille entraîne de la même manière une détection erronée. Il est à noter aussi que les zones sombres sur les images acquises (des cils épais par exemple) produisent des zones de forte concentration de points de contour qui généralement induisent une fausse détection.

La plage de variation du rayon

La CHT est un algorithme qui fait des calculs intensifs, il dessine des cercles de différents rayons sur toute l'image, et prend celui qui convient le mieux (celui qui passe par le plus de points de contour en même temps). Cependant, l'algorithme risque d'avoir un temps d'exécution important car il effectue des calculs inutiles. Avoir des informations sur les plages de valeurs approximatives où le rayon peut se situer s'avère très pratique, car en spécifiant une plage à parcourir, on limite les calculs, ce qui occasionne un gain significatif en termes de temps de calcul. De plus on évite d'éventuelles fausses détections dues aux faux piques dans l'accumulateur qui sont d'autant plus importants lorsque la

plage de rayons parcourue est grande. Le rayon minimum et le rayon maximum sont donc communiqués au bloc de la CHT.

La taille de l'image

Les images étant perçues comme des tableaux sur MATLAB, leur taille est donnée par le nombre de lignes et de colonnes. Ces données sont essentielles à la transformée de HOUGH, car elles représentent les positions potentielles du centre du cercle à détecter. Il est intéressant de réduire l'espace à parcourir lors de la recherche du centre pour optimiser la mémoire et le temps pris par l'algorithme.

Remarque

Il faut tenir compte de la différence de représentation des coordonnées des points dans le plan entre les différents étages écrits en MATLAB et le Bloc de la CHT codé en langage C.

Les coordonnées des points de contour générés par MATLAB sont données sous forme de position dans la matrice de l'image : ligne et colonne, il est nécessaire de faire un changement de repère car la transformée de HOUGH travaille avec les coordonnées dans le repère cartésien (x,y) .

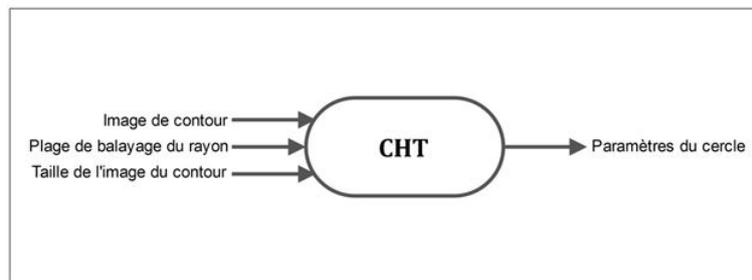


FIGURE 4.2: Bloc de la CHT

4.1.3 Fonctionnement du Bloc de la CHT

La transformée de HOUGH est sensée déterminer à partir de la position des points entrés, le cercle qui conviendrait le mieux, si ces points devaient représenter un cercle.

Dans notre cas, et en fonction de la plage de parcours du rayon, l'algorithme va essayer de trouver le contour circulaire optimal qui localise l'iris ou bien la pupille, et ce à partir de la carte des points de contour générée par les filtres CANNY.

Il est clair que les plages de variation du rayon de la pupille et de l'iris doivent être distinctes. Dans le cas contraire, on risque de détecter qu'un cercle (celui de l'iris ou bien de la pupille) en dépit de l'autre.

L'algorithme crée un accumulateur de la taille de l'image de contour. Généralement la taille de l'image de contour est réduite comparée à celle de l'image originale pour ne pas créer des accumulateurs trop grands.

L'algorithme parcourt l'image de contour en dessinant en chaque pixel de coordonnées (x,y) des cercles dont le rayon R parcourt la plage de variation du rayon, pour calculer ensuite le nombre par lequel ce cercle passe (ce nombre représente le nombre de votes).

Pour chaque point de contour $M_i (x_i, y_i)$, l'accumulateur va être incrémenté pour tous les cercles qui le traversent. A la fin du traitement, les coordonnées dans l'espace de HOUGH de la case de l'accumulateur contenant la valeur maximale représentent le cercle choisis. En réalité, il suffit que : $(x_i - a)^2 + (y_i - b)^2 = r^2$ soit vérifiée (a, b, r sont les coordonnées du cercle dans l'espace de HOUGH et représentent aussi les coordonnées dans l'accumulateur 3D).

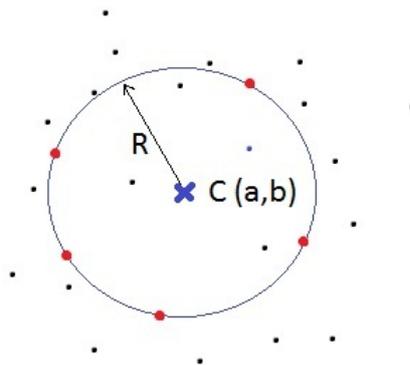


FIGURE 4.3: Système de votes (Le cercle de centre (a,b) et de rayon R obtient 5 votes)

Nous avons constaté que, dans la pratique, les contours sont détectés de manière grossière, il n'existe donc pas un seul cercle qui peut contenir l'iris ou la pupille. Dans ce cas, on pourra trouver plusieurs cases de l'accumulateur contenant la même valeur maximale. Nous proposons de calculer la moyenne des paramètres et de prendre le cercle moyen comme résultat de détection de la pupille.

Pour la détection de l'iris, et en sachant que le centre de l'iris n'est pas très loin du centre de la pupille, on ne parcourt que le carré avoisinant le centre de la pupille.

L'algorithme de la CHT est destiné à être implémenté sur FPGA, donc il faudrait tenir compte des ressources limitées de la carte. Un accumulateur tridimensionnel risque de saturer la mémoire. Pour éviter ce problème, on a dû créer un accumulateur bidimensionnel.

Pour chaque point de contour :

- On incrémente l'accumulateur pour les cercles qui le traversent ;
- On stocke les maximums locaux dans un tableau intermédiaire ;
- On efface l'accumulateur pour le préparer au prochain point de contour.

4.1.4 Spécifications fonctionnelles et optimisations des ressources dans l'algorithme de la CHT

Nous allons détailler les différentes spécifications fonctionnelles et optimisations de ressources prises pour adapter la transformée de HOUGH à la segmentation de l'iris.

Paramétrer le cercle différemment

Habituellement, pour la programmation de la transformée de HOUGH circulaire on utilise une paramétrisation trigonométrique du cercle. C'est le cas de la transformée de HOUGH implémentée par MASEK. A cause des contraintes matérielles, les multiplications en virgule flottantes et le calcul de fonctions trigonométriques (cos, et sin) sont très coûteuses, on a décidé de ne travailler qu'avec les nombres entiers et d'utiliser directement l'équation du cercle qui ne nécessite que deux multiplications.

Accumulateur 2D

Comme on vient de le souligner, les ressources matérielles sont limitées et la création d'un accumulateur tridimensionnel est impossible. Nous avons donc apporté des modifications à l'algorithme pour l'adapter à la nouvelle architecture 2D de l'accumulateur.

Notre démarche consiste à créer un accumulateur 2D de dimensions : $[\text{row}] \times [\text{col}]$ au lieu de l'accumulateur 3D de dimensions : $[\text{row}] \times [\text{col}] \times [\text{nr}]$, avec :

- row nombre de lignes de l'image de contour ;
- col nombre de colonnes de l'image ;
- nr nombre de rayons à parcourir.

Pour chaque rayon on parcourt l'accumulateur pour attribuer les votes aux différents cercles. On stocke dans un tableau intermédiaire les cercles qui obtiennent le maximum de votes pour le rayon en cours, et ensuite on réinitialise l'accumulateur pour le préparer au traitement du rayon suivant. On incrémente le rayon et on refait le traitement pour le nombre de rayon précisé.

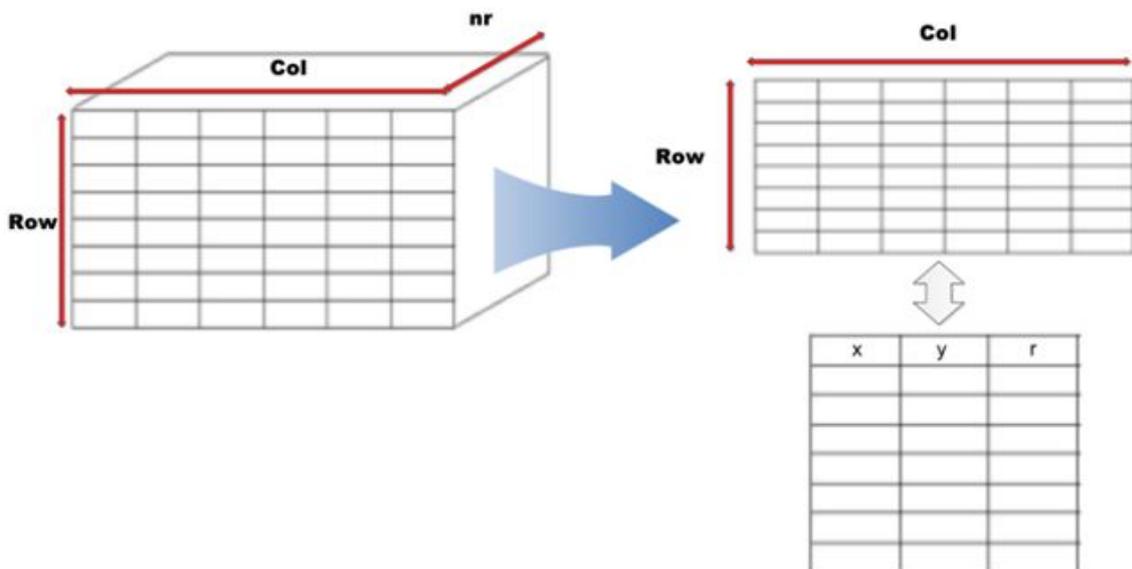


FIGURE 4.4: Passage de l'accumulateur 3D vers l'accumulateur 2D

Système de vote

Les contours de l'iris sont générés sous la forme d'un nuage de pixels d'une certaine épaisseur. Le système de vote classique ne suffit pas pour faire des détections précises, on a donc dû le modifier afin d'avoir de meilleurs résultats.

Comme expliqué dans la partie de la transformée de HOUGH, l'algorithme parcourt l'image de contour en dessinant en chaque pixel de coordonnées (x,y) des cercles dont le rayon R parcourt la plage de variation du rayon, pour calculer ensuite le nombre par lequel ce cercle passe (ce nombre représente le nombre de votes).

Notre idée consiste à augmenter le nombre de points qui participent au vote. Au lieu de prendre en considération seulement les votes des points par lesquels le cercle dessiné passe, exactement, et du fait que ce nombre peut être réduit à cause des mauvais résultats de la segmentation, on prend les votes des points qui appartiennent au disque délimité par les deux cercles concentriques de centre (x_i, y_i) et de rayons respectifs $(R + m)$ et $(R - m)$. Avec m une marge qu'on prend, elle est déterminée de façon expérimentale.

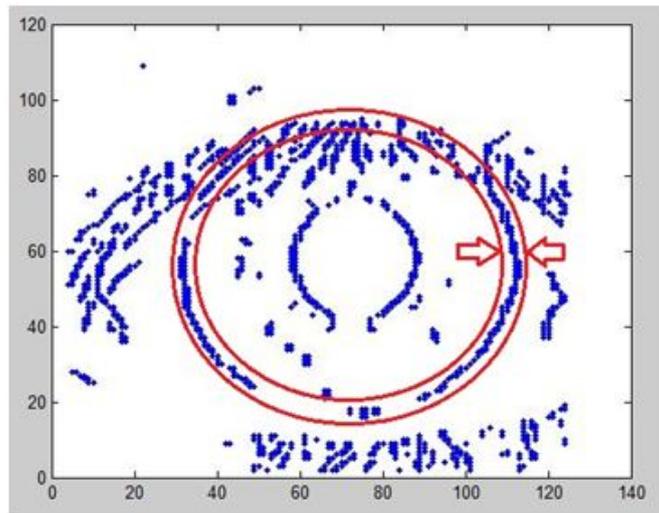


FIGURE 4.5: Nuage de pixels formant le contour de l'iris

Filtrage des détections à l'aide de la distance maximum entre la pupille et l'iris

L'iris et la pupille, sont des cercles non concentriques. Toutefois, la distance entre leurs deux centres ne dépasse pas une certaine valeur. Il est très intéressant d'exploiter cette information pour filtrer les fausses détections.

Ainsi, afin d'affiner les résultats de détection de l'iris, nous proposons un filtrage avant le calcul du cercle moyen, Parmi tous les cercles candidats à représenter le cercle de l'iris, lors du dépouillement des résultats du vote, on exclue les cercles dont la distance du centre par rapport à la pupille détectée dépasse la distance maximale (Cette distance est déterminée de façon expérimentale en pixels).

4.2 Étude détaillée du fonctionnement de l'algorithme de segmentation

Le déroulement de la segmentation se fera comme expliqué dans la figure 4.6 :

L'étage 1 charge d'abord l'image de l'iris et la stocke en mémoire. Il applique ensuite, le filtre CANNY et crée une image de contour globale. Les coordonnées des points de contour, la plage de variation du rayon de la pupille et la taille de l'image de contour générée sont aussitôt transférés au bloc de la CHT qui calcule les coordonnées du centre ainsi que le rayon de la pupille.

Le bloc les renvoie vers l'étage 2 où l'image de contour est créée en appliquant le filtre CANNY à nouveau. Il faut changer les pondérations pour avoir l'image de contour verticale.

Les données sont pour la seconde fois renvoyées vers le bloc de la CHT, qui calcule le centre et le rayon du cercle de l'iris. Il est plus convenable de chercher cette fois au voisinage du centre de la pupille détectée précédemment par souci d'optimisation.

Une fois la région de l'iris détectée, on dessine le résultat sur l'image originale, et on procède au calcul du masque. Le filtre CANNY génère cette fois l'image de contour horizontale et grâce à la transformée de HOUGH linéaire on détecte la ligne qui sépare les paupières de l'iris.

Une communication doit être établie entre les différents étages de la partie logicielle avec

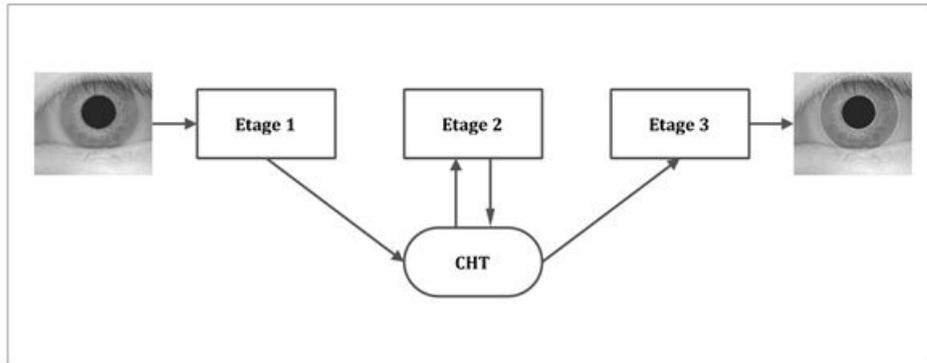


FIGURE 4.6: Déroulement de l'algorithme de segmentation

le bloc de la CHT pour l'échange des données et des résultats.

4.3 Simulation du module de segmentation par la transformée de Hough

Afin de mettre au point l'algorithme et dans le but de l'adapter à l'application de la segmentation de l'iris, nous avons effectué les tests en intégrant le bloc CHT dans l'environnement logiciel implémenté sur MATLAB. L'interfaçage et la communication des données entre MATLAB et le module de la CHT en langage C ont été géré par un échange de fichiers.

4.3.1 Organigramme de l'algorithme

- Initialisation l'accumulateur H, hmax est dynamique et fixé à 15
- Ri variable de parcours des rayons, $r = r_{min} + Ri$
- xi, yi variables de parcours des points de contour
- ai, bi variables de parcours de l'accumulateur toutes sont initialisées à 0
- mi variable de parcours de l'accumulateur transitoire.

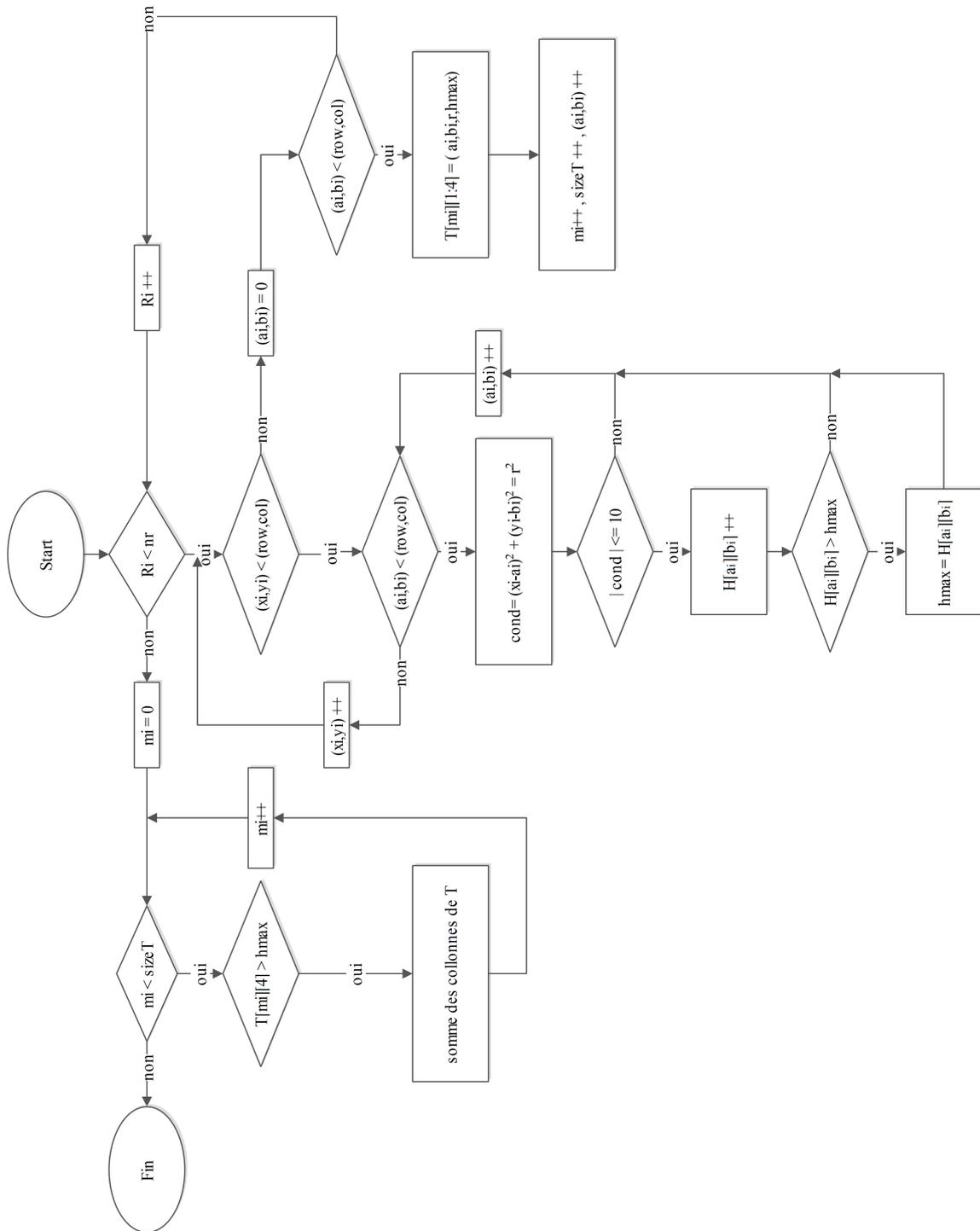


FIGURE 4.7: Organigramme de l'algorithme élaboré

4.3.2 Résultats de la simulation du module de segmentation élaboré

Nous avons appliqué le module de segmentation développé sur 66 images d'iris issues de la base CASIA V1.0.

Le tableau suivant donne les résultats statistiques de la segmentation :

	Bonnes	mauvaises
La pupille	100%	0%
L'iris	65%	35%

TABLE 4.1: Résultats de la segmentation en utilisant l'algorithme élaboré

Pour la pupille

Les résultats ont montré un score parfait de la détection de la pupille. En effet, sur 66 images segmentées, la détection de la pupille est parfaite. L'absence de mauvaises détections de la pupille donne un taux de détection de 100% (figure 4.8).

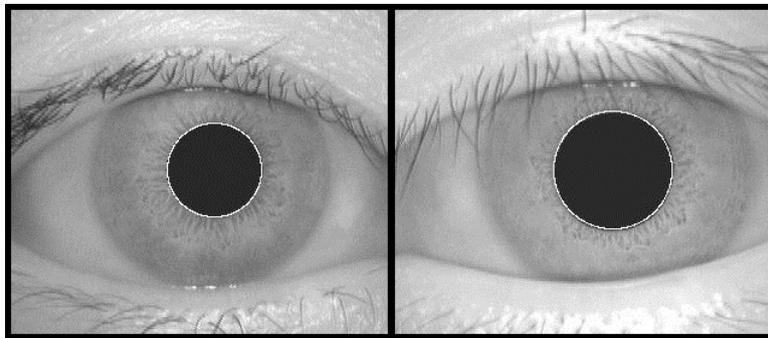


FIGURE 4.8: Résultats de la détection de la pupille

Pour l'iris

Les résultats de la segmentation de l'iris ont montré un certain nombre de détections erronées. En effet, sur les 66 images segmentées, il y'a eu 43 bonnes détections, soit un taux de 65% (figure 4.9).

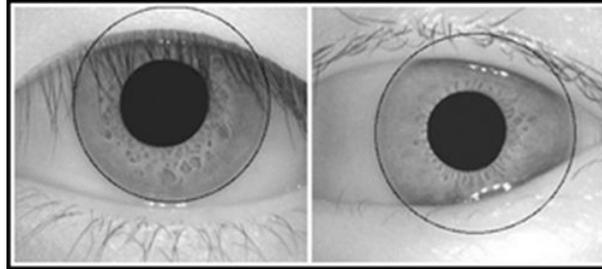


FIGURE 4.9: Résultats de la détection de l'iris

Ce qui laisse un taux d'erreur de 35%. Toutefois, parmi les mauvaises détections, il y'a le cas où seule une petite partie de la texture manque, et le cas où, le cercle détecté de l'iris est complètement faux (Fig 4.10).

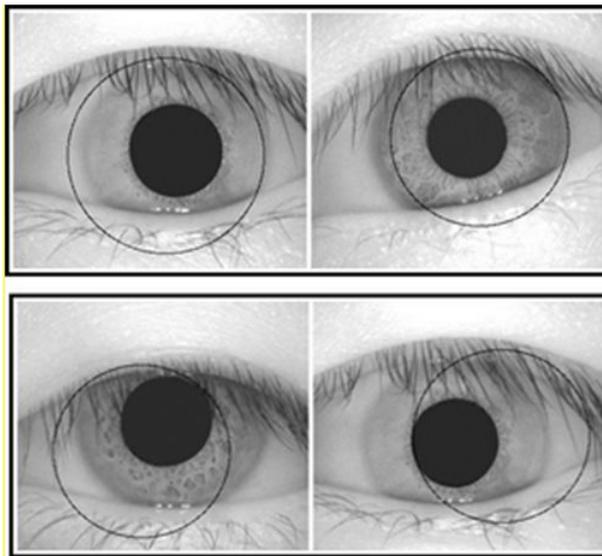


FIGURE 4.10: Les résultats pour la mauvaise détection de l'iris

À l'issue de ces résultats encourageants, on peut aborder l'implémentation de la CHT, sur la plateforme matérielle.

4.4 Partie Implémentation et résultats

Dans cette partie, nous commençons par une présentation de la plate-forme matérielle RC203E mise à notre disposition, de son environnement logiciel et de flot d'implémentation qu'elle supporte.

Par la suite, nous décrivons notre démarche d'implémentation de la CHT sur cette plate-forme. Enfin, nous décrivons des mesures réalisées et faisons ressortir l'influence des paramètres utilisés sur les performances du système, notamment sur le taux d'utilisation des ressources matérielles du FPGA.

4.4.1 Présentation de la plateforme d'implémentation

Le kit de développement RC203E de CELOXICA fournit une excellente plateforme pour l'implémentation matérielle. Elle est basée un FPGA VIRTEX-II de XILINX conçu pour des applications hautes performances. La Fig. 4.11 donne le diagramme simplifié de la carte de développement.

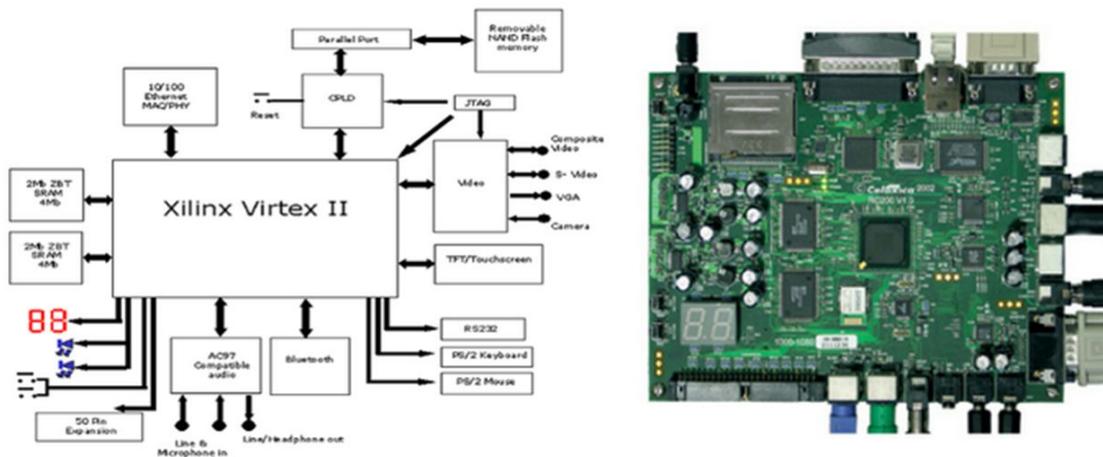


FIGURE 4.11: Le kit de développement RC203E de CELOXICA

L'environnement matériel

L'élément central de la carte est le circuit FPFA XC2V3000-4FG676 de la famille VIRTEX-II de XILINX. Il est doté de 3000 portes logiques réparties sur 14 K tranches du circuit (slices). Le FPGA intègre une mémoire distribuée dynamique (SDRAM) de 448 Kbits et 96 blocs RAM (BRAM) de 18 Kbits chacun. Il comporte 96 multiplieurs embarqués exécutant des opérations sur des mots de 18 bits.

La carte comporte un circuit programmable CPLD pour la configuration/reconfiguration du FPGA à travers le port parallèle et une interface JTAG pour la programmation de l'ISP PROM. Elle offre aussi de multiples interfaces d'entrées et de sorties. Elle possède une interface RS232 , un écran TFT de 640x480 pixels et deux connecteur PS/2 pour la souris et le clavier. Deux LEDs et deux afficheurs 7 segments à cathode commune sont présents sur la carte, et peuvent être utilisés durant la phase de test et de mises au point des programmes (debug).

L'environnement logiciel

Le kit de développement RC203E est livré avec un environnement de développement basé sur le langage HANDEL-C (voir Annexe) : Le DK/PDK design suite de CELOXICA. La dernière version (DK 5) est commercialisée sous le sigle AGILITY.



FIGURE 4.12: Kit de développement DK5

La suite DK est un outil de design ESL (Electronic System Level), c'est à dire qu'il fait abstraction du bas niveau au profit du développement d'architectures et d'algorithmes. En effet, le langage HANDEL-C avec une syntaxe proche du langage C, est une solution alternative à la programmation bas niveau (VHDL) du circuit FPGA.

L'environnement DK dispose d'un module de simulation. Il offre des algorithmes d'optimisation adaptés à une implémentation spécifique, et convertit le code HANDEL-C en éléments logiques standards dans un fichier EDIF servant d'entrée à la suite d'outils de XILINX (ISE). La figure 4.13 donne une représentation du e flot de conception.

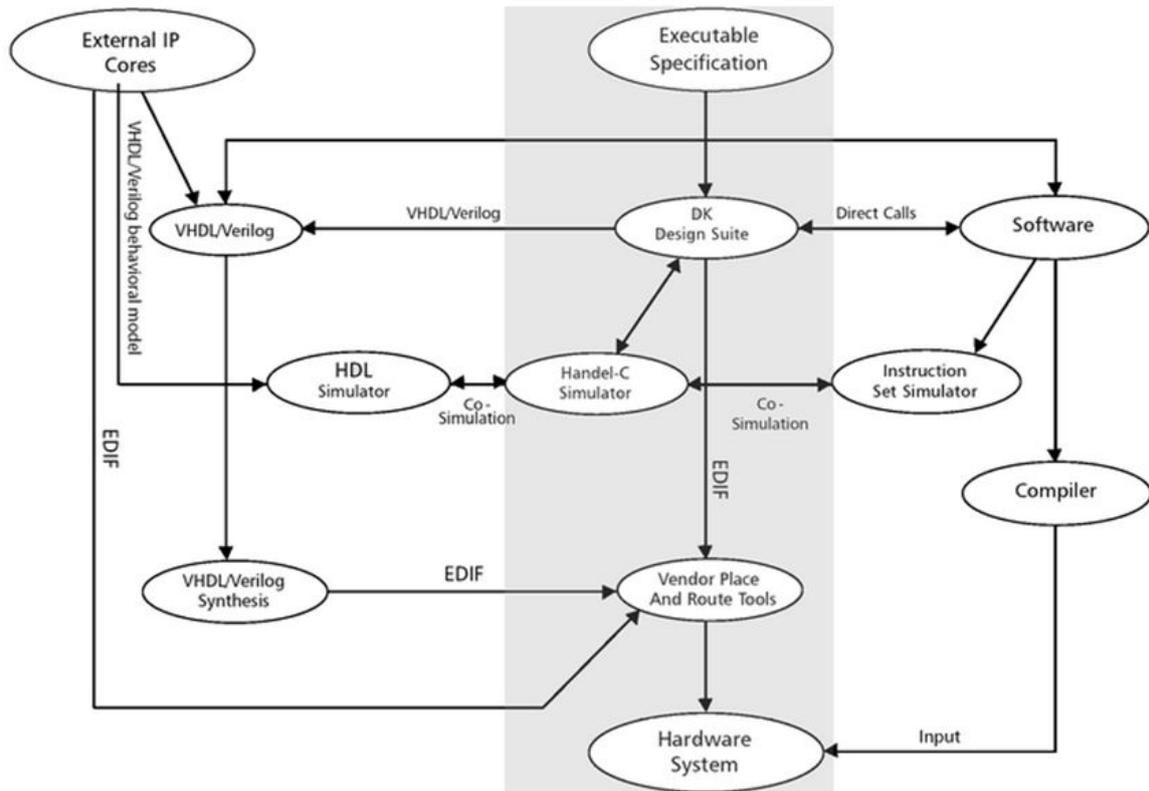


FIGURE 4.13: Flot de conception dans DK5

DK offre une couche d'abstraction fournissant les API d'accès aux composants matériels de la plateforme : la librairie PAL (Platform Abstraction Layer).

L'intérêt de cette couche est d'écrire un code identique alors que les cartes sont différentes, l'application HANDEL-C peut alors être portée sans difficulté sur une autres carte que l'original. Il suffit de bien choisir la configuration adéquate. Comme le fait apparaître la figure 4.14 la couche PAL établit le lien avec l'ensemble de pilotes spécifiques à la plateforme. Cela permet d'obtenir un niveau d'abstraction facilitant l'utilisation du matériel.

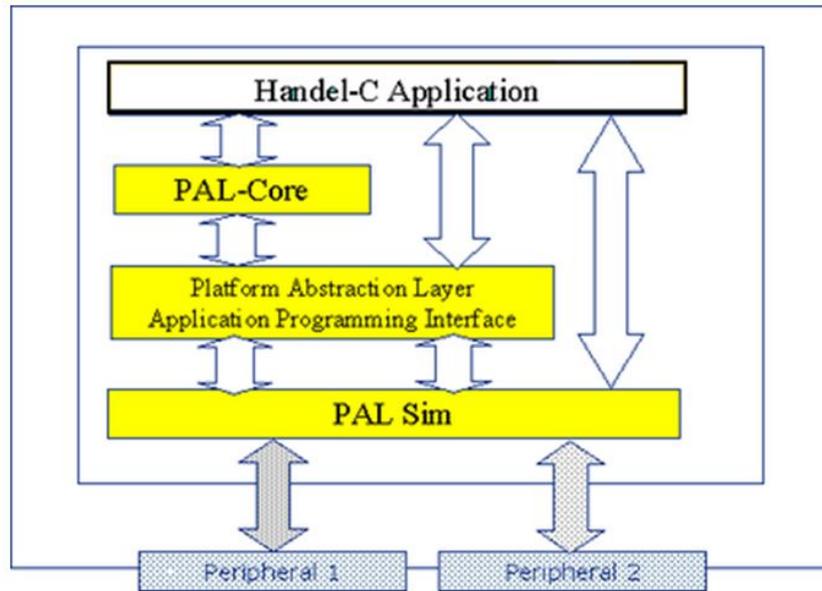


FIGURE 4.14: Couche PAL

Le développement de l'application dans HANDEL-C passe par deux étapes : la simulation et la synthèse. La suite DK a un double rôle :

- En tant que simulateur, elle permet de détecter les erreurs syntaxiques et logiques, pour ensuite simuler l'application sur la plateforme de simulation virtuelle (PALSIM)
- En tant que synthétiseur : il va jusqu'à la création d'un fichier EDIF contenant les informations sur la NETLIST. Le DK intègre des scripts permettant de faire appel à ISE pour générer le « bitstream » du FPGA cible.

4.4.2 Travaux réalisés sur la plateforme de développement

4.4.2.1 Codage Handel-C de la transformée de Hough circulaire

Le passage du langage C standard au langage HANDLE-C, ne se fait pas directement. Car il ne faut pas perdre de vue le fait qu'on s'intéresse à l'implémentation sur FPGA de la CHT et non à l'exécution séquentielle sur un CPU. Il faut penser à introduire du parallélisme pour accélérer l'exécution. Mais plus il y'a du parallélisme, plus on consomme les ressources de la carte.

Parmi les difficultés rencontrées lors du portage de l'algorithme on cite :

- HANDEL C est un langage fortement typé. Un type entier de 8 bits est un type complètement différent d'un type entier de 7 bits ou d'un entier de 8 bits non signé (unsigned 8), il est donc impossible de faire des affectations ni des comparaisons entre des types différents.
- L'affectation entre des variables de tailles différentes doit se faire en utilisant les instructions de manipulations de bits, comme la concaténation avec 0 pour augmenter la taille, et le [:] pour sélectionner un nombre de bits bien défini et l'affecter à une variable de taille inférieure.
- On ne peut pas exécuter deux instructions d'accès mémoire en un cycle d'horloge (opération interdite)
- L'incréméntation par la syntaxe : « H[i][j]++ » dans le cas où H est déclarée comme RAM est tolérée seulement en simulation puisqu'elle fait la lecture et l'incréméntation en même temps .
- Il en est de même pour les tests sur des RAM, il faut passer par une variable intermédiaire.

Ci-après un extrait du code HANDEL C de la CHT, le code complet de l'algorithme est présenté à l'annexe.

4.4.2.2 Implémentation de la CHT sur la plateforme virtuelle PALSIm

Le code HANDEL-C de la CHT est écrit de façon à s'exécuter soit sur la plateforme virtuelle PALSIm, soit sur la carte RC203E. Dans un premier temps nous avons testé notre code sur le simulateur DK. Les résultats obtenus sont conforme à la simulation logicielle.

Sur la figure 4.16 une capture d'écran de l'exécution sur la plateforme PALSIm.

4.4.2.3 Implémentation et gestion de la communication FPGA/CPU

Pour la communication des données et des paramètres issus de la partie logicielle sur MATLAB (les edges map) vers la carte FPGA, et la récupération des résultats du calcul

```
58 static macro proc PAL_Requires()
59 {
60     seq
61     {
62         PalVersionRequire (1, 0);
63         PalSevenSegRequire (1);
64         PalPS2PortRequire (2);
65         #ifndef USE_SIM
66             PalRS232PortRequire (1);
67         #endif
68         PalVideoOutRequire (1);
69         PalPL1RAMRequire (1);
70     }
71 }
72 static macro proc PAL_Resources_Enable()
73 {
74     par
75     {
76         #ifndef USE_CONSOLE
77             PalConsoleEnable (ConsolePtr);
78         #endif
79         #if 0 // Not used
80             PalSevenSegEnable (PalSevenSegCT (0));
81             PalSevenSegEnable (PalSevenSegCT (1));
82         #endif
83     }
84 }
```

FIGURE 4.15: Extrait du code HANDE-C de la CHT

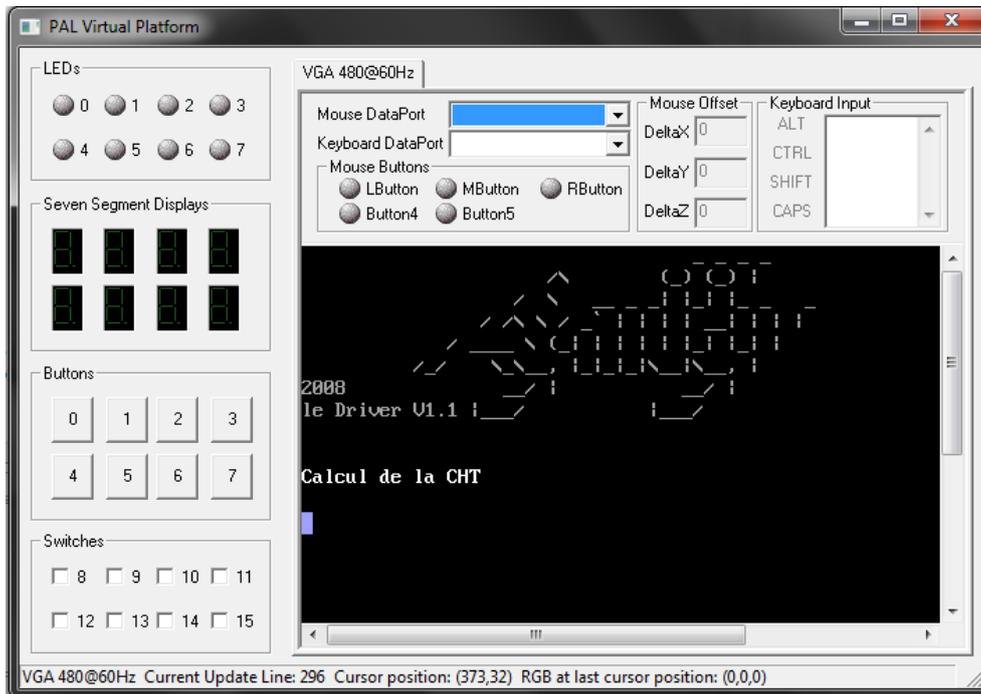


FIGURE 4.16: l'exécution sur la plateforme PALSIm

de la CHT implémentée sur FPGA (paramètres du cercle x_0 , y_0 et R), nous avons opté pour l'utilisation du port série RS232 de la carte.

Le choix du port série RS232 est justifié par le fait qu'il soit adapté aux données de l'application. En effet, les différents paramètres en entrée et en sortie du bloc de la CHT peuvent être écrits sur 8bits, ce qui correspond à la taille des données qu'utilise le port RS232. Nous avons exploré les possibilités de communication de la carte RC203E (les macros de la librairie PAL) et la fonction de gestion de l'interface série de MATLAB (la classe Serial et ses différents modes de configuration), pour établir une communication entre le PC et la carte FPGA. les figures 4.17 et 4.18 présentent un simple programme de test de la communication dans les deux sens, en émission (du PC vers le FPGA) et en réception (du FPGA vers le PC).

Le programme complet HANDLE-C/ MATLAB de la gestion de la communication est donné en annexe.



```

% initialisations
clear all
close all
clc
RS232_FPGA=0;
Terminator='X'; %caractere attendu
disp('Port Serie en matlab ...');
% Disconnect and delete all instrument objects
instrreset
% Create the serial interface object,
%create the serial port
s21 = serial('COM8');

%set some properties
set(s21, 'BaudRate', 2400);
set(s21, 'FlowControl', 'hardware');
set(s21, 'Terminator', '3');
set(s21, 'Parity', 'none'); %pair
fwrite(s,DATA,'uint8')

PalVersionRequire (1, 0);
PalVideoOutRequire (1);
PalRS232PortRequire (1);
par
{
    PalConsoleRun (&ConsolePtr, PAL_CONSOLE_FONT_NORMAL,
    PalVideoOutOptimalCT (ClockRate), ClockRate);
    PalDataPortRun (RS232Port, ClockRate);

par
{
    PalConsoleEnable (ConsolePtr);
    PalDataPortEnable (RS232Port);

par {
    while (1) // Boucle de Réception
    {
        PalDataPortRead (RS232Port,&variable);
        SequenceChannel ! variable;
    }
}
}
    
```

FIGURE 4.17: Communication MATLAB/FPGA

```
PalVersionRequire (1, 0);
PalVideoOutRequire (1);
PalRS232PortRequire (1);
par
{
    PalConsoleRun (&ConsolePtr, PAL_CONSOLE_FONT_NORMAL,
PalVideoOutOptimalCT (ClockRate), ClockRate);
    PalDataPortRun (RS232Port, ClockRate);
par
{
    PalConsoleEnable (ConsolePtr);
    PalDataPortEnable (RS232Port);
}
par {
    while (1) // Boucle de Réception
    {
        SequenceChannel ? variable;
        PalDataPortWrite (RS232Port,variable);
    }
}
}

% initialisations
clear all
close all
clc
RS232_FPGA=0;
Terminator='X'; %caractere attendu
disp('Port Serie en matlab ...');
% Disconnect and delete all instrument objects
instrreset
% Create the serial interface object,
%create the serial port
s21 = serial('COM8');

%set some properties
set(s21, 'BaudRate', 2400);
set(s21, 'FlowControl', 'hardware');
set(s21, 'Terminator', '3');
set(s21, 'Parity', 'none'); %pair
% s21.ReadAsyncMode='continuous';
s21.BytesAvailableFcnCount = 1;
s21.BytesAvailableFcnMode = 'byte';
%s22.BytesAvailableFcnMode = 'terminator';
time = datestr(now,0);
s21.BytesAvailableFcn = {@my_BytesAvailable,time};
%open the interface
fopen(s21)
```



FIGURE 4.18: FPGA vers MATLAB

Conclusion et perspectives

Les travaux présentés dans ce mémoire répondent à une problématique qui se situe à l'intersection de l'informatique et de l'électronique : Simulation et implémentation sur FPGA d'un module de segmentation pour un système de reconnaissance biométrique basée sur l'iris.

Nous nous sommes intéressés en particulier dans un premier temps à l'étude algorithmique du module de segmentation et nous avons élaboré une nouvelle approche pour la localisation des cercles de l'iris et de la pupille ; à partir d'une spécification de haut niveau, en se basant sur le système de référence MASEK.

Notre démarche a consisté à procéder en deux étapes : dans un premier temps, on essaye d'isoler le cercle de la pupille en utilisant la transforme de HOUGH circulaire (CHT), car ce cercle est plus facile à détecter. Par la suite on lance une nouvelle détection au voisinage du centre de la pupille en configurant la CHT avec la plage de variation du rayon de l'iris.

Nous avons aussi proposé un nouveau système de vote pour la CHT, au lieu de prendre en considération seulement le vote des points qui réalisent exactement l'équation du cercle, on tolère le vote des points du voisinage, appartenant à un disque délimité par deux cercles concentriques avec une certaine tolérance sur le rayon. Nous avons aussi introduit un filtrage afin d'affiner les résultats de détection : parmi tous les cercles candidats à représenter le cercle de l'iris, lors du dépouillement des résultats du vote, on exclue les cercles dont la distance du centre par rapport à la pupille détectée dépasse une certaine distance maximale.

Les objectifs initiaux du projet ont été atteints avec succès :

- Une étude et une simulation d'un système de référence a été réalisée, le système MASEK en l'occurrence.
- Le profilage et le partitionnement logiciel/matériel d'un système de reconnaissance biométrique basé sur l'iris a été effectué.
- Le portage sous *HANDEL-C* et l'implémentation sur FPGA de la transformation de *HOUGH* circulaire ont été réalisés.
- Enfin, les problèmes de l'interfaçage FPGA/CPU ont été étudiés et un module de gestion de la communication a été développé.

Le système implémenté a le comportement désiré, mais l'architecture matérielle utilise pratiquement toutes les ressources du FPGA. Alors en perspectives, il est nécessaire d'optimiser l'algorithme de la CHT, afin de réduire le nombre de ressources nécessaires. A notre avis, une nouvelle architecture faisant appel à un processeur *CORDIC* pour le calcul de l'équation du cercle, va permettre l'implémentation de la CHT avec une complexité matérielle raisonnable.

Bibliographie

- [1] [http ://www.biometrie-online.net](http://www.biometrie-online.net).
- [2] Matlab source code for a biometric identification system based on iris patterns. *The University of Western Australia*, 2003.
- [3] PHAN Viet Anh. Developpement d'un module de segmentation pour un système de reconnaissance biométrique basé sur l'iris. *Memoire de fin d'étude, ÉVRY, France*, 2008.
- [4] W.W. Boles. A wavelet transform based technique for the recognition of the human iris. *Proc. of International Symposium on Signal Processing and its Application, Australia*, 1996.
- [5] W.W. Boles. A security system based on human iris identification using wavelet transform. *Proc. of the 1st International Conf. on Knowledge-based Intelligent Electronic Systems, Adelaide, Australia*, 1997.
- [6] W.W. Boles and B. Boashash. A human identification technique using images of the iris and wavelet transform. *IEEE Transactions on signal processing*, 46, N 4, 1998.
- [7] J. Daugman. High confidence personal identification by rapid video analysis of iris texture. *Proc. of the IEEE for the International Conference on Security Technology*, 1992.
- [8] J. Daugman. High confidence recognition of persons by rapid video analysis of iris texture. *Publication of IEE for Conf. on European Convention on Security and Detection*, 1995.
- [9] John Daugman. High confidence visual recognition of persons by a test of statistical independence. *IEEE Trans. Pattern Anal. Mach.*

BIBLIOGRAPHIE

- [10] J.S. Ahrens F. Bouchier and G. Wells. Laboratory evaluation of the iris scan prototype biometric identifier. *Sandia National Laboratories, Albuquerque, NM, Tech, Rep.SAND96-1033*, 1996.
- [11] Mohamed Nadir Khemakhem. Vérification de l'identité multimodale dans des séquences vidéo. *Institut Télécom-ParisTech, CNRS-LTCI*, 2010.
- [12] Emine Krichen. Reconnaissance des personnes par l'iris en mode dégradé. *Thèse de doctorat*, 2007.
- [13] T. Tan L. Ma, Y. Wang. Iris recognition using circular symmetric filters. *Proc. of ICPR02, Québec city, Canada*, 2002.
- [14] A. Witkin M. Kass and D.Terzopoulos. Snakes. Active contour models. *International Journal of Computer Vision*, 1987.
- [15] Libor Masek. Recognition of human iris patterns for biometric identification. *The University of Western Australia*, 2003.
- [16] C. Sanchez Avila R. Sanchez-Reillo and A. Gonzales-Marcos. Improving accesscontrol security using iris identification. *Proc. of the Conf. BMES/EMBS, IEEEPublication, Atlanta*, 1999.
- [17] R.P. Wildes. A system for automated iris recognition. *Proc. of 2nd IEEE Workshop on Applications of Computer Vision*, pp. 121-128, 1994.
- [18] R.P. Wildes. Iris recognition : an emerging biometric technology. *Proc. of theIEEE*, 85, N9, 1997.
- [19] T. Tan Y. Zhu and Y. Wang. Biometric personal identification based on iris patterns. *Proc. of IAPR, International Conf. on Pattern Recognition, II*, pp. 805-808, 2000.

Annexe A

A.1 Les filtres de Canny

Le filtre de CANNY (ou détecteur de CANNY) est une méthode de détection de contours utilisée en traitement d'image, qui par étapes, utilise plusieurs algorithmes et filtres pour détecter un large éventail de contours sur une image.

Elle a été développée par John F. CANNY en 1986, et elle a été conçue pour être optimale suivant trois critères :

- bonne détection : faible taux d'erreur dans la signalisation des contours,
- bonne localisation : minimisation des distances entre les contours détectés et les contours réels,
- unicité de la réponse : une seule détection pour un contour et évite ainsi les effets de rebond.

L'objectif étant de bien mettre en évidence les contours de l'image analysée, la détection de contour se fait sur plusieurs étapes :

Réduction du bruit (Lissage) :

Cette première étape est de réduire le bruit de l'image originale, ce qui permet d'éliminer les pixels isolés susceptibles d'avoir de fortes réponses lors du calcul du gradient, conduisant ainsi à de faux contours. Cette étape est assurée par un filtrage gaussien 2D.

Calcul du gradient :

Après le lissage de l'image, l'étape suivante consiste en l'emploi d'un gradient qui retourne l'intensité des contours. Le gradient est calculé selon les directions de x et y avec les dérivées directionnelles. Cette étape permet aussi de choisir la direction de l'edge map par la suite et en fonction des pondérations des directions x et y, l'edge map verticale pour la détection de la région de l'iris. Pour la détection de la pupille on utilise une edge map avec des pondérations x et y équilibrés.

Suppression des non-maxima :

La carte des gradients obtenue précédemment fournit une intensité en chaque point de l'image. Une forte intensité indique une forte probabilité de présence d'un contour.

Toutefois, cette intensité ne suffit pas à décider si un point correspond à un contour ou non. Seuls les points correspondant à des maxima locaux sont considérés comme correspondant à des contours, et la prochaine étape de la détection. La fonction supprime les non-maxima sur l'image selon l'orientation de l'image. La fonction est appelée sous le nom : « nonmaxsup ».

Seuillage des contours :

La différenciation des contours sur la carte générée se fait par seuillage à hystérésis. Cela nécessite deux seuils, un haut et un bas ; qui seront comparés à l'intensité du gradient de chaque point. Le critère de décision est le suivant :

Pour chaque point, si l'intensité de son gradient est :

- Inférieur au seuil bas, le point est rejeté ;
- Supérieur au seuil haut, le point est accepté comme formant un contour ;
- Entre le seuil bas et le seuil haut, le point est accepté s'il est connecté à un point déjà accepté.

A.2 Transformée de Hough linéaire

Une droite peut être décrite par l'équation : $y = ax + b$. Les paramètres de l'espace de HOUGH sont donc : a et b et forment un espace bidimensionnel.

La droite $y = a_0x + b_0$ va être représentée par un point dans l'espace de HOUGH (a, b) comme montré sur la figure A.1.

Aussi : la droite $b = -ax_0 + y_0$ dans l'espace de HOUGH représente l'ensemble de droites passant par le point (x_0, y_0) . La figure A.1 illustre les deux cas :

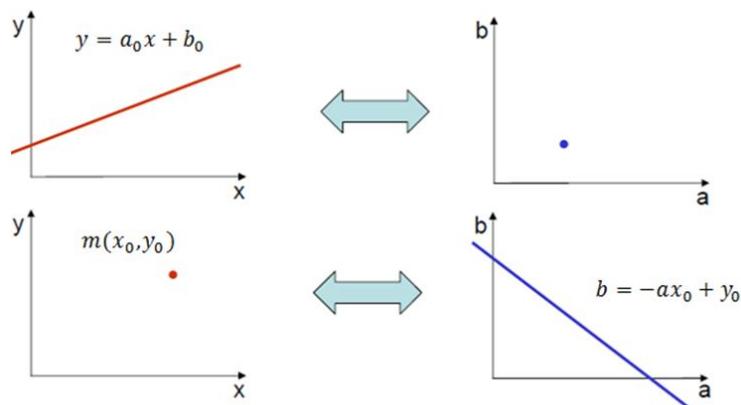


FIGURE A.1: Représentation d'une droite et d'un point dans les espaces cartésien et de HOUGH

La méthode de détection d'une droite sera donc la suivante :

- Appliquer une détection de contour.
- Discrétiser le plan des paramètres (a, b) (dit l'espace de HOUGH) et initialiser un accumulateur.
- Pour chaque point de contour Déterminer sa droite image dans l'espace de HOUGH et incrémenter l'accumulateur sur les points de cette droite.
- Recherche des maximas qui donnent les paramètres des droites sur l'image.

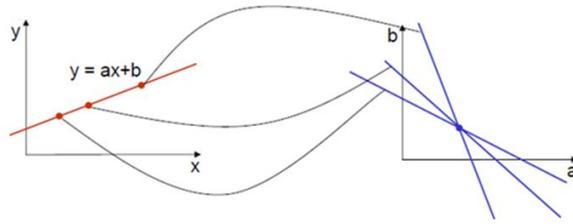


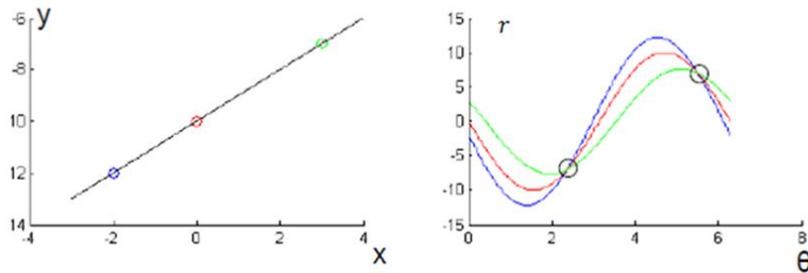
FIGURE A.2: Détection d'une droite à partir de trois points

Remarque :

La paramétrisation des droites par l'équation d'une droite « $y = ax + b$ » a des limites car elle rend la détection des droites verticales impossible. En effet pour une droite verticale : $a \rightarrow \infty$ or l'espace de HOUGH doit être borné.

En prenant « $x = r \cos \theta$ » et « $y = r \sin \theta$ » l'équation devient :

$$x \cos \theta + y \sin \theta - r = 0$$

FIGURE A.3: Détection d'une droite à partir de trois points dans l'espace (r, θ) **A.3 Handel - C**

HANDEL-C a permis de réaliser des applications impressionnantes, comme des jeux vidéo sans ordinateur, en branchant un moniteur VGA directement en sortie d'un circuit XILINX qui pilotait à la fois l'affichage et le jeu. Le même programme, compilé sur une Sun Sparc-5 s'exécute 5 fois plus lentement.

C'est un langage qui permet de générer des circuits logiques, sans être pour autant un HDL (Hardware Description Language). Il est plutôt destiné à compiler des algorithmes

de haut niveau pour en faire des portes logiques (niveau hardware), sans que l'utilisateur ne se préoccupe de savoir exactement comment fonctionne le calculateur. C'est en fait un langage de programmation plutôt qu'un HDL. En un sens, le HANDEL-C est au hardware ce qu'un langage de haut niveau conventionnel est à l'assembleur.

Les programmes en Handel-C

La syntaxe du HANDEL-C est basée sur celle du langage C dont elle reprend de nombreuses structures. L'ordre d'écriture des instructions correspond exactement à leur ordre d'exécution. Bien sûr, comme tout langage qui se respecte, le HANDEL-C contient des structures de contrôle du flot du programme.

Par exemple, l'exécution d'une partie de code peut être conditionnée par la valeur d'une expression, ou un bloc de code peut-être répété un certain nombre de fois grâce à une boucle.

Il est important de noter que les circuits logiques générés par le HANDEL-C sont exactement ceux nécessaires à l'exécution du programme. Il n'y a pas d'interpréteur comme en assembleur ; les portes logiques constituant le circuit HANDEL-C final sont des instructions assembleur du système HANDEL-C.

Les programmes parallèles

Puisque le compilateur HANDEL-C génère du hardware de bas niveau, il est possible d'accroître les performances du système en utilisant le parallélisme. Bien que le séquentiel soit inhérent au HANDEL-C, on peut (et c'est même parfois primordial) demander au compilateur de créer du hardware pour exécuter des instructions en parallèle. Et il s'agit d'un vrai parallélisme : deux instructions seront exécutées exactement au même instant par deux structures hardware bien distinctes (contrairement aux ordinateurs classiques qui en donnent seulement l'illusion).

Si une branche arrive au niveau du canal avant l'autre, elle attend jusqu'à ce que cette dernière soit elle aussi prête pour le transfert de données.

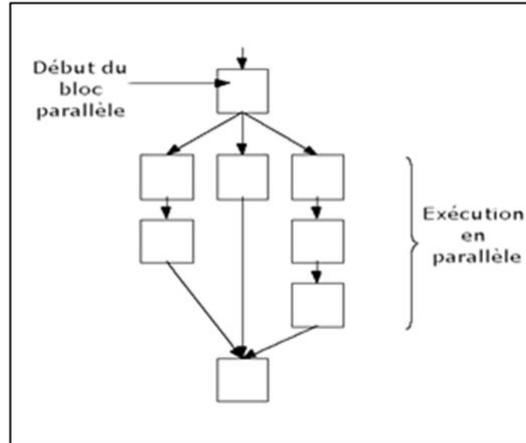


FIGURE A.4: Séparation et regroupage des branches parallèles

Quand un bloc parallèle est rencontré, chacune de ses branches est exécutée simultanément au départ du bloc en question. Les flots d'exécution parallèles se rejoignent à la fin du bloc quand toutes les branches ont été exécutées. Toute branche se terminant très rapidement est obligée d'attendre la plus lente avant de continuer (c'est-à-dire avant que l'on ne sorte du bloc parallèle).

Les canaux de communication

Les canaux constituent le lien entre les branches parallèles. Une des branches fournit des données sur le canal tandis que l'autre la lit. Les canaux servent aussi à synchroniser les différentes branches parallèles car le transfert de données ne peut avoir lieu que si chacune des branches est prête pour le faire : si la branche émettrice n'est pas prête pour la communication alors la réceptrice doit attendre, et vice versa (voir Fig A.2).

Portée et partage des variables

La portée des déclarations est, comme en C traditionnel, basée autour des blocs de code (un bloc étant encadré par des accolades ...). En gros, cela signifie, d'une part, que les variables globales doivent être déclarées en dehors de tout bloc et, d'autre part, qu'une variable déclarée dans un bloc sera valide dans celui-ci ainsi que dans tous ses sous blocs. La A.6 nous en donne une illustration.

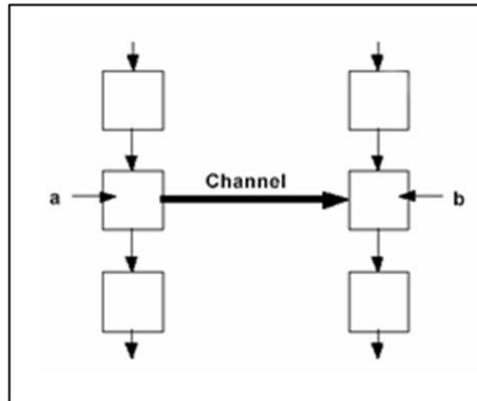


FIGURE A.5: Communication par canal entre branches parallèles

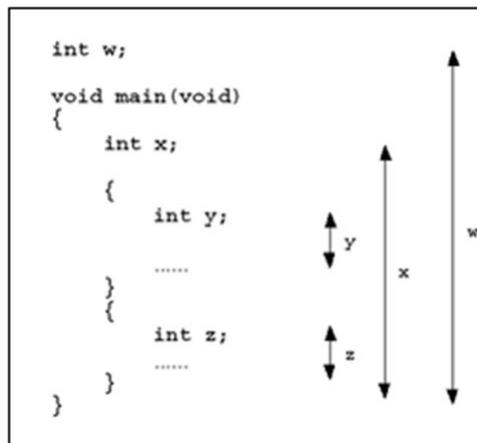


FIGURE A.6: Domaine de validité des variables

Puisque les structures parallèles sont de simples blocs de code, une variable peut-être valide dans deux branches parallèles : il suffit par exemple d'avoir un bloc principal ou on déclare sa variable et d'avoir deux sous blocs parallèles. Ceci peut entraîner des conflits de ressource si plusieurs sous blocs parallèles accèdent en même temps à la variable en question. La syntaxe du HANDEL-C stipule qu'une variable ne peut pas être accédée par plus d'une branche parallèle à la fois.

Différences fondamentales entre le Handel-C et le C

Lors du portage d'un programme du C vers le HANDEL-C, il convient de faire attention, car le HANDEL-C, bien qu'il soit basé sur la syntaxe du C, comporte tout de même quelques particularités importantes. La liste ci-dessous en présente les principales :

- Les flottants (nombres à virgules) n'existent pas en HANDEL-C. Il n'y a que des entiers.
- Les pointeurs non plus (les pointeurs existent en réalité mais n'ont pas la même fonction).
- Dans un cycle d'horloge, les RAM et les ROM ne peuvent avoir qu'une seule entrée accédée. Ainsi deux branches parallèles ne peuvent pas accéder à une RAM en même temps par exemple.

A.4 Architecture et ressource du FPGA Virtex II

Le FPGA VIRTEX II est principalement composé d'une matrice de CLB. Chaque bloc configurable est composé de 4 éléments identiques appelés slices et de deux buffers. Les slices sont identiques et contiennent :

- Deux générateurs de fonctions (composés de LUTs à quatre entrées).
- Deux éléments de mémorisation (bascules D).
- Des portes logiques et des multiplexeurs.
- Une chaîne de cascade horizontale (portes OR).

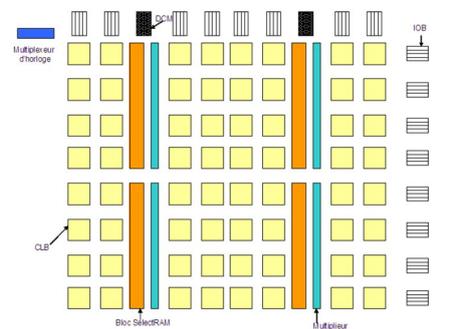


FIGURE A.7: Architecture de la VIRTEX II

Chaque bloc est connecté à une matrice de commutation pour accéder aux ressources de routage général. Les blocs mémoires sont de 18kb assignables de 16k * 1bit à 512*36bits. Chaque mémoire possède deux ports synchrones et indépendants l'un de l'autre. A chaque bloc mémoire est associé un multiplieur 18*18 optimisé pour des opérations sur le bloc mémoire.

Chaque multiplieur et bloc mémoire est relié à quatre matrices de commutation pour accéder aux ressources de routage. Les blocs DCM (Digital Clock Multiplexer) et le multiplexeur d'horloge global fournissent des fonctionnalités d'horloges évoluées. Au total il y a 12 DCM sur l'ensemble du FPGA.

A.5 Matlab

```
% Programme principale :
global image;
global circleiris;
circleiris=[0 0 0];
image_name='002_1_2.bmp';
image=imread(image_name);
etage1;
open hough_bornes_dyn.exe;
fid=fopen('pfe_result.dat','r');
circlepupil=fscanf(fid,'%u')
a=fclose(fid);
etage2;
open hough_bornes_dyn.exe;
fid=fopen('pfe_result.dat','r');
circleiris=fscanf(fid,'%u')
a=fclose(fid);
%a=dessin2(image_name,'-iris.jpg',circleiris(1),circleiris(2),circleiris(3));
%a=dessin2(image_name,'-pupil.jpg',circlepupil(1),circlepupil(2),circlepupil(3));
```

```
% etage3;

% Etage 1
% Fixer les paramètres de la base de données
%CASIA
pupil_min = 28 ; pupil_max = 75 ; scaling=0.4 ; horz=1 ; vert=1 ; sigma=2 ; hithres=0.25 ;
lowthres=0.25 ;
% Changement de l'échelle
scpup_min = round(pupil_min*scaling) ;
scpup_max = round(pupil_max*scaling) ;
rd = round(pupil_max*scaling - pupil_min*scaling) ;
% Détection des points de contour :
[I2 or] = canny(image, sigma, scaling, vert, horz) ;
I3 = adjgamma(I2, 1.9) ;
I4 = nonmaxsup(I3, or, 1.5) ;
edgemap = hysthresh(I4, hithres, lowthres) ;
[rows,cols]=size(edgemap) ;
% Extraction des points de contour depuis l'edgemap
[x,y]=find(edgemap~=0) ;
nxy=size(x) ;
% subplot 211
% plot(x,y,')
% subplot 212
xx=y ;
y=abs(rows-x) ;
x=abs(xx) ;
% plot(x,y,') ;
% Sauvegard des points de contour dans un fichier .dat
fid=fopen('pfe_variable_x.dat','w') ;
fprintf(fid,'%u \n',x) ;
a=fclose(fid) ;
fid=fopen('pfe_variable_y.dat','w') ;
fprintf(fid,'%u \n',y) ;
```

```
a=fopen(fid);
fid=fopen('pfe_variable_other.dat','w');
fprintf(fid,'%u \n',rows,cols,rd,scup_min,scup_max,nxy(1));
a=fopen(fid);

% Etage 2
% Fixer les paramètres de la base de données CASIA
iris_min = 80; iris_max = 150; scaling=0.4; horz=0; vert=1; sigma=2; hithres=0.20;
lowthres=0.19; reflecthres = 240;
% Changement de l'échelle
sciris_min = round(iris_min*scaling);
sciris_max = round(iris_max*scaling);
rd = round(iris_max*scaling - iris_min*scaling);
% Détection des points de contour :
[I2 or] = canny(image, sigma, scaling, vert, horz);
I3 = adjgamma(I2, 1.9);
I4 = nonmaxsup(I3, or, 1.5);
edgemap = hysthresh(I4, hithres, lowthres);
[rows,cols]=size(edgemap);
% Extraction des points de contour depuis l'edgemap
[x,y]=find(edgemap~=0);
nxy=size(x);
xx=y;
y=abs(rows-x);
x=abs(xx);
% Sauvegard des points de contour dans un fichier .dat
fid=fopen('pfe_variable_x.dat','w');
fprintf(fid,'%u \n',x);
a=fopen(fid);
fid=fopen('pfe_variable_y.dat','w');
fprintf(fid,'%u \n',y); a=fopen(fid);
fid=fopen('pfe_variable_other.dat','w');
fprintf(fid,'%u \n',rows,cols,rd,sciris_min,sciris_max,nxy(1));
```

```
a=fclose(fid);

%Etage 3
imagewithcircles = uint8(image);
[x,y] = circlecoords([circleiris(2),circleiris(1)],circleiris(3),size(image));
ind2 = sub2ind(size(image),double(y),double(x));
[xp,yp] = circlecoords([circlepupil(2),circlepupil(1)],circlepupil(3),size(image));
ind1 = sub2ind(size(image),double(yp),double(xp));
imagewithcircles(ind2) = 255;
imagewithcircles(ind1) = 255;
imwrite(imagewithcircles,[image_name,'-segmented.jpg'],'jpg');
```

A.6 CHT Code C

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
int row,col,nr,rmin,rmax,sizex; // les variables externes importées de Matlab
int i,j,k;
FILE* fichier = NULL;
fichier = fopen("pfe_variable_other.dat", "r");
if (fichier != NULL)
{ fscanf(fichier, "%d",&row); fscanf(fichier, "%d",&col); fscanf(fichier, "%d",&nr);
fscanf(fichier, "%d",&rmin); fscanf(fichier, "%d",&rmax); fscanf(fichier, "%d",&sizex);
fclose(fichier); }
else {printf("Impossible d'ouvrir le fichier Var other ");
return 0;}
int x[sizex],y[sizex];
// ***** Fichier VAR X *****
fichier = fopen("pfe_variable_x.dat", "r");
if (fichier != NULL) {
```

```

for(i=0;i<sizeX;i++)
fscanf(fichier, "%d", &x[i]);
fclose(fichier); }
else {printf("Impossible d'ouvrir le fichier Var X ");
return 0;}
// ***** Fichier VAR Y *****
fichier = fopen("pfe_variable_y.dat", "r");
if (fichier != NULL) {
for(i=0;i<sizeX;i++)
fscanf(fichier, "%d", &y[i]); }
else {printf("Impossible d'ouvrir le fichier Var Y ");
return 0;}
// Création et initialisation de l'accumulateur :
int h[row][col];
for(i=0;i<row;i++)
for(j=0;j<col;j++)
h[i][j]=0;
//printf("Initialisation de T\n");
int t[100][4];
for (i=0;i<100;i++)
for(j=0;j<4;j++)
t[i][j]=0;
// ***** Remplissage accumulateur *****
//printf("Remplissage de H\n");
int hmax=8; k=0; int r1,cx,cy,cond,r,a,b,rf;
for(r1=0;r1<=nr;r1++)
{ for(a=(r1+rmin);a<(row-r1-rmin);a++)
{for(b=(r1+rmin);b<(col-r1-rmin);b++)
{ for(i=0;i<sizeX;i++)
{ cx=x[i]; cy=y[i]; r=r1+rmin;
cond =((cx-a)*(cx-a)+(cy-b)*(cy-b)-r*r);
if(cond>=-10 && cond<=10) h[a][b]++;}
// test de tous les points de contour

```

```

if(h[a][b]>hmax)
{hmax=h[a][b];
printf("%d\n",hmax); } }}
for(a=(r1+rmin);a<(row-(r1+rmin));a++)
{for(b=(r1+rmin);b<(col-(r1+rmin));b++)
{ if(h[a][b]==hmax)
{t[k][0]=a; t[k][1]=b; t[k][2]=r; t[k][3]=h[a][b]; k++; }
h[a][b]=0;}}}
int siset; siset=k; k=0;
for(a=0;a<siset;a++)
{ if(t[a][3]==hmax)
{ t[k][0]=t[a][0]; t[k][1]=t[a][1]; t[k][2]=t[a][2]; t[k][3]=t[a][3]; k=k+1; }
// Calcul de la moyenne : 1ère étape la somme :
siset=k; cx=0;cy=0;rf=0;
for(a=0;a<siset;a++)
{ cx=cx+(row-t[a][1]); cy=cy+t[a][0]; rf=rf+t[a][2]; }
cx=(cx/siset)/0.4;
cy=(cy/siset)/0.4;
rf=(rf/siset)/0.4;
printf("resultat final : C[%d,%d], R=%d",cx,cy,rf);
fichier = fopen("pfe_result.dat", "w");
if (fichier != NULL)
{ fprintf(fichier,"%d %d %d",cx,cy,rf);
fclose(fichier);
return 0; }

```

A.7 Code Matlab pour la communication

```

Pour l'émission :
% Code MATLAB pour l'émission série
% Initialisation
clear all

```

```
close all
clc
instrreset
% Créer l'interface pour l'objet série
s21 = serial('COM8');
% Réglage des propriétés de communication
set(s21, 'BaudRate', 2400);
set(s21, 'FlowControl', 'hardware');
set(s21, 'Terminator', '3');
set(s21, 'Parity', 'none'); %pair
% Envoi des données via le port série
fopen(s21)
fwrite(s21,DATA,'uint8') % Donnée sur 8bits
%%%%%%%%%%%%
% Code MATLAB pour la réception série synchrone
set(s21, 'BaudRate', 2400);
set(s21, 'FlowControl', 'hardware');
set(s21, 'Terminator', '3');
set(s21, 'Parity', 'none'); %pair
% s21.ReadAsyncMode='continuous';
s21.BytesAvailableFcnCount = 1;
s21.BytesAvailableFcnMode = 'byte';
%s22.BytesAvailableFcnMode = 'terminator';
time = datestr(now,0);
s21.BytesAvailableFcn = {@my_BytesAvailable,time};
%open the interface
fopen(s21)
s21.BytesAvailable
fclose (s21);
% % % %
function rs232=my_BytesAvailable(obj, event, time)
%display('BytesAvailable event occurred for the object : Serial-COM8')
rs232 = fread(obj,obj.BytesAvailable,'uint8')
```

```
RS232_FPGA=rs232
```

```
end
```

```
%%%%%%%%%
```