Université Mohamed V
Ecole Nationale Supérieure d'Informatique et d'Analyse des Systèmes - RABAT

Rapport du projet de l'ingénieurie des systèmes numériques

Conception FPGA d'un réseau de neurones pour la détection des couleurs

Filière : Ingénierie des Systèmes Embarqués et Mobiles

Réalisé par :

AIT JA Abdellilah
BENHAMMOU Nouhayla
FTERI MGHARI Khadija

Encadré par :

M.ALAOUI ISMAILI Zine Abidine

Année universitaire 2021/2022

Remerciements

En premier lieu, nous tenons à remercier notre professeur M.Alaoui Ismaili Zine Abidine pour son encadrement, pour sa disponibilité et pour la confiance qu'il nous a accordé pour élaborer ce projet librement.

Nous souhaitons ensuite adresser nos remerciements à toute personne ayant contribué de près ou de loin a l'acheminement de ce travail par toute intervention et toute remarque.

Résumé

Le présent projet avait comme but de réaliser un système fpga intelligent permettant à son utilisateur de detecter les couleurs dans une image donnée tout en garantissant la classification exacte des couleurs dans chaque image donnée au système.

Pour réaliser ce projet on a commencé par la construction d'un modèle d'apprentissage automatique basé sur un réseau de neurones capable de décortiquer une image et extraire la couleur désirée.Par la suite

La phase de l'entrainnement du modèle intelligent a été effectuée à l'aide de python et ses bibliothèques. Tandis que le codage de la partie hardware a été réalisé grace au langage vhdl et le logiciel xilinx. Pour simuler les résultats nous avons exploiter Octave et une carte graphique.

Ce projet a abouti alors à avoir un système fpga bien accompli basé sur le machine learning pour détecter une couleur donnée dans une image.

Mot clés : **FPGA, VHDL, Apprentissage automatique, Réseau de neurone, Détection de couleur.**

Abstract

The aim of this project was to create an intelligent fpga system allowing its user to detect the colors in a given image while guaranteeing the exact classification of the colors in each image given to the system.

To carry out this project, we started by building an automatic learning model based on a neural network capable of dissecting an image and extracting the desired color. Subsequently

The smart model training phase was performed using python and its libraries. While the coding of the hardware part was carried out thanks to the vhdl language and the xilinx software. To simulate the results we used Octave and a graphics card.

This project then resulted in having a well accomplished fpga system based on machine learning to detect a given color in an image.

Keywords : **FPGA, VHDL, Machine Learning, Neural Network, Color Detection.**

Table des matières

Table des figures

Introduction générale

L'apprentissage automatique peut être considérée comme un puissant vecteur de changement et de croissance pour plusieurs secteurs d'activités. L'ingénierie des systèmes numériques, cependant, n'a que très peu évolué. Mal préparées pour cette évolution numérique, les acteurs de ce secteur,(les grandes entreprises, investisseurs, ingénieures en systèmes numériques et systèmes embarqués) doivent prendre acte de la situation et adapter leurs modèles d'affaires et leurs anciens modèles de travail s'ils veulent relever le défi.

Le présent travail vise alors la réalisation d'un circuit FPGA qui facilitera la tâche et détecter une couleur donnée sur image. Ce système donnera donc la possibilité d'exploiter les propriétés du machine learning dans une application de detection de couleur implémentée dans un hardware.

Ce rapport est donc élaboré pour montrer les importantes étapes surpassées pour réaliser ce projet, il est donc structuré comme suit :

• Le chapitre 1 intitulé **" Contexte générale du projet "** donnera une vue général sur le module dans lequel s'inscrit ce projet, sa problématique , ses objectifs et son ordonnancement .

• Le chapitre 2 intitulé **" Analyse et conception "** est consacré à l'étude effectuée et la démarche adoptée dans les principales tâhes : le réseau de neurones, la création du modèle du Machine Learning et la programmation FPGA.

• Le chapitre 3 nommé **"Réalisation"** expose dans une première section les outils utilisés dans l'élaboration de ce projet tandis que la deuxième section présentera les differentes étapes de l'implémentation du modèle intelligent dans un circuit FPGA.

• Dans la **conclusion** on va mettre le point sur l'apport de notre projet, ses limites et les perspectives qu'il peut engendrer.

Chapitre 1

Contexte général du projet

1.1 Introduction

Ce chapitre a pour but de contextualiser le projet. Nous allons, tout d'abord, présenter le module pour lequel nous avons effectuer le projet. Ensuite nous allons présenter les objectifs de ce dernier, sa problématique et son ordonnancement.

1.2 Présentation de l'ingénierie des systèmes numériques

1.2.1 Présentation du module ingénierie des systèmes numériques

Les Systèmes numériques développe et exploite des applications et des systèmes informatiques organisés ou non en réseau destinés aux procédés de productions de biens d'équipement et de services techniques.

Cett science des composants complexes intervient sur la partie électronique des systèmes techniques quelle que soit leur technologie dominante. Son domaine de compétence est centré sur le signal, l'acquisition, le traitement analogique, le traitement numérique et son exploitation.

La conception des systèmes FPGA est une discipline importante dans cette spécialité et permet la réalisation de circuits intégrés complexes.

1.2.2 Image illustartive

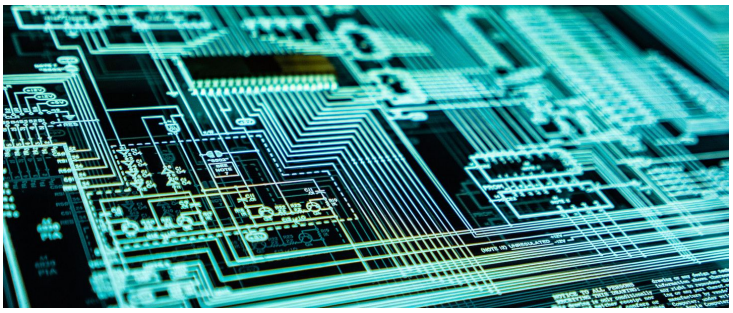

FIGURE 1.1 – Zoom sur un mircoprocesseur

1.3 Présentation du sujet

Avec l'avènement du digital et les petites et grandes révolutions que cela a induit dans les entreprises du matériel éléctronique, les acteurs du secteur des systèmes numériques étaient aussi obligés de s'adapter à la révolution des circuits éléctroniques qu'ils produisent.

Dans ce sens, notre projet consiste à réaliser système de detection de couleur intelligent basé sur un réseau de neurones bien entrainné capable de classifier les couleur dans une image. Grâce à ce modèle intégré dans un circuit FPGA nous aurons un produit avancé , intelligent et efficace dans la tache qu'il réalise.

1.4 Problématique

Les systèmes numériques évoluent de plus en plus. Néomoins le défi est relevé pour les ingénieurs dans ce domaine de pouvoir associer l'intelligence artificielle et ses applications dans les circuits numériques.L'implémentation des modèles intelligents et des algorithmes complexes pour faciliter quelques fonctionnalités d'un hardware(FPGA intégé) de detection de couleur serait alors le défi de notre projet.

1.5 Objectif du projet

L'objectif principal de ce projet est de concevoire et réaliser un circuit FPGA basé sur un modèle d'apprentissage automatique, qui visera à bien detecter les couleurs selectionnées dans n'importe quel image. Le défit s'étend aussi à la bonne implémentation du modèle dans le FPGA tout en assurant une bonne programmation vhdl.

1.6 Planification du stage

Afin de réaliser le projet dans les délais établis, nous avons utilisé le diagramme de GANTT puisqu'il est considéré comme l'un des outils les plus efficaces pour représenter l'état d'avancement des différentes tâches . Comme le montre la figure ci-dessous, nous avons adopté le modèle en cascade pour réaliser notre projet. Ce dernier représente une organisation des activités sous forme de phases linéaires et séquentielles, où chaque phase correspond à une spécialisation des tâches et dépend des résultats de la phase précédente.

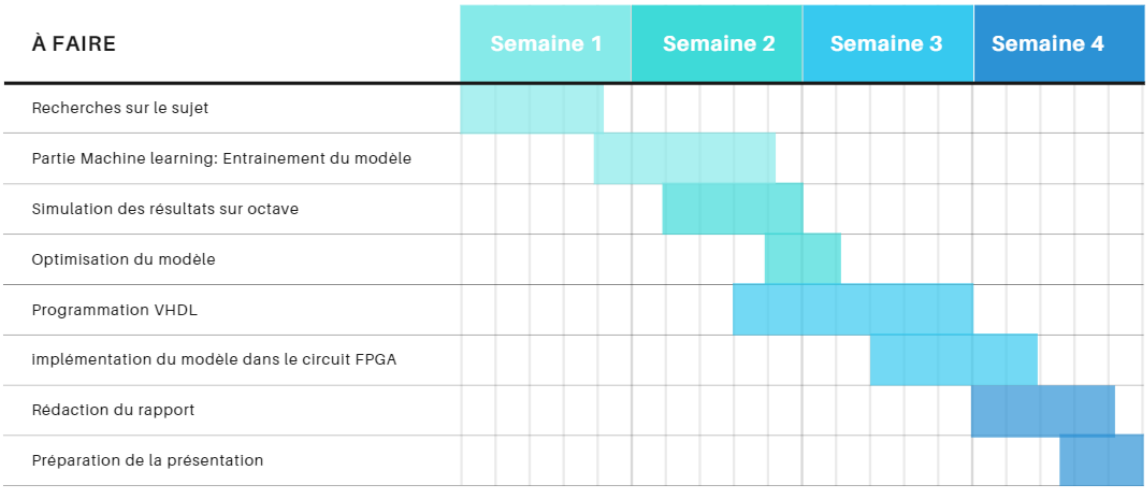

FIGURE 1.2 – Diagramme de GANTT

1.7 Conclusion

Dans ce chapitre nous avons abordé différentes parties. Premièrement, nous avons présenté le module dans lequel s'inscrit ce projet. Ensuite, nous avons expliqué la problématique et précisé l'objectif du projet et la planification suivi. Le deuxième chapitre sera dédié à l'analyse et conception.

Chapitre 2

Analyse et conception

2.1 Introduction

Dans ce chapitre nous allons dresser les fonctionnalités de notre système, en un premier lieu. Puis, nous allons entamer la phase de conception en presentant les différents diagrammes. Ensuite, nous détaillerons la démarche suivie pour la réalisation du projet.

2.2 Analyse des besoins

L'analyse et la spécification des besoins représentent la première phase du cycle de développement d'un logiciel.

2.2.1 Besoin non-fonctionnel

Les besoins non-fonctionnels décrivent toutes les contraintes auxquelles est soumis le système pour sa réalisation et son bon fonctionnement.

— La performance : le système doit être avant tout performant, c'est à-dire àtravers ses fonctionnalités, il doit répondre à toutes les exigences des utilisateurs d'une manière optimale.
— La convivialité :le système doit être facile à utiliser. En effet, il doit être conviviales, c'est-à-dire simple.
— Rapidité : La système doit optimiser les traitements pour avoir un temps de réponse réal.
— Maintenabilité et scalabilité : Le code doit être lisible et compréhensible afin d'assurer son état évolutif et extensible par rapport aux besoins du marché.

2.2.2 Besoin fonctionnel

Les services proposés par notre plateforme se résument en six actions majeures :

— Sélectionner les couleurs : Le système doit être capable de sélectionner une grande plage de couleurs differents.

— Détecter les couleurs : Le système doit être capable de distinguer les couleurs sélectionnées.
— Classifier les couleurs : Le système doit determiner la classe de la couleur détectée.
— Reconnaitre les couleurs : Après avoir classifier la couleur le système doit la reconnaître en se basant d'une base de données d'apprentissage.

2.3 Conception et modélisation

2.3.1 Diagramme de conception du système

Le système comprend principalement quatre parties : le module d'acquisition d'images, le module de mise en cache des données, le module central FPGA et le module d'affichage VGA. Le schéma fonctionnel de la conception globale du système est présenté à la figure ci dessous :

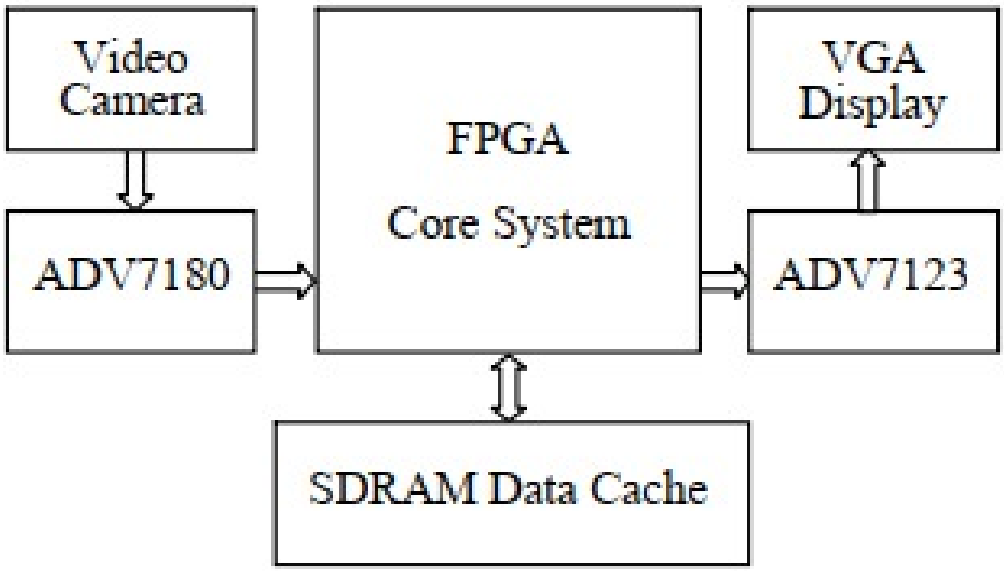

Figure 2.1 – DIAGRAMME DE LA STRUCTURE DU SYSTÈME

2.3.2 Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation est classé aussi avec les diagrammes fonctionnels. Il permet de représenter les fonctions d'un système du point de vue de l'utilisateur. Il montre en fait la relation entre un acteur et ses demandes ou attentes vis-à-vis du système, sans décrire les actions en cours ni les mettre dans un ordre logique.

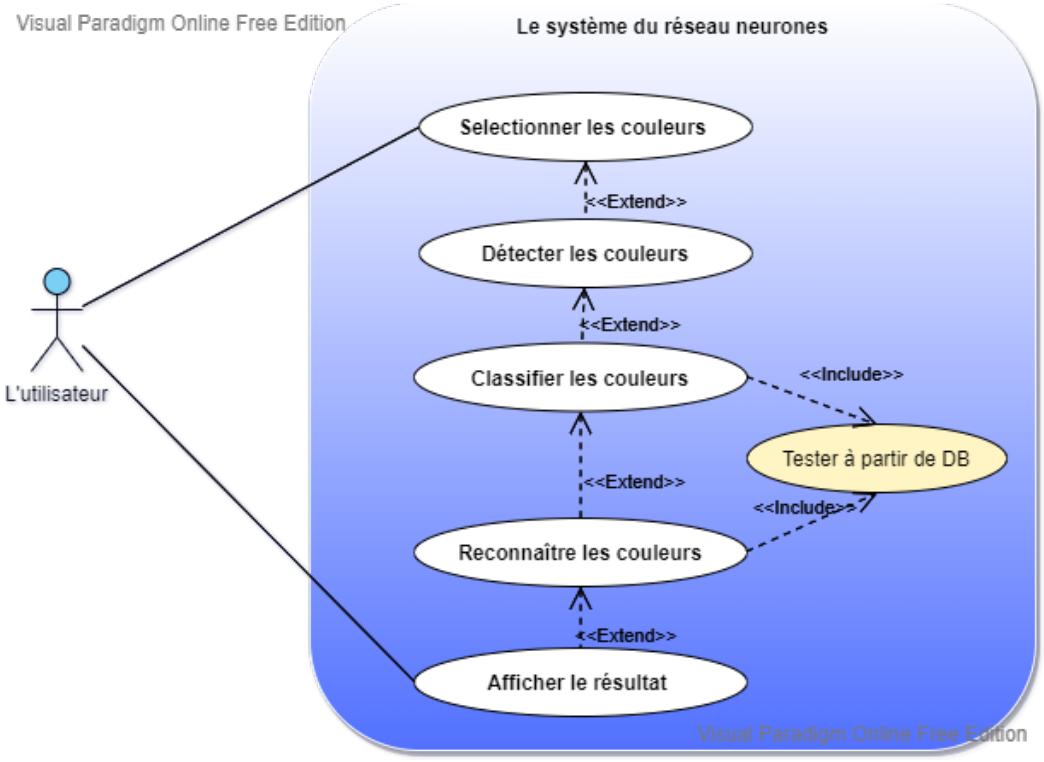

FIGURE 2.2 – Diagramme de cas d'utilisation

2.3.3 Diagramme de séquences

Ce diagramme est classé avec les diagramme dynamique, il permet de représenter temporellement les interactions entre objets.
Prenant l'exemple suivant :

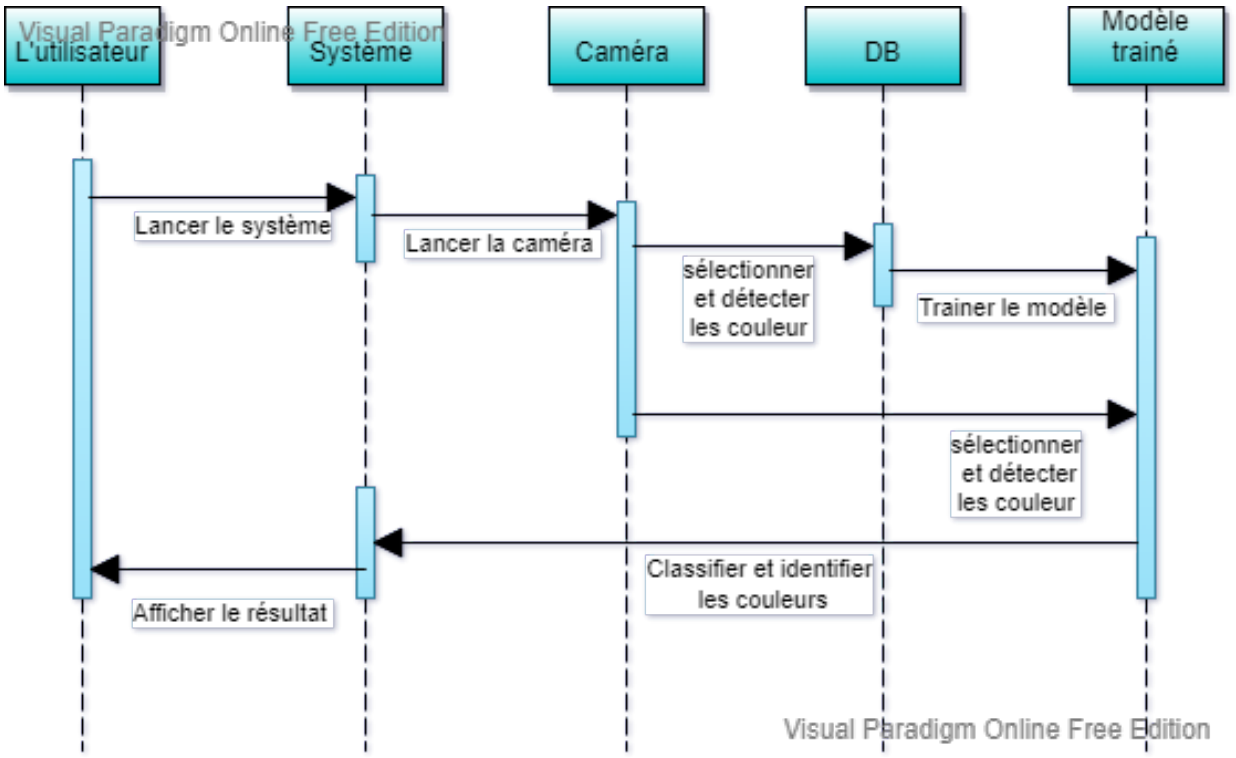

FIGURE 2.3 – Diagramme de séquences

2.4 Conclusion

Ce chapitre a été consacré pour effectuer l'analyse des besoins, la conception et préciser la démarche adoptée lors de la concrétisation du projet. Le troisième chapitre est dédié à la réalisation. Il montrera les outils utilisés, expliquera les algorithmes implémentés et présentera les pages du site web réalisé.

Chapitre 3

Réalisation

3.1 Introduction

Dans ce chapitre nous présenterons les diffèrents outils utilisés dans la réalisation du projet. Puis nous allons expliquer les algorithmes implémentés. Ensuite, nous présenterons les differentes exemples de simulation.

3.2 Outils utilisés

3.2.1 outils de la partie hardware

La partie hardware a été développée avec Xilinx et son ISE Design suit :

FIGURE 3.1 – Logos du principale outil utilisés en Hardware

— ISE est un outil logiciel abandonné de Xilinx pour la synthèse et l'analyse de conceptions HDL, qui cible principalement le développement de micrologiciels embarqués pour les familles de produits de circuits intégrés Xilinx FPGA.[?]

3.2.2 Outils Software

— Python : Python est un langage de programmation interprété, multi-paradigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet.[?]

FIGURE 3.2 – Logo du langage Python

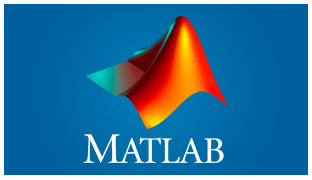

FIGURE 3.3 – Logo du langage de simulation MATLAB

Python est connu par ses bibliothèques riches en fonctionnalités. Dans notre projet,nous nous sommes appuyées sur 4 bibliothèques principales : Pandas, Scikit Learn, Pillow, datetime et Keras.
Matlab quant à lui il est trés utile pour les simulations que nous allons exploiter par la suite.

FIGURE 3.4 – Logos des principales bibliothèques de Python utilisées

— Datetime : Datetime est un module qui est conçu avec une programmation orientée objet pour fonctionner avec la date et l'heure de Python.
newline
— Scikit Learn : Scikit-learn est le principal package de machine learning en python, il possède des dizaines de modèles dont la régression logistique. En tant que package de machine learning, il se concentre avant tout sur l'aspect prédictif du modèle de régression logistique, il permettra de prédire très facilement mais sera pauvre sur l'explication et l'interprétation du modèle. Par contre, pour la validation de la qualité prédictive des modèles, l'ajustement des hyper-paramètres et le passage en production de modèles, il est extrêmement efficace.
newline
— Pillow : Pillow est une bibliothèque basée sur la Python Imaging Library (PIL). PIL est une bibliothèque qui offre plusieurs procédures standards pour la manipulation d'images.
newline
— Keras : C'est une bibliothèque open source qui permet notamment de conce-

voir des services web et développer de nouveaux systèmes d'intelligence artificielle (IA).

3.2.3 GITHUB

GitHub est un service web d'hébergement et de gestion de développement de logiciels, utilisant le logiciel de gestion de versions Git. Le site assure également un contrôle d'accès et des fonctionnalités destinées à la collaboration comme le suivi des bugs, les demandes de fonctionnalités, la gestion de tâches et un wiki pour chaque projet.

FIGURE 3.5 – Logo de la plateforme GITHUB

3.3 Démarche du projet

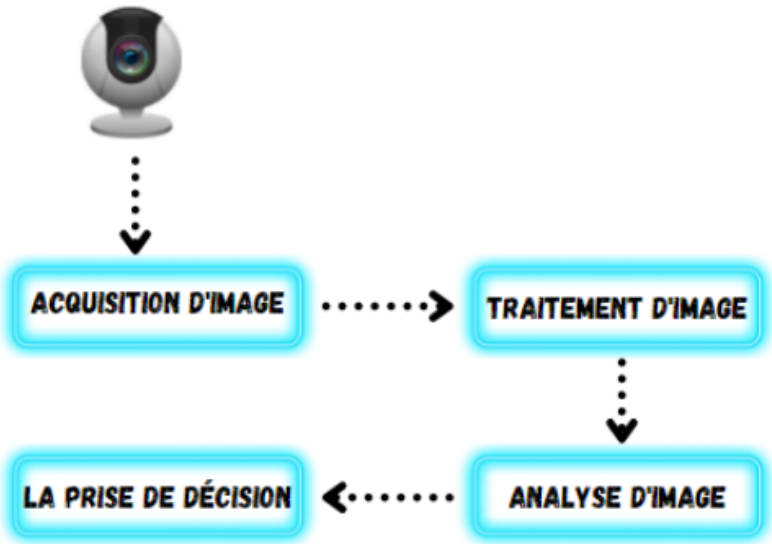

FIGURE 3.6 – le schéma fonctionnel

Dans cette section, nous allons présenter les différentes étapes suivies dans la conception de notre projet.

3.3.1 Acquisition d'Image

L'image ou le cadre "Frame" détecté par la webcam est acquis en tant que représentation numérique des caractéristiques visuelles du monde physique. La webcam est utilisé pour détecter et capturer les informations nécessaires à la création d'une image.

3.3.2 Traitement d'Image

Les images acquises sont ensuite traitées pour être stockées dans la base de données ou comparées par ceux qui existent dedans. Les signaux des images acquises sont filtrés pour éliminer le bruit ou toutes les fréquences irrévérencieuses, afin de les préparer pour le analyse.

3.3.3 Analyse d'Image

L'image traitée est analysée pour extraire des informations utiles. Cette étape implique de nombreuses propriétés d'image importantes comme l'identification des cordonnées, la comparaison avec la base de données, la reconnaissance des couleurs, l'extraction des caractéristiques, le suivi de mouvement et la segmentation d'images.

3.3.4 La Prise de Décision

Les données obtenues à partir de toutes les étapes ci-dessus sont utilisées pour produire des informations numériques significatives, qui conduit à prendre des décisions.

3.4 Partie FPGA

3.4.1 Présentation de la carte Spartan 3E

Le système utilise le FPGA Spartan 3E (XC3S1200E ou XC3S500E) de Xilinx. Cette carte offre un environnement de conception très adapté pour le prototypage d'applications variées dont celles des systèmes numériques à usage général et des systèmes embarqués. Cette carte est de plus idéale pour les applications de traitement vidéo et de traitement de signal en général. La carte Nexys2 regroupe entre autre un FPGA XC3S500E Spartan 3E, un accès USB2 pour la configuration et le transfert de données rapide, une mémoire externe RAM de 128Mbit et une mémoire externe ROM de 128Mbit, un port PS/2, un port VGA, un port RS-232, un port de configuration USB, deux convertisseurs de données, un oscillateur à 50 MHz, un connecteur d'expansion Hirose FX2 permettant 60 entrées/sorties génériques, 8 Led, 4 boutons poussoir et 8 commutateurs (Switch).

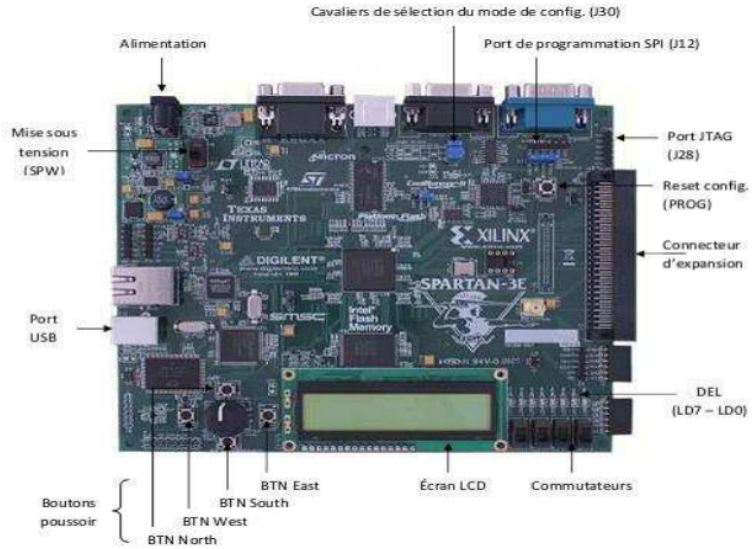

FIGURE 3.7 – Zoom sur spartan 3E

3.4.2 Circuit FPGA

Field Programmable Gate Array (FPGA) : circuit programmable composé d'un réseau de blocs logiques, de cellules d'entrée-sortie et de ressources d'interconnexion totalement flexibles, ce circuit, qui nécessite un outil de placement-routage,

est caractérisé par son architecture, sa technologie de programmation et les éléments de base de ses blocs logiques.

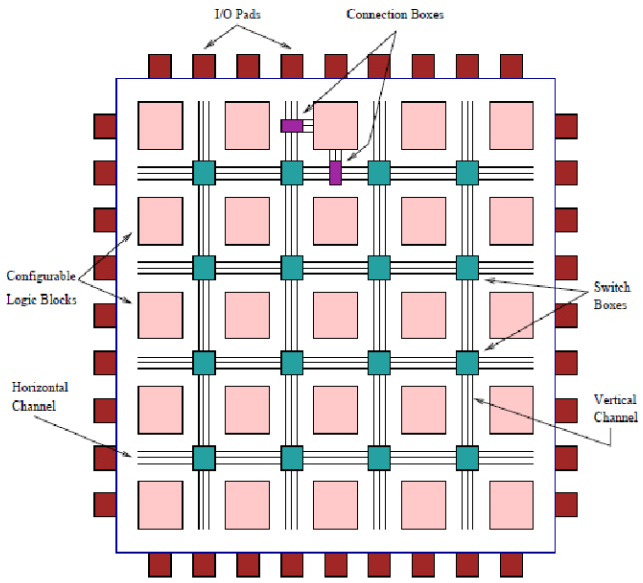

FIGURE 3.8 – schéma de l'architecture d'une FPGA

Le FPGA est une version plus évoluée et offrant plus de flexibilité que les PLD. Ainsi souvent utilisé pour créer un circuit intégré prototype dans la phase préliminaire du développement d'un ASIC, et ceci en raison de sa facilité de programmation ce qui nous fait gagner le temps de développement. Les FPGA sont composés des cellules programmables, les IOB et les CLB. Les IOB (Input Output Blocs) servent comme tampons aux entrées/sorties du FPGA. Les CLB (Control Logic Blocs) sont des cellules programmables qui réalisent les fonctions combinatoires et séquentielles de notre architecture à implanter. La carte supporte trois modes de configuration à la mise sous tension : `[U+F0B7]` Mode Master-Slave utilisant le PROM 4 Mbit, `[U+F0B7]` Un mode SPI utilisant la mémoire Flash sériel `[U+F0B7]` Un mode BPI utilisant la mémoire Flash parallèle. La carte peut aussi être programmée directement à l'aide d'un port JTAG lorsqu'elle est sous tension.

3.4.3 Structure d'une image et la pixellisation

Tout d'abord, l'image est convertie en son fichier de coefficients (.COE) en utilisant python. Au cours de cette étape, le codage de couleurs vraies 24 bits est changé en couleur 8 bits. Le fichier de coefficients de l'image est ensuite chargé dans la RAM de bloc sur le FPGA dans des emplacements de mémoire continue. Un seul port BRAM est utilisé à cette fin.

Nous avons un seul module appelé "Color Detection" qui traite les données du BRAM et applique notre propre algorithme pour classer les cercles en fonction

de leur couleur ainsi que de leur taille. Le module a un seul bloc qui maintient toujours une liste de sensibilité de toutes les variables changeantes comme le nombre de couleurs, la taille des cercles de couleurs différentes, etc. Il existe des variables d'indicateur pour déplacer le flux de contrôle de la détection du point le plus haut pour calculer le rayon de cercles pour effacer le cercle par la suite.

Nous utilisons des instructions de cas pour classer les cercles en fonction de la couleur 8 bits.

Voici les détails de l'algorithme que ce module implémente : Pour compter les cercles, les données d'image sont parcourues ligne par ligne pour trouver le premier pixel qui a une couleur différente de la couleur d'arrière-plan de l'image. Ce pixel est le point le plus haut d'un cercle. Le nombre de cette couleur est augmenté de un.

Pour la détection du rayon, nous descendons d'un pixel un pixel à droite à chaque étape jusqu'à ce que nous obtenions un pixel différent de la couleur du pixel le plus haut du cercle.

La distance parcourue vers le bas à partir du sommet est considérée comme le rayon du cercle.

Le décompte de la taille appropriée pour la couleur est augmenté de un.

Maintenant que nous avons détecté le cercle et que nous ne voulons plus le détecter. Désormais, nous revenons au pixel le plus haut et commençons à colorer le carré dont le milieu du bord supérieur est le point le plus haut du cercle avec la couleur d'arrière-plan.

Des tolérances appropriées sont données car l'image pixélisée n'a pas un sommet net d'un seul pixel.

Ce processus se poursuit jusqu'à ce que le pointeur de déplacement atteigne le dernier pixel de l'image.

3.5 Réseau de neurones pour détection de couleurs

3.5.1 Principe de fonctionnement

Le mode de fonctionnement d'un réseau de neurone est à première vue simple : l'utilisateur fournit en entrée une image sous la forme d'une matrice de pixels.Celle-ci dispose de 3 dimensions :Deux dimensions pour une image en niveaux de gris.Une troisième dimension, de profondeur 3 pour représenter les couleurs fondamentales (Rouge, Vert, Bleu).

L'architecture du Convolutional Neural Network dispose en amont d'une partie convolutive et comporte par conséquent deux parties bien distinctes :

Une partie convolutive : La convolution est une opération mathématique

simple généralement utilisée pour le traitement et la reconnaissance d'images. Sur une image, son effet s'assimile à un filtrage.Son objectif final est d'extraire des caractéristiques propres à chaque image en les compressant de façon à réduire leur taille initiale. En résumé, l'image fournie en entrée passe à travers une succession de filtres, créant par la même occasion de nouvelles images appelées cartes de convolutions. Enfin, les cartes de convolutions obtenues sont concaténées dans un vecteur de caractéristiques appelé code CNN.

.

- Dans un premier temps, on définit la taille de la fenêtre de filtre située en haut à gauche.
- La fenêtre de filtre, représentant la feature, se déplace progressivement de la gauche vers la droite d'un certain nombre de cases défini au préalable (le pas) jusqu'à arriver au bout de l'image.
- À chaque portion d'image rencontrée, un calcul de convolution s'effectue permettant d'obtenir en sortie une carte d'activation ou feature map qui indique où est localisées les features dans l'image : plus la feature map est élevée, plus la portion de l'image balayée ressemble à la feature.

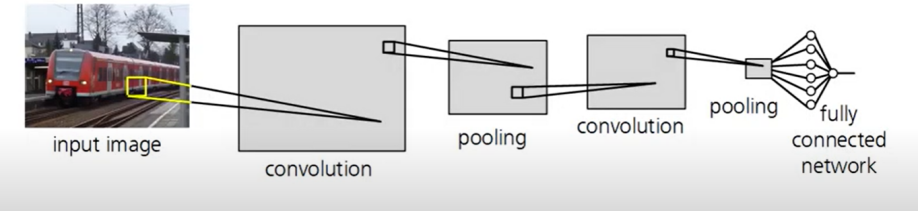

FIGURE 3.9

Une partie classification : Le code CNN obtenu en sortie de la partie convolutive est fourni en entrée dans une deuxième partie, constituée de couches entièrement connectées appelées perceptron multicouche (MLP pour Multi Layers Perceptron).Le rôle de cette partie est de combiner les caractéristiques du code CNN afin de classer l'image. Pour revenir sur cette partie, n'hésitez pas à consulter l'article sur le sujet .

3.5.2 Détection de couleurs

Dans cette partie nous allons voir en détail comment nous avons construit notre engin de détection intelligent.

- Définir l'image :Nous avons choisi une image précise. Notre image est enregistré dans le même dossier que notre programme , ce qui facilite la recherche et l'importation.

— Reconnaissance des couleurs :Par principe de pixelisation , nous avons pu délimiter les bordures des différentes couleurs dans notre image choisie.
— Enseigner les couleurs :Tout d'abord, nous devons leur apprendre les couleurs. Pour ce faire, nous avons besoin de données comprenant des noms de couleurs et des valeurs correspondant à ces couleurs. Étant donné que la plupart des couleurs peuvent être définies en utilisant le rouge, le vert et le bleu. C'est 3 couleurs seront les input de notre réseau de neurones.Les différents layers de combinasons seront traités par les 3 input pour donner comme output la bonne détéction des couleurs.Ce process s'inscrit sous la technique Perceptron multicouche, qui est une architecture prédictive des ANN, ayant une couche d'entrée (non neuronale), des couches cachées et une couche de sortie. Ce réseau est entraîné par algorithme de rétropropagation, réalisant un apprentissage supervisé (apprentissage par exemples).

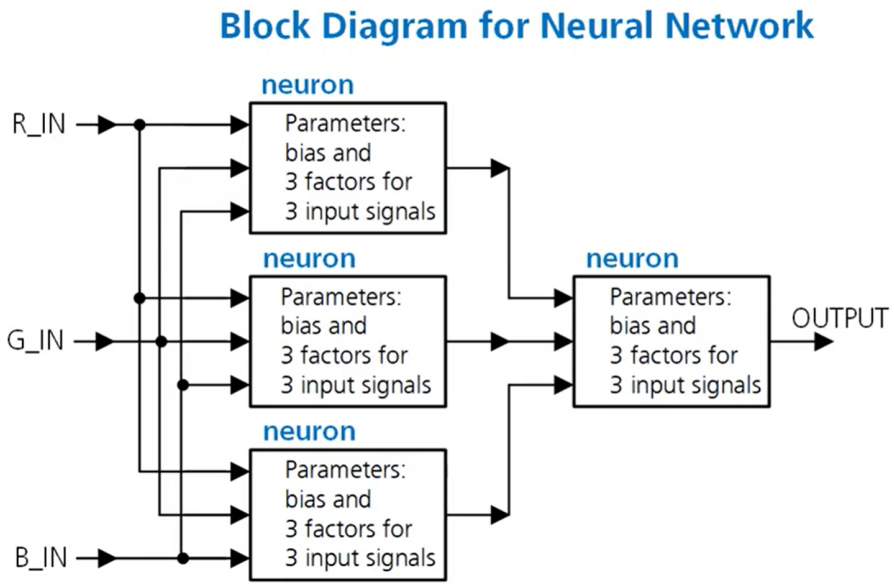

Figure 3.10

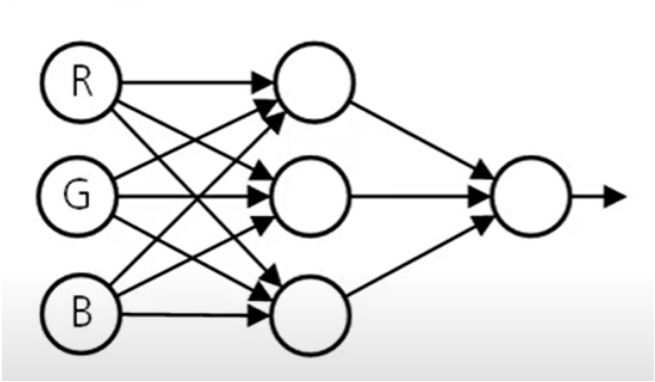

Figure 3.11

Training #	Epochs	Accuracy
1	1565	67%
2	**2353**	**95%**
3	3315	80%
4	4239	93%
5	680	40%

FIGURE 3.12

3.6 Implémentation du modèle IA dans le circuit FPGA

3.6.1 Résultats de simulation

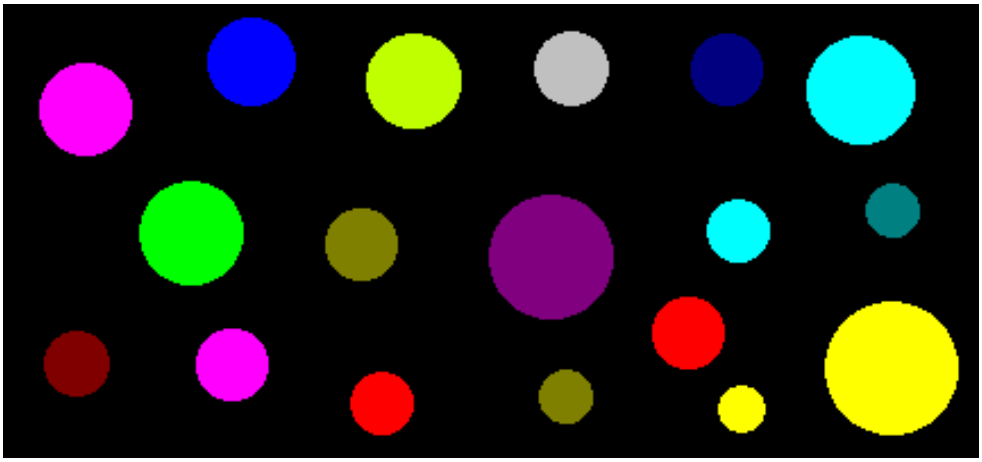

FIGURE 3.13 – Input pour tester la couleur noire

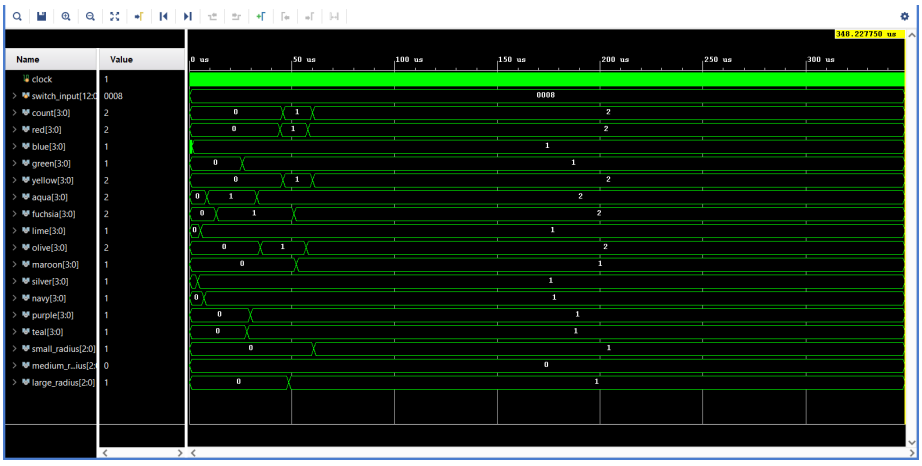

FIGURE 3.14 – Output de simulation

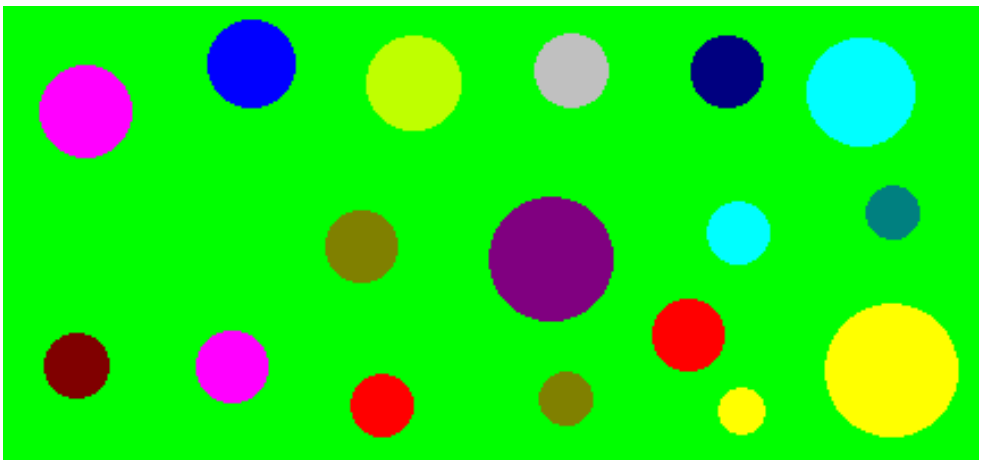

FIGURE 3.15 – Input pour tester la couleur verte

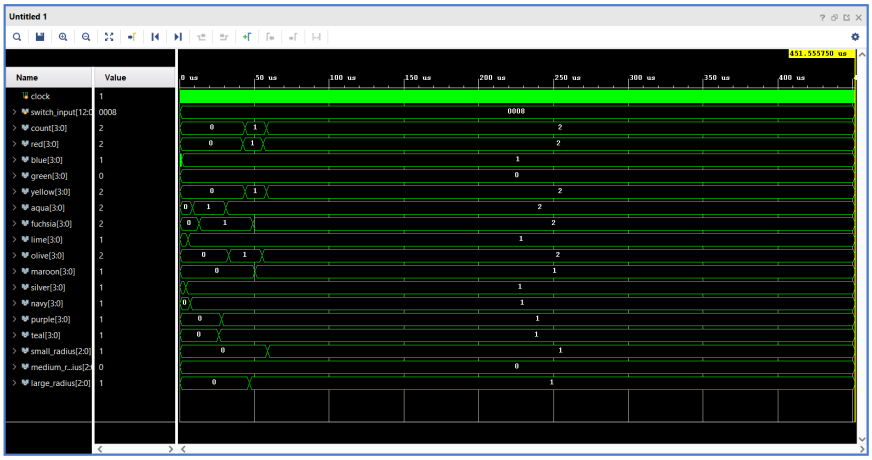

FIGURE 3.16 – Output de simulation

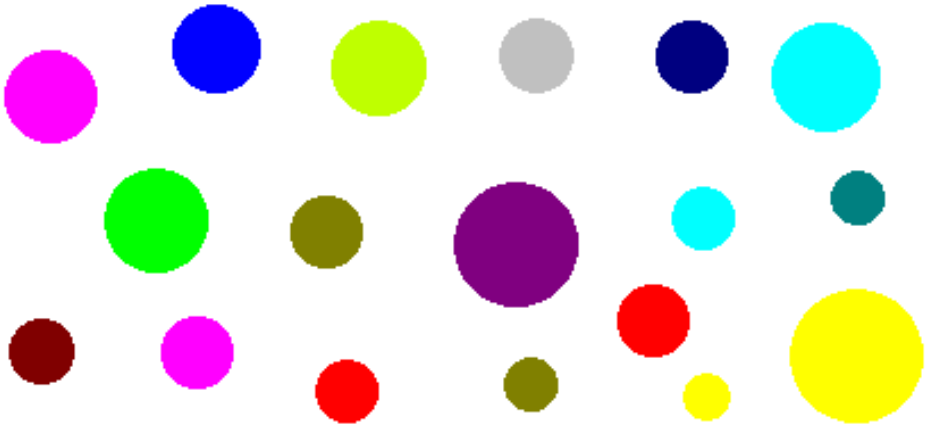

FIGURE 3.17 – Input pour tester la couleur blanche

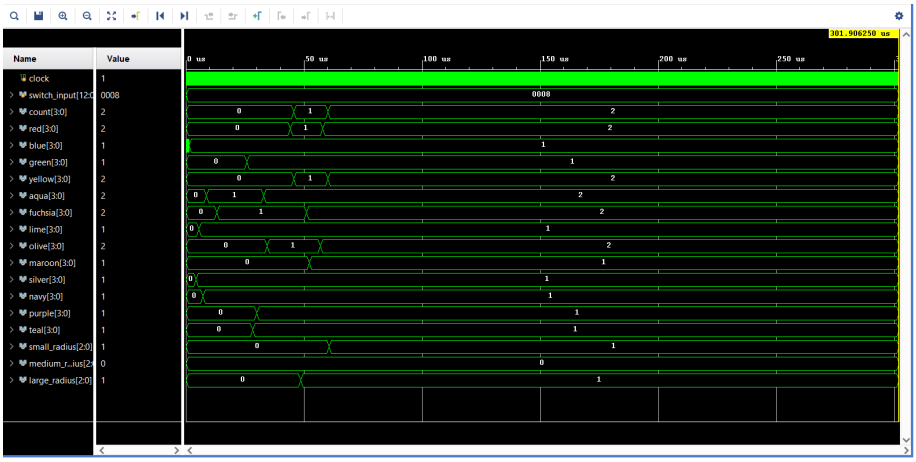

FIGURE 3.18 – Output de simulation

FIGURE 3.19 – Input pour tester la couleur rouge

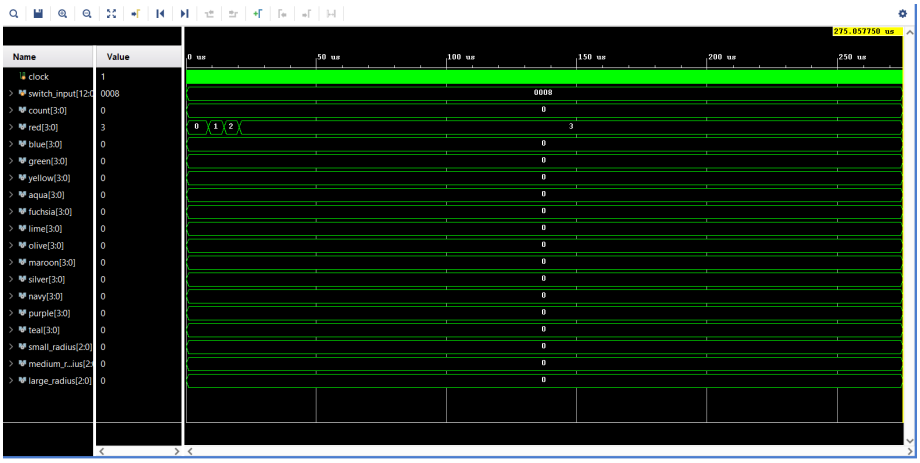

FIGURE 3.20 – Output de simulation

3.7 Conclusion

Nous avons abordé dans ce dernier chapitre trois parties. La première partie s'agit d'une présentation des outils utilisés lors de la réalisation du projet, la

deuxième explique les algorithmes implémentés et la troisième a été consacré à la démonstration des différentes pages web de notre site web.

Conclusion générale

En définitive, ce projet a été une très bonne introduction à notre domaine d'études, et une mise en pratique de nos connaissances acquises durant les séances de systèmes numériques.

À terme de ce projet, nous avons pu donc réaliser un système intelligent capable de détecter les couleurs dans une image. Ce système fpga basé sur l'apprentissage automatique est d'une grande efficacité et performance.

Ce projet fait aussi appel à un véritable travail de réflexion sur la manière de le concevoir pour remplir à bien les besoins d'un projet de systèmes numériques. Cela nous a permis de comprendre les difficultés d'un projet entre les choses que nous voulions réaliser et les contraintes que nous devrions surmonter.

Bibliographie

[1] B. Balland. Optique géométrique : imagerie et instruments, 1e éd., lausanne,. *Presses polytechniques et universitaires romandes*, 2007.

[2] Commission Internationale de l'Éclairage. Vocabulaire international de l'éclairage.
`http://www.standardsstore.ca.`, 1987.

[3] Foveon. Direct image sensors., 2001-2021Consulté le 6 août 2009.

[4] R. Lukac. Single-sensor imaging : methods and applications for digital cameras.
`http://www.crcnetbase.com/isbn/978-1-4200-5452-1.`, 2009.

[5] D. Metz. Comprendre la couleur et ses profils.
`http://www.profil-couleur.com.`, Consulté le 25 mars 2009.

[6] National Institute of Standards and Technology. Codata internationally recommended values of the fundamental physical constants.
`http://physics.nist.gov/cuu/Constants/`, 2008.