ECOLE ROYALE MILITAIRE
BRUXELLES
ROYAUME DE LA BELGIQUE

ACADEMIE MILITAIRE
FONDOK JDID
REPUBLIQUE TUNISIENNE

Travail de fin d'études

Implémentation d'une méthode de détection et suivi de visage en temps réel

Par le sous-Lieutenant
ZRELLI Mohamed

Sous la direction du Dr. Ir. G. DE CUBBER

Bruxelles, 21 Juin

Table des matières

Table des figures

Remerciements

Ce travail de fin d'étude s'est déroulé au sein de l'Ecole Royale Militaire Belge.

Au nom de ce travail, je profite de cette occasion de présentation du projet pour exprimer mes profonds respects, mes sincères remerciements et toutes mes gratitudes à mes encadrant : Mr **BAUDOIN Yvan** professeur à l'ERM, MR **DE CUBER Greet** et Mr **BERRABAH Sid Ahmed** chercheurs à l'ERM qui ils m'ont guidé avec grande résignation tout au long de la préparation de ce travail. Merci pour tous leurs conseils, leurs persévérances et leurs bienveillances.

Je remercie vivement le commandant **COLON Eric**, Melle **VERBIEST Kristel** et Mme **DOROFTEI Daniela** pour leurs aides.

Comme je suis profondément reconnaissant à mes professeurs en Tunisie qui ont participés à ma formation, et surtout Mme **LENGLIZ Ilhem** chef de département informatique à l'Académie Militaire tunisienne qui a suivie l'avancement de mon travail.

Je remercie sincèrement mes parents, qui m'ont soutenu avec leurs encouragements durant toute la durée de mon projet en Belgique.

Mes remerciements aussi aux membres de jury pour l'honneur qu'ils ont fait de bien vouloir analyser ce travail et apporter leurs critiques et suggestions.

Mohamed.

Glossaire

OpenCV : Open Computer Vision
USB : Universal Serial Bus
RS : Recommended Standard
PTU : Pant-Tilt- Unit
IPP : Integrated Performance Primitive
PCA : Principal Composant Analyse
PMC : Perceptron Muli-Couches
RVB : Rouge Vert Bleu
VS : Visual Studio
HT : Hyper Terminal
ASCII : American Standard Code for Information
CV : Computer Vision
MinGW : Minimalist GNU for Windows
RGB : Red Green Blue
CAPI : Custom Application Programming Interface
FIFO : First In First Out

1. Introduction

1.1 Asservissement visuel : (Visual Servoing)

Dans les dernières années les chercheurs ont montré leurs volontés de reproduire les capacités humaines de perception et d'action dans des systèmes ce qui a mené à l'intégration des données issues de capture extéroceptifs, et plus particulièrement de celle issues d'une caméra. D'un point de vue méthodologique, l'asservissement visuel consiste à intégrer directement dans la boucle de commande des robots des informations extraites des images fournies par des caméras afin de réaliser l'action souhaitée, ce ci est important car il permet d'élargir le domaine d'application de la robotique et qui donne une amélioration considérable de la précision obtenue.[1]

1.1.1 Applications de l'asservissement visuel [2]

Les techniques d'asservissement visuelles consistent à utiliser les informations fournies par une ou plusieurs caméras d'acquisition affin de contrôler les mouvements d'un système en robotique. On n'oublie pas encore que le domaine de l'asservissement visuel est porteur de nombreuses applications potentielles et en dehors da la robotique.

- **Robotique médicale :**

Les premières applications significatives réalisées portent sur l'aide au geste chirurgical par asservissement visuel en chirurgie robotisée. Dans ce cadre les instruments de chirurgie et l'endoscope sont tenus par des bras robotique esclaves.

Fig. 1.1 – Robot pour chirurgie médicale

- **Réalité virtuelle :**

Les techniques d'asservissement visuel s'appliquent assez directement au domaine de la réalité virtuelle car il est également possible de générer des asservissements visuels des mouvements spécialisés de type cinématographie pour la caméra virtuelle de restitution virtuelle, ou encore de contrôler ces mouvements en évitant les obstacles et les occultations tel que le cas de l'application classique de visite de musée virtuel.

FIG. 1.2 – visite virtuelle d'un musé

1.1.2 Les types d'asservissement [3]

- **Asservissement 3D :**

Le contrôle de déplacement du robot dans l'espace cartésien est le but de ce type. Dans ce cas la grandeur asservie correspond à l'altitude d'un repère lié rigidement à l'effecteur du robot par rapport à un autre repère attaché à l'objet d'intérêt (objet cible).

L'inconvénient de ce type est la nécessité d'une étape de reconstruction 3D permettant de fournir une mesure d'altitude, cette dernière peut par exemple être obtenue par triangulation en utilisant un système stéréoscopique calibré, ou par des techniques de reconstruction de pose en cas d'utilisation d'un capteur monoculaire. Il résulte après convergence de l'asservissement lorsque le régime permanant est atteint, un biais entre la projection du robot et l'objet d'intérêt à atteindre, ceci constitue le problème majeur de l'asservissement visuel 3D.

- **Asservissement 2D :**

La grandeur asservie est exprimée sous la forme des primitives visuelles dans l'image, dans ce cas on parle de " asservissement référencé image ". Contrairement à l'asservissement visuel 3D, le contrôle de robot n'est pas fait dans l'espace cartésien mais directement dans l'image car les primitives extraites de l'image sont généralement des formes géométriques élémentaires qui proviennent de la cible ce qui rend les coordonnés des points les plus couramment utilisées dans l'image, ces primitives peuvent également être des droites,

des ellipses, cylindres, des invariants projectifs... dans le cas des scènes complexes qui ne contiennent aucune forme géométrique simple, l'information du mouvement dans l'image ou l'information photométrique d'une région d'intérêt de l'image peuvent être utilisés comme primitives visuelles.[3]

- **Asservissement hybride :**

Comme son nom l'indique ce type d'asservissement consiste à faire la combinaison entre les deux types précédents 2D et 3D donc il utilise forcément 2 types de primitive et il est connu sous le nom de : Asservissement **2D1/2**

1.2 Détection de visage : (Face detection) :

Nous vivons actuellement dans l'ère de la technologie et nous essayons d'attribuer les facultés et les capacités humaines aux machines. L'avance scientifique est telle que l'intelligence artificielle est utilisée pour gérer d'une manière optimale des systèmes et des équipements complexes afin de les aider à prendre des décisions pertinentes. Pour parvenir à un tel résultat on doit passer par la détection des formes, des objets et des êtres humains dans leurs milieux naturels et réels.

L'image du visage est un avantage externe incontestable pour l'identification des individus autour de nous. L'être humain a des capacités naturelles pour reconnaitre et différencier les visages, car notre cerveau très évolué peut mémoriser un nombre important d'individus avec les différentes caractéristiques. Aujourd'hui la capacité du système de reconnaissance de l'homme est considéré comme la plus efficiente puisque l'identification est efficace sous différente circonstances : changements lumineux, angles de prise des vues différentes, porte des lunettes ou pas, avec ou sans barbe ou moustache, le style des cheveux et l'expression émotionnelle etc.

La détection de visage est une nouvelle technologie informatique qui détermine les endroits et les tailles des visages humains dans des images (numériques) arbitraires. Elle détecte les dispositifs faciaux et ignore toute autre chose, tel que des bâtiments, des arbres, des corps et tout dispositif autre que le visage etc.[4] Cette technologie est utilisée dans plusieurs domaines comme la biométrie pour l'identification et la reconnaissance faciale, elle est également employée dans les systèmes de surveillance visuelle ainsi que plusieurs appareils photo numériques récentes emploient la détection de visage pour la mise au point automatique.
Cette technologie tire son importance des différentes applications qui existent soit dans l'industrie vue son emploie soit dans la sécurité physique et même dans la vie courante.

1.3 Problématique

Les questions qui se posent en ce moment sont : Comment parvenir à détecter le visage d'une personne dans une image prise par une caméra et comment le suivre en temps réel s'il sort du champ de vision de dispositif d'acquisition ?

1.4 Le but de travail

Le but de ce travail de fin d'études est :
Etudier dans un premier lieu le principe de fonctionnement d'un algorithme de détection de visage en Matlab appelé " Algorithme de Yifan Shi ".

Dans un second lieu, faire l'implémentation d'un algorithme de détection de visage en utilisant OpenCV (Open Computer Vision) en C++ vue la rapidité d'exécution.
Dans un dernier lieu, Implémenter un algorithme permettant de commander un pan-tilt affin de pouvoir suivre le visage détecté dans ces mouvement et le gardé toujours dans le champ de vision de la loupe de la caméra utilisée.

1.5 Méthodologie de travail :

Avant toute chose on commence, au chapitre 2, par donner un aperçu sur le matériel dont on a besoin (Hard et Soft) pour la réalisation de ce travail.

L'étude d'un algorithme de détection de visage en Matlab ainsi que quelques méthodes de détection qui vont servir à la bonne compréhension de ce domaine et l'implémentation d'un programme en C++ utilisant OpenCV seront présentées au chapitre 3.

Nous traitons dans le chapitre 4 le contrôle du pan-tilt ainsi que l'explication du principe de la méthode de suivi et l'implémentation de l'approche de suivi de visage détecté.

Afin de bien interpréter le fonctionnement de ce programme un ensemble de testes sera exécuté pour connaitre les limites de cette méthode puis on clôture ce projet par une conclusion.

2. Le matériel utilisé

2.1 Introduction

Pour la réalisation de l'application " détection et suivi de visage" on a besoin d'avoir quelques logiciels ainsi que quelques appareils qui vont être utilisés soit pour la détection des visages soit pour leurs suivis, dans ce chapitre on se contente de donner une idée générale à propos de ce que on a utilisé comme matériel ou logiciel (Hardware et Software) tout au long de la réalisation de ce projet.

2.2 Hardware

2.2.1 Webcam

Pour la détection des visages en temps réel on a besoin d'un dispositif d'acquisition des images, pour ceci on dispose de la webcam CREATIVE suivante :[7]

- Creative Webcam CT6840 (Part no. 1168403000)
- USB connection
- Résolution pas très élevée pour avoir des images des tailles moyennes qui facilitent le traitement et minimisent le temps de calcule.

Fig. 2.1 – Caméra CREATIVE ct6840

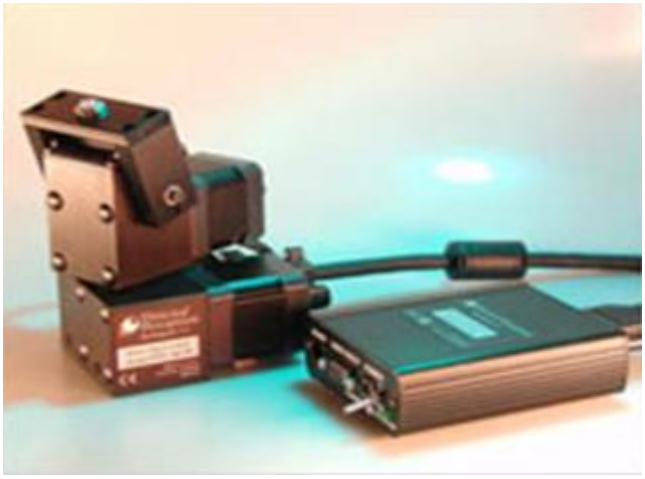

FIG. 2.2 – Le PANT-TILT et Contrôleur

2.2.2 Computer Controlled Pan-Tilt Unit[8]

Ils représentent deux dispositifs liés l'un à l'autre et ils sont entièrement commandés par ordinateur et offrent la gestion de plusieurs paramètres indispensables pour le mouvement et le contrôle de la rotation du moteur de la tourelle parmi les quels on trouve :

- Vitesse de rotation du moteur.
- l'accélération.
- la puissance.
- la résolution.

Le contrôleur D46
Le contrôleur est un dispositif intermédiaire entre l'ordinateur utilisé et la tourelle. Il permet de manipuler les commandes (ordres) des mouvements cinématiques précises, selon des paramètres utilisateurs donnés directement soit par un Hyper terminal ou par un pupitre de commande, mais dans notre programme les paramètres d'entrés pour le mouvement du PAN-TILT sont calculés dans le programme implémenté de détection des visages et ils sont transmis au contrôleur pour faire leurs asservissement grâce à un programme de suivi implémenté sur la base de programme de test de la tourelle (voir figure1.3).

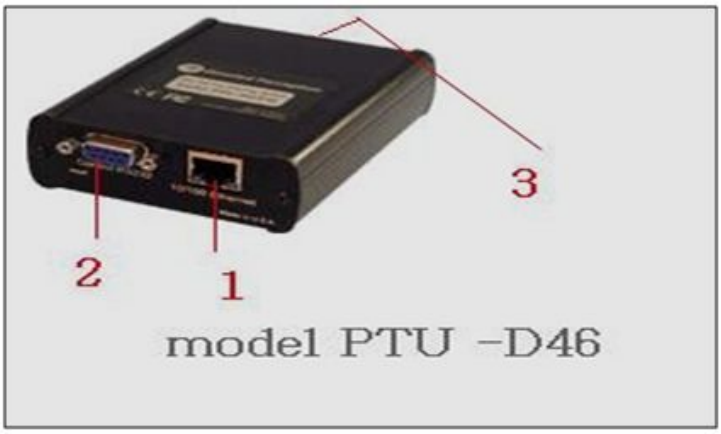

FIG. 2.3 – Le contrôleur D46

– 1 : Port RS 485.
– 2 : Host port RS 232.
– 3 : Port pour le câble de PAN-TILT.

Le Pan-Tilt
Le rôle du Pan-tilt est de recevoir les ordres provenant du contrôleur et de les traduire en mouvement rotatoire de son moteur qui va assurer de sa part un mouvement suivant ses deux axes dans les différents sens selon le besoin en faisant la gestion de plusieurs paramètres d'entrés simultanément (voir figure 1.4).

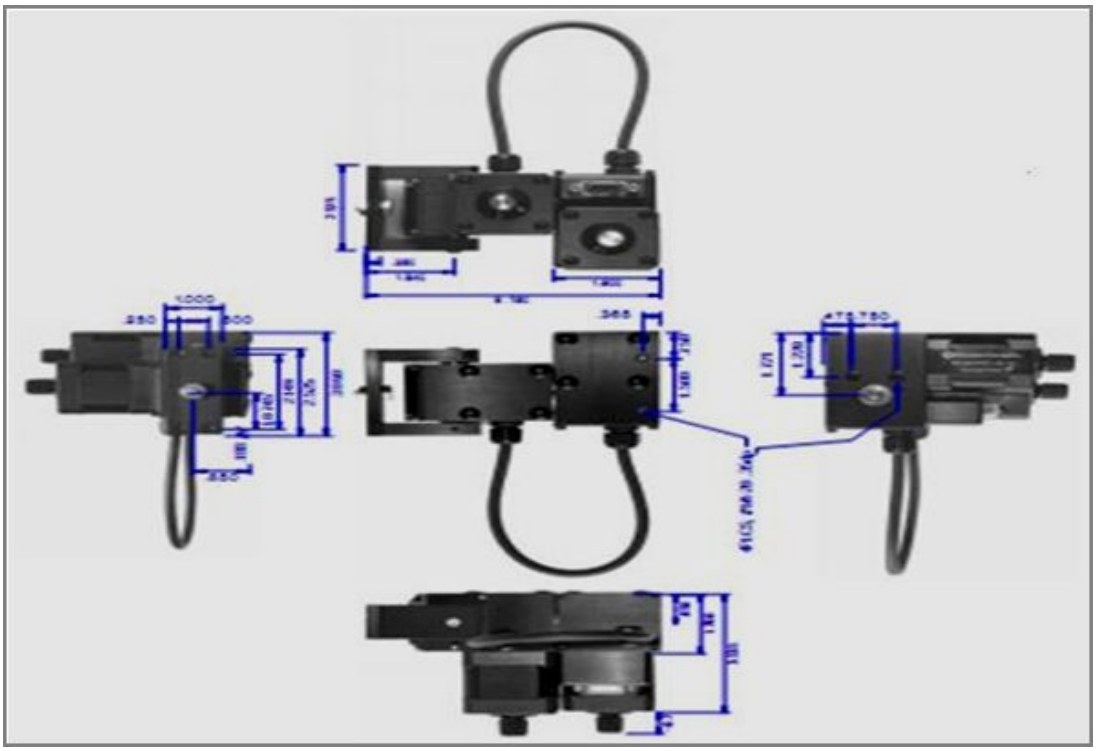

FIG. 2.4 – Les différentes vues de la tourelle

Caractéristique	Valeur
Vitesse de rotation MAX	300°/seconde
Liaison	Liaison série RS-232
Résolution	0.0514°/pas
Calibrage au démarrage	Automatique
Charge maximale	2,72kg

Tab. 2.1 – Caractéristiques de Pan-Tilt

Le câble :
Le pan-tilt D46 (PTU-D46) est relié au contrôleur avec le câble **PT-CABLE-CE-7** ci-dessous assurant la transmission des ordres provenant de l'ordinateur via le port -RS232. Le câble suivant se trouve avec des différentes longueurs. Le choix de la longueur dépond du type de l'application et ses conditions. Pour notre application on utilise la longueur standard de 7 pieds qui est suffisante pour une transmission à temps des données(voir figure 1.5).

Fig. 2.5 – Le câble de connexion de PAN-TILT et le contrôleur

2.3 Software

Dans le but de réaliser l'application de " Détection et suivi de visage ", à côté de ce matériel on a besoin d'utiliser les logiciels présentés ci-dessous :
Matlab : on utilise précisément la version :7.3.0.267(R2006b) afin d'exécuter le programme existant pour la détection des visages dans une image pour visualiser ce qu'il donne comme résultat et tester le bon fonctionnement de ces sous fonctions et leurs performances.
Microsoft Visual C++ : on utilise exactement la version 2005 complète pour :

- L'implémentation de l'algorithme de la détection en temps réel des visages dans une image prise par la webcam utilisée.
- L'implémentation de l'algorithme pour suivre le visage ainsi que le contrôle et l'asservissement visuel en utilisant le Pan-Tilt.

OpenCV2.0 (Computer Vision) : [10] Open source computer vision library in $C/C++$.
C'est une bibliothèque pour la vision par ordinateur permettant de :

- Traiter et manipuler les données image (modifier, redimensionner, transformer, extraire, filtrer...).
- Optimisation des fonctions en temps réel. (Utiliser directement des images prises par la caméra ou précisément une image extraite d'une vidéo).
- Présente plusieurs bibliothèques contenant des fonctions prédéfinies permettant le traitement facile et direct des images.
- Présente une interface pour IPP (Integrated Performance Primetive).

Parmi les modules les plus importants dans OpenCV on trouve :

- **CV-** fonction principale pour OpenCV (main).
- **Cvaux-** fonctions auxiliaires permettant de faire toute opération expérimentale assurant le teste de quelques types de fonction sur différentes images.
- **Cxcore-** assure la définition de la structure des données et présente toutes les opérations arithmétiques et logiques indispensables de traitement.
- **Highgui-** autre types de fonction appelées (GUI functions).

La nouvelle version de " **OpenCV2.0a** " présente beaucoup de changement par rapport aux versions précédentes :
On peut directement télécharger et installer cette version de Open CV sur internet. Après l'installation on remarque qu'il manque les bibliothèques " ***.lib** " et la présence de quelques fichiers " ***.dll** " uniquement.

La génération des bibliothèques de OpenCV2.0 se fait avec l'utilisation d'un programme appelé " CMAKE " permettant de formuler les nouvelles bibliothèques à partir des fichier DLL ou précisément à partir du programme binaire.
Pour cette version de OpenCV on remarque la modification des "***.lib** " en "***200.lib** "ou " ***200d.lib** " ainsi que la génération des nouvelles bibliothèques ayant les noms de la forme de *200d.lib permettant d'avoir plus de fonction prédéfinies.

2.4 Conclusion

Dans ce chapitre on a présenté le matériel nécessaire pour la réalisation du travail demandé en montrant les différentes caractéristiques des différents dispositifs (résolution d'image, vitesse d'acquisition, vitesse de rotation, type de liaison... etc) ce qui aide à bien comprendre leurs fonctionnements pour pouvoir implémenter un programme qui sera compatible avec leurs paramètres ce qui permet d'eviter tout problème provenant du Hard ware.

3. Détection des visages

3.1 Introduction

Les visages constituent une catégorie de stimulus importante par la richesse des informations qu'ils véhiculent. Ils sont à la fois des vecteurs visuels principaux à l'identité des individus et des vecteurs essentiels de communication (verbale et non verbale). La détection automatique de visages est un problème très important. Du point de vue application, elle est à la base de tous système de reconnaissance des visages, car avant de reconnaître n'importe quelle personne il est indispensable de localiser son visage, et vue l'importance de ce domaine les recherches ont donné plusieurs méthodes ayant des principes différents.

3.2 Les méthodes de détection des visages : [5]

Depuis quelques années la détection des formes, des objets ainsi que celle des visages sont prises comme domaines de recherche par plusieurs personnes et sociétés ce qui amène à l'existence de plusieurs approches pour la détection des visages dans une image, parmi ces approches on cite les suivantes :

- Méthode basée sur les caractéristiques invariables :
  - La couleur
  - La texture
- Méthode basée sur la connaissance.
- Méthode basée sur la correspondance avec les modèles (Template Matching).
  - Template déformable
  - Template à face prédéfinie
- Méthode basée sur les apparences
  - Eigen face
  - Distribution-based
  - Réseau de neurones

Dans ce qui suit, on donne un petit aperçu à propos de chacune des méthodes ci-dessus dans le but de bien vouloir comprendre mieux le domaine.

3.3 Caractéristiques invariables (Feature Invariant)

Cette méthode a pour objectif de trouver les caractéristiques structurelles même si le visage est dans différentes positions, conditions lumineuses ou changement d'angle de vue.
Le problème avec cette méthode est que la qualité des images peut être sévèrement diminuée à cause de l'illumination, le bruit ou l'occlusion ce qui empêche l'algorithme de fonctionner correctement.
Cependant, Il existe plusieurs propriétés ou caractéristiques invariables du visage dont les principales sont les suivantes :

3.3.1 Caractéristiques du visage (Facial Feature)

Cette méthode utilise les plans d'arrêtes appelés " Canny detector " et des heuristiques pour supprimer tous les groupes d'arrêtes sauf celles qui représentent les contours du visage. Une ellipse est déduite comme frontière entre l'arrière-plan et le visage. Celle-ci est décrite comme étant formée des points de discontinuité dans la fonction de luminance (intensité) de l'image. Le principe de base consiste à reconnaitre des objets dans une image à partir de modèles de contours connus aux préalables. Pour réaliser cette tache, deux méthodes seront présentées : la transformée de **Hough** et la distance de **Hausdorff**.

Distance de Hausdorff : Cette méthode utilise quant à elle les arêtes comme données de base ainsi qu'un algorithme spécial de 'Template Matching'. En effet, la distance de Hausdorff vise à mesurer la distance entre deux ensembles de points séparés, qui sont la plupart du temps une carte d'arêtes (image de recherche) et un modèle prédéfini.
L'algorithme de base effectue la recherche des meilleurs endroits de correspondance partout dans l'image (translation) et aussi pour différentes rotations. Cette recherche peut également inclure un facteur d'échelle afin de détecter des variations du modèle original. Cette méthode a été utilisée avec succès pour la détection des visages qui présentent des vues frontales (faciale).

L'adaptation de celle-ci pour pallier à différentes poses (rotation axiale de la tête) amène cependant à avoir certains problèmes, car différents modèles devraient être utilisés (nombre important de calibre). De plus, une étape très complexe de décision aurait pour mission de départager les fausses détections ainsi que les détections multiples. Cette méthode est très couteuse en mémoire et en temps de calcule.

Transformée de Hough : La transformée de Hough est une méthode permettant d'extraire et de localiser des groupes de points respectant certaines caractéristiques, équation d'une forme bien déterminée.
Par exemple, les particularités recherchées peuvent être des droites, des arcs de cercle, des formes quelconques etc. Dans un contexte de détection de visage, ce dernier est représenté par une ellipse dans la carte d'arêtes. L'application de la transformée de Hough circulaire produirait donc une liste de tous les candidats (objets probables) étant des cercles ou des dérivées.

L'algorithme de base a également été modifié pour voir ainsi apparaitre plusieurs variantes, dont la " Randomized Hough Transform " (la transformée de Hough aléatoire), qui peut être appliquée à la recherche des formes quelconques tout comme des cercles ou des ellipses (exp :Visages).
Finalement, la transformée de Hough peut être utilisée pour détecter les yeux et les iris. Par contre, cette méthode échouera lorsque l'image est trop petite ou lorsque les yeux ne sont pas clairement visibles. [4]

3.3.2 Texture

La texture de l'être humain est distinctive et peut être utilisée pour séparer les visages par rapport à d'autres objets. Augusteijin et Skufca ont développé une méthode de détection de visages sur une image en se basant uniquement sur la texture.
Le calcule de la texture se fait en utilisant les caractéristiques de second ordre sur des sous-images de $17 * 16$ pixels. Dans cette méthode trois types de caractéristiques sont pris en considération : la peau, les cheveux et le reste des composants de visage.

3.3.3 Couleur de la peau (Skin Color)

Pour les visages la couleur de la peau de l'être humain est une caractéristique spécifique c'est pour sa elle a été utilisée pour la détection des visages. En effet la couleur de la peau est différente selon les personnes et leur origine (Africain, Européen, Asiatique...). Dans ce contexte plusieurs études ont démontré que la plus grande différence s'étend largement entre intensité plutôt que la chrominance lumineuse. [13]

3.3.4 Caractéristiques multiples (Multiple Feature)

Il existe plusieurs méthodes qui combinent les différentes caractéristiques faciales pour la détection des visages. La majorité ente eux adopte des propriétés globales : couleur de la peau, la forme, la taille du visage, pour trouver les candidats puis les vérifier localement en se basant sur les caractéristiques détaillées comme les yeux, les sourcilles, le nez et les cheveux.
Les méthodes standards commencent toujours par détecter les régions qui représentent la couleur de peau, ensuite regrouper les pixels représentant la peau à l'aide de "**connected component analysis** " ou " **clustring algorithme** ".
Si la région regroupée a la forme d'une ellipse elle est alors considérée comme candidate et sur laquelle on appliquera les caractéristiques locales pour vérifier et valider l'existence de visage.

3.4 Appariement de gabarits :(template matching)

La détection des visages se fait à travers un apprentissage d'exemples standards de visages ou d'images frontales contenant des visages. La procédure se fait en corrélant les images d'entrées et les exemples enregistrés (gabarits) et le résultat donne la décision finale soit de l'existence ou non d'un visage.
On trouve 2 types de corrélation suivant le type des gabarits :

- Faces de visages prédéfinies (predefined face template).
- Modèles déformables (Deformable Template).

3.4.1 Modèle prédéfini

Les sous-gabarits souvent utilisés pour la détection frontale sont les yeux, le nez, la bouche et les contours du visage. Chaque sous-gabarit est défini par une segmentation des lignes. On commence par une comparaison entre les sous-images et les contours de gabarit pour détecter les visages candidats. Puis on translate les sous-gabarits comme les yeux, le nez ... sur les positions candidates.
On peut conclure que la détection s'effectue en deux étapes : la première consiste à la détection des régions candidates, la deuxième est l'examen des détails pour déterminer les caractéristiques du visage. Cette méthode est utilisée par Sakai et Al

3.4.2 Modèle déformable

Cette approche est utilisée dans le but de modéliser les caractéristiques faciales qui s'adaptent élastiquement par rapport au modèle du visage présent. Dans cette méthode, les caractéristiques faciales sont décrites par des gabarits paramétrés. Une fonction est définie pour relier les contours, les sommets et les angles dans l'image d'entrée, pour faire correspondre les paramètres sur les gabarits. La meilleure adaptation du modèle élastique est de trouver la fonction énergétique en minimisant les paramètres. Cette méthode a été utilisée par Yuille et Al.

3.5 Méthode basée sur les apparences (Appearance-based method)

La différence entre cette méthode et " Template matching " est que les modèles (Template) sont lus à partir des images d'apprentissage qui doivent être représentatives et faites à différentes positions du visage.
Généralement " appearence-based method " se base sur des techniques d'analyse statistiques (pourcentage d'existence des modèles dans l'image) et d'apprentissage automatique pour trouver des caractéristiques significatives des visages et des non visages.
Pour l'approche suivante il existe plusieurs méthodes où chacune d'entre elles se base sur une des caractéristiques du visage ou plus précisément une partie de visage qui peut être interprétée sous le cadre probabiliste.

3.5.1 Eigen faces

La PCA (Analyse en Composantes Principales) n'a besoin d'aucune connaissance préalable de l'image, son principe de fonctionnement s'appui sur la construction d'un sous espace vectoriel retenant que les meilleurs vecteurs propres, tout en gardant le maximum d'information utile non redondante (méthode très efficace pour réduire la dimension des données et le temps de calcule). L'exemple Le plus récent de l'utilisation d'(eigenvectors) est dans la reconnaissance des visages de Kohonen dans laquelle un simple réseau de neurone a démontré sa performance dans la détection des visages passant par la normalisation des images de visage.

Le réseau de neurones estime la description des visages par l'approximation des vecteurs propres et l'auto-corrélation matricielle de l'image.
Kirby and sirovich ont démontrés que les images de visage peuvent être codées linéairement en nombre modeste d'image d'apprentissage. La démonstration est basée sur la transformée de Karhumen-loeve, appelée aussi " analyse des composants principales "ou " la transformée de Hottlling ".

La première idée proposée par Pearson en 1901 et par Hotteling en 1933 est d'avoir une collection d'image d'apprentissage de $n * m$ pixels représentés comme des vecteurs de taille $m * n$, le principe des vecteurs est de trouver le sous-espace optimal pour ajuster l'erreur entre la projection des images d'apprentissage dans ces sous-espaces et les images originales miniaturisées.

3.5.2 Distribution based

Sung et Poggio ont développés la méthode basée sur la distribution pour la détection des visages, elle démontre comment la distribution de l'image appartenant à une seule classe d'objet peut être classifiée comme exemple de classe positive ou négative. Ce système est constitué de deux composants, le premier est constitué d'un modèle d'exemple basé sur la distribution de visage et de non visage, le deuxième est un classifieur à perceptron multicouche. Chaque exemple de visage et de non visage est tout d'abord normalisé, puis redimensionner l'image de taille ($19 * 19$ pixels) pour la traiter comme 361 vecteurs bidimensionnels, ces structures sont regroupées en six bouquets de visages et de non visages en utilisant l'algorithme modifié K-means.

3.6 Réseau de neurones

Le réseau de neurones est utilisé pour classifier les pixels de l'image, en tant que visage ou non-visage. Dans toute utilisation de réseaux de neurones, il faut définir une topologie du réseau. La topologie de réseau est déterminée par des testes successifs et il n'y a aucune méthode standard à suivre pour définir la meilleure.
Un visage se distingue surtout par des yeux, un nez et une bouche. La topologie de base sera donc d'une unité finale fournissant une réponse binaire ou probabiliste. On mettra derrière cette unité les couches cachées du réseau, on appelle notamment

cela une topologie de base car le nombre d'unités, leur taille et leur position restent non empiriques et ne peuvent jamais être exactement fixés.
Les réseaux de neurones les plus répandus et les plus simples à la fois restent les perceptrons multicouches **(PMC)** qui consistent à une succession de 9 couches, interconnectées totalement ou partiellement.
L'algorithme d'apprentissage total reviendra à transmettre à tous les PMC, traitant chacun une unité, le résultat attendu. Si l'exemple à apprendre est un visage, la première étape est de transmettre aux PMC les yeux, la bouche et le nez. De là, chaque PMC applique son algorithme d'apprentissage.
L'inconvénient de cette approche réside dans le temps de calcul qui ne permet pas souvent de faire des traitements en temps réel. Le schéma suivant explique bien cette approche.

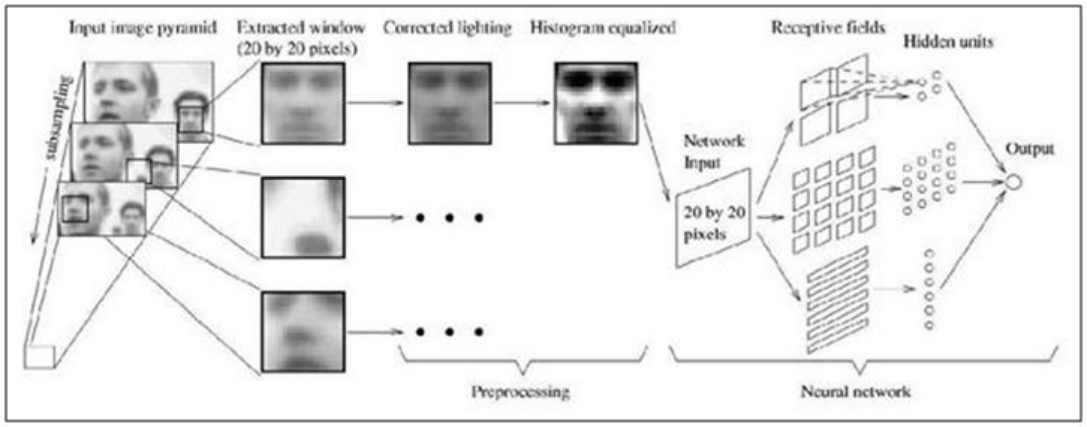

Fig. 3.1 – Principe du réseau de neurone pour la détection des visages

3.7 Comparaison de quelques méthodes

Méthode	Avantages	Inconvénients
Les couleurs	Rapidité,détection de la peau	Les yeux,l'arrière-plan
Templates	simple, Mesure de similarités	précision, multi-échelle
Les arêtes	invariance aux rotations	la rotation (cercle), Les yeux
Réseau de neurones	Apprentissage automatique	précision,recherche multi-échelle

Tab. 3.1 – Comparaison de quelques méthodes

3.8 Méthode Implémentée :[6]

Il existe de nombreuses techniques de détection qui diffèrent aussi bien par les approches qu'elles emploient que par les techniques d'apprentissages qu'elles utilisent. L'approche implémentée dans ce travail de fin d'étude est connue sous le nom de la méthode de "**YIFAN SHI** ".

3.8.1 Principe de la méthode

L'approche ou la méthode proposée est basée sur l'utilisation de deux méthodes faites au part avant :

La méthode de"**détection de la peau**" pour éliminer les sections n'ayant pas la couleur de cette dernière utilisant (un dispositif invariant avec la couleur de la peau comme caractéristique spécifique de visage) et celle de " **template matching** " pour éliminer les sections n'ayant pas les caractéristiques d'un visage qui sont précisées par l'algorithme.
Elle exige un ensemble d'opération affin d'appliquer plusieurs traitements permettant la détection des visages ayant un ensemble des caractéristiques bien déterminé. Cette méthode utilise deux principales ressources d'information pour fonctionner :

- Les résultats d'interactions entre l'image et les exemplaires (template) présentés pour l'algorithme.
- La couleur de la peau dans l'image qui est comparée à un exemple présenté à l'algorithme.

3.8.2 Les étapes de la détection de visage

Charger et redimensionner l'image :Charger et redimensionner l'image : tout d'abord la photo originale est chargée et redimensionnée tel que sa taille maximale soit 800px. Ceci est fait pour diminuer le volume des calcules et le temps de traitement de l'image, tout en veillant que les plus petits visages qui apparaissent dans des scènes normales sont assez grands pour produire une crête significative ayant une amplitude au-dessus du bruit quand ils sont corrélés avec un visage des calibres (templates).

Conversion des images et des photos :Les photos et les images des calibres sont converties en espace chromatique couleur de laboratoire (préparation à la corrélation).

Comparaison des couleurs :La couleur de la photo est comparée à la couleur de la peau, calculée à partir des calibres donnés à l'entré du programme.

Combinaison des résultats : les résultats de la couleur et de la corrélation sont combinés pour déterminer les positions probables des centres des visages.

Optimisation de la position :les positions initiales sont groupées et raffinées de sorte que les finales devinent tellement les positions exactes des visages.
Dans ce qui suit on présentera les principales fonctions assurant le bon fonctionnement de l'algorithme de " face detection ".

3.9 Traitement de photo

Dans cette partie du rapport on essaye de détailler les différentes parties des sous fonctions du programme en montrant les caractéristiques spécifiques de chacune : [10]

3.9.1 Les calibres (Templates)

Cette méthode nécessite l'utilisation d'un ensemble de calibre pour corréler la photo prise par la caméra avec un ensemble des images des visages définis initialement. Les calibres doivent avoir les caractéristiques suivantes :

- **Plusieurs visages sont employés** : des visages multiples sont employés pour capturer la moindre geste qu'un visage peut avoir en photo normale. D'ailleurs, toutes les photos sont différentes mais tous ont des caractéristiques de visage donc, si un certain secteur est semblable à tous, nous pouvons être certains que ce soit un visage, alors que si un autre secteur est par hasard semblable seulement à une d'entre elles et différent aux autres, il recevra toujours une valeur basse de corrélation.
- **La même partie du visage** : tous montrent approximativement le même secteur du visage. C'est important car les corrélations avec les différents calibres sont ajoutées et il est nécessaire qu'elles produisent les crêtes maximum dans les mêmes endroits
- **Les photos sont normales** : tous les modèles regardent directement l'appareil-photo, c'est à dire les visages ne sont pas tournés de point de vue de vision-neuses.
- **Photos prises avec flash** : la majorité des calibres sont prises avec flash, la chose qui rend l'algorithme plus efficace puisque l'endroit des nuances et de la similitude des couleurs détectées est commun.

3.9.2 Espace couleur

Chaque image est représentée par trois matrices des pixels. Au départ elles sont chargées dans le format de RVB mais cet espace chromatique n'est pas optimal pour l'identification des visages car dans ce cas chaque image est représentée par 3 matrices qui rendent le traitement très couteux en mémoire et en temps de calcule. Les 3 matrices sont corrélées avec celles des modèles. Les résultats sont comme suit :

- Grande similitude : → résultat proche de **1**.
- Pas de similitude : → résultat = **-1** .

Une correction est faite en ne gardant que les valeurs qui sont égales dans les trois matrices de corrélation sinon on affecte -1 à cette case.
Le schéma suivant explique le principe de la détermination de l'espace couleur de visage.

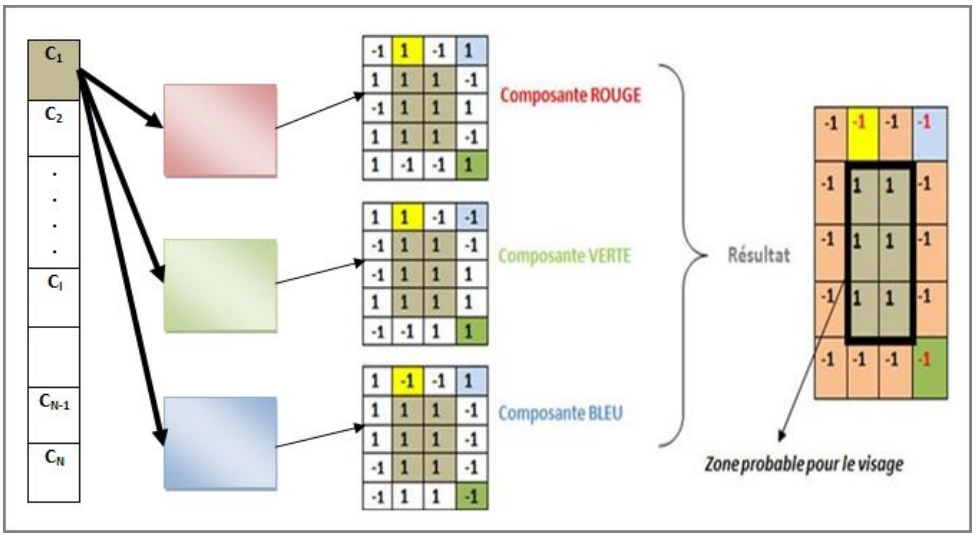

FIG. 3.2 – Principe de détermination de l'espace couleur

Pour éviter le problème de la luminance, l'espace chromatique de l'image doit être changé en espace chromatique de laboratoire. Cet espace est optimal parce que la luminance est séparée des composants de couleur ce qui permet la comparaison facile entre le calibre et les couleurs de la photo. Par conséquent, toute l'information de couleur est contenue dans les deux matrices qui ont de différentes valeurs relatives (lèvres, yeux et sourcils).

3.9.3 Corrélation normale

L'astuce principale de la détection de visage est la corrélation normale entre la photo et chacune des composants de couleur et des composants de chacun des calibres de visage.
Pour cette étape on emploi la fonction de Matlab normxcorr() qui revoit 2 valeurs uniquement :

- 1 : corrélation maximale
- -1 : corrélation minimale

On ajoute aussi les corrélations des 3 composants de couleur en veillant qu'une valeur ne sera élevée que si elle est la même dans les trois composantes sinon elle reçoit une basse valeur (-1).
Les corrélations combinées venant de tous les calibres sont ajoutées, donnant des valeurs élevées dans les endroits où un visage est présent et des faibles valeurs où il n'y a pas de visage.

3.9.4 Identification automatique de la taille

Cette tache est indispensable dans notre algorithme car ce dernier sera très compliqué et couteux en mémoire et en unité de calcule en corrélant la photo avec un calibre de taille quelconque. Ceci est dû à ce que l'espacement absolu des pixels dans les images diffère de celui dans la photo ce qui donne toujours des valeurs basses de corrélation.
La solution trouvée pour ce problème est de définir une douzaine de tailles qui sont les plus probables ou les plus utilisées ensuite avec le 1er calibre on calcule les corrélations pour toutes les tailles et on ne garde que la taille qui donne le maximum de corrélation et on sauvegarde la valeur trouvé dans un vecteur. Le calcule se répètera avec tout les calibres et on ne garde toujours que la valeur maximale de corrélation et la taille correspondante dans un vecteur. La moyenne des tailles de ce vecteur est calculée et les corrélations des trois tailles les plus étroites sont exécutées. Enfin on choisie la valeur la plus élevée et le vecteur des tailles est mis à jour.
Malgré que cet algorithme optimise la fonction de corrélation du programme mais il est évidemment une exécution contre le compromis de calcul-coût vu le nombre d'itérations faites pour déterminer uniquement la taille.

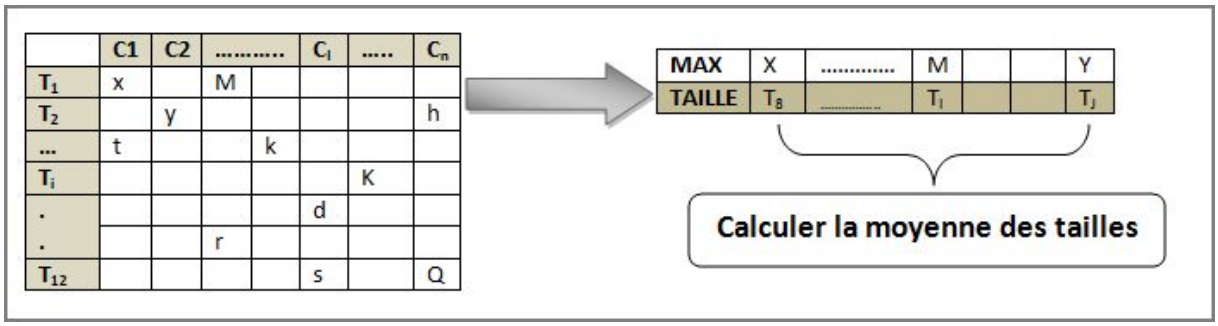

Fig. 3.3 – Principe d'identification de la taille

3.9.5 Identification automatique des couleurs

L'identification automatique de couleur se base sur l'idée que la couleur de la peau des visages dans une image sera déduite de celle des calibres. En premier lieu on charge les visages calibres, on les converties en couleur de laboratoire et on les ramène à une moyenne pour obtenir une seule paire de (a, b) qui sera la couleur de la peau de référence. (Voir la méthode de conversion en annexe) En second lieu on calcule la distance dans (a, b), espace de chaque pixel dans la photo de couleur de la peau, pour obtenir une image de différence. Dans un dernier lieu une fonction sigmoïde est appliquée à cette image de différence donnant les résultats suivants :

- 2 dans le secteur élevé
- -2 dans le secteur inférieur.

Ces valeurs semblent basses elles ont été optimisées par plusieurs photos normales pour fournir un bon perfectionnement en évitant les faux positifs.

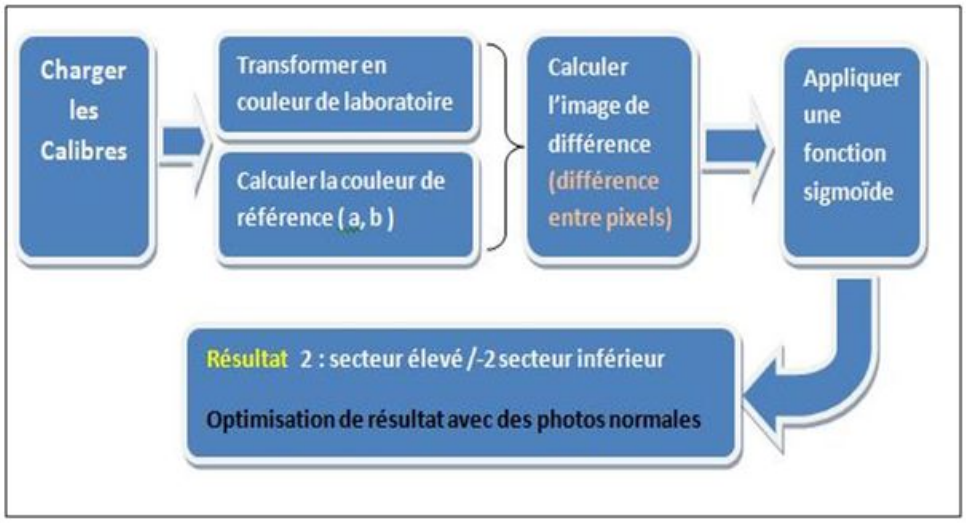

FIG. 3.4 – Les étapes de la détermination automatique de la couleur

FIG. 3.5 – Résultat de la fonction de détermination de la couleur

Grâce à cette fonction, les valeurs élevées de corrélation dans le secteur de visage augmentaient encore, alors que le bruit dans la fonction de corrélation est atténué dans les secteurs qui n'ont pas la couleur de la peau. Ces images montrent bien le résultat obtenu après l'exécution de la fonction précédente(Fig 3.5).

Cet algorithme fonctionne même si les visages dans les modèles sont caucasiens. Car la majeure partie d'information changée se trouve dans la matrice de luminance et ceci permet de détecter la couleur de la peau de plusieurs personnes des races différentes.

3.9.6 Détection des visages

Elle correspond à l'avant dernière étape dans cet algorithme, elle s'exécute automatiquement dés que la somme des corrélations combinées est faite et la détection automatique de la couleur est effectuée. Généralement la détection des visages se compose d'une série d'opération et de seuils assurant cette fonction.

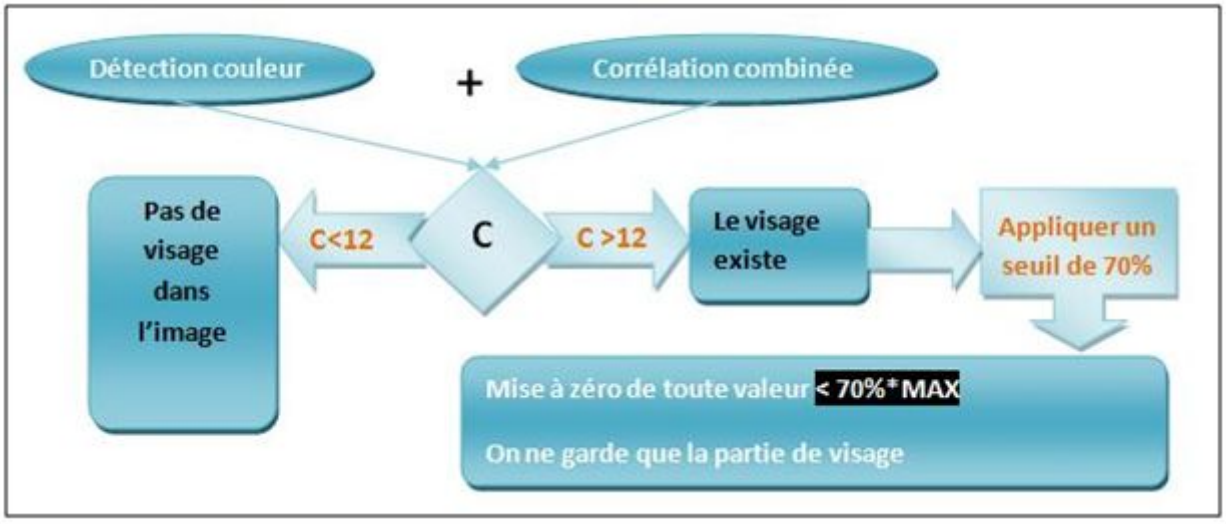

FIG. 3.6 – Principe de la méthode de détection de visage

FIG. 3.7 – Résultat de la détection de visage sur une Image

Fonctionnement : soit C la valeur de corrélation combinée, selon C on distingue les cas suivants :

- $C \prec 12$: Pas de visage dans la photo.
- Théoriquement :
  - $C = 47$: la valeur maximale possible.
  - $C = -47$: la valeur minimale possible.
- Pour les photos normales contenant des visages $C \geq 22$.
- Pour les photos normales ne contenant pas des visages $C \leq 8$.

Si l'algorithme détermine qu'il y a un visage dans la photo, la prochaine étape est d'appliquer un seuil relatif à tous les Pixel dans l'image. Les Pixel en-dessous de **70%** de la valeur du maximum sont rendus égaux à 0. Dans ce cas nous appliquons un seuil relatif parce que les images avec des visages se sont avérées avoir des variations significatives en passant de l'une à l'autre (voir figure 3.7).

3.9.7 Groupement automatique

Suivant les résultats de la fonction de détection des visages, plusieurs maximums locaux dans l'image peuvent correspondre à un visage simple. Cet effet est principalement produit par l'algorithme automatique de détection de couleur puisque la somme des corrélations produit une crête autour du milieu du visage, donc ces valeurs sont atténuées par l'algorithme de la détection de la couleur de la peau. Par conséquent, quand le résultat du sigmoïde est ajouté à la somme des corrélations combinées, au lieu d'une crête simple, des multiple maximum locaux sont obtenus.

Fig. 3.8 – Résultat du groupement automatique

Le principe de la fonction de groupement automatique : L'algorithme de groupement automatique détecte tous les faisceaux des points dont le diamètre est plus petit que la taille principale (on assume qu'elle est environ **40%** plus grande que la taille de visage). La moyenne pondérée des points de chaque faisceau est calculée, ayant pour résultat les évaluations finales des positions des visages en donnant la position exacte du centre.

3.10 Implémentation de la méthode

Pour cette méthode il existe un code 'Matlab' qui détaille les différentes fonctions de cet algorithme de détection des visages mais le problème reste toujours les exigences de l'être humain d'avoir les résultats le plus rapide possible et s'approcher du temps réel mais ceci est difficile avec Matlab car ce dernier nécessite un énorme temps de calcule surtout si la taille de l'image est importante (une matrice de taille N*M avec N et M sont respectivement le nombre des ligne et le nombre des colonnes de la matrice image nécessite N*M opérations) ainsi que ce programme devient non efficace si la fréquence de capture des photos est supérieure à l'inverse du temps du traitement des images.

La solution est de traduire ce programme Matlab en programme C++ ou de trouver un programme C++ assurant presque la même fonctionnalité en utilisant le même principe présenté dans la partie précédente pour répondre au besoin désiré.

3.10.1 Etapes de réalisation

Installation OpenCV2.0

Il existe plusieurs version de OpenCV mais on choisi d'installer une nouvelle version donnée gratuitement sur le site de Open Computer Vision Library :
(http ://sourceforge.net/projects/opencvlibrary/) intitulée : **OpenCV-2.0.0a-win32.exe**.
Après l'installation du programme téléchargé on remarque le non existence des fichiers ***.LIB** et quelques fichiers ***.DLL**, dans ce cas la une seul question qui se pose :
Comment faire pour les générés de nouveau ?
Solution :
On remarque que OpenCV2.0 est compilé par défaut avec (MinGW) au lieu de Visual Studio donc pour Pouvoir utiliser OpenCV dans Visual Studio on doit compiler sa source pour VS2005, pour cela on doit avoir le programme **CMAKE2.4** qu'on peut le téléchargé et l'installé gratuitement. [11]
CMake est un générateur de système de construction pour différente plate-forme qui permet de générer les fichiers pour différent compilateur selon la configuration précisée. Il présente une interface conviviale et facile à configurer pour obtenir les fichiers pour la plate- forme désirée.

Etapes de recompilation d'OpenCV2.0 pour VS

On doit configurer CMAKE dans le but de recompiler OpenCV pour VS 2005. Cette interface montre bien les différentes étapes de configuration :

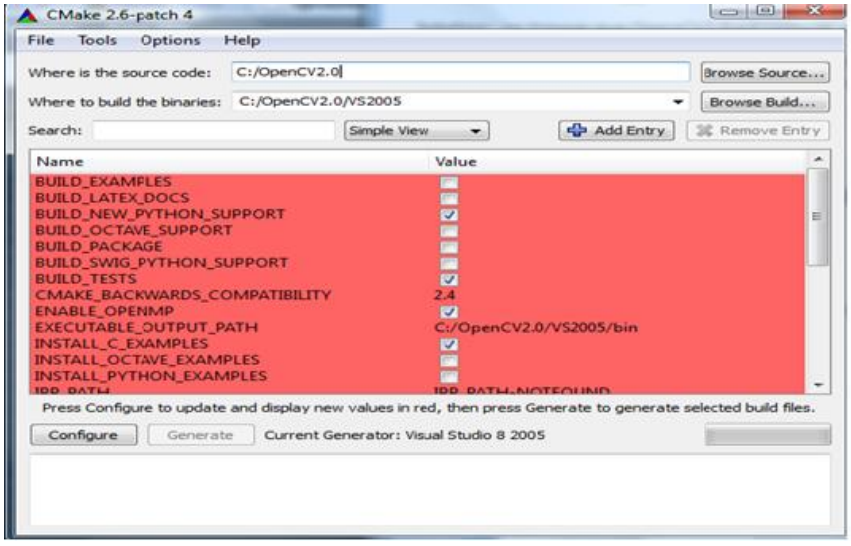

Fig. 3.9 – interface de CMAKE pour la compilation avec VS

- On lance le programme cette interface s'affiche(voir figure3.9).
- On donne l'emplacement source (fichier binaire à recompiler).
- On donne le fichier de sortie où le logiciel va faire la génération des $*.LIB$ et $*.DLL$. si le fichier n'existe pas il sera crée automatiquement au moment de la vérification des données.
- On choisi configurer : le programme va s'auto-configuré en utilisant les paramètres donnés à l'entré.
- Choisir les paquets qu'on veut installer.
- Choisir le bouton générer.
- Compiler et exécuter le programme solution qui se trouve dans le fichier VS2005.
- Copier les $C : /OpenCV/bin/ * .dll \leftarrow C : /OpenCV/bin$
- Copier les $C : /OpenCV/lib/ * .lib \rightarrow C : /OpenCV/lib$

Configuration initiale du programme

Après avoir fait tous ces étapes on doit ajuster les paramètres de notre programme $C++$ selon nos besoins pour utiliser correctement OpenCV et les bibliothèques offertes.

- Ajouter les $*.h$ files et spécifier le chemin exacte : $C : "'/OpenCV2.0/include/opencv/* .h$
- Donner l'emplacement de bibliothèques dans la fenêtre de propriétés.
- Pour notre programme de détection des visages dans la zone de debugging de la fenêtre de configuration on doit introduire un fichier XML qui contient les principales caractéristiques d'un visage et plus précisément il contient les gabaries ou les modèles (des photos frontales pour la corrélation) qui vont être utilisées par l'algorithme au moment de la détection et exactement à l'étape de " Template Matching ".
  $--cascade = "C : /OpenCV2.0/data/haarcascades/haarcascade\_frontalface\_alt.xml"$

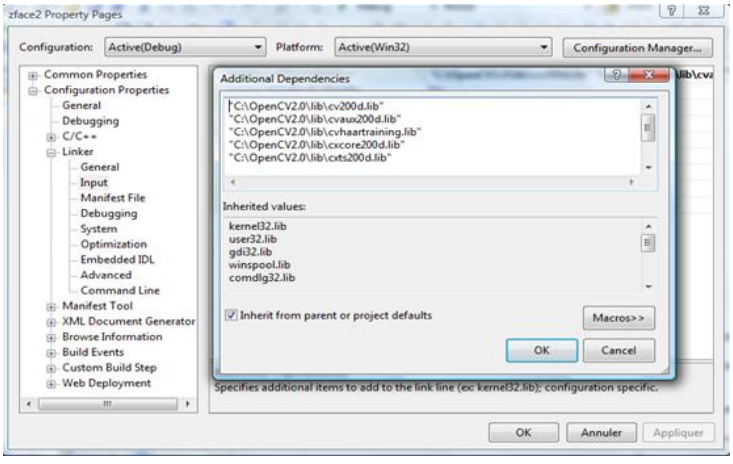

FIG. 3.10 – interface de configuration du programme C++ de face détection

3.11 Les fonctions principales de face détection

Les opérations de capture

- *cvCaptureFromAVI () :* la capture correspond à prendre des images de la vidéo prise par la caméra CREATIVE utilisée pour être traitées après des intervalle de temps bien déterminés, c'est-à-dire avec la fréquence de capture nécessaire et suffisante pour faire la détection en temps réel.
- *cvReleaseCapture(capture)* : cette fonction permet de libérer la capture c'est à dire vider l'espace mémoire réservé à la capture des images qui vont être traitées. Cette fonction ne s'exécute que s'il y'a une capture déjà faite sinon une erreur s'affiche.

La fonction de détection

- *detectAndDraw(Image, Temlate...)* : elle correspond à la fonction principale du programme implémenté, permet d'effectuer tout les traitements nécessaires pour la détection des visages comme :(redimensionner, comparaison des histogrammes ,conversion de la couleur en nivaux de gris et l'appariement des modèles utilisés avec l'image traitée).

NB : les calibres(Templates) sont appelés par leurs noms qui sont sauvegardés dans un fichier chacun sur une ligne ce qui permet des les utilisés par la suite par la fonction de Matching.

Le traçage de cercle autour du visage

- *Circle(Image, Centre, Rayon, Couleur, Epaisseur, Type ligne, Décalage)*

La structure de cette fonction est définie dans le programme implémenté et elle ne s'exécute qu'après avoir déterminé ses paramètres qui se déduisent à partir du traitement effectué sur les images car les rayons des cercles sont proportionnels aux tailles des visages trouvés ainsi que les centres sont confondus avec ceux des visages. Le résultat de la détection est affiché dans une fenêtre distincte de celle de la compilation du programme.

La fenêtre résultat : Cette fenêtre est destinée à afficher la capture (vidéo) de la caméra utilisée au cours du temps ainsi que le résultat de détection de l'algorithme utilisé. Parmi les actions faites sur cette fenêtre on trouve :

- *cvNamedWindow ()* : consiste à initialiser la fenêtre de capture comme interface de sortie du programme.
- *Imshow (nom fenêtre, image)* : affichage dans l'interface résultat.
- *cvDestroyWindow (nom de la fenêtre)* : permet de détruire la fenêtre de capture, elle permet aussi de libérer la capture mais elle s'exécute après la fonction cvReleaseCapture() qui libère la capture.

3.12 Test et résultat

3.12.1 Les résultats obtenus

Dans cette partie on se contente de montrer les domaines de succès de l'algorithme implémenté lors de son exécution ainsi que ces avantages. Cette photo montre bien que l'algorithme détecte tout les visages présents à condition que ces derniers répondent aux conditions de fonctionnement de cette application.
Les avantages :

- Détection presque en temps réel qui permet de réutiliser cet algorithme dans une des applications pratiques par la suite (réponse rapide).
- Bonne précision de la détection en cas d'absence de perturbation.
- 100% efficacité dans le cas ou les visages sont tous normales
- Erreur presque nulle de la position exacte de centre de visage.

3.12.2 Les limites de cette méthode

Malgré le grand succès connu par cet algorithme, il présente plusieurs déficiences dans certaines condition ce qui donne quelques limites détectées à l'étape de teste de l'application.

Angle de visage

le programme ne détecte rien et ne fonctionne pas quand des têtes présentées dans l'image sont inclinées latéralement ou verticalement. Ce problème est dû à ce que tous les visages utilisés comme références (modèles) regardent directement l'appareil-photo, ainsi les corrélations avec les images quand les têtes sont inclinées sont relativement basses car la plupart des photos sont normales, il y a au moins un visage regardant directement à l'appareil-photo qui aurait des corrélations élevées, donc, quand le seuil relatif est appliqué les basses corrélations dans les têtes inclinées tomberont au-dessous de ce lui qui les rendent indétectables par notre algorithme.

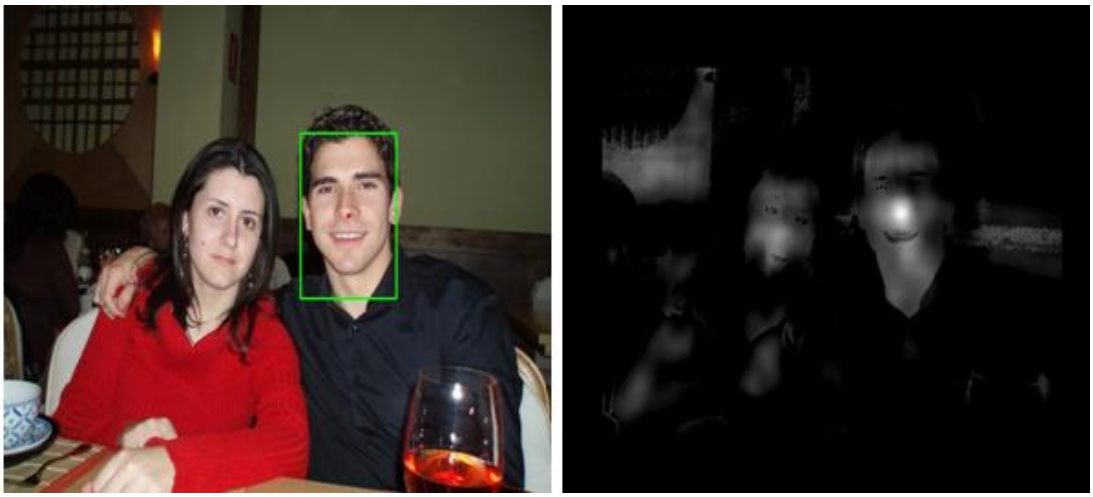

FIG. 3.11 – résultat de détection de visage incliné

Occlusion de visage :(une partie du visage est cachée)

Dans ce cas l'algorithme s'exécute mal si un visage est partiellement caché par un autre ou par un objet quelconque. Ce mal fonctionnement est dû à ce que la corrélation dans la partie cachée sera faible car elle ne correspond pas à un visage, ce qui donne une valeur totale de corrélation plus basse. Donc cette partie analysée ne sera pas considérée comme visage ce qui diminue la performance de l'algorithme.

FIG. 3.12 – Résultat de la détection des visages partiellement cachés

les faux Positifs : (false positives)

Bien que la valeur des faux positifs fixée pour le logiciel ait une probabilité relativement basse ils apparaissent toujours en quelques photos. Généralement ils apparaissent dans des mains car ils ont une couleur presque exacte a celle de la peau et si le font du côté du visage est petit il produit aussi des corrélations significatives.

FIG. 3.13 – Photo montrant la détection des faux positifs

Photos de jour

La performance de l'exécution du logiciel est dégradée pour quelques photos de jour. Ce phénomène est dû à la combinaison de deux facteurs suivants : En premier lieu, la variété des nuances possibles dans des conditions de jour constitue un défi pour ce genre d'algorithmes parce que la plupart des calibres sont des photos normales prises avec le flash et elles ne montrent pas toutes les nuances possibles. En second lieu, la couleur de la peau peut changer avec la lumière, et comme la plupart des calibres sont prises sous la lumière instantanée (celle du flash), les nuances de couleur peuvent être différentes.

FIG. 3.14 – Résultat de la détection dans une image de jour

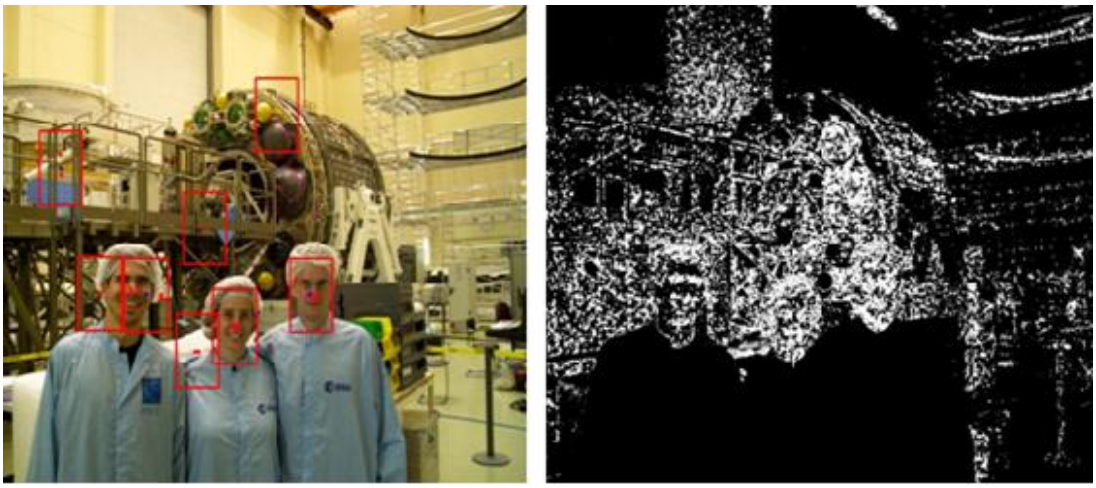

Fig. 3.15 – Résultat de la détection en basse lumière

Photo de basse lumière

Les photos prises dans des basses conditions légères lancent un défi pour l'identification des visages. Le problème est que ces photos sont très bruyantes, ainsi les couleurs sont dégradées. Cette dégradation a deux effets : d'abord, la couleur de la peau n'est pas correctement identifiée, car l'identification automatique de couleur peut atténuer les corrélations au lieu de les augmenter. En second lieu, les deux matrices 'a' et 'b' sont très bruyantes causant la perte de plusieurs dispositifs, ce qui donne des valeurs basses de corrélation.

Échec d'identification automatique de la taille des visages

Fig. 3.16 – Résultat montrant l'échec de la méthode de détection automatique de la taille

Ce programme est censé pouvoir découvrir les tailles des visages dans la photo, Cependant, l'algorithme suppose que tous les visages ont les mêmes tailles, ce qui est le cas habituel dans la plupart des photos normales, mais il ne l'est pas dans certaines photos d'entre elles. D'ailleurs, l'algorithme essaye seulement quelques tailles différentes, ainsi si la taille d'un visage est égale ou comprise entre deux de ces

tailles, le logiciel tend à employer la plus petite d'entre elles pour identifier la taille correcte et par ce que il a les variations rapides qui sont habituellement présentes dans les photos et induisent des valeurs plus élevées de corrélation. Dans ces cas l'image de résultat est habituellement au-dessous du seuil absolu d'identification de visage.

3.13 conclusion

Dans ce chapitre intitulé " face detection " on a vue quelques méthodes de détection des visages dans une image et on a étudié une d'entre elles appelée : " Méthode de YI Fan SHI " qui combine le principe d'appariement des gabaries et la détermination de la couleur de la peau et dans le but de mieux faire la détection en temps réel on recourt à utiliser OpenCV en C++ pour diminuer le temps nécessaire de traitement affin d'utiliser ce programme pour un asservissement visuel.
Puisque la détection des visages et la base de plusieurs application de nos jours comme la reconnaissance, le tracking etc. on choisie le programme suivant pour être comme base pour l'implémentation d'un programme de tracking avec le PAN-TILT qui va servir à présenter un cas d'asservissement visuel.

4. Suivi de visage

4.1 Introduction

Dans la partie précédente et après avoir expliquer l'implémentation de la méthode de détection des visages présents dans une image prise en temps réel par une caméra de type CREATIVE on désire dans cette partie exploiter les résultats obtenus précédemment pour l'implémentation d'un programme intitulé " suivi de visage " et comme son nom l'indique il va servir à suivre le visage dans son mouvement et qui peut être utilisé par la suite pour résoudre quelque problèmes de suivi des formes ou des objets bien déterminés.

4.2 Problématique

Notre caméra CREATIVE utilisée pour la détection des visages ne peut couvrir qu'un angle presque égal à 47° uniquement ce qui ne rend l'algorithme de détection des visages efficace que dans cette zone mais on désire détecter le visage de la personne même si elle se déplace en sortant de cet arc de détection qui représente le champ de vision de la loupe de la caméra.

4.3 Proposition

Pour assurer la fonction de suivi on désire fixer notre caméra CREATIVE CT 6840 sur un mécanisme appelé PAN-TILT PTU-D46-17,5 une sorte de tourelle pour lui donner un mouvement suivant deux axes de déplacement PAN (horizontalement) et TILT (verticalement) pour que sa loupe puisse suivre le centre de visage et garde ce dernier toujours au milieu de la fenêtre et dans son champ de vision tout au long de la période de détection.

4.4 Contrôle de PAN-TILT

4.4.1 Présentation générale

Dan le modèle représenté ci-dessous on remarque que le PTU ou l'unité de PAN-TILT est connectée à un contrôleur. La puissance pour le contrôleur peut être assurée à partir d'une alimentation d'énergie facultative d'AC/DC ou d'une source de puissance de batterie.

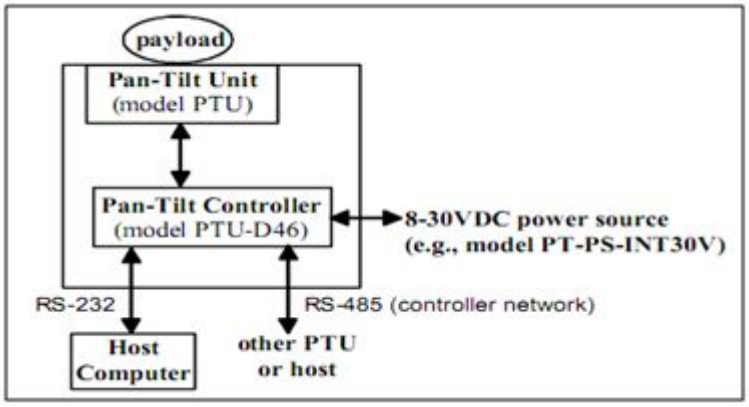

FIG. 4.1 – Modélisation de l'environnement de la tourelle

Le " PTU CONTROLER " accepte des commandes par l'intermédiaire de RS-232 à partir d'un ordinateur et il permet de bouger le PTU à la position désirée. Le contrôleur de pan-tilt peut être relié à d'autres contrôleurs par l'intermédiaire d'un réseau du multi-drop RS485 de sorte qu'une porte série simple d'ordinateur serveur puisse commander plusieurs PTU.

4.4.2 La liaison RS-232

C'est une norme standardisant un bus de communication de type série sur 3 fils (électrique, mécanique et protocole) qui est disponible sur tous les PC jusqu'à l'année 2000, appelé aussi port série. Sur les systèmes d'exploitation MS-DOS, Windows le port RS-232 est connu sous le nom de **" port COM "**jusqu'à nos jours, cette liaison est fréquemment utilisées dans l'industrie pour la connexion des appareils électroniques (appareil de mesures, automates..etc)

4.4.3 Type des données sur le port série

Le contrôleur utilisé est connecté au port COM1 et communique avec un 'baudrate' de 96OO. Pour tester l'envoi des caractères sur le port série il suffit juste d'ouvrir un logiciel d'écoute sur ce port puis exécuter quelque commandes de contrôle de la tourelle via l'hyper Terminal. On remarque que le logiciel d'écoute a détecté '1000' qui est '3 E8' en hexadécimal. Donc on peut conclure que les informations qui passe dans le port série sont en hexadécimal avec quelque exceptions du codage dans certain cas.

4.4.4 Les modes de contrôle de PAN-TILT

Contrôle par commandes ASCII
Le PAN-TILT peut être contrôlé à travers les ports séries intégrés (RS-232 et RS-485) en utilisant des commandes ASCII qui sont documentées dans le manuel d'utilisation. Ceci peut être fait utilisant une interface de communication tel que l'HyperTerminal, ou d'une application faite sur la base des commandes prédéfinies.
Ce mode permet l'exécution de plus de 10 commandes /seconde.

Le schéma suivant illustre bien ce modèle.

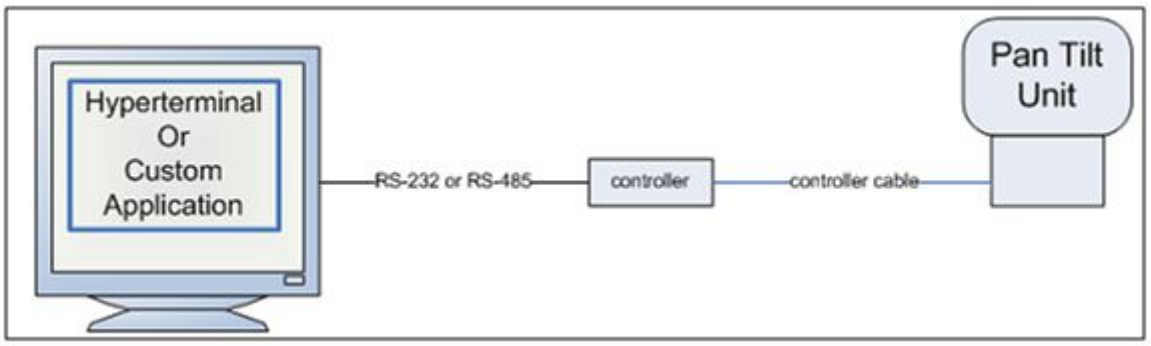

FIG. 4.2 – Schéma explicatif de contrôle par commande ASCII

Commandes de base de contrôle de PAN-TILT
Dans cette partie on se contente de donner une idée à propos les commandes de base de contrôle de PTU et on précise la syntaxe utilisée :

– Commandes de contrôle de la position absolue :

Permettent de spécifier ou demander la position actuelle suivant les axes PAN et TILT par rapport à la position initiale.
TP/PP$\prec$ *delim* $\succ$: demander la position absolue actuelle TILT/PAN
TP/PP$\prec$ *position* $\succ$: donner la position absolue désirée TILT/PAN

– Commandes de contrôle de la position relative :

Permettent de spécifier ou demander la position relative du PTU suivant les deux axes par rapport à la position actuelle.
TO/PO$\prec$ *delim* $\succ$ **:** demander la position relative actuelle TILT/PAN
TO/PO$\prec$ *position* $\succ$ **:** donner la position relative désirée TILT/PAN

– Commande pour avoir les positions limites :

PN* Minimum PAN position : -3062
NX* Maximum PAN position : 3062
TN* Minimum TILT position : -836
TX* Maximum TILT position : 451

Contrôle par du code binaire :
En plus du format ASCII, le Pan-TILT acceptera les formes binaires des commandes. Ces formats binaires sont soutenus par l'intermédiaire de notre bibliothèque d'interface de langage C (PTU-CPI) qui est fournie en tant que code source de la norme ANSI C il peut être compilé dans l'application sur beaucoup de plates-formes de calcul (CPU/OS). Permet l'exécution de plus de 60 commandes/ seconde et il est recommandés pour des applications de haute performance, généralement pour application en temps réel. Le schéma suivant illustre bien ce modèle :

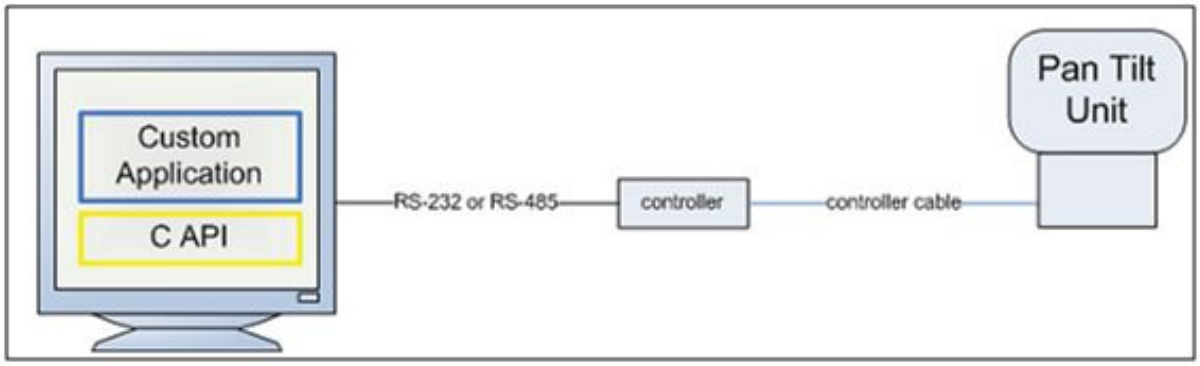

Fig. 4.3 – Schéma explicatif de contrôle par commandes binaires

Pour ce mode de contrôle de PAN-TILT il faut fixer les paramètres de configuration telle que la vitesse de transmission, le nombre des bits par trame, le format des données etc.. au moment d'établissement de la connexion entre le terminal et le PTU.

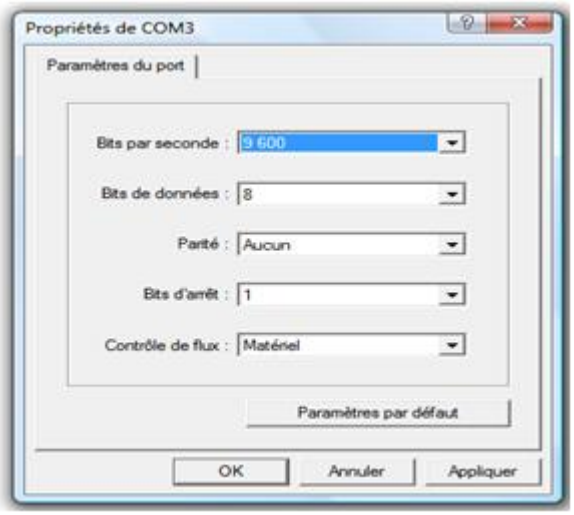

Fig. 4.4 – Interface des caractéristiques du port COM

Au moment de la connexion au port COM spécifié la transmission des données ne peut se faire qu'après avoir donner ses paramètres pour configurer le port série choisi. L'interface de l'Hyper Terminal représenté ci-dessus montre bien les valeurs à précisées pour l'initialisation du port com3.

4.5 Méthode pour de suivie des visages

La tourelle est fournie avec des sources qui contiennent les commandes de bases ainsi qu'un programme permettant de tester ses performances présent pour différents systèmes d'exploitation Windows, Unix etc. Elle sera donc pilotée grâce à un contrôleur (-D46) et un PC via une liaison série RS-232. Le programme de test nous donne à son exécution les positions limites de la tourelle (PAN-TILT) ainsi que quelques autres informations comme la vitesse de transmission des données, le format des données ainsi que le numéro de port utilisé.

4.5.1 Etude des limites du matériel : [12]

Déplacement horizontale :

On remarque qu'au cours du déplacement horizontal nous aurons un angle mort où le PAN-TILT ne pourra pas y aller donc le balayage de toute la zone (360°) par la tourelle est impossible ce qui constitue un problème qu'on doit résoudre. C'est ce que montre le schéma suivant :

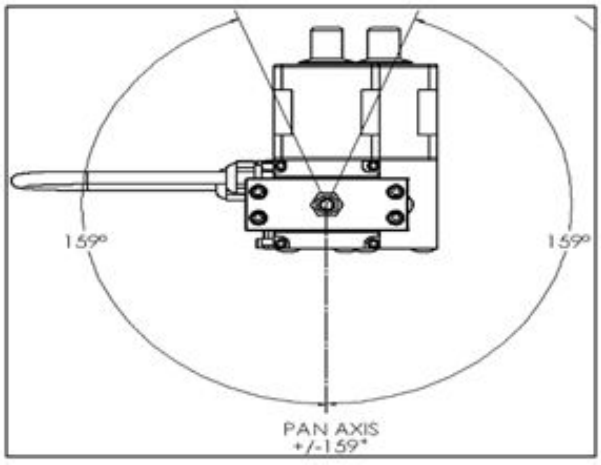

Fig. 4.5 – Vue de dessus de PAN-TILT

Ces valeurs sont calculées suivant les données existantes dans le manuel du PAN-TILT
Les valeurs obtenues après le test sont :

- Max Pan : 3063
- Min Pan : -3062
- Max Tilt : 541
- Min Tilt : -836

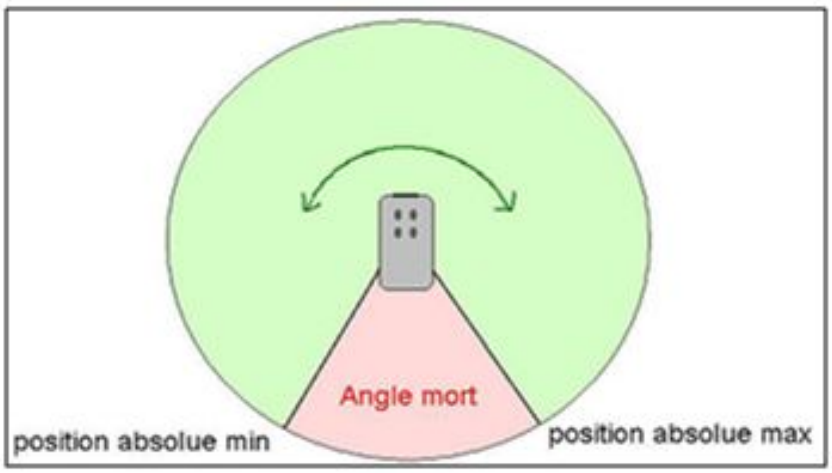

Fig. 4.6 – Schéma de l'angle mort de la tourelle

L'angle mort trouvé est de 43°.
Dans ce cas on doit passer pour mesurer l'angle d'ouverture de la caméra CREATIVE qui va être fixée sur la tourelle (PAN-TILT) pour pouvoir compenser cette partie inaccessible.

Mesure de l'angle d'ouverture

Une simple manipulation sera faite pour pouvoir déterminer l'angle d'ouverture de la caméra utilisée et cette méthode est décrite dans le schéma suivant (Fig 4.7) :
b : connu (une longueur mesurée sur un mur).
c : connu (distance entre le mur et la caméra).
Il est devenu possible avec une simple opération mathématique d'obtenir une valeur arrondie de l'angle d'ouverture **a** de la caméra. On obtient un angle d'ouverture $a = 47°$.
L'angle mort est donc de 43°. Cela peut sembler beaucoup, mais cette contrainte technique est en partie compensée par l'angle d'ouverture que possède la caméra (47°). Il n'y a pas donc de régions totalement inaccessibles mais il se peut que le système rencontre des difficultés de détection et de suivi lorsque la cible se trouve dans les positions extrêmes.

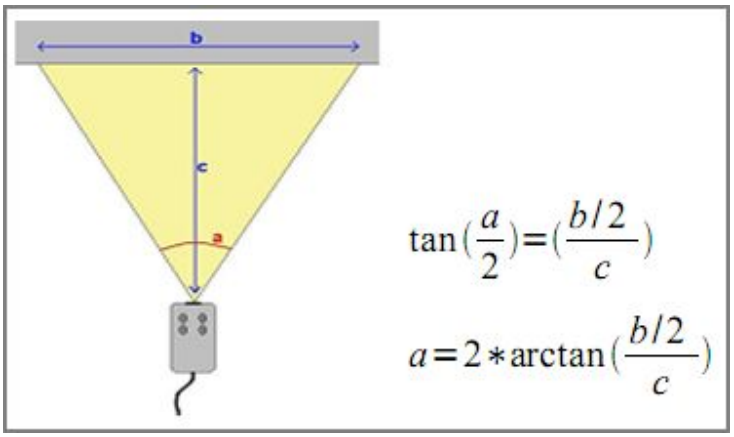

FIG. 4.7 – Méthode de mesure de l'angle d'ouverture de la caméra

Déplacement vertical

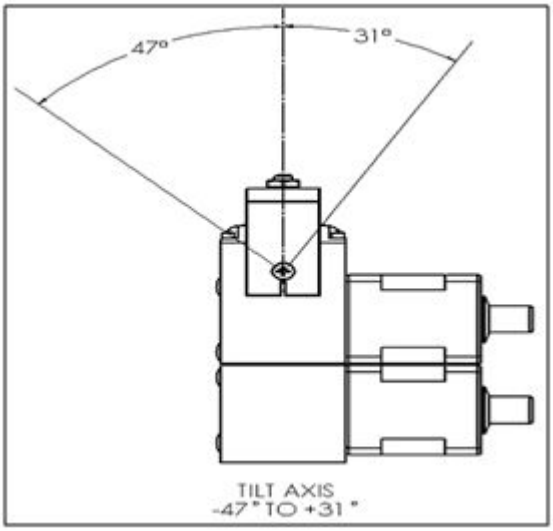

FIG. 4.8 – Vue de gauche de la tourelle

Les mêmes caractéristiques sont disponibles pour les déplacements verticaux " TILT ", mais ceci ne va pas être un grand problème car les images doivent être frontalement prises pour que les visages seraient détectables,(contrainte de Face Detection) donc le mouvement suivant le TILT ne va pas atteindre les positions maximales. (Voir les limites de la méthode détection des visages).

4.5.2 Principe de la méthode de suivi

Ajustement du programme " Face Detection " : Pour faire le programme de suivi on doit faire quelques modifications sur la fonction de " face detection " tel que la restriction de nombre des visages détectés à un seul uniquement qui va être suivi par la suite et on choisit de déterminer le visage par un cercle qui va être identifié par son centre et son rayon proportionnel à la surface détectée (nombre de pixels qui représentent le visage dans la photo).

Dans la situation où plusieurs visages se présentent dans une même image, le visage signalé ou détecté est celui qui présente la plus grande valeur de corrélation et comme la valeur de corrélation est proportionnelle à la surface sur laquelle le visage est représenté (nombre de pixel ayant la couleur de la peau) alors l'algorithme détectera le plus grand visage qui répond aux autres conditions comme la position, la couleur de la peau, l'orientation etc.
Les paramètres :
Les paramètres d'entrés de cette méthode sont eux même les paramètres de sortie du programme de détection des visages et qui correspondent au coordonnés du centre de visage détecté qui doivent être dans les intervalles suivants :
X et Y sont deux entiers tel que :$X[0..640]etY[0..480]$. La position du centre de l'image est :

- Xcentre=Xmax/2= 320
- Ycentre=Ymax/2=240

NB : les coordonnées du centre de visage ne peuvent pas être très proches des valeurs extrêmes (0, 480, 640) car le visage ne peut pas être détecté dans ce cas là puisqu'il n'est pas totalement présent dans l'image et il ne peut pas être identifié comme visage par l'algorithme " Face Detection " et ceci correspond à une des limites de cette méthode.
Le suivi :
On désire suivre le visage dans son mouvement aléatoire soit qu'il se trouve dans le champs de vision de la caméra ou non donc on doit le garder toujours dans la zone de couverture de la loupe ou plus précisément au milieu de l'angle de détection de notre caméra CREATIVE ct6840 utilisée. Dans ce cas on doit faire bouger la tourelle dans les différentes directions (axes de libertés du PAN-TILT) pour qu'elle garde toujours le centre de visage détecté et le centre de l'image presque confondus. Ce qui permet de déduire brièvement que le principe consiste à appliquer un asservissement de la tourelle dans le but de superposer le centre fixe de l'image et le centre du visage.

4.5.3 Les étapes de la méthode

Calcule de l'écart par rapport au centre

On doit calculer l'écart entre la position du visage (Centre.X, Centre.Y) et le centre fixe de l'image qu'on suppose qu'il correspond au point O (320, 240) tel que :

- EcartX=Centre.X-320
- EcartY=Centre.Y-240

Ceci est bien expliqué dans le schéma suivant :(Fig 4.9)

- Si EcartX $\succ$ seuilX : on doit faire une correction en faisant bouger la caméra suivant l'axe horizontale de la tourelle.
- Si EcartY $\succ$ seuilY : on doit faire une correction en faisant bouger la caméra suivant l'axe verticale de la tourelle.

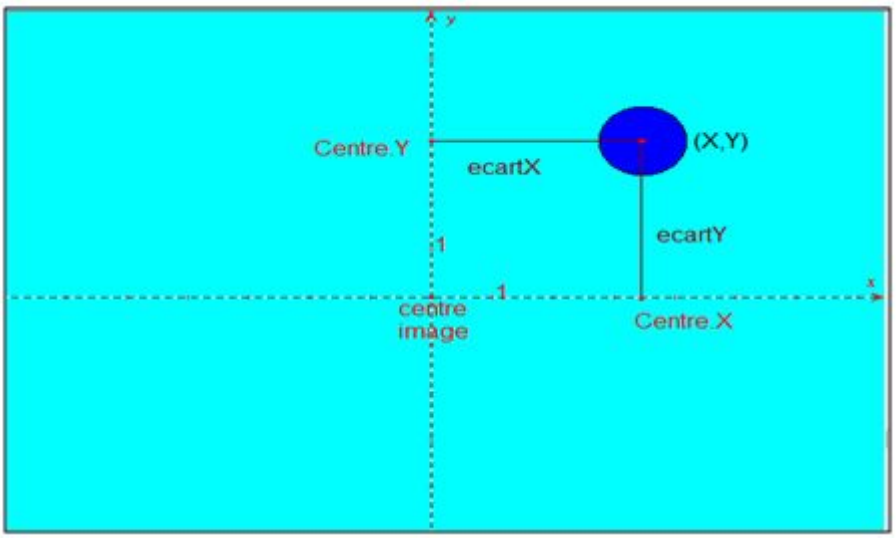

FIG. 4.9 – Principe de la méthode de suivie

Détermination du sens du mouvement

Le sens du mouvement de la tourelle va être déterminé suivant le signe de l'écart calculé entre les coordonnés du centre de visage détecté et le centre fixe de l'image :

- EcartX $\prec 0 \rightarrow SensX = 1$.
- EcartX $\succ 0 \rightarrow SensX = -1$.
- EcartY $\prec 0 \rightarrow SensY = 1$.
- EcartY $\succ 0 \rightarrow SensY = -1$.

Pour l'affectation des valeurs aux sens des déplacements on doit tenir compte de l'effet miroir qui se présente dans la fenêtre résultat au moment de la capture des images par la caméra ce qui nécessite qu'on inverse le signe de déplacement suivant les deux axes de la tourelle.

Détermination des constantes K et K'

La résolution de la tourelle est de 0,0514 / pas, ce qui fait 19,45 pas par degrés. Pour obtenir l'angle absolu (avec comme repère fixe le socle de la tourelle) il suffit de lire la valeur instantanée de la position de la tourelle (en " pas ") et de multiplier par 19,45. Pour l'asservissement, on sait que l'angle d'ouverture de la caméra est d'environ 47° ce qui correspond à 640 pixels horizontalement et 480 pixels verticalement.
Sachant qu'il faudra déplacer la tourelle de 19,45 pas par degrés souhaités, il suffit de faire la correspondance avec le déplacement souhaité exprimer en nombre de pixels.

$$K = 19.45 \frac{pas}{degre} \frac{47}{640} \frac{degre}{pixel} = 1.5(pas/pixel)$$

$$K' = 19.45 \frac{pas}{degre} \frac{47}{480} \frac{degre}{pixel} = 2(pas/pixel)$$

Calcule des paramètres du PAN-TILT

Dans cette partie on se contente de calculer les 2 paramètres qui vont être donnés au contrôleur pour exécuter ces ordres dans le but de faire bouger la tourelle qui va suivre le visage par la loupe de la caméra fixée la dessus.

- La valeur du pan : $Valpan = |EcartX| * K * SensX$.
- La valeur du tilt :$Valtilt = |EcartY| * K' * SensY$

Les paramètres calculés (valpan & valtilt) vont servir comme des valeurs d'entrés (input) qui vont être exécutées par le PAN-TILT pour ajuster le centre de visage presque au milieu de l'image affichée dans la fenêtre résultat.

Exécution des ordres

Cette étape consiste à faire passer les paramètres calculés dans l'étape précédente **Valpan** (déplacement horizontale) et **Valtilt** (déplacement verticale) à la fonction du contrôleur qui fait bouger la tourelle suivant les deux directions et dans le sens qui convient pour ajuster le centre de visage à sa nouvelle position correspondant presque au centre fixe de l'image.
Le schéma suivant traduit bien le principe de fonctionnement de notre application de détection et suivie des visages.

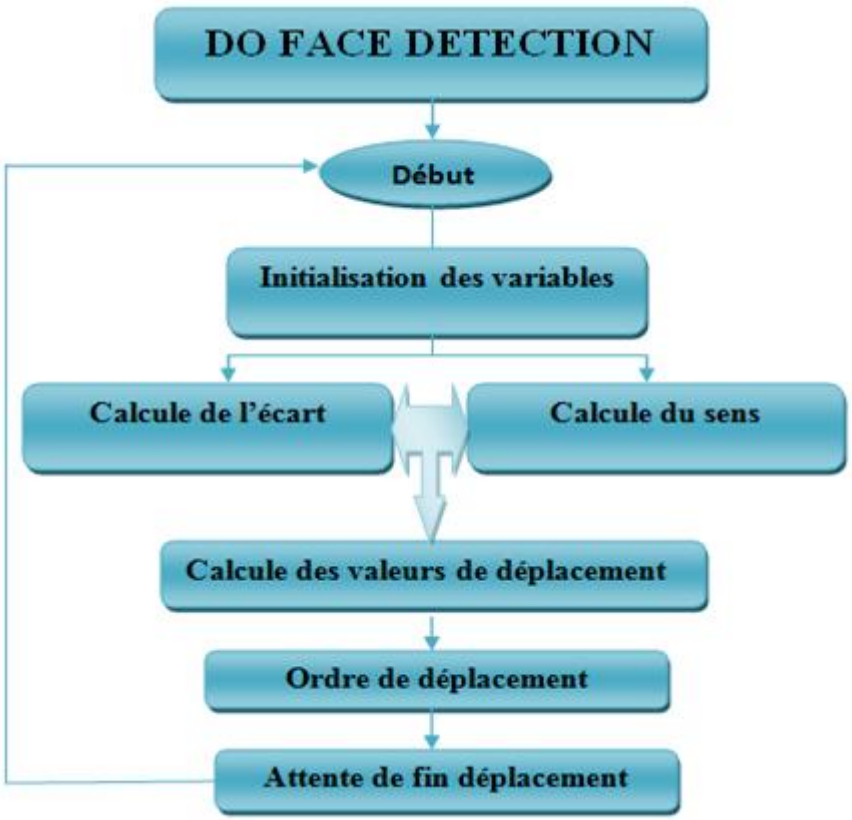

Fig. 4.10 – Les étapes de l'application de détection

4.6 Implémentation de la méthode

Dans le but d'implémenter un programme C++ qui fait le suivi du visage par la tourelle on choisit de se baser sur le programme de test de PAN-TILT qui se trouve aussi avec le manuel d'utilisation et la fiche technique de ce dernier. Pour résoudre notre problème on utilise les fonctions prédéfinies dédiées à faire bouger la tourelle dans ses différents sens et directions.

4.6.1 Les principales fonctions à utilisées

Les fonction d'asservissement [12]

– Set_desired(paramètres)

On peut dire que la fonction suivante suffit dans le projet pour réaliser toutes les actions nécessaires et les mouvements suivant les axes de PAN-TILT c'est-à-dire faire bouger la tourelle horizontalement et verticalement et régler sa vitesse et son accélération au cours du déplacement.

char **set_desired** (char axis, char kinematic_property, PTU_PARM_PTR *value, char movement_mode)		
PARAMETRE	**VALEUR**	**COMMENTAIRES**
axis	PAN (ou TILT)	Sélectionne un axe
kinematic_property	POSITION / SPEED	Sélectionne un paramètre d'action
value	(3090/-3090) / (0/3000)	Valeur à donner au paramètre
movement_mode	ABSOLUTE / RELATIVE	Mode de mouvement

– Get_desired(paramètres)

Dans le programme implémenté on a besoin d'avoir quelques informations concernant le PAN-TILT pour analyser son fonctionnement, la fonction suivante nous permet d'obtenir en temps réel les différents paramètres. Dans notre cas elle est utilisée pour obtenir à tout instant la position absolue de la tourelle. La valeur retournée sera fonction des paramètres entrés.

long **get_desired** (char axis, char kinematic_property)		
PARAMETRE	**VALEUR**	**COMMENTAIRES**
axis	PAN (ou TILT)	Sélectionne l'axe
kinematic_property	POSITION	Paramètre à obtenir

– Await_completion()

Dans notre programme la vitesse de détermination des coordonnées du centre de visage est supérieure a la vitesse d'exécution des ordres pour bouger la tourelle c'est pour cette raison qu'on utilisera dans certains cas cette fonction qui bloque en quelque sorte le système tant que le mouvement n'est pas terminé. C'est à dire on ne passe d'une commande de déplacement à une autre que si la précédente est achevée.

```
char await_completion (void)
```

La fonction initialisation du port série

Cette fonction est intitulée " **Init_port_Com()** ", elle correspond à la première étape du programme implémenté elle consiste exactement à faire : En premier lieu, choisir le port série pour la communication de la tourelle avec l'ordinateur utilisé. Dans notre cas on utilise le port "$COM1$"
En second lieu, définir la vitesse de transmission (échange) des données qui se calcule en nombre des bits transmis par seconde (bits/sec). Pour échanger des donnés entre la tourelle et l'ordinateur utilisé on choisit la vitesse de $9600 bits/sec$.
Ces paramètres sont donnés par l'utilisateur au moment de l'exécution du programme au système qui commence par les vérifiés puis il initialise le port correspondant et le réserve uniquement pour commander le PAN-TILT.

La fonction de "Face Detection "

Cette fonction traduit bien le fonctionnement de l'algorithme de détection des visages présenté dans le chapitre 2 et qui se base sur l'approche de détection hybride : couleur de la peau et l'appariement des gabaries, comme son nom l'indique ce programme se contente de faire la détection des visages et il retourne les coordonnés du centre du plus grand visage détecté après quelques changement des paramètres et des seuils de détections des sous fonctions du programme standard.

4.6.2 Pseudo-codes du programme

– Programme principal (détection et suivi)

```
PROCEDURE Tracking_face ( )
{ x,y entier
        TANT QUE (capture ≠ 0) REPETER
   (x,y)← Face_detection()
   Correction_horizontale (x)
   Correction_verticale (y)
   Fin TANT QUE
   FIN PROCEDURE        }
```

Dans ce qui suit on se contente de détailler le fonctionnement des sous fonctions du programme principale affin de mieux comprendre comment ils ont étaient implémentées.

– Correction suivant l'axe horizontale

```
PROCEDURE Correction_horizontale( entier val, entier k)
 { valpan, ecartx, sensx, k entier
   SI (val >320) Alors
    { Ecartx ← val – 320
    Sensx ← 1 }
        SINON
          { Ecartx ← 320 – val
            Sensx ← -1 }
  Valpan ←ecartx * k* sensx
  Bouger _la_tourelle ( valpan )
  Attendre_fin_deplacement ()
    FSI
  FIN PROCEDURE }
```

– Correction suivant l'axe vertical

```
PROCEDURE Correction_verticale( entier val, entier k)
 { valtilt, ecarty, sensy, k entier
   SI (val >240) Alors
        { Ecarty ← val – 240
          Sensy ← 1 }
          SINON
            { Ecarty ← 240 – val
              Sensy ← -1 }
     Valpan ←ecarty * k* sensy
     Bouger _la_tourelle ( valtilt )
     Attendre_fin_deplacement ()
   FSI
FIN PROCEDURE }
```

4.7 Tests et résultats de l'application

Pour montrer le degré de réussite du notre dispositif à suivre le visage détecté on désire faire quelques testes sur l'application implémentée pour s'assurer de son bon fonctionnement, pour ceci on choisit de déterminer l'erreur en distance par rapport au centre fixe de l'image. Soit :

– **(X, Y)** : les coordonnés du centre de l'image afficher dans la fenêtre résultat
– **(centre.x, centre.y)** : coordonnés du centre de visage.

$$e = \sqrt{(center.x - X)^2 + (center.y - Y)^2}$$

Le temps de la détection est déterminé directement par l'intermédiaire d'une fonction prédéfinie de notre application. Dans ce cas on désire tracer la courbe $\mathbf{e = f(t)}$ pour les deux cas suivants :

– Visage immobile (pas de déplacement) :

Dans cette partie on place le visage dans la le champ de vision de la loupe de web-cam avec un petit décalage par rapport au milieu et on lance le programme de suivi en relevant les valeurs de l'erreur durant 30 secondes dans un fichier.
Le traçage des valeurs enregistrées donne les courbes suivantes :

Résultat1 :

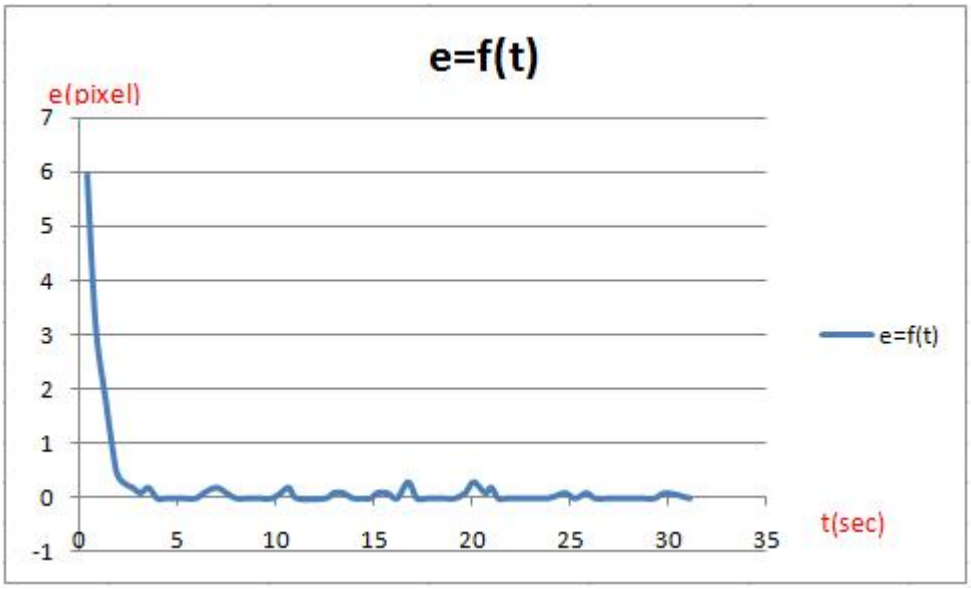

FIG. 4.11 – courbe représentative de variation d'erreur en fonction de temps (visage fixe)

L'allure de cette courbe traduit bien le fonctionnement de l'application de détection et suivi de visage ce qui fait, après avoir détecté la position de visage et calculer l'écart par rapport au centre, le programme donne l'ordre à la tourelle de corriger le décalage trouvé ce qui explique bien l'allure de la courbe dans la première partie [0,4s].

Dans la seconde partie on remarque presque une stagnation de la valeur de l'erreur ainsi qu'une allure de courbe au voisinage de 0 ce qui s'explique par l'immobilisation du visage détecté.
La légère variation de la valeur de l'erreur autour de 0 s'explique par la stabilité partielle du visage à cause du mouvement respiratoire de l'être humain ainsi qu'aux erreurs de détection.

Résultat2 :

Pour les mêmes conditions si on refait l'expérience une autre fois pour confirmer le résultat obtenu, la courbe (Fig. 4.12) garde son allure pour la première partie mais on remarque dans la deuxième partie qu'elle change en notant l'apparition d'un pic au milieu du temps de détection ±15secondes malgré que le visage reste stable et la personne ne bouge pas.

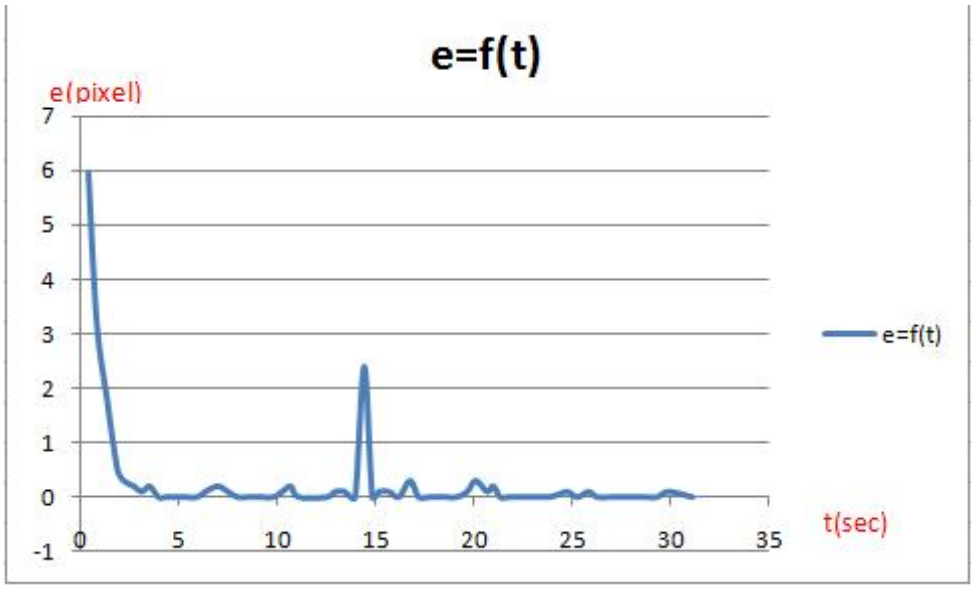

FIG. 4.12 – Courbe montrant l'apparition d'un faux positif lors de l'asservissement

Cette crête apparue au milieu de la courbe ne correspond pas à la valeur exacte de la position du visage car on est sûr qu'il est stable au moment du prélèvement des valeurs, mais elle s'explique par une limite de l'algorithme de détection qu'on appelle les faux positifs et elle correspond à un objet détecté ayant une valeur de corrélation proche de celle des visages ce qui le rend détectable.

– Visage mobile (déplacement aléatoire) :

Dans cette expérience on désire mesurer l'erreur par rapport au centre fixe lorsque le visage est en mouvement aléatoire dans toutes les directions mais en essayant de le garder toujours en face de notre dispositif de détection.

On se contente de ne pas faire des mouvements brusque pour minimiser la perturbation du programme de détection affin de vouloir obtenir des valeurs significatives.

En analysant l'allure de la courbe trouvée(Fig. 4.13) on remarque que chaque pic est suivi d'une valeur proche de 0 ce qui traduit bien le principe de fonctionnement de la tourelle (PAN-TILT) qui consiste à ramener le visage détecté toujours au centre de la fenêtre.

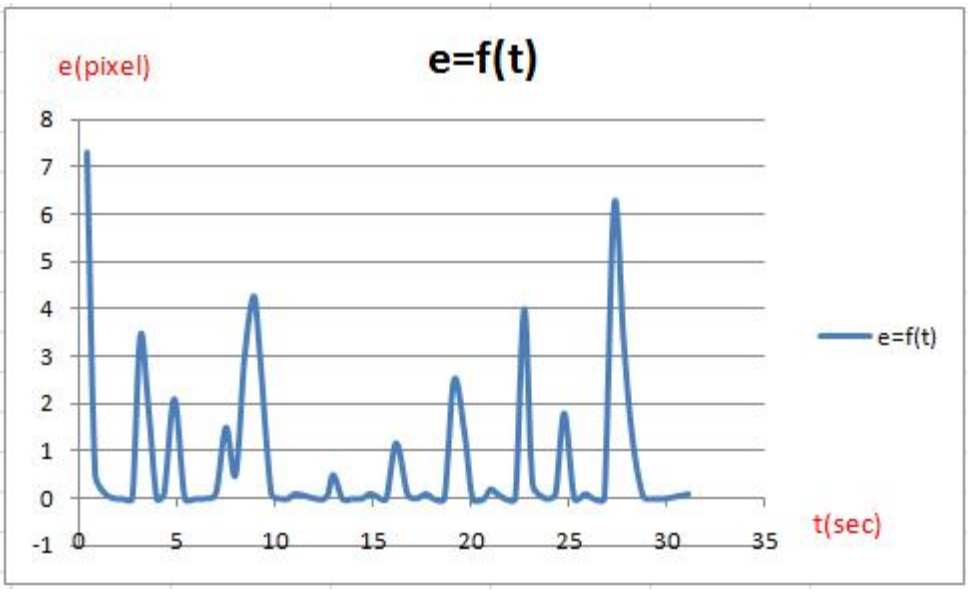

FIG. 4.13 – Courbe représentative de la variation d'erreur en fonction de temps (visage en déplacement)

4.8 Optimisation du programme

Après avoir analysé les résultats de la méthode de correction proportionnelle on remarque une instabilité qui est due à une sorte d'oscillation du PAN-TILT autour de la valeur du centre fixe de l'image, pour résoudre ce problème on propose l'approche suivante.

4.8.1 Dead zone

Dans cette partie on se contente d'améliorer le fonctionnement du PAN-TILT pour le rendre mieux stable au moment de l'asservissement ou correction de la position de visage dans la fenêtre de détection ce qui permet de suivre le mouvement du visage.
Pour ceci on désir au lieu de garder le visage au centre fixe et exacte de l'image de tolérer son existence dans une zone centrée au milieu en appliquant des seuils de déclenchement de la correction suivant les deux axes du mouvement dans le but de minimiser les petits mouvements de la tourelle pour la rendre plus stables et éliminer les oscillations. Le schéma(Fig 4.14) présenté ci-dessous explique bien cette approche.

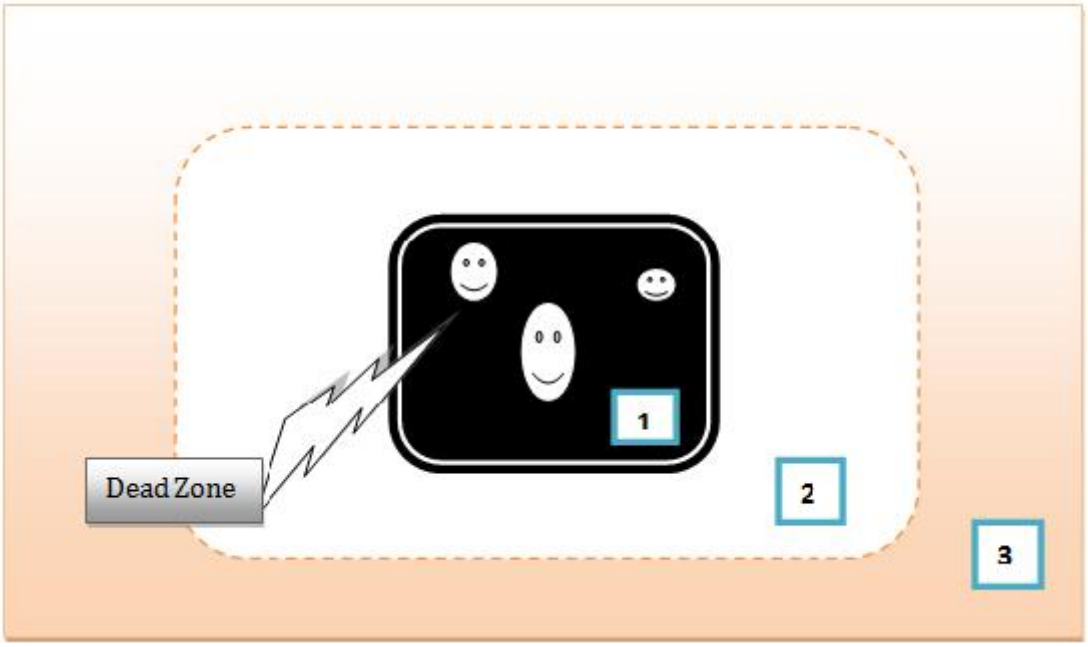

FIG. 4.14 – approche de la zone morte

On remarque que l'image est devisée en 3 zones :

- **Zone 1 :**

Si la position de visage détectée est dans cette zone on est dans un cas stable ou la tourelle ne doit pas bouger et elle n'effectue aucun mouvement de correction et le visage reste toujours dans la fenêtre de détection.

- **Zone 2 :**

Si la prochaine valeur détectée correspond à une position de la zone 2 (le visage vient juste de sortir de la zone 1) alors la tourelle va intervenir dans ce cas en ramenant le visage au centre de l'image qui correspond dans ce cas à la zone morte, pour l'empêcher de sortir du champ de vision de la loupe de la caméra.

- **Zone 3 :**

Dans le cas ou la prochaine valeur détectée passe brusquement de la zone 1 à la zone 3 ce qui confirme bien qu'il ne s'agit pas d'une vraie détection mais un fausse car cette position n'est pas probable puisque dans la position précédente le visage étais juste au milieu de l'image. Dans ce cas cette valeur est rejetée et n'est pas asservie par la tourelle c'est à dire qu'on applique un filtre qui ne tient compte que de la valeur précédente pour éliminer les faux positifs (fausse détections).

Tests et résultats :
Après avoir implémenté cette approche on remarque au moment de l'exécution du programme que le (PAN-TILT) devient plus stable ainsi que les vibrations autour de la valeur du centre sont éliminées et que l'asservissement des faux positifs écartés est très réduits.

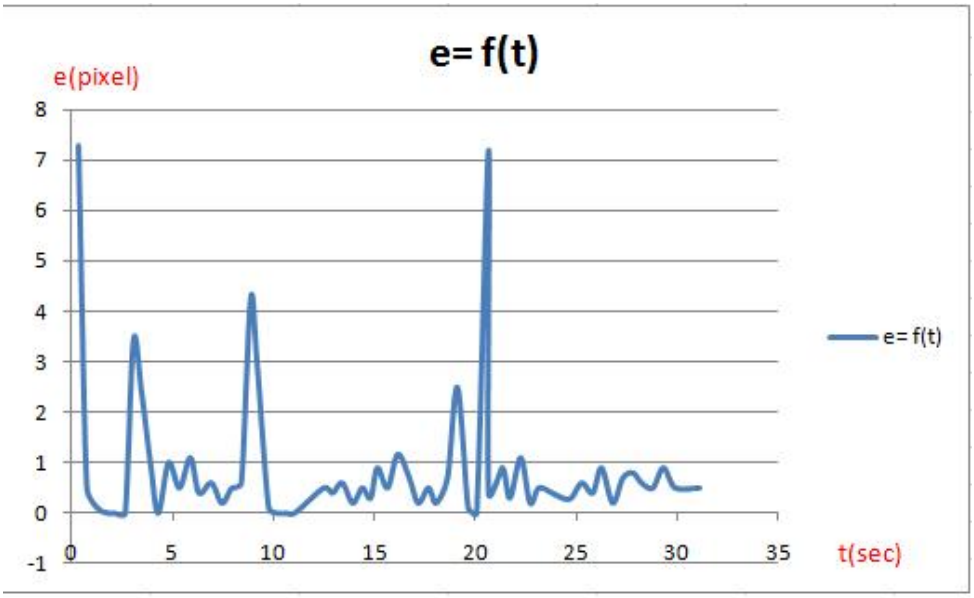

FIG. 4.15 – Courbe d'analyse des résultats de la zone morte

Cette courbe (Fig. 4.15) traduit bien le fonctionnement de l'approche " Dead zone " en effet les corrections sont faites toujours par rapport au centre de l'image et en analysant l'erreur en fonction du temps durant une période de 30 seconde on déduit :

- [2.5..5s] : le programme fait la correction en ramenant le visage au centre fixe (erreur =0)
- [5..8s], [11..17s]et[21..31s] : on remarque une légère variation de l'erreur dans une marge bien déterminée et que la tourelle n'intervient pas pour faire une correction de la position ce qui illustre le principe de cette approche.
- 20s : on remarque la détection d'un faux positif mais le programme ne le prend pas en considération puisque il n'a pas pris du temps pour faire l'asservissement.

4.8.2 Aspect intégrative

On désire dans cette partie mieux optimiser le fonctionnement du dispositif en améliorant l'asservissement visuel avec la tourelle (PAN-TILT) en essayant d'éviter le traitement des fausses valeurs détectées de la position de visage par le programme de " Face Detection "qui sont dues soit à la nuance des couleurs du fond soit à la basse luminosité ou à des images de jour, ce qui présentent beaucoup des perturbations pour la distinction entre visage ou autre forme (non visage).
Pour résoudre ce problème on désire implémenter l'aspect intégrative dans notre application de détection et de suivi de visage.
Le but :
Le but de cette méthode consiste à introduire la notion du passé pour prévoir la position future probable du visage et déterminer si la position affichée correspond à une vraie ou fausse détection (faux positif) pour améliorer la qualité de l'asservissement.

Principe de la méthode :
On choisit de tenir compte des 10 dernières positions affin de prévoir la position suivante, pour mettre en oeuvre cette méthode on procède comme **suit :**
Implémenter une sorte de mémoire qui va contenir à chaque instant t les 10 dernières positions détectées de visage (liste chainée circulaire) où chaque valeur qui arrive, fait sortir la plus ancienne. Stratégie de FIFO (First In First Out).

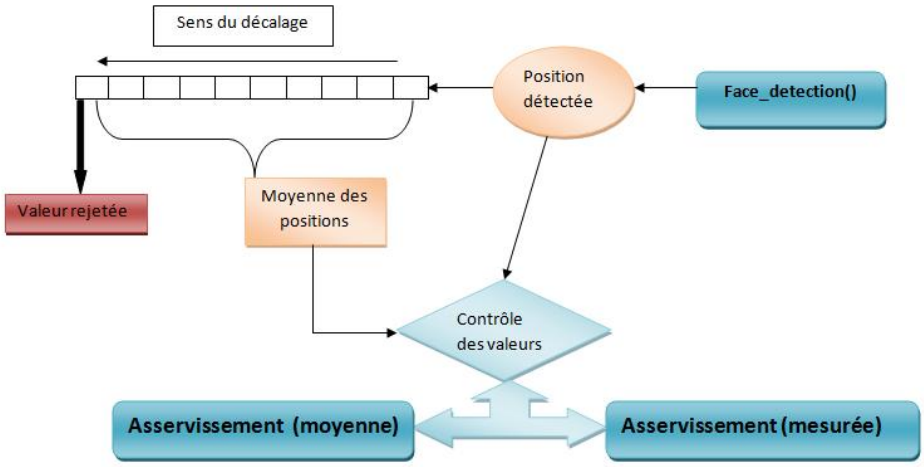

FIG. 4.16 – Principe de la méthode intégrative

Le schéma (Fig 4.16) traduit bien le fonctionnement de cette approche implémentée qui s'explique par les étapes suivantes :

- Détection de la position par " Face Detection ".
- Décalage des valeurs de la mémoire réservée.
- Insertion de la position détectée à la dernière position.
- Calcule de la moyenne des valeurs enregistrées en mémoire
- Comparaison avec la valeur mesurée :

Si**(mesurée- moyenne)** $\succ$ **Seuil** :
$\rightarrow$ Asservissement avec la valeur **calculée (prévue)** :
Rejeter la valeur mesurer car elle correspond à un faux positif.
Sinon $\rightarrow$ Asservissement avec la valeur **mesurée**.
Cette valeur peut probablement être une position suivante du visage .

Tests et résultats :
En exécutant le programme après avoir effectué ces changements on remarque que dans la fenêtre Result (Fig 4.17)'il s'agit bien d'une fausse position trouvée qui ne correspond pas à un visage, dans ce cas le programme doit rejeter la valeur détectée de la position affichée et il ne la prend pas en considération pour l'asservissement mais il exécute la correction avec la valeur estimée qui est calculée à partir des valeurs des 10 dernières positions qui résident en mémoire.

Les deux figures (Fig 4.17 et Fig 4.18) expliquent bien le résultat obtenu et ce qui a été expliqué :

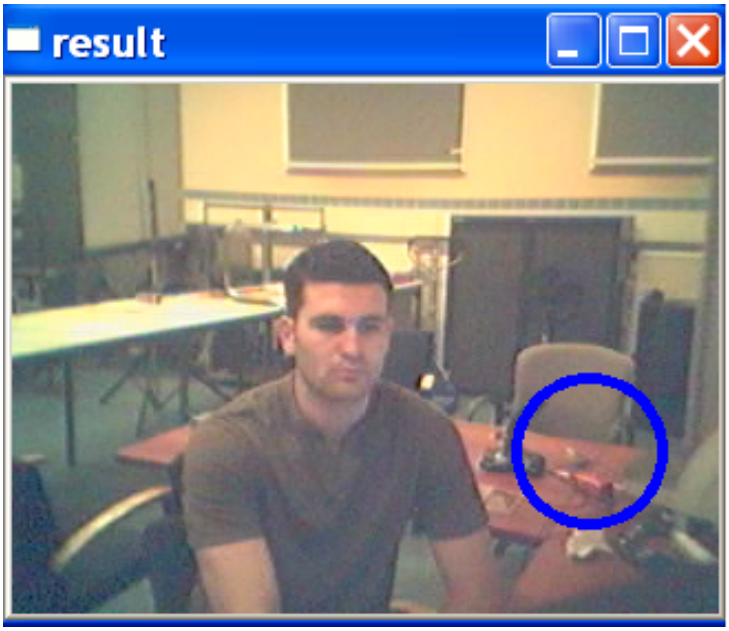

FIG. 4.17 – détection d'un faux positif

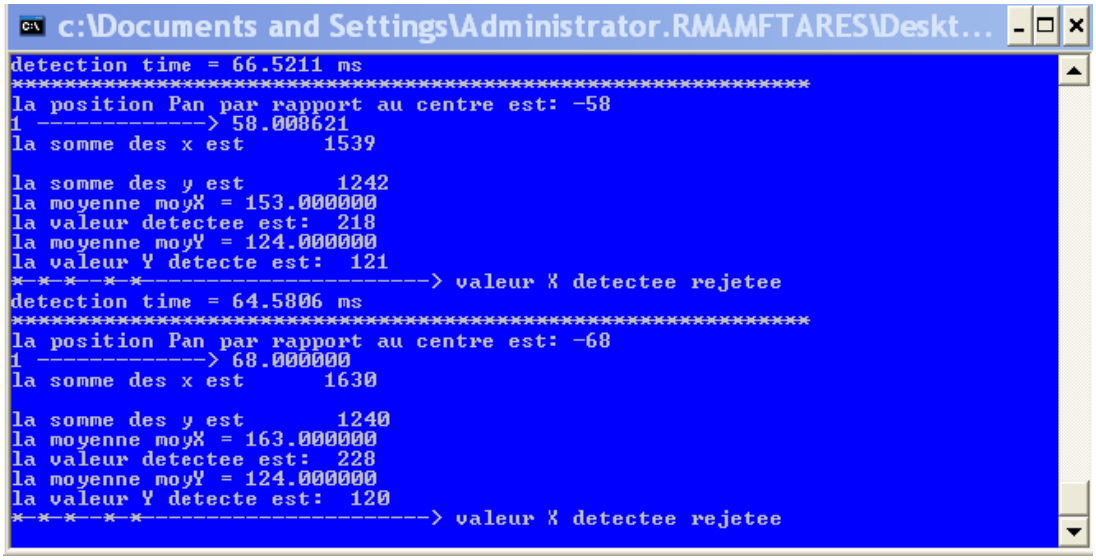

FIG. 4.18 – Réponse du programme pour la fausse détection

Pour ce résultat on remarque que le programme néglige la valeur de X détectée (218) car elle est trop écartée de la valeur estimée (153) en fonction des 10 dernières valeurs, ce qui confirme qu'il s'agit bien d'une fausse détection. Cette capture d'écran de l'interface d'exécution confirme le résultat obtenu et illustre bien le fonctionnement.

Avec cette approche on garantit que l'asservissement ou la correction de position ne se fait que lorsque on est sûr à 80% que la détection correspond bien à un vrai visage détecté et que la position trouvée est probable la chose qui rend le mouvement du PAN-TILT plus fluide ce qui garantie le bon suivi ou ce que on appelle dans ce cas le Tracking car on prévoit la position probable du visage dans le futur en prenant en considération le mouvement effectué dans le passé.

4.9 Conclusion

Dans ce chapitre on a implémenté un programme en C++ qui permet de faire le suivie en temps réel des visages humains détectés en utilisant une caméra et la tourelle (PAN-TILT),Ce programme fonctionne correctement dans des conditions bien précises de détection et en cas d'absence des perturbations extérieures tel que la forte ou basse luminosité qui cause la détection des faux positifs qui le rend instable ainsi qu'on à réussi à résoudre ce problème en introduisant en premier lieux le principe de " Dead Zone " et en second lieu une approche intégrative qui permet de prévoir la prochaine position probable du visage et d'éliminer les mauvaise détection ce qui amène à avoir un bon asservissement visuel (Visual Servoing). . Avec cette application on a réussi à atteindre l'objectif fixé au début du projet mais il peut être encore optimisé.

5. Conclusion

Ce rapport de fin d'etudes explique et détaille le fonctionnement de la tourelle (PAN-TILT) ainsi que les étapes suivies pour aboutir à la réalisation de l'application intitulée " **détection et suivi de visage en temps réel** " qui consiste à implémenter un programme pour suivre la cible"visage" par une caméra en se basant sur les commandes de teste de la tourelle et le programme de détection des visages étudié.

Il faut noter que les objectifs tracés dès le début sont atteints puisque le PAN-TILT suit bien la cible détectée indépendamment des différentes conditions et des objets dans le milieu extérieur ainsi qu'on a ajouté deux approches permettant d'améliorer la performance de l'asservissement visuel en filtrant les fausses valeurs détectées et en empêchant le programme de les prendre en considération au moment de l'asservissement, ce qui a permis d'avoir aussi un déplacement plus fluide au tour des deux axes de la tourelle.

Il faut ajouter aussi que le programme implémenté n'est pas la solution optimale pour faire l'asservissement et le suivi de visage car on peut encore implémenter l'approche proposée dans la partie futur travail mais le facteur temps ne m'a pas permis d'atteindre ce stade.

Cette application parait très utile dans le futur car elle peut être combinée avec le programme " **Go to Goal** " déjà implémenté sur le robot qui existe dans le laboratoire MECA en faisant une combinaison entre plusieurs types d'asservissement et on travaillant sur des cibles multiples.

Ce projet me parait très intéressant sur de nombreux points, tout d'abord il m'a permit de traiter des multiples sujets dans des domaines différents comme la commande d'un appareil via le port série ou encore l'analyse d'un flux d'image en continue ce qui constitue un grand apport pour ma formation personnelle car il m'a donné l'occasion aussi de toucher de prêt le domaine de traitement d'image et de rentrer en profondeur dans l'environnement des langages de programmation ainsi que d'être en face des vrais problèmes dont on doit évoquer une solution et de l'optimiser en tenant compte aussi du facteur temps.

6. Perspectives : (Future Work)

Dans l'application implémentée, pour faire l'asservissement et filtrage des valeurs correspondant à des fausses détections on ne prend en considération que les positions détectées du passé par le programme de " Face Detection ", comme proposition pour mieux améliorer les performances du programme dans le futur on doit tenir compte aussi des commandes (ordres) du PAN-TILT qui viennent juste de s'exécuter dans les derniers instants pour prévoir le sens du déplacement ce qui permet de mieux estimer la position probable du visage.

Pour inclure cette approche on propose d'intégrer le filtre de Kalman (**KALMAN FILTER**) dont le principe se base sur un fonctionnement purement récursive ce qui permet de l'exploiter pour déterminer ou mieux estimer la position du futur en tenant compte du passé de plusieurs facteurs (positions du visage, sens du déplacement précédent, axe du déplacement..) simultanément.

On peut conclure qu'avec cette approche on peut prévoir la meilleure prochaine position probable du visage avec beaucoup plus de précision car ce principe intègre plusieurs facteurs qui affectent la qualité de l'asservissement visuel fait par la tourelle, la chose qui permet d'optimiser la fonction de Tracking.

Pour tout ce qui concerne le programme de détection des visages on peut l'exécuter avec une autre caméra plus performante ce qui permet d'utiliser le principe de zoom lorsque on détecte le visage qu'on désir suivre, ce qui améliore la qualité de détection et diminue la surface qui peut faire apparaitre des fausses détections.

7. Annexe

De l'espace RVB à l'espace Lab :[15]
La première étape consiste à passer des composantes RVB aux composantes XYZ. On utilise pour cela une matrice de conversion. Ensuite, il s'agit de passer de l'es-

$$\begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \begin{bmatrix} 0.618 & 0.177 & 0.205 \\ 0.299 & 0.587 & 0.114 \\ 0 & 0.056 & 0.944 \end{bmatrix} . \begin{pmatrix} R \\ V \\ B \end{pmatrix}$$

pace XYZ à l'espace Lab. On utilise alors les formules de conversion suivantes :

$$L = 116(\tfrac{Y}{Y_n})^{1/3} - 16 \qquad \text{pour } fracYY_n \succ 0.008856$$

$$L = 903.3(\tfrac{Y}{Y_n}) \qquad \text{pour } \tfrac{Y}{Y_n} \leq 0.008856$$

$$a = 500(f(\tfrac{X}{X_n}) - f(\tfrac{Y}{Y_n})$$

$$b = 200(f(\tfrac{Y}{Y_n}) - f(\tfrac{Z}{Z_n})$$

Oû

$$\text{Pour } t \succ 0.008856 f(t) = t^{1/3}$$

$$\text{Pour } t \leq 0.008856 f(t) = 7.7787t + \tfrac{16}{116}$$

Xn, Yn et Zn correspondent au blanc décrit dans l'espace XYZ. On les obtient pour RGB=(255,255,255).

Fonction initialisation du port COM1 :
Cette fonction est indispensable pour le fonctionnement de notre programme car elle assure la communication entre l'ordinateur utilisé et le mécanisme de tracking.

```
   /********** FONCTION INITIALISATION PORT COM ***************/
void Init_Port_Com()
{
       int COMportNum, BaudRate ;
      char COMportName[256], tmpChar;
      portstream_fd COMstream;
      char COMportPrefix[10] = "COM";

     /* Entré des arguments*/
//numéro de port
 COMportName[0] = ' ';
    while (COMportName[0] == ' ') {
        printf("\nEntrer le %s le numéro de port: ", COMportPrefix);
        scanf("%d", &COMportNum);
        //vérifier le port choisi
                                    }
//vitesse de transmission des données
        printf("\nEntrer la vitesse(default: 9600): ");
        scanf("%d", &BaudRate);
       //vérifié la vitesse choisie

      /* initialisation du port série */

        set_baud_rate(BaudRate);
        COMstream = open_host_port(COMportName);

        if ( COMstream == PORT_NOT_OPENED )
              { printf("\nSerial Port setup error.\n");
                goto abnormal_exit;  }
        printf("\nSerial port %s initialized\n", COMportName);

.............
```

Support de la caméra :
Pour le suivi on a utilisé le PAN-TILT -D46 pour les déplacements et la webcam CREATIVE CT6840 pour l'acquisition des images mais cette dernière ne présente pas une morphologie compatible avec celle de la tourelle pour être facilement fixée la dessus. Pour résoudre ce problème on doit fabriquer un support qui va permettre de donner une position stable à la caméra sur le PAN-TILT ce qui permet de synchroniser son mouvement avec celui de la tourelle.

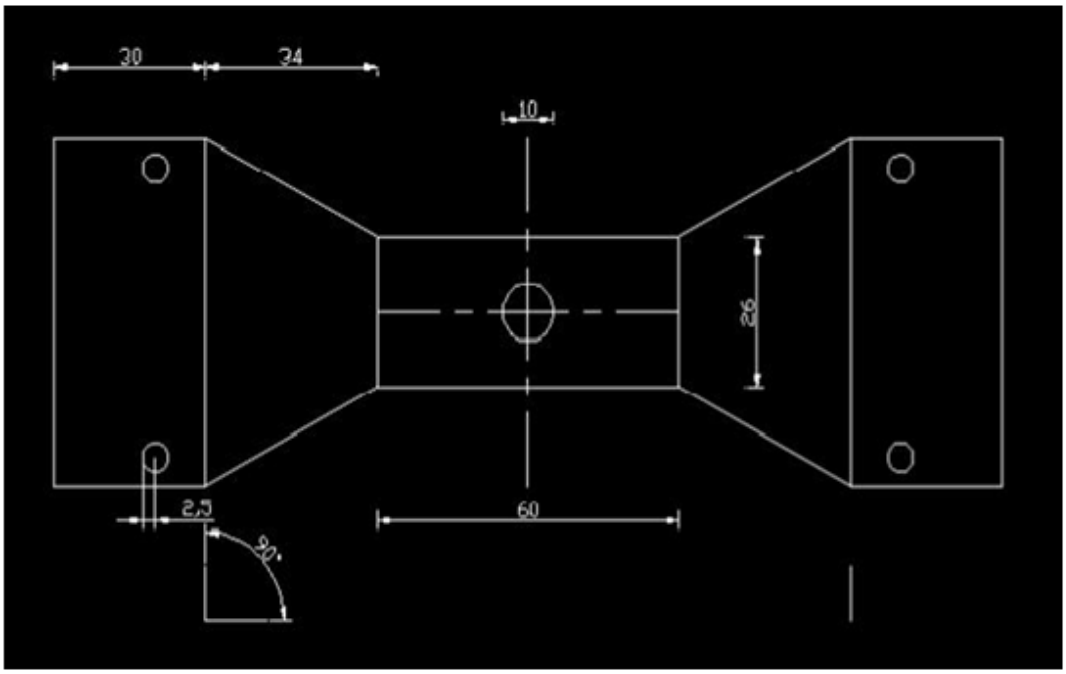

Fig. 7.1 – Schéma technique de support de la caméra construit

Ce schéma représente un model éclaté du support, et pour aboutir la forme désiré on doit plier cette plaque au niveau de toutes les arêtes trouvées d'un angle de 90° comme il est noté sur la figure.

Dispositif final d'acquisition et de suivi :

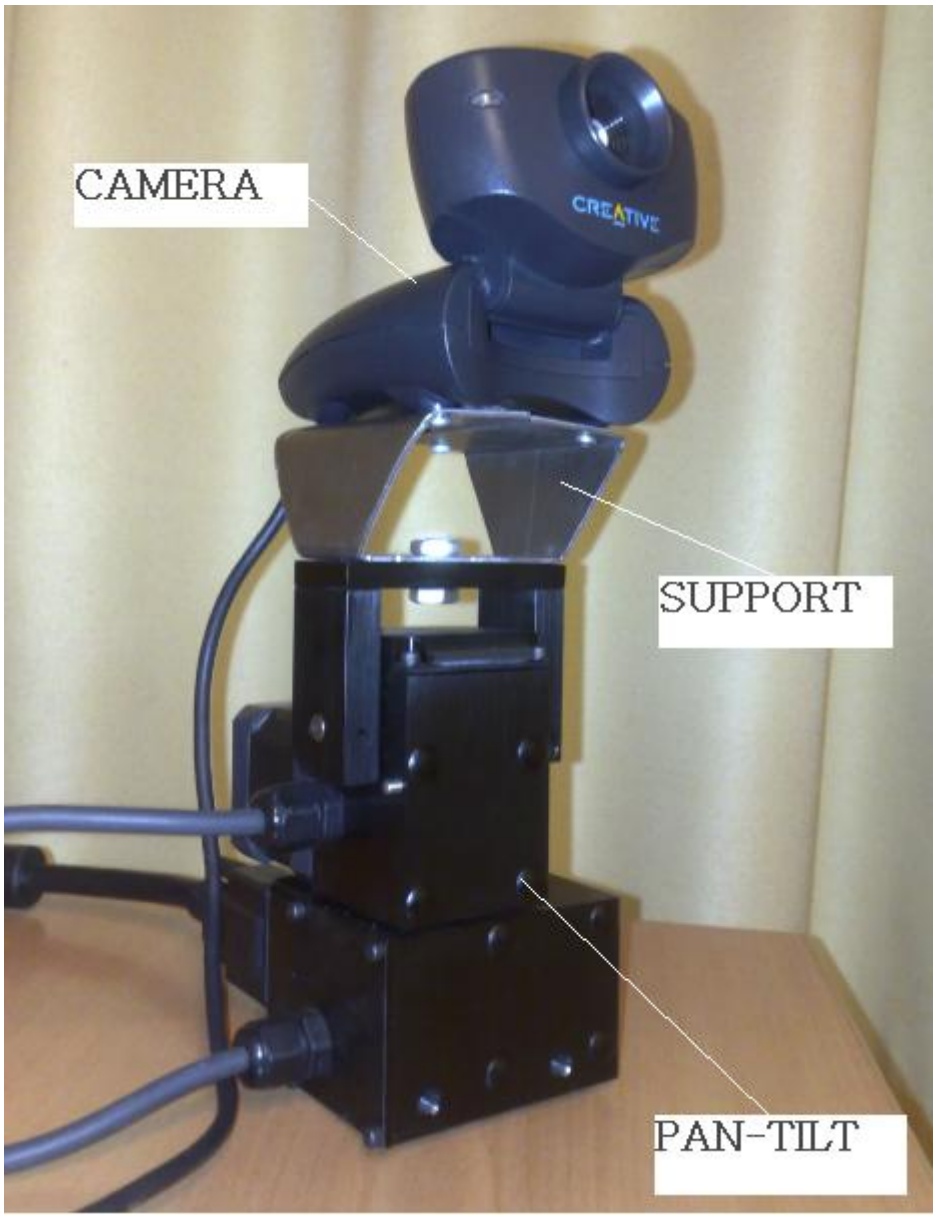

FIG. 7.2 – Dispositif final de détection et suivi

8. *Bibliographie*

[**1**] : **H.Hadi Abdelkader, Y.Mezouar, P.Martinet, F.chaumette** : *Asservissement visuel en vision omnidirectionnelle à partir de droites*

[**2**] : *François Chaumette* IRISR/ INTRA Rennes Campus de Beaulieu ; article : *Avancées récentes en asservissement visuel.*

[**3**] : *Alexandre Krupa* : IRIA Rennes Bretagne Atlantique/IRISA compus universitaire de Beaulieu, article : *Asservissement visuel par imagerie medicale*

[**4**] : *La détection des visages dans une image*, Ecole nationale Supérieure en Informatique (E.S.I)
http ://share.esi.dz/index.php ?option=com_docman&task=doc_download
&gid=248&Itemid=1

[**5**] **OUAFEK REDOUANE, MEGHALSI MOHAMED DEDA** 2008/2009 :
la détection des visages dans une image :
http ://share.esi.dz/index.php ?option=com_docman&task=doc_download&gid=248
&Itemid=1

[**6**] face recognition software in matlab : where is Waldo ?
http ://scien.standard.edu/pages/labsite.2007/psych221/projects/07/face detection

[**7**] : site CREATIVE webcam :
www.clubic.com/article-18466-1-creative-webcam-live-pro.html - France -

[**8**] COMPUTER CONTROLED PAN-TILT UNIT models PTU -46 :
www.suprevision.com.tw/ptu-d46.pdf

[**9**] : *Manuel d'utilisation de la tourelle (PAN-TILT -D46 manual)* :
www.dperception.com/products/ptu-d46/

[**10**] :*Ch. Bencheriet, A/H. Boualleg* & **H. Tebbikh B. Guerzize** & **W. Belguidoum** : Détection de Visages par Méthode Hybride :
Couleur de Peau et Template Matching Open Source Computer Vision Library

[**11**] Swintching to OpenCV2.0 withVS 2005 :
http ://mirror2image.wordpress.com/2009/10/20/switching-to-opencv-2-0-with-vs2005/

[**12**] :LAHOUEL Radia VERMEIL Yohan master professionnel ;
Asservissement d'une caméra sur le centre de gravité d'une cible via une tourelle 2 axes

[**13**] : BENCHERT,BOUALLEG, TEBIKH :
Segmentation de la couleur de peau par seuillage selon différents espaces couleurs,
3ème journée internationale sur l'informatique graphique en TUNISIE

[**14**] : World Academy of Science, Engineering and Technology 60 2009,
Article : *Real-time target tracking using a Pan and Tilt platform*

[**15**] : Les espaces couleurs RVB et Lab.
http ://www.tsi.enst.fr/tsi/enseignement/ressources/mti/RVB_ou_LAB/html/
colorspace.htm