République Algérienne Démocratique et Populaire
Université Abou Bakr Belkaid– Tlemcen
Faculté des Sciences
Département d'Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en Informatique

Option : Système d'Information et des Connaissances (S.I.C)

Thème

La détection et suivi des objets en mouvement dans une scène vidéo en utilisant la bibliothèque OpenCV

Réalisé par :

- ***M^{elle} HACHEMI Fouzia***

Présenté le 05 Octobre 2016 devant le jury composé de MM.

- *Mr. BENAMAR. A* *(Président)*
- *Mr .BENAISSA .M* *(Encadreur)*
- *Mr. BENZIAN. Y* *(Examinateur)*
- *Mr. MANA. M* *(Examinateur)*

Année universitaire: 2015-2016

REMERCIEMENTS

Louange à الله*, seigneur de l'univers.*

Au terme de ce travail je tiens tout d'abord à exprimer ma profonde gratitude à mon encadreur ***Monsieur Benaissa Mohamed****.*
Qui ma guidé tout au long de ce travail.

Mes remerciements s'adressent à tous les membres du jury pour l'honneur qu'ils m'ont fait en acceptant.

Je remercie toutes les personnes ayant contribué et faciliter la réalisation de ce travail dans de bonnes conditions.

Dédicace

Je dédie ce modeste travail

À mes très chers parents

À toute ma famille et mes amis

À tous ceux qui m'ont encouragé et

Soutenu …

FOUZIA

ملخص

الرؤية بالكمبيوتر هي فرع من فروع الذكاء الاصطناعي الذي يهدف إلى السماح للآلة بان تفهم ما ترى عند الاتصال بكاميرا أو أكثر.يمكن استخدامها في معرفة الأشكال، و الذي يتمثل في التعرف على شكل في صورة بعد الحفظ. مع الانتشار الواسع للصور الرقمية تم الإثبات بان تحليل الحركة في فيديو هو وسيلة لا غنى عنها في تطبيقات متنوعة مثل المراقبة بالفيديو، ضغط الفيديو، والتصوير الطبي، والروبوتات، تفاعل الإنسان مع الحاسوب .

هذا المشروع يسمح بدراسة الوسائل المختلفة لتتبع جسم متحرك واحد أو أكثر في فيديو. نحن مهتمون في طريقة الكشف عن الأجسام المتحركة حسب لونها.

الكلمات المفتاحية :الكشف عن الأجسام،OpenCV،طريقة اللون، طريقة الحركة،تحديد موقع الأجسام المتحركة.

Résumé

La vision par ordinateur est une branche de l'intelligence artificielle dont le but est de permettre à une machine de comprendre ce qu'elle «voit » lorsqu'on la connecte à une ou plusieurs caméras. Elle peut servir entre autre à la reconnaissance de formes, qui consiste à reconnaitre une forme dans une image après l'avoir enregistrée.

Avec la généralisation de l'utilisation des images numériques, l'analyse du mouvement dans les vidéos s'est révélée être un outil indispensable pour des applications aussi diverses que la vidéo surveillance, la compression vidéo, l'imagerie médicale, la robotique, l'interaction homme machine.

Ce projet de fin d'étude consiste en une étude de différentes méthodes de suivi d'un ou plusieurs objets dans une vidéo. Nous sommes intéressés par la méthode de détection des objets en mouvement selon leur de couleur.

Mot clés : détection objet, OpenCV, méthode de couleur, méthode de mouvement, localisation d'un objet en mouvement (tracking).

Abstract

Computer vision is a branch of artificial intelligence that aims to allow a machine to understand what it "sees" when it connects to the one or more cameras. It can be used among other things to the recognition, which is to recognize a shape in an image after recording.

With the widespread use of digital images, motion analysis in video proved to be an indispensable tool for applications as diverse as video surveillance, video compression, medical imaging, robotics, human computer interaction.

This final project is a study of different methods of tracking one or more objects in a video. We are interested in the detection method of moving objects by their color.

Key words: object detection, OpenCV, color method, method of movement, locating a moving object (tracking)

Table des matières

Chapitre III : Les méthodes de la détection d'objets en mouvement

Chapitre IV: Résultat de détection d'un objet en mouvement selon la couleur

Table des figures

Chapitre I

Chapitre II

Chapitre III

Chapitre IV

Liste des abréviations

3D	Images A **Trois D**imensions.
AVI	**A**udio **V**ideo **I**nterleave.
BMP	**B**it**M**a**p**
BSD	**B**erkeley **S**oftware **D**istribution
CCD	**C**harged **C**oupled **D**evice
CD	**C**ompact **D**isc
CD-ROM	**C**ompact **D**isc - **R**ead **O**nly **M**emory
Codec	En**co**deur/**Déc**odeur
DivX	**D**igital **V**ideo **E**xpress
DVD	**D**igital **V**ertical**D**isk
EPS	**E**ncapsulated **P**ost**S**cript
Flv	**F**lash **V**ideo
GIF	**G**raphical**I**nterchange**F**ormat
IDE	**I**ntegrated **D**evelopment **E**nvironment
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers
JPEG	**J**oint**P**hotographic**E**xperts**G**roup
Mac OS	**Mac**intosh **O**perating **S**ystem
MKV	**M**atroska **V**ideo
Mov	Quicktime **Mov**ie
MP3	**MP**EG-1/2 Audio Layer **3**

MP4	**M**PEG-**4P**art 14
MPEG	**M**otion**P**icture**E**xpert**G**roup
NTSC	**N**ational **T**elevision **S**ystem **C**ommette
OpenCV	**Open**Source**C**omputer**V**ision
PAL	**P**hase **A**lternation **L**ine
Pixel	**Pi**cture **EL**ement
RGB	**R**eed **G**reen **B**lue
RM	**R**eal **M**edia
RVB	**R**ouge **V**ert **B**leu
SECAM	**Sé**quentiel **C**ouleur **AM**émoire
WMV	**W**indows **M**edia **V**ideo
XML	e**X**tensible **M**arkup **L**anguage
YCbCr	Luminance *(Y)*, Chrominance *(**R**ed-**Y**ellow);* **C**hrominance *(**B**lue-**Y**ellow)*

INTRODUCTION GENERAL

Introduction générale

La vision par ordinateur est une branche de l'intelligence artificielle dont le but est de permettre à une machine de comprendre ce qu'elle «voit » lorsqu'on la connecte à une ou plusieurs caméras. Elle peut servir entre autre à la reconnaissance de formes, qui consiste à reconnaitre une forme dans une image après l'avoir enregistrée.

Avec la généralisation de l'utilisation des images numériques, l'analyse du mouvement dans les vidéos s'est révélée être un outil indispensable pour des applications aussi diverses que la vidéo surveillance, la compression vidéo, l'imagerie médicale, la robotique, l'interaction homme machine, l'analyse de séquences sportives...etc. En effet, les zones de mouvement d'une séquence d'images correspondent souvent à des événements sur lesquels un système de vision doit se focaliser. L'analyse du mouvement est un vaste sujet qui englobe un certain nombre de problématiques :

– la détection des objets en mouvement, c'est-à-dire la détection d'un ensemble de régions d'intérêt en mouvement dans la scène observée,

– le suivi de primitives ou de régions, dont le but est de déterminer la position de chaque primitive ou région dans l'image à chaque instant

La problématique détection des objets en mouvement, est en général une première étape pour des outils automatiques de vision par ordinateur. Ces outils peuvent avoir pour vocation, soit uniquement de détecter, soit de détecter et reconnaître, soit de détecter et suivre des objets pour, par exemple, analyser le comportement ou la trajectoire de ces objets.

L'autres problématiques est aussi importante et nécessite la mise en place de méthodes simples et robustes. Tous ces sujets font l'objet d'un grand nombre de travaux, mais il n'existe pas, à l'heure actuelle, d'algorithmes aboutis s'adaptant à n'importe quelle situation.

Nous avons étudiés quatre méthodes de détection des objets en mouvement dans une sciène vidéo.

Nous sommes intéressés par la méthode de détection des objets en mouvement selon leur couleur

Notre mémoire est décomposé en quatre chapitres :

Chapitre1 : Nous parlons de généralités sur la bibliothèque OpenCV. Dans le présent chapitre, nous décrivons cette bibliothèque en donnant son Présentation et sa configuration dans le système d'exploitation Microsoft Windows avec Visual Studio 2008.

Chapitre 2 : Nous présentons une étude générale sur les caractéristiques fondamentales des vidéos.

Chapitre 3 : Présent les méthodes de détection du mouvement. Ainsi leurs propres avantages et leurs inconvénients.

Le **chapitre 4** conclut ce mémoire en présentant l'ensemble des résultats expérimentaux obtenus pour la détection et le suivi d'objets en mouvement.

Ce travail est terminé par une conclusion générale et une bibliographie

.

Chapitre I

La configuration d'OpenCV avec Visual Studio

1. Introduction

Dans ce chapitre, nous allons expliquer comment installer OpenCV sur Windows et d'interagir avec Microsoft Visual Studio 2008 pas à pas.

La bibliothèque OPENCV sera utilisée dans notre projet pour suivre des objets en mouvement en utilisant une webcam.

2. Présentation de la librairie OpenCV

2.1 Historique

Officiellement lancé en 1999, le projet OpenCV est développé initialement par Intel pour optimiser les applications gourmandes en temps processeur. Cela faisait partie d'une série de projets tels que l'affichage d'un mur en 3 dimensions. Cette bibliothèque est distribuée sous licence BSD.

Les principaux acteurs du projet sont l'équipe de développement de bibliothèque de chez Intel ainsi qu'un certain nombre d'experts dans l'optimisation de chez Intel Russie.

Les objectifs de base du projet étaient :

- Faire des recherches sur la vision par ordinateur en vue de fournir un logiciel libre et optimisé.
- Établir une infrastructure commune s'appuyant sur les développeurs pour obtenir un code plus lisible et transférable.
- Continuer à développer en rendant le code portable et permettre des performances optimisées gratuites avec une licence qui est libre de toutes contraintes commerciales.

La première version alpha d'OpenCV fut présentée lors de la conférence IEEE sur la vision par ordinateur et la reconnaissance de formes en 2000. Après cela, cinq versions bêta ont été publiées entre 2001 et 2005 et la première version 1.0 a été publiée en 2006.
Au milieu de l'année 2008, OpenCV obtient l'appui de la société de robotique Willow Garage et la bibliothèque est encore développée à ce jour. Une version 1.1 est sortie en Octobre 2008

et un livre écrit par deux auteurs d'OpenCV, publié par O'Reilly Media est sorti sur le marché ce même mois.

La deuxième version majeure d'OpenCV née en octobre 2009. Il s'agit d'OpenCV 2 incluant des changements majeurs au niveau du langage C++ servant à faciliter le développement de nouvelles fonctions et améliorant les performances. **[S1][11]**

2.2 Qu'est-ce que OpenCV

OpenCV (Open Source Computer Vision) est une bibliothèque libre de vision par ordinateur. Cette bibliothèque est écrite en C et C++ et peut être utilisée sous Linux, Windows et Mac OS X. Des interfaces ont été développées pour Python, Ruby, Matlab et autre langage. Open CV est orienté vers des applications en temps réel.

Un des buts d'OpenCV est d'aider les gens à construire rapidement des applications sophistiquées de vision à l'aide d'infrastructure simple de vision par ordinateur. La bibliothèque d'OpenCV contient plus de 500 fonctions. **[10] [22]**

2.3 Fonctionnalités

La bibliothèque OpenCV met à disposition de nombreuses fonctionnalités très diversifiées permettant de créer des programmes partant des données brutes pour aller jusqu'à la création d'interfaces graphiques basiques. Elle propose la plupart des opérations classiques en traitement bas niveau des images et des vidéos.

- **Traitement d'images**

Elle propose la plupart des opérations classiques en traitement bas niveau des images :

- lecture, écriture et affichage d'une image.
- calcul de l'histogramme des niveaux de gris ou d'histogrammes couleurs.
- lissage, filtrage.
- seuillage d'image (méthode d'Otsu, seuillage adaptatif).
- segmentation (composantes connexes, GrabCut).
- morphologie mathématique. **[S3]**

- **Traitement vidéo**

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

  - lecture, écriture et affichage d'une vidéo (depuis un fichier ou une caméra).
  - détection de droites, de segment et de cercles par Transformée de Hough.
  - détection de visages par la méthode de Viola et Jones.
  - cascade de classifieurs boostés.
  - détection de mouvement, historique du mouvement.
  - poursuite d'objets par mean-shiftou Camshift.
  - détection de points d'intérêts.
  - estimation de flux optique (Méthode de Lucas–Kanade).
  - triangulation de Delaunay.
  - diagramme de Voronoi.
  - enveloppe convexe.
  - ajustement d'une ellipse à un ensemble de points par la méthode des moindres carrés. **[S3]**

2.4 Modules OpenCV

OpenCV a une structure modulaire. Les principaux modules d'OpenCV sont énumérés :

- **CxCore (**Cœur d'OpenCV) :

Contient les structures de donnée et les opérations mathématiques de base. Elle permet de réaliser les fonctions d'algèbre matriciel, la transformation de données, la gestion de mémoire, la manipulation d'erreurs, le chargement dynamique de code et permet aussi la réalisation des graphiques.

- **HighGui (librairie d'interfaçage graphique)**

OpenCV intègre sa propre bibliothèque haut-niveau pour ouvrir, enregistrer et afficher des images et des flux vidéo. Celle-ci contient aussi un certain nombre de fonctions permettant de réaliser des interfaces graphiques très simples, mais largement suffisantes pour tester nos programmes.

- **ImgProc (traitement d'image) :**

Ce module comprend des algorithmes de traitement d'image de base, y compris le filtrage d'image, transformations d'image, les conversions d'espace couleur et etc.

- **Vidéo (traitement de flux vidéo)**

Ceci est une vidéo module d'analyse qui inclut des algorithmes de suivi (tracking) d'objets, des algorithmes de soustraction de fond et etc.

- **ObjDetect (détection d'objets)**

Cela comprend l'objet de détection et de reconnaissance des algorithmes pour les objets standard.

- **ML (librairie d'apprentissage automatique)**

Comporte les fonctions de classification, d'analyse de donnée et des outils de clustering.

- **Calib3D**

Calibrage caméra et reconstruction 3D. **[S4] [10]**

3. Installation et configuration OpenCV2.3.1 avec Visual Studio 2008

3.1 Téléchargement et extraction OpenCV

- Télécharger OpenCV pour Windows
- Extraire le programme C: \

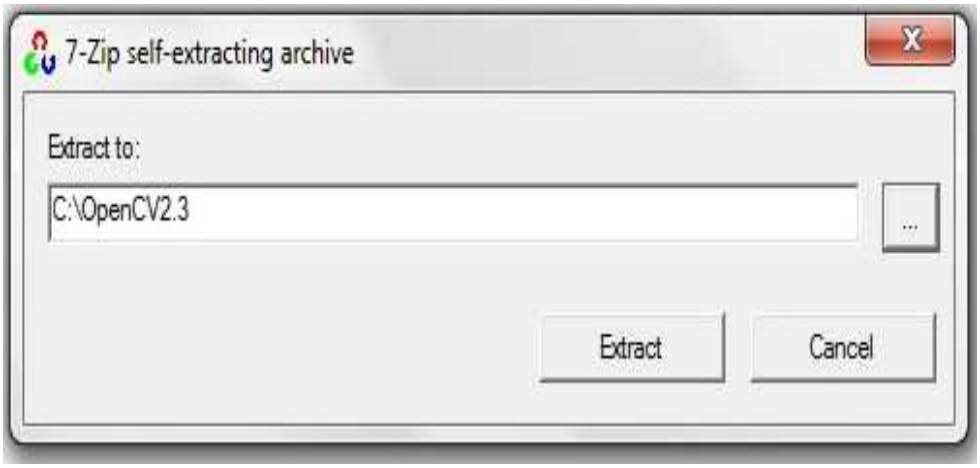

Figure I.1 : Extraction d'OpenCV

3.2 Mise en place des variables d'environnement

- Afin de définir les variables d'environnement, aller sur :

Menu démarrer → clic droit sur ordinateur →propriétés

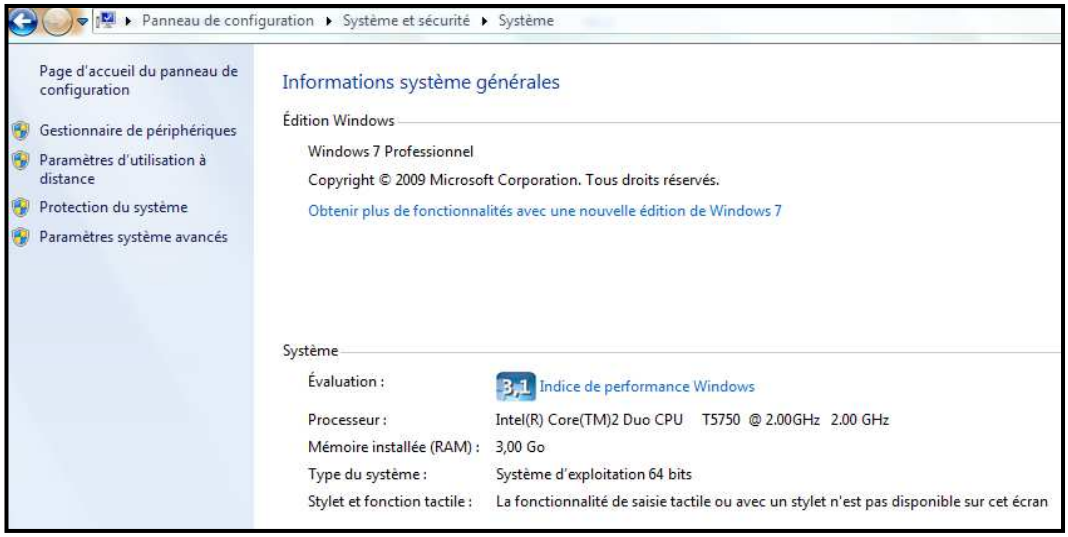

Figure I.2 : propriétés du système

- Choisir Paramètres système avancés (voir **Figure I.2**)
- Choisir variables d'environnement (**Figure I.3**)

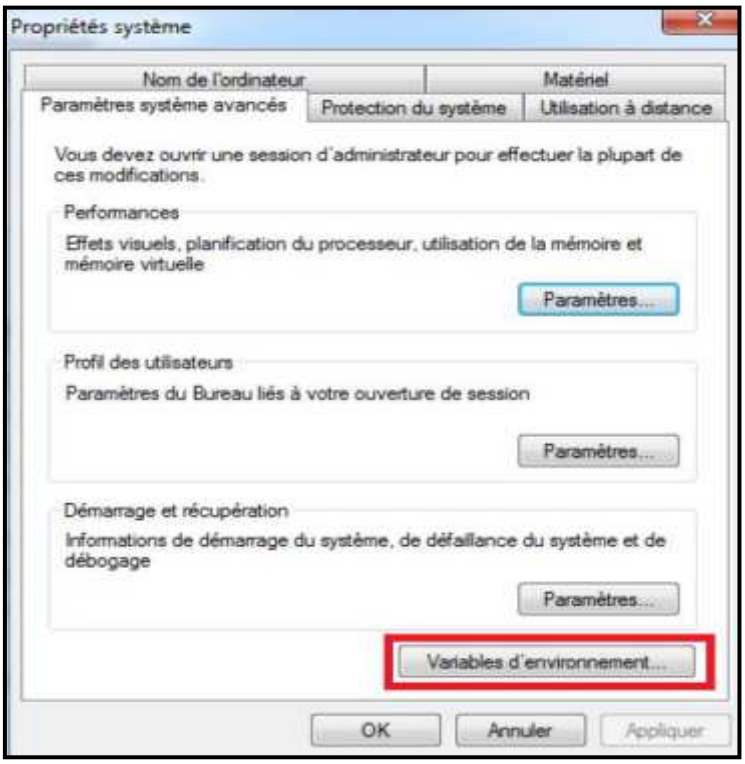

Figure I.3 : propriétés système

- Trouver la variable PATH sous Variables système, sélectionnez- le puis cliquez sur Modifier .... Allez à la fin de Valeur de la variable, puis ajouter le répertoire ; « C:\opencv\build\x64\vc09\bin »
  (**Figure I.4**).

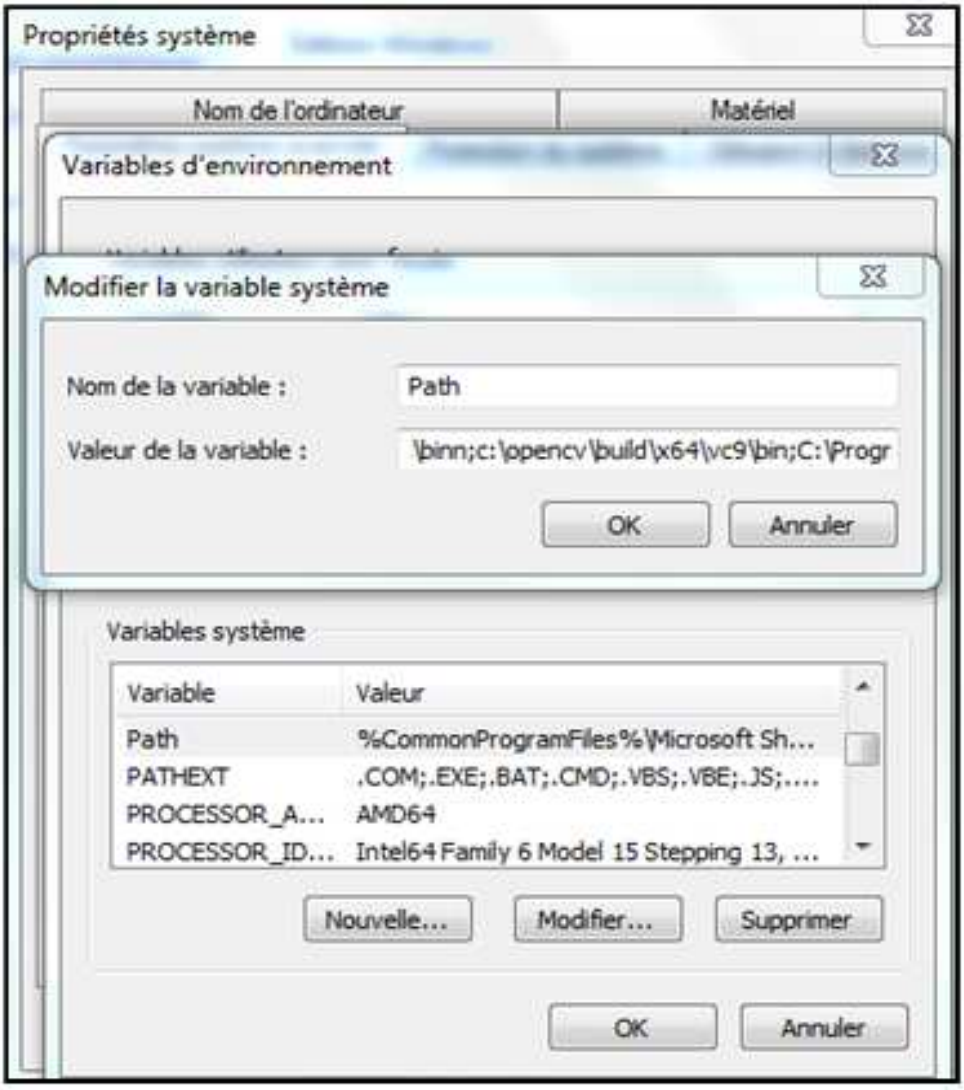

Figure I.4 : définir le path

- Ce répertoire contient les DLL OpenCV nécessaires pour exécuter votre code.
- Si votre type de système est 32 bits remplacer x64 par x86.
- (C:\opencv\build\x86\VC09 \bin). A noter la partie de VC09 du chemin est la version de Visual Studio installé sur le système (VC09 = Microsoft Visual Studio 2009).
- Le processus d'installation d'OpenCV est terminé, la prochaine étape est la configuration de Visual Studio

3.3 Microsoft Visual Studio 2008

Microsoft Visual Studio est une suite de logiciels de développement pour Windows conçue par Microsoft. La dernière version s'appelle Visual Studio 2015.

Visual Studio est un ensemble complet d'outils de développement permettant de générer des applications webASP.NET, des services web XML, des applications bureautiques et des applications mobiles. Visual Basic, Visual C++, Visual C# utilisent tous le même environnement de développement intégré (IDE), qui leur permet de partager des outils et facilite la création de solutions faisant appel à plusieurs langages.

Par ailleurs, ces langages permettent de mieux tirer parti des fonctionnalités du Framework .NET, qui fournit un accès à des technologies clés simplifiant le développement d'applications web ASP et de services web XML grâce à Visual Web Developer. **[s5]**

3.4 Configuration de Visual Studio 2008

- A créé un nouveau projet dans Visual Studio 2008:

  Fichier→ Nouveau → Projet

- Sélectionnez "Visual C++" Template →"Win32 Console Application" dans le volet central. Entrez le nom et l'emplacement du projet. Cliquez OK. (**Figure I.5**)

- Cliquez sur "suivant" dans la nouvelle fenêtre, puis choisissez "Application console"→"Projet vide" puis "Terminer".

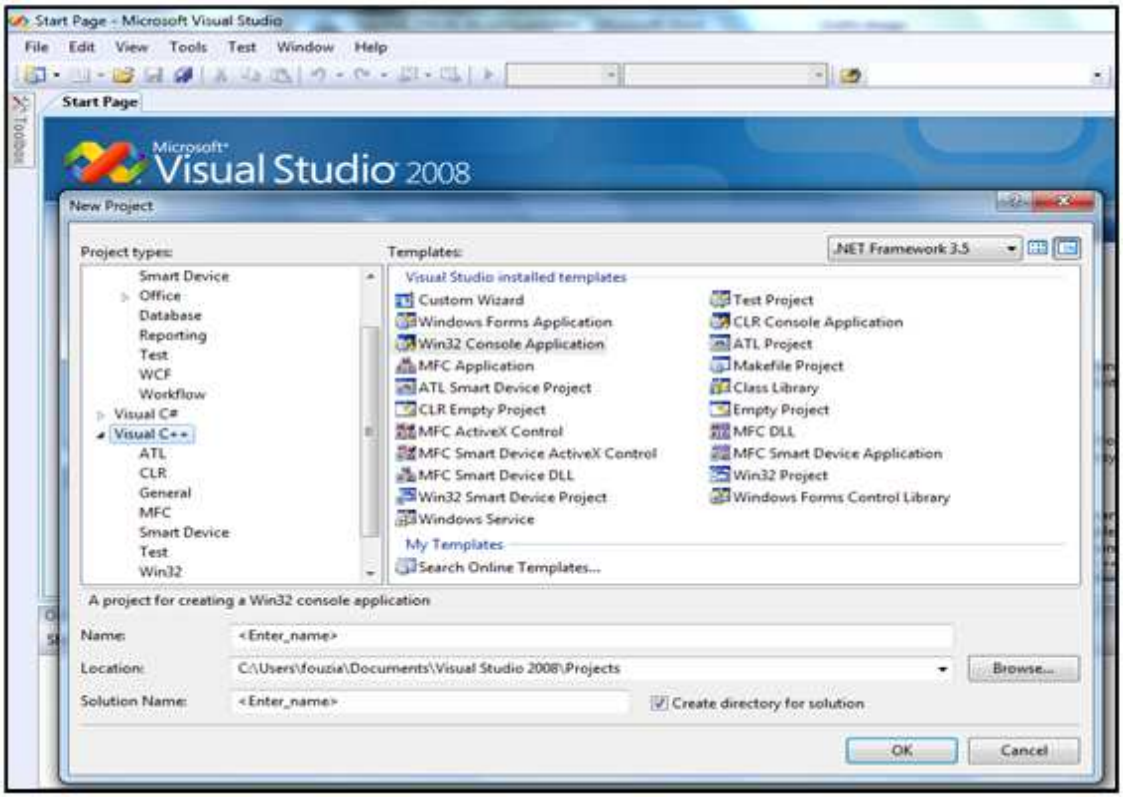

Figure I.5 : Création d'un nouveau projet

3.5 Configuration générale

Il s'agit d'indiquer à Visual Studio les nouveaux répertoires où sont disponibles les fichiers .h et .lib d'OpenCV.

- Choisir All Configuration pour Configuration.
- Choisir (x64) pour platform (si le type de système d'exploitation 64 bits) ou bien win32 (si le type de système d'exploitation 32bits).
- Dans le volet de gauche, cliquez sur
  Configuration→ properties→ VC++ Directories→ general
- Dans Additional include Directories ajouter les sous répertoires suivants :
  - C:\openCV\build\include
  - C:\openCV\build\include\openCV
  - C:\openCV\build\include\openCV2 (**Figure I.6**)

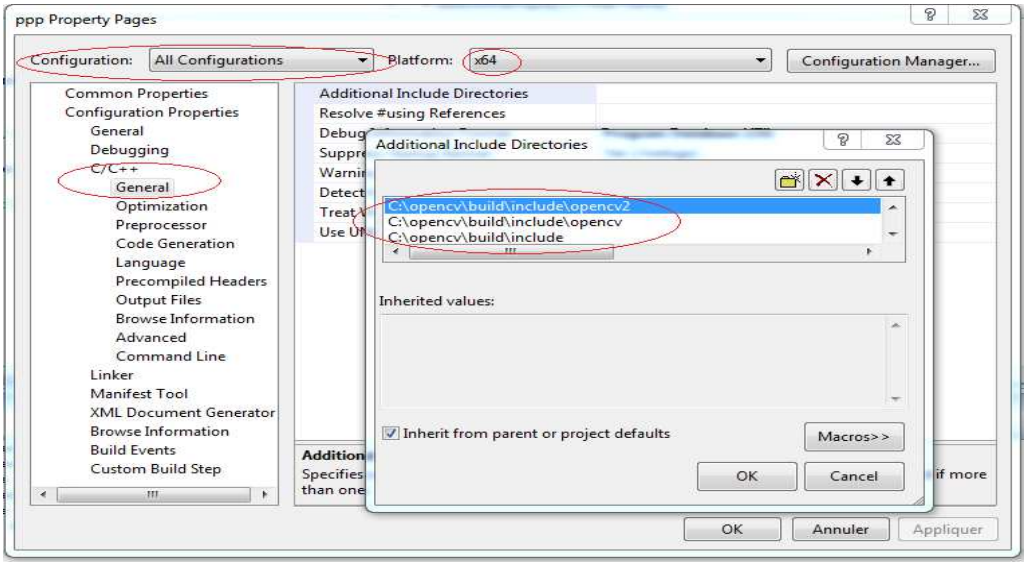

Figure I.6 : Configuration général de Microsoft Visual Studio 2008 pour OpenCV

- Choisir ensuite configuration properties ---> Linker ---> General
- Dans Additional Library Directories ajouter la ligne <<C:\OpenCV\build\x64\VC09\lib>> (**Figure I.7**)

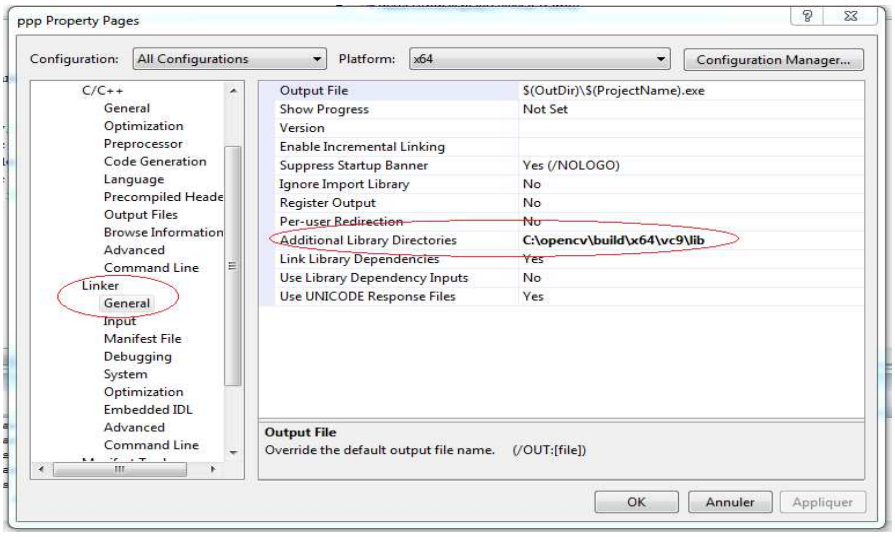

Figure I.7 : Configuration de Microsoft Visual Studio 2008 pour la librairie OpenCV

- ***Remarque :*** le chemin C:\OpenCV peut varier en fonction de ce que vous avez choisi lors de l'installation.

3.6 La configuration nécessaire

Cette étape est à recommencer pour chaque nouveau projet.

- Aller dans Additional Dependencies ajouter les fichiers lib suivants :
  - opencv_calib3d231d.lib
  - opencv_contrib231d.lib
  - opencv_core231d.lib
  - opencv_features2d231d.lib
  - opencv_flann231d.lib
  - opencv_gpu231d.lib
  - opencv_haartraining_engined.lib
  - opencv_highgui231d.lib
  - opencv_imgproc231d.lib
  - opencv_legacy231d.lib
  - opencv_ml231d.lib
  - opencv_objdetect231d.lib
  - opencv_ts231d.lib

Comme sur la **Figure I.8**

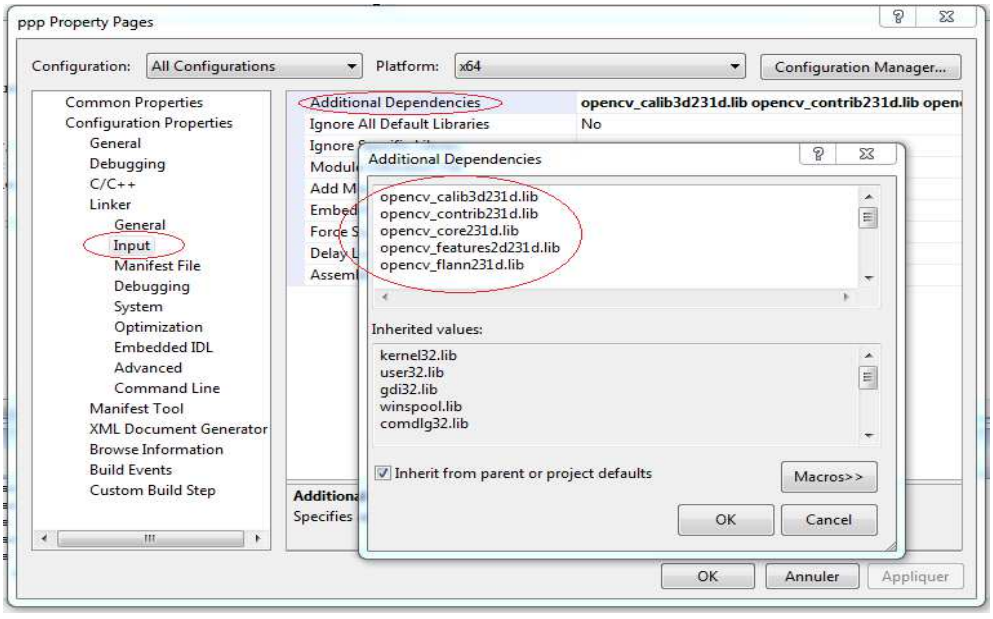

Figure I.8 : Configuration nécessaire de Microsoft Visual Studio 2008 pour OpenCV

4. Conclusion

Bibliothèque OpenCV permet aux utilisateurs de créer des applications de vidéo et d'images complexes. De nombreux exemples de code sont disponibles en ligne qui permettrait de créer facilement des applications. Cette note d'application devrait avoir donné aux utilisateurs des informations sur la manière d'inclure les bibliothèques externes dans Visual Studio.

Chapitre II

Notion de base sur les images et les vidéos

1. Introduction

Le traitement des vidéo est une science récente qui a pour but d'offrir aux spécialistes de différents domaines, comme au grand public, des outils de manipulation de ces données digitales issues du monde réel.

Dans ce chapitre, nous allons présenter les différentes phases de la formation de vidéo ainsi que sa définition, de ces différents types, de ces caractéristiques et de ces formats.

2. L'image et sa représentation

L'image est définie comme étant une fonction f(x, y) à deux dimensions, où x et y sont les coordonnées spatiales, et l'amplitude à tous points f(x, y) correspondant à l'intensité ou au niveau de gris. Lorsque les points (x, y) et l'amplitude sont discrétisés, on parle d'image numérique ou digitale. Dans ce dernier cas la fonction f est remplacée par la lettre I et le couple (x, y) par le couple (i, j). **[23]**

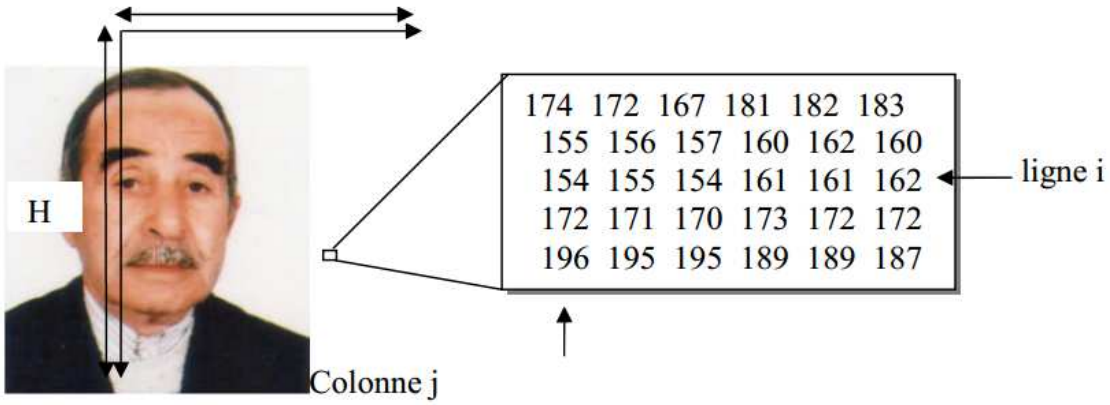

Figure II.1 : Image couleur, son repère et un extrait de pixels. [1]

L'image numérique, désignée aussi par le terme scène, possède un repère comme indiqué **Figure II.1**, il est différent de celui d'une fonction mathématique. Elle a une hauteur (H) et une largeur (W). **[1]**

2.1 Types d'images

Les images peuvent être en plusieurs types :

- **Image binaire**

Une image binaire est une matrice rectangulaire dont les éléments valent 0 ou 1. Lorsque l'on visualise une telle image, les zéros sont représentés par des noirs et les une par des blancs**. [2]**

- **Image en niveau de gris**

Chaque pixel est un niveau de gris, allant de 0 (noir) à 255 (blanc). Cet intervalle de valeur signifie que chaque pixel est codé sur huit bits (un octet). 256 niveaux de gris suffisent pour la reconnaissance de la plus part des objets d'une scène. **[S11]**

- **Codage d'une image en niveaux de gris**

Si on code chaque pixel sur 2 bits on aura 4 possibilités (noir, gris foncé, gris clair, blanc). L'image codée sera très peu nuancée.
En général on code chaque pixel sur 8 bits = 1 octet. On a alors 256 possibilités (on dit 256 niveaux de gris).
L'image de 10 000 pixels codée occupe alors 10 000 octets en mémoire.
Exemple d'image en 72 pixels par pouce (environ 30 pixels par cm), codée en 256 niveaux de gris.
Cette image de 303 x 303 pixels occupe 303 x 303 = 91809 octets puisque chaque pixel occupe 1 octet en mémoire. **[S11]**

- **Image couleur (RGB)**

Chaque pixel possède une couleur décrite par la quantité de rouge (R), vert (G) et bleu (B).

Chacune de ces trois composantes est codée sur l'intervalle [0, 255], ce qui donne

255^3=16 777 216 couleurs possibles. Il faut 24 bits pour coder un pixel. **[S11]**

- **Codage d'une image en couleurs 8 bits**

Dans ce cas on attache une palette de 256 couleurs à l'image.
Ces 256 couleurs sont choisies parmi les 16 millions de couleurs de la palette RVB. Pour chaque image le programme recherche les 256 couleurs les plus pertinentes.
Chaque code (de 0 à 255) désigne une couleur.
L'image occupe 3 fois moins de place en mémoire qu'avec un codage 24 bits. L'image est moins nuancée : sa qualité est bonne mais moindre. **[S11]**

- **Codage d'une image en couleurs 24 bits**

Il existe plusieurs modes de codage de la couleur. Le plus utilisé est le codage Rouge, Vert, Bleu (RVB). Chaque couleur est codée sur 1 octet = 8 bits. Chaque pixel sur 3 octets c'est à dire 24 bits : le rouge de 0 à 255, le vert de 0 à 255, le Bleu de 0 à 255.
Le principe repose sur la synthèse additive des couleurs : on peut obtenir une couleur quelconque par addition de ces 3 couleurs primaires en proportions convenables.
On obtient ainsi 256 x 256 x 256 = 16777216 (plus de 16 millions de couleurs différentes). **[S11]**

2.2 Caractéristiques de l'image

2.2.1 Pixel

Une image est constituée d'un ensemble de points appelés pixels (voir l'extrait de la **figure II.1**). Le pixel (Picture élément) représente ainsi le plus petit élément constitutif d'une image numérique. La quantité d'information que véhicule chaque pixel donne des nuances entre images monochromes et images couleurs. Pour les images 3D le «pixel» est alors appelé un voxel, et représente un volume élémentaire. Des exemples d'images de ce type se rencontrent dans les images médicales. Les images tomographiques axiales sont ainsi des images construites à partir de plusieurs radiographies faites sous des angles de vue différents. **[S20]**

2.2.2 Poids de l'image

C'est la taille de l'image. Etant donné que cette dernière est représentée sous forme d'une matrice dont les valeurs représentent l'intensité (pixels), le nombre de colonne (W) multiplié par le nombre de ligne (H) donne le nombre total de pixels dans l'image.

Pour une image de 640x480 en couleur :

- ✓ Nombre de pixel = 640x480 = 307200
- ✓ Poids de chaque pixel = 3 octets
- ✓ Le poids de l'image = 307200x3 = 921600 octets = 900 Ko. **[S7]**

2.2.3 Bruit

Un bruit (parasite) dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins, il provient de l'éclairage des dispositifs optiques et électroniques du capteur. **[3]**

2.2.4 Transparence

La transparence est une caractéristique définissant le niveau d'opacité des éléments de

L'image, c'est la possibilité de voir à travers l'image des éléments graphiques situés derrière celle-ci. **[S21]**

2.2.5 Luminance

C'est le degré de luminosité des points de l'image. Elle est définie ainsi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface. **[24]**

2.2.6 Contraste

Est une propriété intrinsèque d'une image qui désigne et quantifie la différence entre les parties claires et foncées d'une image (elle différencie les couleurs claires des couleurs foncées).

En photographie on le définit le contraste comme la différence entre la densité la plus forte et la plus faible d'une image. Le contrôle du contraste est un élément important de la pratique photographique. Le contraste final de l'image dépend à la fois du sujet, de la nature et du traitement du négatif et du positif. **[S22]**

2.2.7 Histogramme

Un histogramme est un graphique statistique permettant de représenter la distribution des intensités des pixels d'une image. Il fournit diverses informations comme les statistiques d'ordre (moyenne, variance,…), l'entropie, et peut permettre d'isoler des objets. **[S8]**

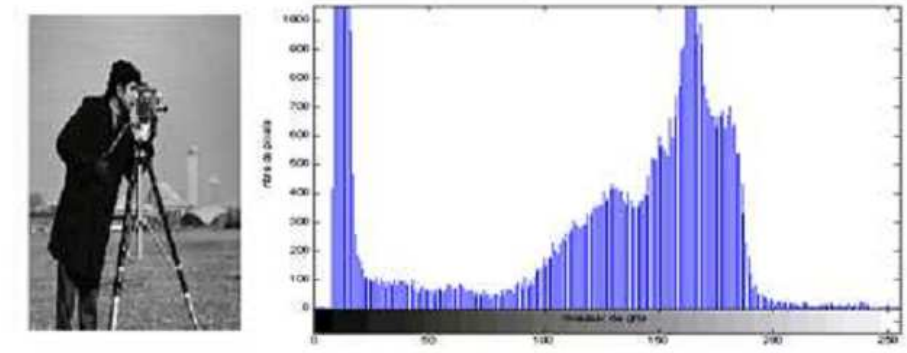

Figure II.2 : Exemple d'image en niveaux de gris. [S9]

Pour les images en couleur plusieurs histogrammes sont nécessaires.

Par exemple pour une image codée en RGB:

- un histogramme représentant la distribution de la luminance
- trois histogrammes représentant respectivement la distribution des valeurs respectives des composantes rouges, bleues et vertes**. [S10]**

2.3 Les différents formats d'images

On peut classer les images en deux formats :

2.3.1 Format vectorielle

Dans une image vectorielle les données sont représentées par des formes géométriques simples qui sont décrites d'un point de vue mathématique.

Par exemple, un cercle est décrit par une information du type (cercle, position du centre, rayon). Ces images sont essentiellement utilisées pour réaliser des schémas ou des plans. **[S23]**

2.3.2 Format matricielle

Une image matricielle est formée d'un tableau de points ou pixels. Plus la densité des points sont élevée, plus le nombre d'informations est grand et plus la résolution de l'image est élevée. Corrélativement la place occupée en mémoire et la durée de traitement seront d'autant plus grandes.

Les images vues sur un écran de télévision ou une photographie sont des images matricielles. **[S23]**

On obtient également des images matricielles à l'aide d'un appareil photo numérique, d'une caméra vidéo numérique ou d'un scanner. Parmi ces formats on peut citer :

- **BMP (BitMap) :** Le format BMP est le format par défaut du logiciel Windows. C'est un format matriciel. Les images ne sont pas compressées. Son logiciel d'origine.
- **Le format EPS :** matriciel n'est pas très différent du EPS vectoriel. En fait seules les données contenues dans le fichier sont différentes. Ainsi un logiciel de retouche de photos tel que Photoshop permet l'importation, la modification et l'exportation de fichiers en format EPS.
- **GIF (GraphicalInterchange Format)** : Le format GIF est un format qui a ouvert la voie à l'image sur le World Wide Web. C'est un format de compression qui n'accepte que les images en couleurs indexés codé sur 8 bits, C'est un format qui perd beaucoup de son marché suite à une bataille juridique concernant les droits d'utilisation sur Internet.
- **JPEG (Joint Photographique Experts Group) :** Les images JPEG sont des images de 24 bits. C'est-à dire qu'elles peuvent afficher un spectre de 16 millions de couleurs. C'est la meilleure qualité d'images disponible. **[S6]**

3. Frame

Image composants la vidéo, les photogrammes **[S2]**

4. Objet

Les objets physiques sont les objets du monde réel qui apparaissent dans les scènes observées par les caméras. **[4]**

Les objets physiques sont divisés en deux types : les objets de contexte et les objets mobiles.

- **Les objets de contexte**

Sont des objets physiques qui sont habituellement statiques (p. ex. les murs). Dans le cas où ils ne sont pas statiques, leurs mouvements peuvent être prédits par les informations contextuelles p.ex. Les chaises, les portes sont des objets de contexte. **[4]**

- **Les objets mobiles**

Sont des objets physiques qui peuvent être perçus dans les scènes par leurs mouvements **[4]**. Il est cependant difficile de prédire leurs mouvements P. ex. les personnes, les véhicules.

Les objets dans les vidéos sont des objets mobiles. Ils sont habituellement détectés et suivis dans plusieurs frames.

5. La détection de l'objet

La détection de l'objet comprend la détection d'objets et la reconnaissance de formes dans le cadre d'une séquence vidéo. Un mécanisme de détection d'objet est nécessaire dans ne importe quel procédé de suivi soit dans chaque trame ou lorsque l'objet apparaît en premier dans la vidéo. **[5]**

6. La vidéo

6.1 Définition

La vidéo est une succession d'images animées défilant à une certaine cadence afin de créer une illusion de mouvement pour l'œil humain.
Elle peut être analogique (signal continu d'intensité de luminance) ou numérique (suite de trames ou images). **[6]**

6.2 Les fondamentaux physiques et techniques

6.2.1 La vision

Dans le phénomène de la vision humaine, un flux lumineux composé de photons frappe l'œil.
Dans l'œil, les bâtonnets vont réagir à l'intensité de la lumière (la luminance ou Y) et les

cônes vont réagir à la couleur (la chrominance ou C).

Le cerveau effectue la synthèse de ces informations pour composer une image. L'œil humain est davantage sensible à l'intensité lumineuse (Y) qu'à la couleur (C). **[S13]**

6.2.2 La couleur

La vidéo utilise ce qui est identifié comme la synthèse additive de la couleur. Dans le système colorimétrique additif, les trois couleurs primaires sont le rouge, le vert et le bleu (RVB ou RGB dans la terminologie anglophone). En combinant ces trois couleurs primaires, il est possible de reproduire tout le spectre visible par l'œil humain. L'écran de visualisation vidéo sera donc composé d'une série de triades rouge-vert-bleu. L'activation de l'ensemble de ces triades formera l'image **[S13]**

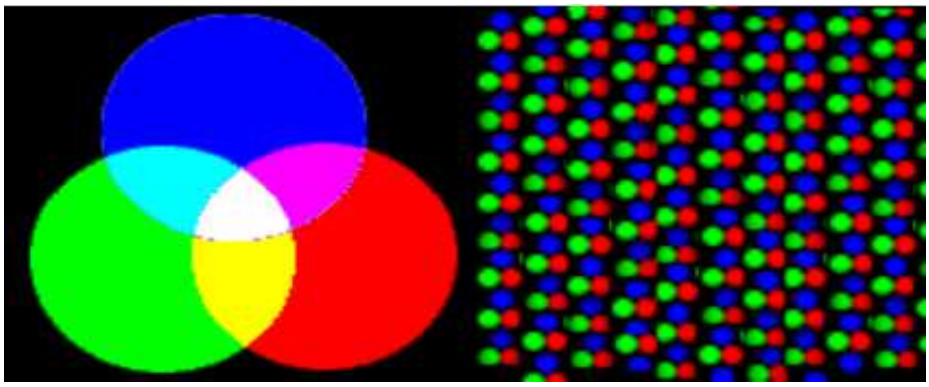

Figure II.3 : codage de couleur en RVB [S12]

6.2.3 Le signal

En vidéo, c'est la caméra qui transforme l'information lumineuse (photons) en signal électrique (électrons). En vidéo analogique, l'intensité de ce signal électrique varie de façon continue. Le processus de transformation de l'information lumineuse en signal électrique est le suivant :

- L'élément optique de la caméra, l'objectif, sépare la lumière en trois composants le rouge, le vert, le bleu. Cette opération est réalisée en faisant passer le flux lumineux par une succession de filtres dichroïques réfléchissant certaines couleurs et en laissant passer d'autres.

- En vidéo professionnelle, les trois images sont projetée sur trois capteurs photo sensibles distincts formés chacun de centaines de milliers de points

généralement entre 400 000 et 700 000. Ces capteurs sont nommés CCD (charged coupled device) ou dispositifs à transfert de charge. Les caméras domestiques ne sont généralement équipées que d'un seul capteur CCD. **[S13]**

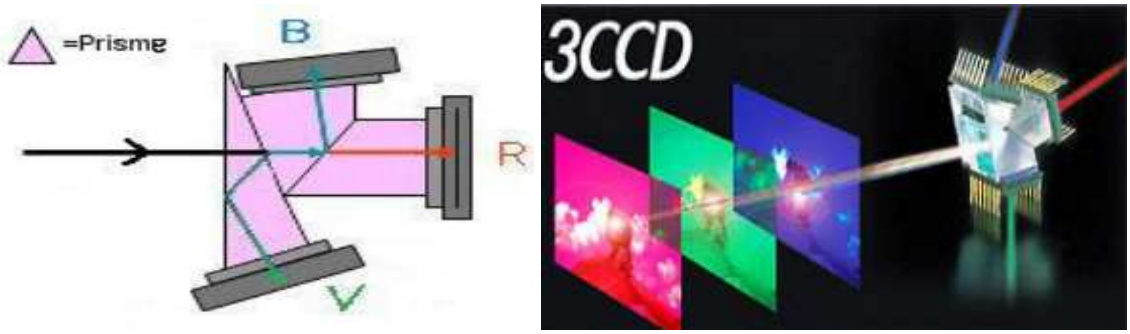

Figure II.4 : Schéma de principe d'un Capture Tri-CCD (charged coupled device) [S14]

- Pour chacun des points de chacun des capteurs, l'énergie lumineuse sera transformée en énergie électrique. Ainsi, à la sortie des capteurs, trois signaux électriques d'intensité variable, un signal pour chacune des trois composantes. La lumière blanche est formée par la somme des trois composantes RVB. Toutefois, elle n'est pas composée des trois couleurs primaires en quantité égale. Les propositions sont les suivantes :

  100% Y = 29.9%(R) +58,7%(V) +11,4%(B)

- Une information supplémentaire devra être ajoutée à ces signaux. Comme nous l'avons dit, une valeur doit être établie pour chacun des points des capteurs. C'est cette valeur pour chacun des points de chacun des pixels qui permettra de recomposer l'image sur un écran de visualisation, chacun des points de la surface d'affichage pouvant alors prendre la même valeur qu'au moment de la capture par la caméra. La restitution de l'information pour chacun des pixels se fait par balayage ligne par ligne de la surface de l'écran de reproduction, les valeurs enregistrées pour chacun des pixels au moment de la capture étant transférées aux pixels de la surface de reproduction. Pour synchroniser ces balayages, un signal de fin de ligne

est ajouté pour chacune des lignes. Un signal de fin d'image est également inséré. **[S13]**

- **Les caractéristiques d'un signal vidéo**

Chaque fichier vidéo a des attributs qui décrivent ce qui constitue le signal vidéo. Ces caractéristiques comprennent:

- **Châssis de taille**: Il s'agit de la dimension de pixel de l'image
- **Le ratio d'aspect**: C'est le rapport entre la largeur et la hauteur
- **Vitesse de défilement**: C'est la vitesse à laquelle les images sont capturées et destinés à la lecture.
- **Débit**: Le taux de débit ou de données est la quantité de données utilisées pour décrire la partie audio ou vidéo du fichier. Il est généralement mesuré en unités par seconde et peut être en kilo-octets, méga-octets ou giga-octets par seconde. En général, plus la vitesse de transmission, meilleure est la qualité.
- **Le taux d'échantillonnage audio:** C'est à quelle fréquence le signal audio est échantillonné lors de la conversion d'une source analogique à un fichier numérique. **[7]**

6.2.4 Affichage vidéo

La technique de la composition d'image par balayage a une conséquence : contrairement à l'image cinéma, l'image vidéo ne s'affiche pas d'un seul coup, elle s'affiche ligne par ligne, en commençant par le haut à gauche pour finir en bas à droite. Afin d'assurer une uniformité dans la luminosité de cet affichage, en d'autres termes afin que l'intensité lumineuse des pixels du haut de l'écran ne commence pas à diminuer alors que ceux du bas viennent d'être activés, une technique spécifique a été développée. Il s'agit de la technique d'entrelacement Un premier balayage est effectué sur les lignes impaires, puis un deuxième sur les lignes paires. Il faut donc deux passages pour recomposer une image complète. La fréquence d'entrelacement s'appuie sur la fréquence du réseau électrique. Comme en Amérique du Nord la fréquence des réseaux est de 60Hz, le système de balayage est donc de 60 demi-images, soit 30 images par seconde. De façon précise, le taux de rafraîchissement est plutôt de 29,97 images par seconde.

La norme Nord-Américaine en télévision est le NTSC (National Television Standard Committee). L'image NTSC est formée de 525 lignes. De ces lignes, seulement 486 sont

utilisées pour afficher l'image. Les deux autres principales normes sont européennes. Il s'agit des normes PAL (Phase Alternation by Line) et SECAM (Séquentiel Couleur à Mémoire). L'image PAL ou SECAM est formée de 625 lignes dont 576 sont utilisées pour l'image. Comme la fréquence des réseaux électriques européens est de 50 Hz, le système de balayage est de 50 demi-images, soit 25 images par seconde. **[S13]**

6.2.5 Le son

Le son fera l'objet d'un traitement distinct. Le signal sonore sera distinct du signal vidéo. Il sera également enregistré sur des pistes distinctes sur le ruban vidéo. **[S13]**

7. Types de vidéo

On distingue deux grandes familles de systèmes vidéo : les systèmes vidéo analogiques et les systèmes vidéo numériques.

7.1 La vidéo analogique

La caméra balaye l'image bidimensionnelle qu'elle a devant elle par un faisceau d'électrons qui se déplace très rapidement de gauche à droite et plus lentement de haut en bas et produit une tension en fonction du temps. Elle enregistre ainsi l'intensité lumineuse, et à la fin du balayage, on a alors une trame. Le faisceau revient à l'origine pour recommencer. Le récepteur va recevoir cette intensité en fonction du temps, et pour reconstruire l'image, va répéter le processus de balayage. **[8]**

7.2 La vidéo numérique

Le processus de captation de l'image vidéo en mode numérique est essentiellement le même que pour la vidéo analogique. Un système optique sépare la lumière en trois composants. À la sortie toutefois, plus d'enregistrement d'un signal électrique mais plutôt l'enregistrement d'une valeur numérique définie pour chacune des trois couleurs de chacun des pixels (RVB). Le processus de transformation d'un signal électrique et valeurs numériques s'appelle la numérisation ou l'échantillonnage. Deux paramètres servent à échantillonner un signal électrique : la fréquence d'échantillonnage et la quantification de l'échantillonnage ou le nombre de bits utilisés pour le codage. **[S13]**

7.2.1 La fréquence d'échantillonnage

Il existe un théorème appelé le théorème de Shannon. Ce théorème stipule que «l'information véhiculée par un signal dont le spectre est à support borné n'est pas modifiée par l'opération d'échantillonnage à condition que la fréquence d'échantillonnage soit au moins deux fois plus grande que la plus grande fréquence contenue dans le signal. La reconstitution du signal original peut être effectuée par un filtre passe-bas idéal de fréquence de coupure égale à la moitié de la fréquence d'échantillonnage.»

La bande passante d'un signal vidéo peut atteindre 6Mhz. Pour numériser ce signal, une fréquence d'échantillonnage de 13,5Mhz sera donc utilisée. En d'autres termes, le signal vidéo sera mesuré 13 500 000 fois par seconde. Comme nous savons que l'œil est sensible à la lumière (luminance) qu'à la couleur (chrominance), souvent, la fréquence d'échantillonnage de la chrominance sera moins importante. Dans les faits, l'on utilise un échantillonnage à 13,5Mhz pour Y (luminance) et 6,75 pour C (U et V). Cet échantillonnage est donc un échantillonnage à 16 bits par pixel. **[S13] [S15]**

7.2.2 La quantification de l'échantillonnage

Un certain nombre de bits serviront à coder chacune des mesures. Nous savons que l'œil humain discerne un maximum de 256 niveaux différents de luminance. Ainsi, un codage à 8 bits est suffisant pour reproduire l'ensemble des variations perceptibles

(8 = 256 combinaisons possibles) Comme chacune des composantes couleurs de l'image sera quantifiée sur 8 bits (8 bits Rouge, 8 bits Vert, 8 bits Bleu) (224= 16 777 216 combinaisons possibles) un total de 24 bits (3 octets) seront utilisés pour coder les informations relatives à un seul pixel d'une image. **[S13] [25]**

8. Composition des fichiers vidéos

Un fichier vidéo se compose généralement de 2 éléments :

8.1 Le conteneur

Il correspond généralement au format du fichier. Son rôle est de rassembler et d'organiser dans un fichier, différents types de données (flux audio, vidéos, sous-titres, meta-données).

Exemple de conteneurs:

- ✓ L'Audio Video Interleave (.AVI), Quicktime (.MOV), Real Media (.RM), MP4
- ✓ FLV (Flash Vidéo), MKV (Matroska), le WMV (Windows Media Video),

Chaque conteneur possède ses spécificités en termes de nombre de pistes acceptées pour la vidéo et l'audio, de codecs reconnus. **[S16]**

8.2 Le(s) contenu(s)

Ils se composent essentiellement de flux audios et/ou vidéos. Ceux-ci sont généralement compressés à l'aide d'un codec (algorithme de compression/décompression) comme le Divx, le H264, le mp3.

Exemple: un flux vidéo au format Divx peut être « encapsulé » dans un conteneur AVI ou QuickTime. **[S16]**

9. Les formats de fichiers vidéo

Décrit l'ordre et la structure de ces images. Les données du flux vidéo, qui peuvent être accompagnées de sons sous la forme de flux audio, sont très volumineuses : elles doivent impérativement être compressées (codées) à l'aide d'un codec pour être stockées (sur disque dur ou sur les supports d'enregistrement : CD, DVD) ou/et transmises (et donc être adaptées au débit des réseaux).

Les flux vidéo (et le(s) flux audio éventuellement associé(s)), une fois encodés, sont généralement encapsulés dans des fichiers conteneurs : ces derniers permettent, notamment, leur lecture simultanée. [S18]

9.1 Les codecs vidéo

Il existe de nombreux codecs permettant de compresser et de décompresser une vidéo dans son conteneur.

- **MJPEG** (Motion JPEG)

Le codec vidéo MJPEG compresse la vidéo image par image, en utilisant la technologie JPEG appliquée à l'image fixe, et réunit ces images en mouvement et le son dans un même format de fichier.

Le MJPEG est le codec le plus utilisé pour les captures vidéo des ensembles cartes d'acquisition et logiciels d'édition vidéo. La conservation d'une bonne qualité d'image produit toutefois de gros fichiers.

Le format MJPEG est un format non normalisé. Les solutions M-JPEG, au début des années 90, ont été développées sans concertations par des fabricants, conduisant à MPEG des solutions propriétaires et à des fichiers très souvent incompatibles entre eux.

- **MPEG** (Moving Picture Experts Group)

Les formats MPEG sont des formats de compression avec pertes pour les séquences vidéo.

Le groupe MPEG (Moving Picture Experts Group), est un groupe d'experts créé en 1988 et chargé du développement de normes internationales pour la compression, la décompression, le traitement et le codage de la vidéo, de l'audio et de leur combinaison, de façon à satisfaire un large panel d'applications. Les formats produits par MPEG sont ouverts, mais non libres

- **DivX** (Digital Video Express)

  - **Extension : .avi**
  - Codec vidéo propriétaire et fermé proposé par DivX Inc., conçu à partir de MPEG-4 part 2, ce dernier ayant étant modifié afin d'y ajouter la possibilité de compresser le son au format MP3. Cela permet ainsi d'obtenir des vidéos compressées très peu volumineuses avec une perte de qualité raisonnable. Ainsi le format DivX permet de stocker un film complet de plusieurs heures sur un CD-ROM de 650 ou 700 Mo. **[S17]**

9.2 Les Conteneurs vidéo

Les conteneurs les plus courants sont les suivants :

- **AVI** (Audio Video Interleave)

  - **Extension : .avi**
  - Format propriétaire et ouvert.

- Le format AVI, développé par Microsoft, très répandu et lisible sur tous les lecteurs vidéo, c'est le format d'encapsulage le plus populaire. Dans un fichier

AVI, chaque piste audio et/ou vidéo peut théoriquement être compressée par n'importe quel codec.

➢ RealMedia

- **Extension : .rm, .ram, .rpm**
- Format développé par Real Networks.
- Format propriétaire et fermé.
- Lisible uniquement avec Realplayer.
- Conteneur de flux audio (notamment : RealAudio) et vidéo (notamment : RealVideo)

➢ MP4

- **Extension : .mp4, .mp4a, .mp4v, .m4P**
- Conteneur officiel pour la norme MPEG-4. **[S17]**

10. Hiérarchie des données vidéo

La hiérarchie des données vidéo est présentée dans la **Figure II.5**

10.1 Séquence vidéo (Video Sequence)

Qui commence par un en-tête de séquence, contient un ou plusieurs groupes d'images et s'achève par un code de fin de séquence.

10.2 Groupe d'images (Group of Pictures)

Qui regroupe un en-tête et une série d'une ou plusieurs images facilitant l'accès direct.

10.3 Image (Picture)

Qui est l'unité élémentaire pour le codage de la séquence vidéo. Une image est un groupe de trois matrices rectangulaires qui représentent la luminance (Y) et la chrominance (Cb et Cr).cette représentation YCbCr est équivalente à celle RGB (Red, Green, Blue). Elle lui est préférable, car l'œil étant plus sensible à la luminosité qu'a la chrominance, il n'est pas nécessaire de stocker autant d'informations dans les matrices Cb et Cr que dans la matrice Y, alors qu'en RGB, dimension deux fois plus petite que la matrice Y.

10.4 Tranche (Slice)

Macroblocs ordonnés de gauche à droite, puis de haut en bas. En cas d'erreur dans la tranche, le décodeur passe à la suivante. Plus il y a de tranches, meilleur est le traitement des erreurs, mais fait perdre de la place.

10.5 Macroblocs (Macroblocks)

C'est une matrice rectangulaire de dimension 2 et constituée de blocs.

10.6 Blocs (Block)

C'est un ensemble des valeurs de luminance et chrominance de 8 lignes de 8 pixels. **[S19]**

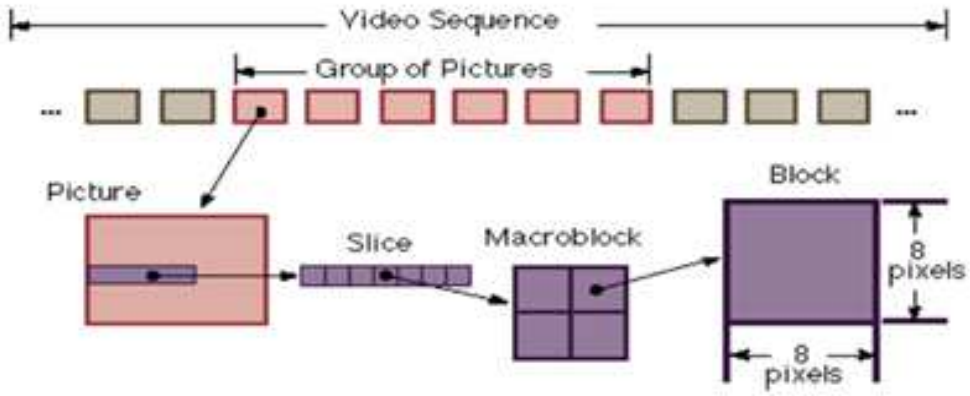

Figure II.5 : Hiérarchie des données dans le flux vidéo. [S19]

11. Représentation d'une séquence vidéo

Une séquence vidéo brute est une suite d'images fixes, qui peut être caractérisée par trois principaux paramètres :

11.1 La résolution en luminance

Détermine le nombre de nuances ou de couleurs possibles pour un pixel. Celle-ci est généralement de 8 bits pour les niveaux de gris et de 24 bits pour les séquences en couleurs.

11.2 La résolution spatiale

Définit le nombre de lignes et de colonnes de la matrice de pixels.

11.3 La résolution temporelle

Est le nombre d'images par seconde. La valeur de ces trois paramètres détermine l'espace mémoire nécessaire pour stocker chaque image de la séquence. Cet espace mémoire est caractérisé par le débit, qui est le coût de stockage pour une seconde (capacité mémoire nécessaire pour stocker une seconde de vidéo)**. [12]**

12. Conclusion

Dans ce chapitre, nous avons présentés une étude générale sur les caractéristiques fondamentales des vidéos dans le but de les améliorer ou d'extraire des informations pertinents. Donc, L'objectif de notre travail est la détection et suivi d'objet en mouvement dans une vidéo, les prochains chapitres sont consacrés à la présentation de différentes méthodes de détection des objets dans une sciène vidéo.

Chapitre III

Les méthodes de détection des objets en mouvement

1. Introduction

La plupart des algorithmes de suivi d'objets prenant en entrée les images fournies par une caméra fixe effectuent une première étape de détection de mouvement afin de déterminer parmi les pixels de l'image courante lesquels appartiennent à l'arrière-plan de la scène, et lesquels représentent des objets en mouvement. Ce domaine de recherche est très actif depuis les débuts de l'analyse de séquences vidéo à la fin des années 1970. Depuis cette date, le nombre d'articles publiés chaque année sur ce sujet ne cesse de croître, et en particulier depuis le milieu des années 1990, lorsque la puissance des ordinateurs grand public a permis d'envisager sérieusement un traitement en temps réel des données vidéo.

2. Méthode de Détection d'objets en mouvement

Pour pouvoir suivre les mouvements dans la vidéo, la première étape est de détecter les mouvements. Cette étape joue un rôle très important dans l'analyse vidéo.

Dans cette partie, on va étudier quatre méthodes pour détecter le mouvement :

2.1 La différence temporelle des images (méthode de mouvement)

La différence temporelle détecte la région de mouvement grâce à la différence de pixel par pixel de deux trames consécutives dans un flux vidéo **[13][15]**. Cette méthode adapte le changement de la scène.

Mais elle est moins d'efficacité parce que dans une durée du temps Δt, peut être, on détecte seulement une partie d'objet, par exemple : la main, la tête etc. Et le vide (la région où l'objet s'est déplacé l'autre lieu) est aussi détecté.

Dans ce cas là, c'est très difficile à extraire des propriétés de mouvement (la taille, la position, la vitesse etc.) et à suivre le mouvement. **[14]**

(a) **(b)**

Figure III.1: Un exemple de la différence temporelle [14]. (a) une scène simple avec deux objets, (b) les régions rouges sont la différence entre deux images consécutives.

Voici, c'est l'idée principale de cette méthode : Soit I_t l'image à l'instant t et I_{t-1} l'image à l'instant t-1. L'objet du mouvement se compose les pixels qui satisfont l'équation suivante :

$$\max (| I_t(x, y)* c - I_{t-1}(x, y)*c |, c=(R,G,B)) >= \text{seuil}$$

Comme ci-dessus a déjà expliqué, on peut améliorer cette méthode en façon suivante au lieu de soustraire l'image à l'instant t-1, on soustrait la moyenne de N images dernières [15] Soit I_{tm} la moyenne de N dernières images à l'instant t. L'objet du mouvement se compose les pixels qui satisfont l'équation suivante :

$$\max (| I_t(x, y)* c - I_{tm}(x, y)*c |, c=(R,G,B)) >= \text{seuil}$$

La moyenne de N images à l'instant t +1 est mise à jours :

$$I_{t+1}(x, y)*c = \alpha I_t(x, y)*c + (1-\alpha) I_{tm}(x, y)*c, c=(R, G, B).$$

Où $\alpha \in (0,1)$ est une constante et est décidé par la pratique.

- **Avantages**

+ Adapte le changement de la scène.

+ Détecte la région de mouvement.

- **Inconvénients**

– Ne permet pas de détecter le mouvement dans les zones uniformes intérieures à l'objet

- Ne fonctionne pas dans plusieurs cas, pour différentes raisons, telles que : la présence de bruit du capteur et les changements de luminosité de la scène qui modifient les intensités des pixels.

2.2 Double de différence temporelle et caractère de contour

Dans cette manière, on utilise aussi la différence de pixel par pixel des trames consécutives dans un flux vidéo comme la 1ère méthode. Mais on va utiliser trois trames consécutives. **[16]**Cette façon nous donne le résultat meilleur que celui de la 1ère méthode, tandis qu'elle adapte aussi le changement de la scène.

Voici, c'est l'idée principale de cette méthode : Soit I l'image à l'instant t, I l'image à l'instant t-1 et I l'image à l'instant t-2.

L'objet du mouvement se compose les pixels qui satisfont l'équation suivante :

$$I_1(x, y) = \max (| I_t(x, y)* c - I_{t-1}(x, y) * c |, c=(R,G,B)) >= \text{seuil}$$

$$I_2(x, y) = \max (| I_{t-1}(x, y)* c - I_{t-2}(x, y) * c |, c=(R,G,B)) >= \text{seuil}$$

$$I_{\text{résultat (x, y)}} = I_1(x, y)\, I_2(x, y).$$

Nous extrayons la région des objets mouvants par la double méthode de différence. Dans le cas, l'objet se déplace lentement ou il n'y a qu'une partie d'objet se déplace, nous ne pouvons pas obtenir complètement la forme d'objet. Peut-être, un objet sera divisé en plusieurs régions. Nous utilisons donc le caractère de contour pour combiner ces régions.

Après avoir masqué l'image à l'instant t-1 avec la région mouvante obtenue à partir de la double méthode de différence, le contour est calculée dans la région où le mouvement est produit.

Le contour est calculé à partir de l'image F qui est l'image à l'instant t-1 masquée avec

$R = I_1 (x, y) \cup I_2 (x, y)$. Le contour est représenté par :

$$G = \begin{bmatrix} F_x^2 & F_x F_y \\ F_x F_y & F_y^2 \end{bmatrix}$$

En utilisant un caractère de gradient avec l'intensité de l'image masquée F. En chaque pixel que le mouvement est détecté par l'intermédiaire de la double méthode de différence dans l'armature courante, le coefficient est calculé. C'est-à-dire, les pixels commandés par le caractère maximal de contour sont choisis en tant que points de caractère de contour parce que les valeurs propres minimum plus grandes sont les caractères plus forts de contour.

Avec cette façon, à l'instant t, on va obtenir des objets mouvant à l'instant t-1. C'est-à-dire, on est en retard de 0,5 seconde. Autre chose, si l'objet ne se déplace pas pendant quelques secondes, cette manière ne peut pas détecter le mouvement. Dans ce cas, nous utilisons le contour d'objet avec le résultat de dernière étape pour détecter le mouvement. **[17]**

- **Avantages**

+ Le résultat de cette méthode est meilleur par rapport la méthode précédant.
+ Détecte la région de mouvement.
+ Utilise trois trames.

- **Inconvénient**

– Si l'objet ne se déplace pas pendant quelques secondes, cette manière ne peut pas détecter le mouvement.

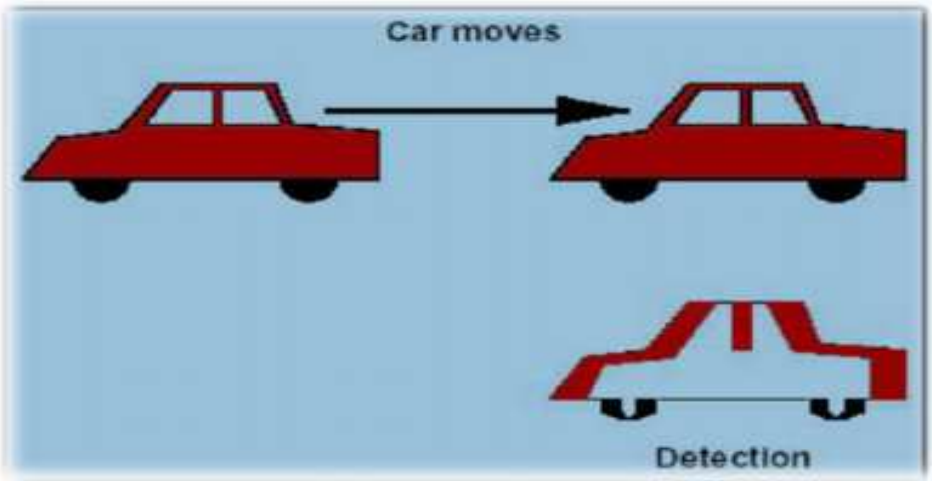

Figure III.2 : Détection de mouvement par différence temporel [21]

2.3 La soustraction de l'image de fond (méthode de différence)

La troisième méthode, on utilise une image de fond. Cette méthode est très populaire et elle est utilisée par plusieurs applications. Cette façon détecte la région de mouvement en soustrayant pixel par pixel l'image courante à l'image de fond.

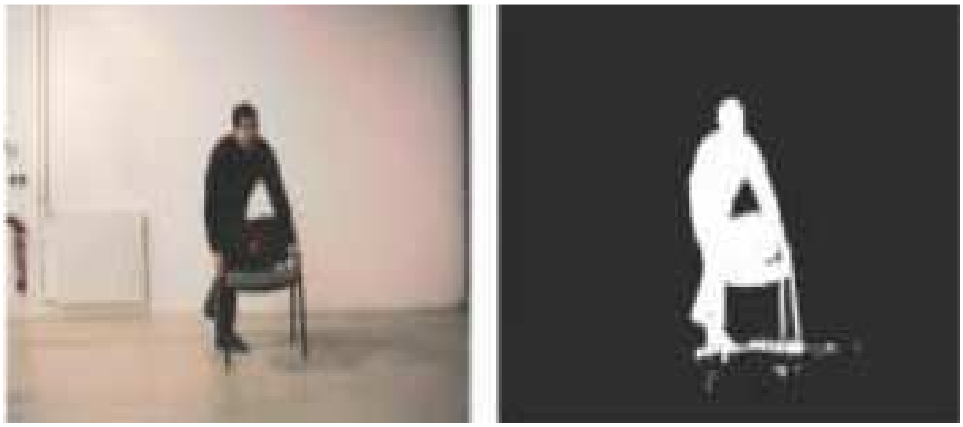

Figure III.3 : Exemple soustraction du fond [19]

Soit I_t l'image à l'instant t. B_t est l'image de fond à l'instant t. L'objet du mouvement se compose les pixels qui satisfont l'équation suivante :

$$\max (| I_t(x, y)*c - B_t(x, y)*c|, c = (R,G,B)) > = S(x, y)$$

Où S_t (x, y) est seuil de la position (x, y) à l'instant t. **[17]**

- **Création d'image de fond**

Il existe trois manières pour créer l'image de fond :

- Le calcul de moyenne de N premières images.
- Choisir l'image de fond qui a peu de changement ou qu'il n'y a pas de mouvement pendant une durée longe.
- Choisir la première image dans le flux vidéo. **[S24]**

- **<u>Les Avantages</u>**

+ Cette méthode est très populaire.
+ Utilisée par plusieurs applications.
+ Détecte l'objet complètement.

- **Inconvénient**

– Cette technique se limite aux caméras en position fixe.

2.4 Méthode de détection des objets selon la Couleur

La quatrième méthode est basée sur la couleur d'objets détectés.

Lecture pixel par pixel de la valeur de celui-ci en rouge vert et bleu. Puis test sur le pixel, suivant la couleur à reconnaitre le rouge, le vert et le bleu doivent être supérieur ou inferieur au seuil impose. S'ils respectent ces conditions alors on incrémente. **[9]**

- **Test**

Condition de supériorité ou d'infériorité sur chaque couleur.

- **Paramètres**

Les paramètres principaux sont les différents seuils des couleurs (RGB).

Dans l'exemple utilise, les conditions sont R>120, G<100, B<100. **[S2]**

- **Résultats**

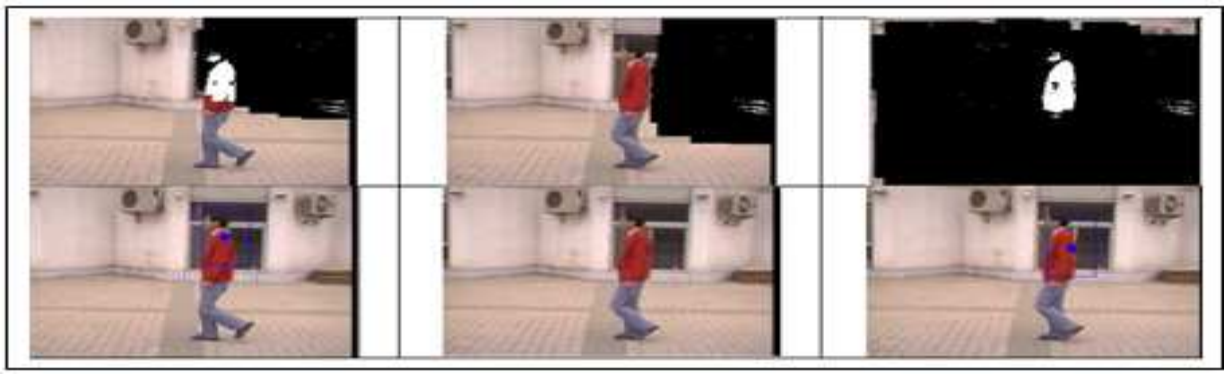

Figure III.4 : Exemple1, pixels de couleur rouge reconnus dans les particules et reconnaissance de l'objet. [S2]

Sur les images ci-dessus, une erreur apparait, l'objet n'est pas reconnu sur le frame du Milieu, cette erreur est du a la reconnaissance dans une particule.

En ajoutant un paramètre, cette erreur peut être évitée, ce paramètre est un seuil applique a la moyenne de la particule, en effet si la moyenne est trop faible alors elle est considérée comme nulle.

Le même exemple repris avec l'ajout de ce seuil donne le résultat suivant. **[S2]**

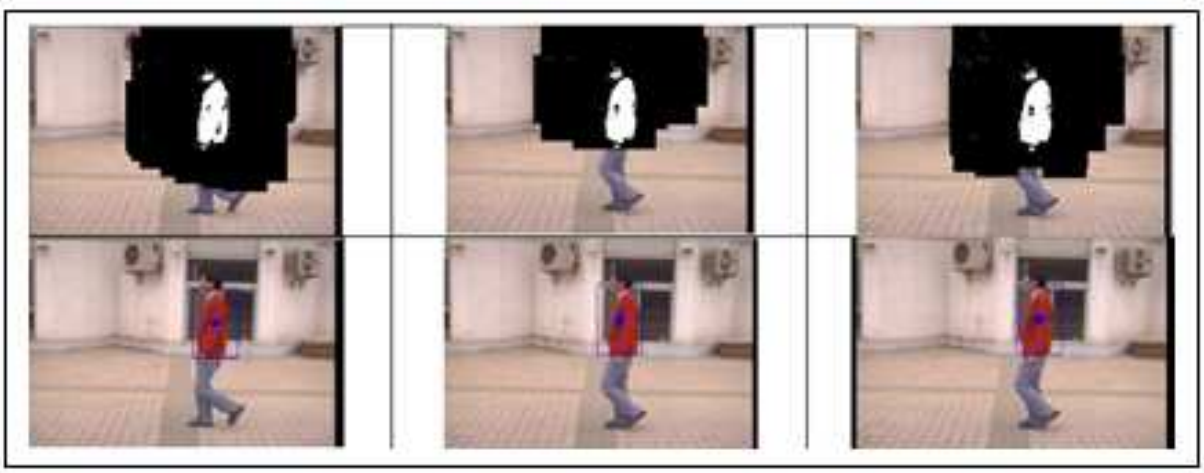

Figure III.5 : Exemple 2, reconnaissance d'un objet à l'aide de la couleur après modification d'un paramètre. [S2]

- **Avantage**

\+ C'est une méthode assez simple à mettre en œuvre, il suffit de reconnaitre une couleur qui ressort par rapport au reste de l'image.

- **Inconvénients**

– Il existe beaucoup de nuance de couleur (256^3) il faut alors avoir une grande base de données puis choisir la couleur ou définir a chaque fois la couleur souhaitée ce qui sous entend que ces valeurs soit connues.

– Si un autre objet possède la même couleur on ne peut les différencier, excepte s'il y a une grande différence de taille entre les deux objets et que l'objet à reconnaitre est le plus grand.

– Si notre objet change de couleur au cours de la vidéo, la couleur à reconnaitre devra également être modifiée.

3. Suivi d'objets en mouvement (tracking)

Le tracking est un procédé de localisation d'un (ou plusieurs) objet en mouvement en temps réel en utilisant une caméra. Un algorithme analyse les photogrammes de la vidéo et localise les cibles en mouvement sur la vidéo.

La principale difficulté dans le tracking sur une vidéo est d'associer la localisation des cibles dans les photogrammes successifs, particulièrement lorsque les objets bouge rapidement par rapport au frame rate. Les systèmes de tracking sur une vidéo normalement utilisé, utilise un modèle en mouvement qui décrit comment l'image de la cible peu changer en tenant compte du mouvement possible de l'objet traquer. **[S2]**

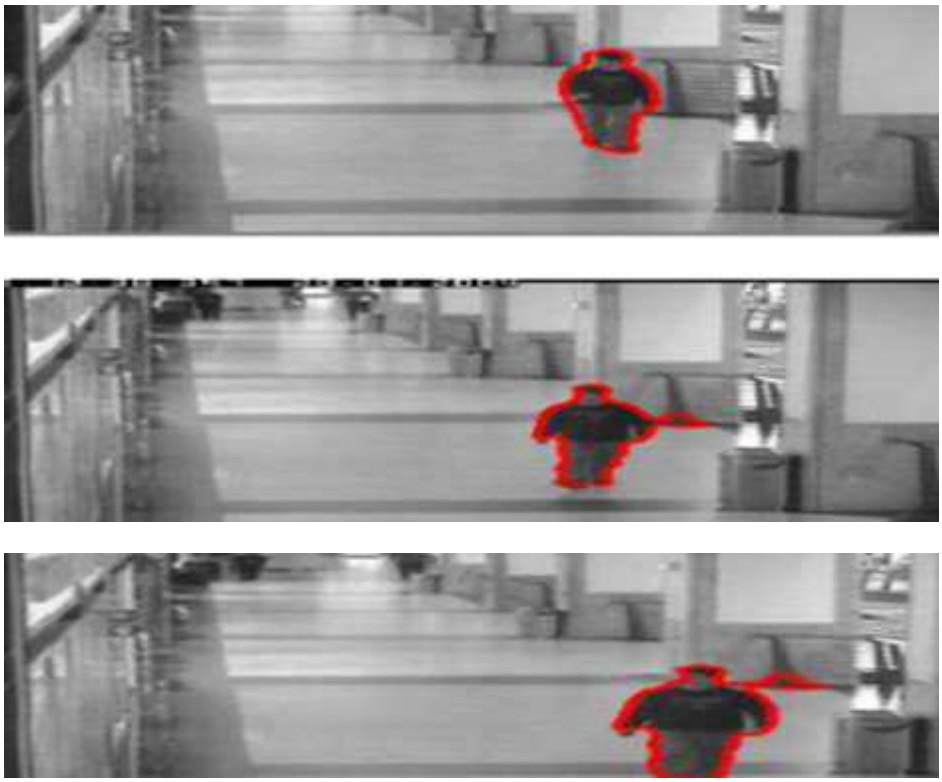

Figure III.6 : Suivi un objet. [20]

4. Conclusion

Les différents algorithmes étudies présentent des avantages et des inconvénients pour différentes situations. L'algorithme de couleur est idéal pour le suivi d'un objet d'une couleur unique.

Nous sommes intéressés par la méthode de détection des objets selon la couleur qui sera le but de notre travail dans le chapitre qui suit.

Chapitre IV

Résultat de détection d'un objet en mouvement selon la couleur

1. Introduction

Nous sommes intéressés dans notre projet de fin d'étude par la conception d'une application de détection des objets en mouvement selon leur couleur dans une scène vidéo capturée par une webcam en utilisant la bibliothèque visuelle OpenCV et le logiciel Visual studio.

2. Environnement du travail

Dans cette section, nous présentons les environnements matériels et logiciels de notre travail.

2.1 Environnement matériel ou le hardware:

Un ordinateur TOSHIBA avec les caractéristiques suivantes:

- Processeur: Intel (R) Core (TM)2 Duo CPU T5750 @2.00 GHz 2.00GHz
- RAM: 3.00 GO.
- Disque Dur: 500 GO.
- Webcam intégrée de marque TOSHIBA Chicony USB 2.0 Camera.

2.2 Environnement immatériel ou le software :

- Système d'exploitation : Microsoft Windows 7 - 64 bits
- Logiciel : Microsoft Visual Studio version 2008
- La bibliothèque : OPENCV version 2.3.1

3. Fonctions utiles de la bibliothèque OpenCV

Dans la suite de ce chapitre, les différentes fonctions et classes utilisées de la bibliothèque OpenCV sont détaillées et expliquées.

3.1 Pour le traitement d'image

Pour la lecture de l'image, il est nécessaire de connaitre certaines fonctions

IplImage* cvLoadImage(const char* filename, int iscolor=CV_LOAD_IMAGE_COLOR)

Cette ligne charge l'image. La fonction **cvLoadImage** détermine le format de l'image a partir du nom passe en argument, réserve la mémoire nécessaire pour la structure de données de l'image et renvoi un pointeur. Ce pointeur s'utilise pour manipuler tant l'image que les données. Le second argument spécifie la couleur de l'image chargée (RGB, noir et blanc ou échelle de gris).

cvNamedWindow("Name", CV_WINDOW_AUTOSIZE);

cvNamedWindow() ouvre une fenêtre qui peut contenir et montrer une image. Le premier argument correspond au nom à donner a la fenêtre. Le second défini les propriétés de la fenêtre. Ici CV_WINDOW_AUTOSIZE, signifie que la fenêtre prendra la taille de l'image.

cvShowImage("Name", img);

Cette fonction permet d'afficher une image de type ≪ IplImage ≫ dans la fenêtre créée au préalable.

cvWaitKey(0);

Cette ligne de code interrompt, momentanément ou indéfiniment, l'exécution du programme. La valeur positive, entrée en argument, correspond au temps en millisecondes avant que le programme ne s'exécute de nouveau. S'il s'agit d'une valeur nulle ou négative, le programme attend que l'utilisateur tape une touche du clavier pour s'exécuter de nouveau.

cvReleaseImage(&img);

Cette fonction libère la mémoire réservée pour contenir les données de l'image.

cvDestroyWindow("Exemple");

Cette fonction permet de détruire la fenêtre.

IplImage* cvCloneImage(const IplImage* image)

Cette fonction permet de créer une copie complète de l'image passée en argument.

IplImage* cvCreateImage(CvSize size, int depth, int channels)

Cree un emplacement et enregistre les données de l'image. Le premier argument spécifie la taille de l'image (hauteur et largeur). Le second spécifie la profondeur en bits des éléments (pixels) de l'image. Le troisième spécifie le nombre de caractères par pixel.

void cvCvtPixToPlane(const CvArr* src, CvArr* dst0, CvArr* dst1, CvArr* dst2, CvArr* dst3)

Cette fonction sépare une matrice multi-canaux(src) vers plusieurs matrices n'ayant qu'un canal (dst0, dst1, dst2 y dst3).

3.2 Classe ImageOpenCV

La classe ImageOpenCV a été crée afin de faciliter et regrouper certaines fonctions. Elle permet de déterminer les caractéristiques d'une image, son nombre de lignes, de colonnes.

inline T & GetPixel(const int &c, const int &r)

Cette fonction est une des plus importantes fonctions utilisées. Elle permet, grâce aux paramètres rentres en argument, d'accéder aux données du pixel, par exemple les composantes RGB du pixel. Les paramètres rentres en argument correspondent respectivement a la colonne et a la ligne du photogramme.

int GetNCols()

Renvoi le nombre de colonnes de l'image.

int GetNRows()

Renvoi le nombre de lignes de l'image.

3.3 Pour lire la video

Pour lire une vidéo et la travailler, les fonctions suivantes sont utiles

CvCapture* capture = cvCreateFileCapture(argv[1]);

La fonction cvCreateFileCapture() prend en argument le nom de du fichier AvI à être charge et renvoi un pointeur a une structure de données CvCapture. Cette structure contient toute les informations sur le fichier AVI qui a été lu. Le pointeur pointe au début de la video.

cvReleaseCapture(&capture);

Libère la mémoire utilisée avec la structure CvCapture. Cette fonction ferme également tous les fichiers ouvert et qui font référenceur fichier AVI.

CvVideoWriter* cvCreateVideoWriter(const char* filename, int fourcc, double fps, CvSize frame_size, int is_color=1)

Cette fonction crée une structure pour écrire une video.

Les arguments nécessaires sont

- Filename : Nom de la video de sortie.

Fourcc : Code de quatre caractères du codec utilise pour comprimer le photogramme

- Fps.Nombre : de photogrammes par seconde de la video que l'on veut créer.
- Frame_size : Taille du photogramme.
- Is_color : Permet de définir l'échelle de couleur de l'image. RGB, gamme de gris ou noir et blanc.

IplImage* cvQueryFrame(CvCapture* capture)

Lit puis renvoi le photogramme d'une camera ou d'un fichier.

double cvGetCaptureProperty(CvCapture*capture,int property_id);

Obtient les propriétés de la video. Propriété et video spécifiées en argument. Voici quelques exemples de propriétés utiles au programme :

- CV_CAP_PROP_FPS. Nombre de photogramme par seconde.

- CV_CAP_PROP_FOURCC. Code de quatre caractères du codec utilise

- CV_CAP_PROP_FRAME_COUNT. Nombre de photogrammes dans la video

int cvWriteFrame(CvVideoWriter* writer, const IplImage* image)

Cette structure permet d'ajouter un photogramme à la video.

void cvReleaseVideoWriter(CvVideoWriter** writer)

Cette fonction finalise l'enregistrement de la video et libère la structure.

3.4 Classe ImageSourceVideo

La classe ImageSourceVideo, comme la classe Image OpenCV regroupe quelques fonctions, dans ce cas ces fonctions renvoi en majorité les caractéristiques de la video.

IplImage * QueryFrame(bool &stop)

S'il s'agit du dernier photogramme de la video, la variable ≪ stop ≫ est mise à vrai, sinon la fonction cvQueryFrame est appelée.

- *int GetNCols() ;*
- *int GetNRows() ;*
- *int GetNumFrames() ;*
- *double GetFramesPerSecond() ;*
- *double GetFourcc() ;*

Ces cinq fonctions renvoient, respectivement, le nombre de colonnes, de lignes, de frame, de frames par seconde et FOURCC.

FourCC est le code du codec utilise pour l'écriture, sous forme d'entier.

4. Implémentation

4.1 Détection d'un objet en couleur rouge dans image par la bibliothèque OpenCV

Nous avons dans cette image (**figure IV.1**) 4 objets de couleur différents, nous sommes intéressé par la détection de l'objet qui une couleur rouge (avec rvb = **`(255, 0, 0))`**

Nous utilisons la fonction **inrange** de la bibliothèque OpenCV qui permettre de détecté un objet avec une couleur bien précise avec les paramètres suivante :

inRange (src, Scalar(0, 0, 0), Scalar(255, 0, 0), redOnly);

La fonction **inrange de OpenCV** est définie comme suite :

inRange(src, Scalar(lowBlue, lowGreen, lowRed), Scalar(highBlue, highGreen, highRed), redColorOnly);

tel que :

src : image entrée qui contient des objets avec des couleurs differents

Scalar(lowBlue, lowGreen, lowRed): interval de couleur bas

Scalar(highBlue, highGreen, highRed): interval de couleur haut

Redcoloronly : résultat de sortie qui contient l'objet qui a la couleur sélectionné dans l'interval bas et haut.

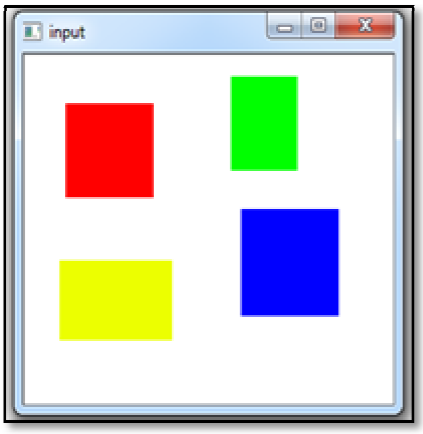

Figure IV.1 : plusieurs objets avec différents couleur

- **Voila le résultat de l'objet détecté qui a une couleur rouge**

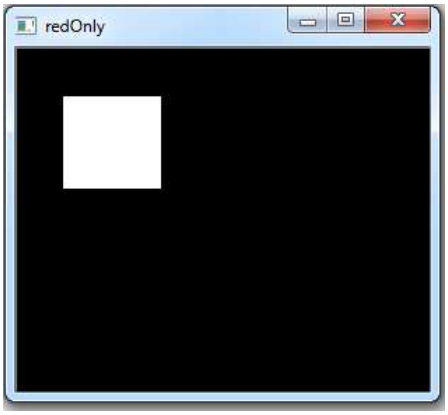

Figure IV.2 : détection de l'objet de couleur rouge

- **Voila le programme**

```
#include <opencv2/core/core.hpp>
#include <opencv2/highgui/highgui.hpp>
#include <opencv2/imgproc/imgproc.hpp>
#include <iostream>
using namespace std;
using namespace cv;
Mat redFilter(const Mat& src)
{
    assert(src.type() == CV_8UC3);
 Mat redOnly;
    inRange(src, Scalar(0, 0, 0), Scalar(0, 0, 255), redOnly);
```

```
    return redOnly;
}
int main(int argc, char** argv)
{
    Mat input = imread("test.png");
    imshow("input", input);
    waitKey();
    Mat redOnly = redFilter(input);
    imshow("redOnly", redOnly);
    waitKey();
    return 0;
}
```

4.2 Détection de plusieurs objets de couleur différents dans une image

Avec le même principe et la même la fonction inrange d'opencv nous pouvant détecter plusieurs objets de différents couleurs dans la même image

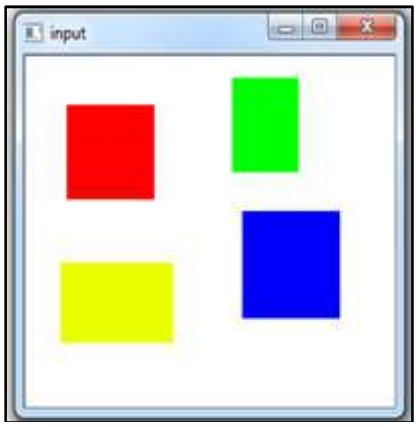

Figure IV.3: Image source contient 4 objets de couleur différents

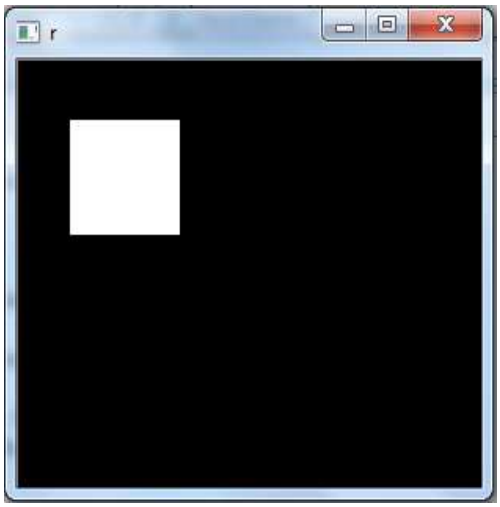

Figure IV.4 : Objet rouge détecté

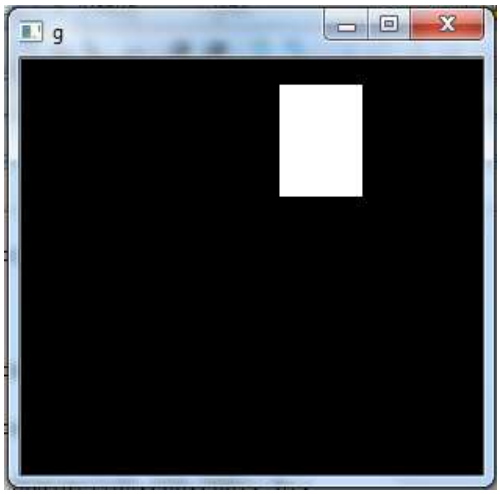

Figure IV.5 : Objet vert détecté

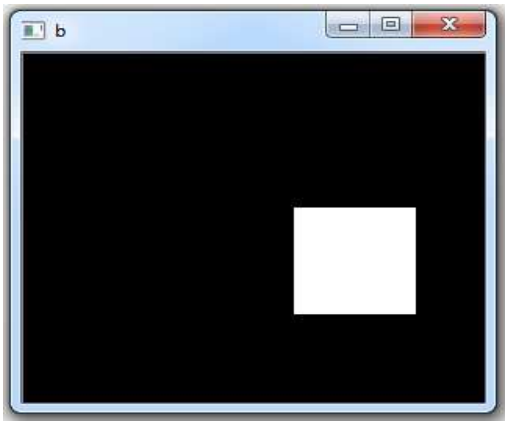

Figure IV.6 : Objet bleu détecté

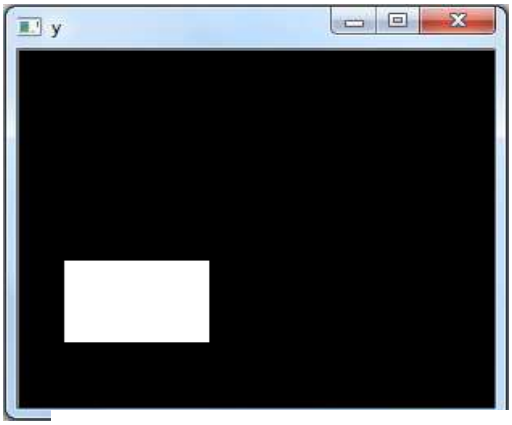

Figure IV.7: Objet jaune détecté

- **Lorsqu'en exécute le masque voila les trois objets détectés avec différents couleurs**

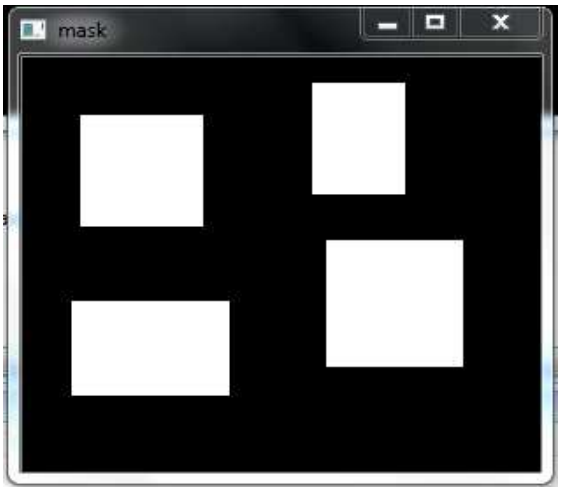

Figure IV.8 : Masque des 4 objets détectés

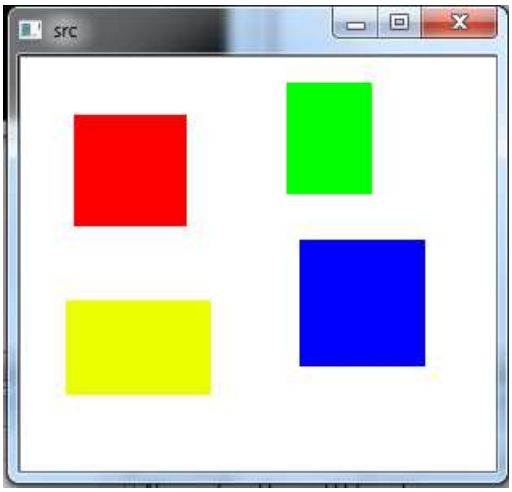

Figure IV.9: Détection de plusieurs objets de couleurs différents

- **Voila le programme**

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
```

```
using namespace cv;
using namespace std;

int main()
{
   Mat src;
   Mat hsv;
   Mat dst;
   Mat r,g,b,y;

      src=imread("color.png");

   Mat mask=Mat::zeros(src.rows,src.cols, CV_8U);

   cvtColor(src,hsv,CV_BGR2HSV);

   inRange(hsv,Scalar(0,100,120) , Scalar(10,255,255), r);
   inRange(hsv,Scalar(50,100,120) ,Scalar(70,255,255), g);
   inRange(hsv,Scalar(110,100,120) , Scalar(130,255,255), b);
   inRange(hsv,Scalar(20, 20, 20), Scalar(35, 255, 255), y);

   mask=y+r+g+b;

   src.copyTo(dst,mask );
imshow("r",r);
waitKey();
imshow("g",g);
waitKey();
imshow("b",b);
waitKey();
imshow("y",y);
waitKey();
```

```
imshow("mask",mask);
waitKey();
imshow("src",src);
waitKey();
imshow("result",dst);


   waitKey();
}
```

4.3 Détection d'un objet en mouvement de couleur rouge dans une scène vidéo

Pour la détection d'un objet en mouvement dans une scène vidéo nous utilisons la fonction `cvInRangeS` de la bibliothèque opencv qui est définie comme suite :

cvInRangeS (p_originalImage,CV_RGB(175, 0, 0),CV_RGB(256,100,100),p_processedImage);

tel que :

p_originalImage : la scène vidéo d'entré

CV_RGB(175, 0, 0),CV_RGB(256,100,100) : l'intervalle de couleur RGB

p_processedImage: scène vidéo de sortie qui contient l'objet détecté

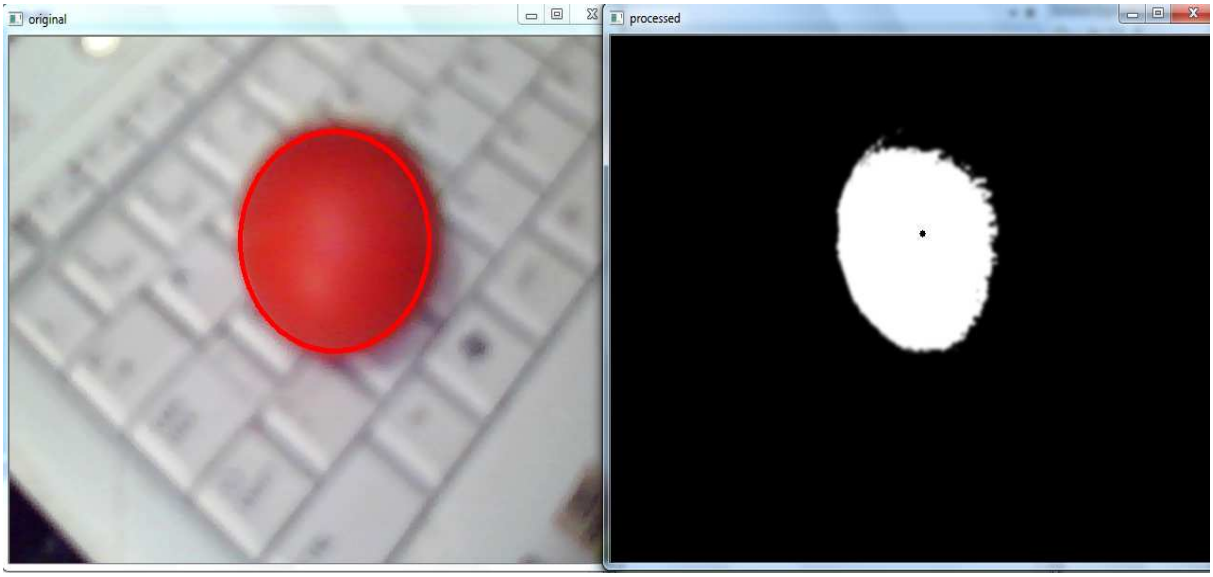

Figure IV.10 : détection d'un objet de couleur rouge dans une scène

- **Résultat de deux objets en mouvement qui contient des couleurs différents**

- **Résultat deux objets contient presque la même couleur**

Un seul objet qui sera détecté selon l'intervalle de couleur RGB indiqué dans la fonction `cvInRangeS`

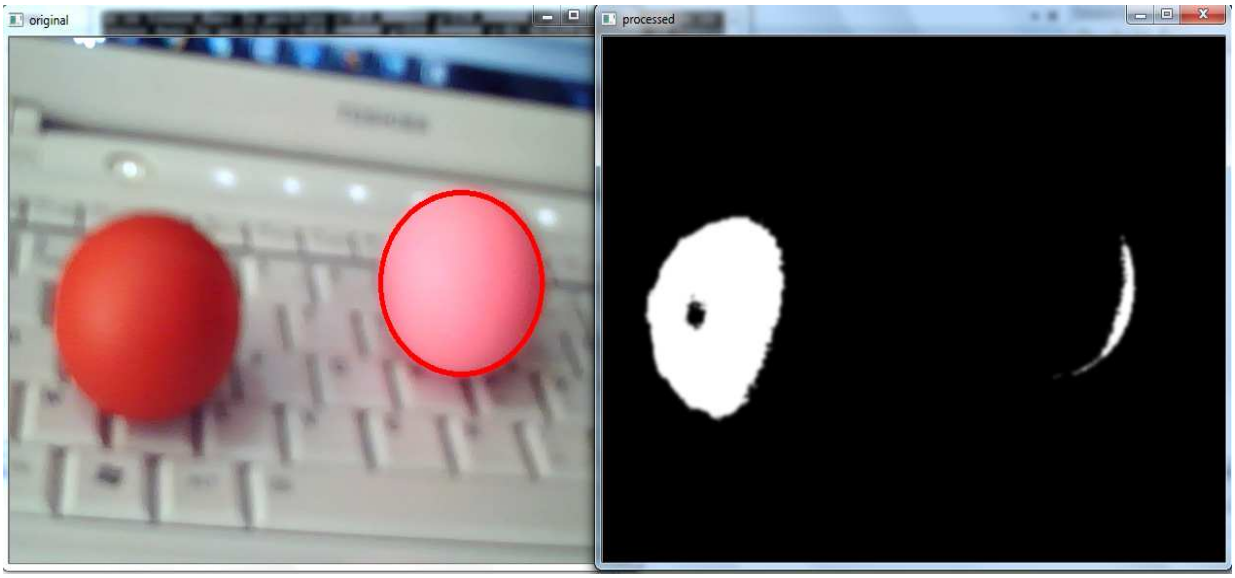

Figure IV.11 : détection d'un objet dans une scène vidéo selon l'intervalle de couleur

- **Résultat deux objets complètement différents dans leur couleur**

Un seul objet qui sera détecté selon l'intervalle de couleur RGB indiqué dans la fonction `cvInRangeS`

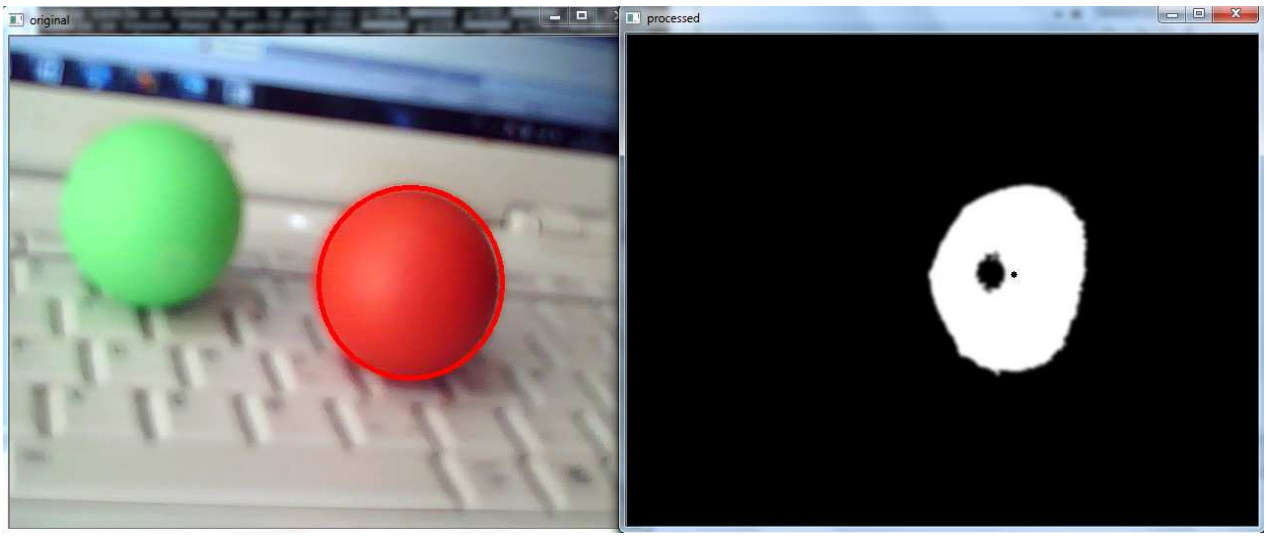

Figure IV.12 : deux objets de couleur différents.

- **Voila le programme**

```
#include<opencv\cvaux.h>
#include<opencv\highgui.h>
#include<opencv\cxcore.h>
#include<stdio.h>
#include<stdlib.h>
int main(int argc, char* argv[])
{
CvSize size640x480=cvSize(640,480);
CvCapture* p_capWebCam;
IplImage* p_originalImage;
IplImage* p_processedImage;
CvMemStorage* p_strStorage;
CvSeq* p_seqCircle;
float* p_fltRadiusXY;
int i;
char checkforEsc;
p_capWebCam=cvCaptureFromCAM(0);
if(p_capWebCam==NULL){
printf("erreur de capture video");
return -1;
}
cvNamedWindow("original",CV_WINDOW_AUTOSIZE);
cvNamedWindow("processed",CV_WINDOW_AUTOSIZE);
p_processedImage=cvCreateImage(size640x480,IPL_DEPTH_8U,1);
while(1)
{
p_originalImage=cvQueryFrame(p_capWebCam);
if(p_originalImage==NULL)
{
printf("probleme d'ouverture video");
return -1;
}
cvInRangeS(p_originalImage,CV_RGB(175, 0,0),CV_RGB(256,100,100)
```

```
,p_processedImage);
p_strStorage=cvCreateMemStorage();
cvSmooth(p_processedImage,p_processedImage,CV_GAUSSIAN,9,9);
p_seqCircle=cvHoughCircles(p_processedImage,p_strStorage,CV_HOUGH_GRAD
IENT,2,p_processedImage->height/4,100,50,10,400);
for(i=0;i<p_seqCircle->total;i++)
{
p_fltRadiusXY=(float*)cvGetSeqElem(p_seqCircle,1);
printf("le cercle se trouve dans la position x=%f y=%f
r=%f",p_fltRadiusXY[0],p_fltRadiusXY[1],p_fltRadiusXY[2]);
cvCircle(p_processedImage,cvPoint(cvRound(p_fltRadiusXY[0]),cvRound(p_
fltRadiusXY[1])),3,CV_RGB(0,255,0),CV_FILLED);

cvCircle(p_originalImage,cvPoint(cvRound(p_fltRadiusXY[0]),cvRound(p_f
ltRadiusXY[1])),cvRound(p_fltRadiusXY[2]),CV_RGB(255,0,0),3);
}
cvShowImage("original",p_originalImage);
cvShowImage("processed",p_processedImage);
cvReleaseMemStorage(&p_strStorage);
checkforEsc=cvWaitKey(10);
if(checkforEsc==27) break;
}
cvReleaseCapture(&p_capWebCam);
cvDestroyWindow("original");
cvDestroyWindow("processed");
return(0);
}
```

4.4 Détection d'un objet en mouvement de couleur jaune dans une scène vidéo

Dans cette vidéo, nous avons deux objets de couleur différents, Il y a un objet de couleur jaune et un autre objet de couleur bleu.

Pour détecter l'objet de couleur jaune en modifiant la fonction *cvInRangeS de la façon suivante :*

cvInRangeS(p_originalImage,cvScalar(23,41,133),cvScalar(40,150,255), p_processedImage);

- **Voila le résultat de la détection de l'objet jaune capturé par la webcaméra**

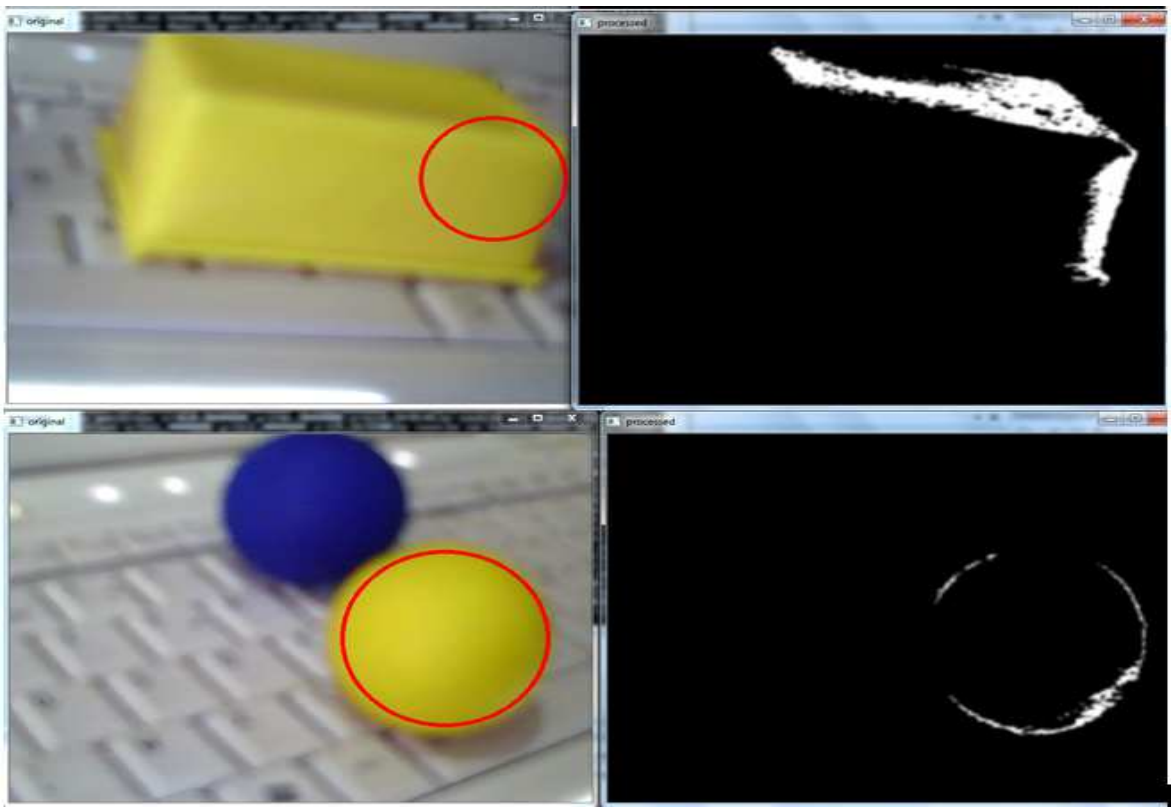

Figure IV.13 : détection d'un objet de couleur jaune

4.5 Détection d'un objet en mouvement de couleur bleu dans une scène vidéo

Pour détecter l'objet de couleur bleu en modifiant la fonction *cvInRangeS de la façon suivante :*

cvInRangeS(p_originalImage,cvScalar(100,10,10),cvScalar(255,100,100), p_processedImage);

- **Voila le résultat de la détection de l'objet bleu capturé par la webcaméra**

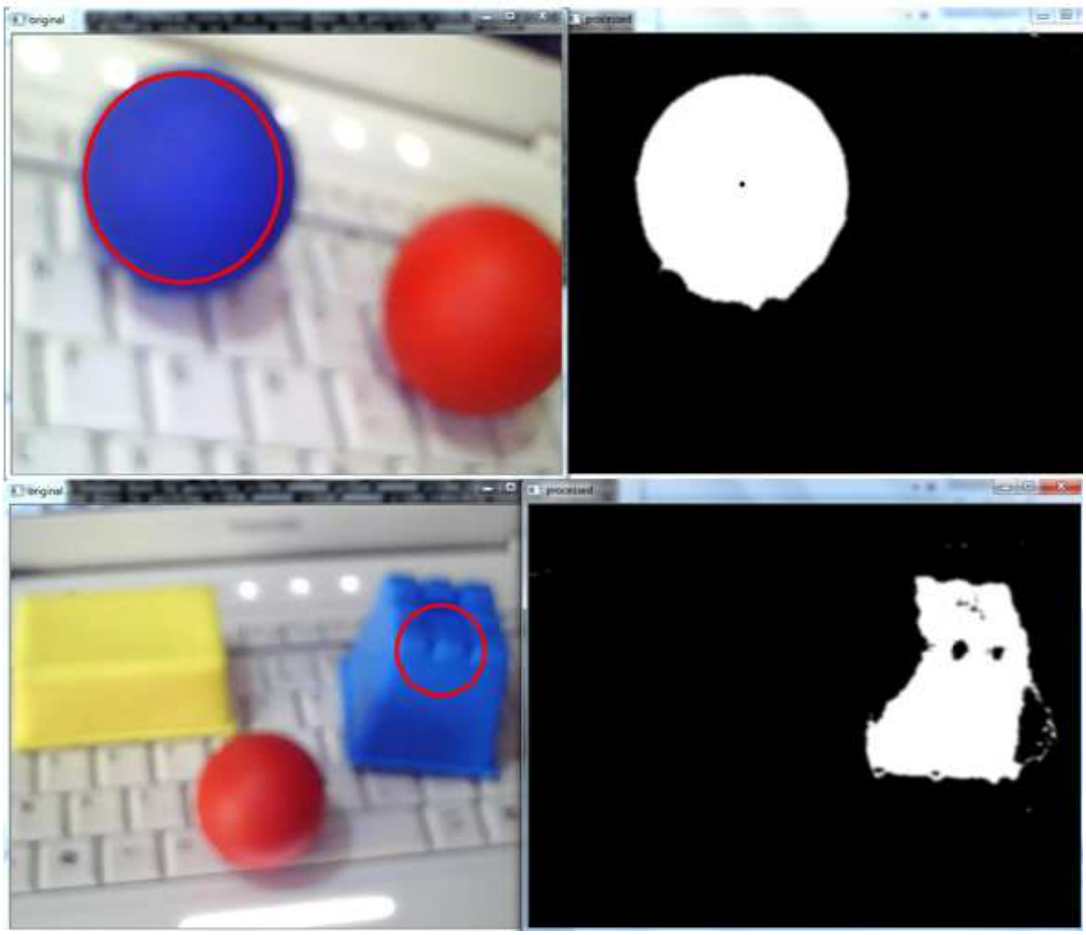

Figure IV.14: détection d'un objet de couleur bleu

5. Conclusion

Le but du projet était de suivre plusieurs objets en même temps d'une vidéo. Les différents algorithmes étudies ont tous montre des avantages et des inconvénients. Certains algorithmes sont simples d'utilisation et de compréhension mais ont des limites.

L'algorithme de reconnaissance des couleurs est simple et pratique mais limite car il ne reconnait que les objets a couleur unique.

Dans le futur, il sera intéressant de terminer le projet, de reprendre le travail effectue et de le compléter afin de pouvoir suivre le mouvement de plusieurs objets de couleurs différents à la fois.

CONCLUSION GENERALE

Conclusion générale

Tout au long de la préparation de notre projet de fin d'études, nous avons essayé de mettre en pratique les connaissances acquises durant nos études universitaires et cela dans le but de réaliser une application de détection et suivi un objet en mouvement sur une sciène vidéo.

Ce projet de fin d'étude consiste en une étude de différentes méthodes de suivi d'un ou plusieurs objets dans une vidéo. Nous sommes intéressés par la méthode de détection des objets en mouvement selon leur de couleur.

La bibliothèque OpenCV est un élément important dans la réalisation et le développement de notre projet et cela revient à la richesse de différentes fonctions qui le compose surtout dans le traitement des images et des vidéos.

Les différents algorithmes étudies ont tous montre des avantages et des inconvénients. Certains algorithmes sont simples d'utilisation et de compréhension mais ont des limites.
L'algorithme de reconnaissance des couleurs est simple et pratique mais limite car il ne reconnait que les objets a couleur unique.

Chacun des algorithmes étudies sont plus efficaces que les autres dans des cas précis, du fait de leur simplicité ou de leur précision. Ils ont néanmoins un point commun, il est nécessaire de faire des compromis, il est nécessaire de reconnaitre l'objet a chaque instant, mais sans commettre d'erreurs (fausse détection).

Comme perspective de notre travaille, nous souhaitons pour les prochains projets de fin d'étude, élargir notre travail de détection et suivi d'un objet en mouvement mais selon différents couleurs.

Il est aussi intéressant de faire une étude comparative entre les différentes méthodes de détection des objets en mouvement pour voir leur performance réel dans le domaine de la vision par ordinateur

BIBLIOGRAPHIE

Articles, livres et thèses

[1]: A. Gagalowic, de cours dispensés à l'ESIEA, 2002.

[2]: Benchrife Ali, " Traitement d'image ", Université Abou Bakar Belkaid Tlemcen, 2008.

[3]: Chérif TAOUCHE, " Implémentation d'un Environnement Parallèle pour la Compression d'Images à l'aide des Fractales", Université Mentouri Constantine, 2005.

[4] : Thi-Lan le, "Indexation et recherche de vidéo pour la vidéo surveillance", université de NICE-SOPHIA ANTIPOLIS, 3 février 2009.

[5] : Djamila Mokhtari, " Détection des chutes par calcul homographique ", Université de Montréal Département d'Informatique et de Recherche Opérationnelle Faculté des Arts et des Sciences, Août 2012.

[6] : MEDJAHED Fatiha, " Détection et Suivi d'Objets en Mouvement Dans Une Séquence d'Images ", Université Mohamed Boudiaf Oran, 2011.

[7]: David Salomon, "Data-compression-the-complete-reference ", 2007.

[8] : Ahmed Ben Atitallah, "Etude et Implantation d'Algorithmes de Compression d'Images dans un Environnement Mixte Matériel et Logiciel", Université Bordeaux 1, Juillet 2007.

[9] : Taoufik EL Kabir, Amine Boukouta, " Reconstruction du mouvement humain par système de vision ", INRIA, Christine Azevedo, 2013.

[10]: G.BRADSKI & A.KAELHER, "Learning OpenCV Computer Vision with the OpenCV Library" O'Reilly, Sebastopol, 2008.

[11] : Samarth Brahmbhatt, "Practical OpenCV", 2013.

[12] : Nicolas DUMOULIN, "Compression de séquences vidéo choix des images de références ", Université de Rennes1-IFSIC, Juin 2003.

[13] : Cao Tien Dung, " La vidéos surveillance ", Institut de la Francophonie pour l'Informatique, Janvier 2007.

[14] : Centre de recherche informatique de Montréal, "La vidéosurveillance intelligente", Avril 2009.

[15] : HORAUD, Radu, MONGA, Olivier, "Vision par ordinateur", 1995.

[16]: Sung Wook Seol, Jee Hye Jang, Hyo Sung Kim, Chul Hun Lee, and Ki Gon Nam "Detection and Tracking System of Moving Objects Using Double Difference based Motion Estimation", 2003.

[17] : Centre de recherche informatique de Montréal, "La vidéosurveillance intelligente", Avril 2009.

[18]: Yiğithan Dedeoğlu. "Moving Object detection, Tracking and Classification for Smart Video Surveillance", Août 2004.

[19] : Thierry Chateau, Antoine Vacavant, " Real Time tracking of human gestures using color image processing", Université Blaise Pasca France, Juin 2004.

[20] : Wassima AIT FARES," Detection Et Suivi D'objets Par Vision Fondes Sur Segmentation Par Contour Actif Base Region ", Université De Toulouse, Janvier 2014.

[21] : R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, O. Hasegawa, P. Burt et L. Wixson, "A System for Video Surveillance and Monitoring", Robtics Institute, May 2000.

[22]: Samuel Rouxel, " La bibliothèque de traitement d'images OPENCV ", CRESITT Industrie, 2010.

[23]: Rafael C. Gonzalez, Richard E. Woods et Steven L. Eddins, "Digital Image Processing Using MATLAB", Pearson Prentice-Hall, 2004.

[24]: Zeroual Djazia, "Implémentation D'un Environnement Parallèle Pour La Compression D'Image A L'Aide Des Fractales", Université de Batna, 2006**.**

[25]: Ahmed Ben Atitallah, " Etude et Implantation d'Algorithmes de Compression d'Images dans un Environnement Mixte Matériel et Logiciel", thèse de doctorat en électronique, Université Bordeaux 1, Juillet 2007.

Sites internet

[S1]: http://www.memoireonline.com/

[S2]: http://www.vision.uji.es/~montoliu/docs/pfm/AnneClaireMaheo.pdf

[S3]: http://fr.wikipedia.org/wiki/OpenCV

[S4]: http://master-ivi.univ-lille1.fr/fichiers/Cours/pje-semaine-3-opencv.pdf

[S5]: https://fr.wikipedia.org/wiki/Microsoft_Visual_Studio

[S6]:http://tecfa.unige.ch/tecfa/teaching/staf13/fiches-mm/formatfichier.htm

[S7]:www.acnice.fr/ia06/iencagnes/file/tice/tuto/Image_numerique.pdf

[S8]:http://www.foad-mooc.auf.org/IMG/pdf/D226_Chapitre-5.pdf

[S9]:www.lagis.univ-lille1.fr/formations/sii/06-07/projets/cmucam-rapport.pdf

[S10]:http://S2-e-monsite.com/2010/03/16/85124207image-numérique.pdf.pdf.

[S11]: www.crdp.ac-grenoble.fr/image/general/general.htm.

[S12]:http://www.nymphomath.ch/info/images/images.pdf.

[S13]:http://gillesboulet.ca/textes/VideoNumerique.pdf.

[S14]:http://www.tdechardin.org/web/page.php?id=1350.

[S15]:http://www.techno-science.net/?onglet=glossaire&definition=7376.

[S16]:icar.univlyon2.fr/projets/corinte/documents/Acquisition_et_compression.ppt.

[S17]: https://climat63.files.wordpress.com/2012/04/format_vidc3a9o.pdf.

[S18]: http://lucbor.fr/formats_videos_et_codecs.pdf.

[S19]: http://benjlaiel.e-monsite.com/medias/files/video-numerique.pdf.

[S20]: www.ac-nice.fr/ia06/iencagnes/file/tice/tuto/Image_numerique.pdf,

[S21]:webcache.googleusercontent.com/search?q=cache:http://obligement.free.fr/articles/codagedesimages.php.

[S22]: www.mediatheque.mc/dotAsset/1bc69e0a-28a1-4399-8193-6a831697f818.pdf

[S23]:www.st-tremeur.fr/courstechno/imagenum/www.crdp.ac-grenoble.fr /image/general/img_num.htm.

[S24]:http://docslide.fr/download/link/27714afb8a63911930a39ea51e575c0e-pfe-oussama