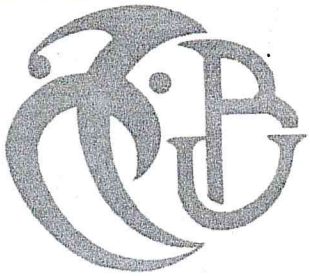
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur

Et de la Recherche scientifique

Université SAAD DAHLAB de BLIDA



Faculté des sciences

Mémoire de fin d'études

Pour l'obtention du diplôme de Master en informatique

THÈME :
Conception et réalisation d'un outil de détection de scènes sanglantes dans une séquence vidéo

Réalisé par :
- Mr. Lazli Hassen

Encadré par :
- Mr. Cherif-Zahar Amine

Présenter le 23 juin 2016 devant le jury composé de :

Président du jury: Mr A.Chemchem

Examinateur : Mr H.Derrar

Année universitaire 2015/2016

Résumé

L'objectif principal de ce travail consiste à réaliser un outil de détection de scènes sanglantes dans une séquence vidéo suivant plusieurs techniques de visions par ordinateur (Image histogramme, haar cascade …etc.). En utilisant pour cela la bibliothèque Open Cv et le langage de programmation python.

La caractéristique principale de cette application est qu'elle est entièrement automatisée, et permet un gain de temps considérable. Les résultats obtenus sont satisfaisants de l'ordre de 86% malgré le faible taux de faux positif généré. Cette méthode présente l'avantage d'être évolutive ou elle peut être combinée avec d'autres méthodes dans le but de détecter d'autres caractéristiques de la violence autre que le sang.

Mots clés : Haar cascade, Histogramme comparaison, détection violence, détection scène sanglante, détection changement scène

Abstract

The main objective of this work is to make a bloody scene detection tool in a video sequence according to several computer vision techniques (histogram picture haar cascade ... etc.). Using OpenCV library and python programming language.

The main feature of this application is that it is fully automated and provides significant time savings. The results are satisfactory in the order of 86 % despite the low rate of false positives generated. This method has the advantage of being scalable or it can be combined with other methods to detect other characteristics of violence other than blood

Keywords: Haar cascade Histogram comparison, detection violence, bloody scene detection, scene change detection

Remerciements

Tout d'abord, je tiens à remercier Mr. Cherif-Zahar Amine, pour m'avoir proposée ce sujet passionnant, et surtout pour ses précieux conseils.

Je remercie vivement les membres de jury, d'avoir accepté d'évaluer mon travail. Ainsi qu'à tous les professeurs qui ont partagé leur savoir avec moi.

Je souhaite remercier mes proches, mes parents qui ont toujours su être là dans tous les Moments, leur soutien et leur générosité et leur volonté ont toujours mérité mon plus profond respect.

Un merci particulier à mes frères et sœur Abdelghani, Wassila, Youcef, Amine

Qui ont su gérer mon stress, me soutenir et me conseiller, comme ils le font si bien dans la vie.

Ainsi qu'à mes deux belle-sœur Lydia, Yassmine pour leur soutien morale et les conseils qu'ils m'ont apporté.

A mes Oncles et tente et a tout ma familles qui ont su toujours m'orienter dans le droite chemin.

Enfin, Je remercie mes chers amis Ihcen, Fayçale, Abd El jalil, Houssem Abdou, Ayoub, Sofian, Youcef, Sarah, Yassmine Imane ainsi que tous les autres ces 5 années passer avec eux étais parmi les plus belles années.

Et un grand Merci à l'homme qui m'a tout appris et à qui je dédicace tout ce travail mn défunt grand père Allah Yarhmou.

Sommaire

Table des figures

Liste des Équations

Liste des tableaux

Introduction Générale

Les contenus multimédias sont devenus l'un des principaux moyens de communication dans le monde. La popularisation des moyens de création a entrainé une expansion du nombre de contenus multimédias disponibles ainsi que le développement de nouveaux types de contenus par la même occasion. A titre d'exemple, la production moyenne annuelle aux Etats-Unis est de 500 films par ans et le double pour l'inde et l'Union Européenne **[1]**

On peut aussi noter des services de vidéo à la demande (VOD), ou l'apparition des sites de partage de vidéo amateurs et professionnelles comme la plateforme YouTube qui enregistre maintenant plus de 500 heures de vidéo par minute dans le monde **[2]** explosant ainsi la quantité de données audio-vidéo disponibles .

Cette démocratisation de la création et de l'accès aux contenus audio-vidéo notamment grâce à internet a aussi augmenté la quantité de contenus pouvant heurter la sensibilité d'un jeune public à travers des contenus de violence ou bien de contenus inapproprié.

Beaucoup de personne souffrent d'hématophobie (peur du sang) qui est, selon l'organisation mondiale de la santé (OMS), la troisième phobie la plus répandue dans le monde, qui engendre un évanouissement à la simple vue d'une image comportant du sang. Par conséquent, renforcer le besoin de détection des contenus sensibles ainsi que la classification des contenus est primordiale.

Ainsi, l'augmentation du nombre de contenus multimédias et la diversification des moyens de diffusion de ces contenus a engendré un problème significatif pour le traitement de la détection de la violence dans les séquences vidéo. En effet, une personne chargée de ce travail se retrouverait immédiatement dépassée par le contenu exorbitant des vidéos existantes et se verrait heurtée à un problème de subjectivité, puisque la notion d'interprétation de la violence varie entre les individus, tout en rajoutant la marge d'erreur humaine causée par la fatigue ou bien par l'avancement rapide. Toutes ces contraintes ont fait que la détection de la violence est devenue un problème majeur auquel des solutions doivent être proposées.

Le but de ce travail est la réalisation d'un outil de détection des scènes violentes automatiquement suivant des caractéristiques bien précises. Ces dernières peuvent prendre plusieurs formes (physiques, verbales, psychologiques, …, etc.).

Dans un premier temps, notre outil de détection de violence devrait pourvoir détecter les changements de scène automatiquement dans une vidéo, afin de pouvoir travailler sur les scènes et avoir un résultat plus compréhensif. Ensuite, cet outil devrait détecter des scènes de violences, mais un outil complet qui engloberait toutes les caractéristiques définissant une violence serait très difficile à réaliser et demanderait des années de travail. Ce qui nous a forcés à nous focaliser sur une caractéristique bien précise, qui est le sang, puisque la présence abondante du sang est très souvent significative de violence et l'impact occasionné par de telles scènes sanglantes sur de nombreuses personnes font de notre choix un choix judicieux.

Le présent mémoire est organisé en plusieurs chapitres. Le premier chapitre est destiné à présenter un état de l'art sur les méthodes de détection de scènes particulières. Nous y présenterons les grandes lignes des systèmes d'indexations déjà connus en décrivant les différentes techniques ainsi que les travaux spécifiques .Le deuxième chapitre abordera la conception de notre outil, les différentes recherches et méthodes utilisées pour la détection de la violence ainsi que la méthode choisie. Dans le troisième chapitre, nous montrerons comment implémenter notre méthode et les différentes étapes détaillées, suivi alors du chapitre quatre, résultats et discussion, ou nous présenterons quelques résultats ainsi qu'une étude statique et comparative des résultats obtenus suivi d'une discussion. Nous terminerons notre travail par une conclusion générale.

Chapitre 1 : Etat de l'art sur les méthodes de détection de scènes particulières

1.1 Introduction

Depuis ces dernières années, le développement des logiciels et matériels technologiques ont permis la création d'une grande quantité de contenus vidéos. L'indexation des contenus multimédias consiste à poser des étiquettes sur des documents multimédias, de façon à pouvoir les trier et les retrouver plus simplement par la suite. L'explosion de la quantité de contenus disponibles a rendu l'automatisation indispensable dans de nombreux domaines applicatifs. On peut l'employer par exemple dans la détection automatique des publicités, l'extraction de caractéristiques, la recherche d'image par contenu ou bien la segmentation en plans.

Dans ce mémoire, nous nous intéressons en particulier à la segmentation en plans qui est une étape importante dans l'indexation et la recherche de données dans une vidéo, elle peut être utilisé dans différentes approches selon plusieurs contextes dont le but général est de segmenter un contenu vidéo en plans constitutifs et d'identifier et de classer les différentes transitions selon un besoin souhaité.

L'idée principale à la base des méthodes de segmentation en plans est que les images au voisinage d'une transition soient fortement dissemblables, On cherche alors à repérer les discontinuités dans la vidéo.

Plusieurs méthodes existent, les plus connues sont l'histogramme comparaison et le standard MPEG-7 qui offrent des outils de comparaison d'images par mouvement.

1.2 Précédents travaux

Jusqu'à ce jour, peu de travaux ont été réalisés en matière de détection automatique de scènes violentes.

Certains travaux ont utilisé des caractéristiques audio, c'est-à-dire certains sons qui pouvaient être révélateurs de scènes violentes. Nous pouvons citer à titre d'exemple les travaux de Giannakopoulos et all. **[3],** Où ils ont considéré six caractéristiques audio et ont supposé, qu'au préalable, le signal audio a été segmenté en scènes et que l'information sémantique sur le contenu a été extraite. Ils ont suivi la façon dont les séquences changeaient de cadre en utilisant des quantificateurs comme, par exemple, la valeur moyenne de l'intensité sonore. Une fois les caractéristiques extraites, ils ont utilisé le support vecteur machine afin de classifier les différents résultats.

D'autres recherches se sont basées sur l'aspect visuel uniquement. Bermejo et all. **[4],** ont utilisé la technique bien connue du sac-de-mots-visuel, pour la reconnaissance de l'action de combat ainsi que les deux meilleurs descripteurs d'actions. A savoir, STIP (space time interest points) et MoSIFT (motion Scale-invariant feature transform). Ils ont pour cela, introduit une base de données vidéo contenant 1000 séquences divisées en deux groupes : les combats et les non-combats. Les résultats obtenus présentent une précision de près de 90 %. Mais une étude menée ultérieurement, dans la même université, a permis de constater que le coût de calcul et de l'extraction des caractéristiques était prohibitif pour une éventuelle application dans les systèmes de surveillance ou dans les médias. Pour pallier à ce problème, une des méthodes d'accélération consiste à choisir une caractéristique principale qui a permis une détection 15 fois plus rapide des points d'intérêt **[5].**

Une autre étude a été réalisée par des étudiants de l'université de Toulon, où sept approches avaient été testées : la première est basée sur le niveau ontologique des mots parlés. Ils avaient supposé que les termes employés dans une scène violente seraient des termes de bas niveau ontologique ; La deuxième approche, qui considère le signal audio, est scindée en trois techniques : analyse de la parole ; approche par décomposition du spectre audio ; et analyse de l'audio en mono. Les résultats obtenus révèlent que la parole et les variations audio, jouaient un faible rôle dans la détection de la violence, puisque seulement 6 % de résultats positifs ont été obtenus ; La troisième approche est basée sur le signal vidéo, deux techniques étaient testées (dynamique des visages et dynamique de la vidéo). Les deux techniques ont utilisé une liste de caractéristiques générées ainsi qu'un réseau de neurones. les résultats obtenus avoisinaient les 75% de réussite **[6].**

1.3 Indexation de vidéo par segmentation de plans

1.3.1 MPEG-7

Le standard MPEG-7 a pour objectif de fournir des descripteurs standardisés des contenus multimédias et de supporter un large éventail d'applications potentielles. MPEG-7 standardise plusieurs ensembles de descripteurs (D), des schémas descripteurs (SD) ainsi qu'un langage de définition de descripteur (DDL).

Pour l'indexation de vidéo mpeg-7 utilisent les descripteurs permettant la description des contenus multimédias :

- Descripteurs visuels, encodant des informations de couleur, texture, forme ou mouvement.
- Descripteurs audio, encodant le timbre, le spectre, ou des éléments de plus haut niveau comme la parole

1.3.2 Descripteur d'activité de mouvement

L'algorithme descripteur d'activité de mouvement (Motion Activity Descriptor) offre la mesure de la quantité de mouvements présents dans les séquences vidéo.

Premièrement, Il découpe l'image en macro-block (blocs carrés de dimension 16*16) comme présenté dans la figure [1.1]

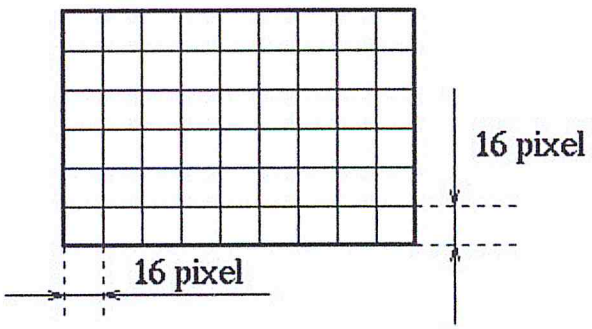


Figure 1.1 Macro-block de taille 16*16

Puis l'algorithme calcule le déplacement de chaque block (block bleu N+1 image de la figure [1.2]) par rapport à sa position initial (block bleu N image de la figure [1.2]) suivant une fenêtre de recherche ainsi qu'un algorithme de block matching.

Il existe plusieurs algorithmes de block matching afin de parcourir l'image et de trouver les deux blocs similaires ou bloc cible au bloc initial.

Plusieurs algorithmes de BMA ont été proposés :

- Exhaustive search (ES)
- Diamond Search(DS)
- Three-step search(TSS)
- New three-step search(NTSS)
- Simple and effective search(SES)
- Four Step Search (4SS)
- Adaptive Rood Pattern Search(ARPS)

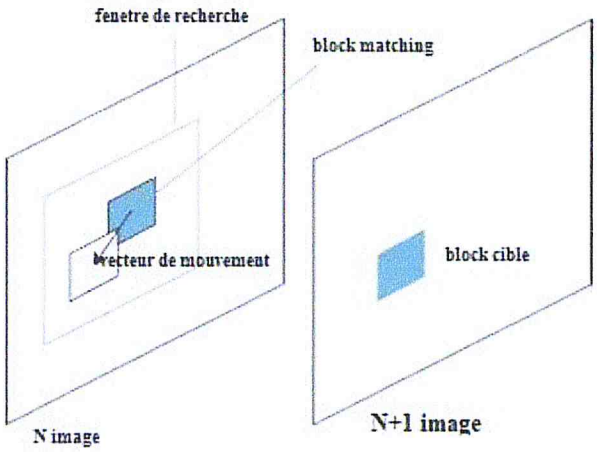


Figure 1.2 : Extraction de vecteur de mouvement

Une fois le vecteur de mouvement extrait (figure 1.2) les caractéristiques de mouvement sont extraites, ou X (i, j) et Y (i, j) indiquent le vecteur de mouvement extrait et (i, j) les indices de bloc

Le calcul de l'intensité du mouvement nous permet de représenter l'intensité de l'activité du mouvement qui est représenté sous forme d'un nombre entier, où une faible valeur désigne une activité de mouvement faible alors qu'une valeur élevé nous montre une très grande quantité de mouvement

Voici un exemple montrant les résultats d'une étude effectuée par Abdelati Malek et all [7] pour la détection de limite vidéo en utilisant le descripteur de mouvement d'activité, ou le résultat vert nous montre une forte activité liée au changement de scène alors que les deux autre résultats respectivement marron et rouge nous montrent une activité moyenne et faible de mouvement

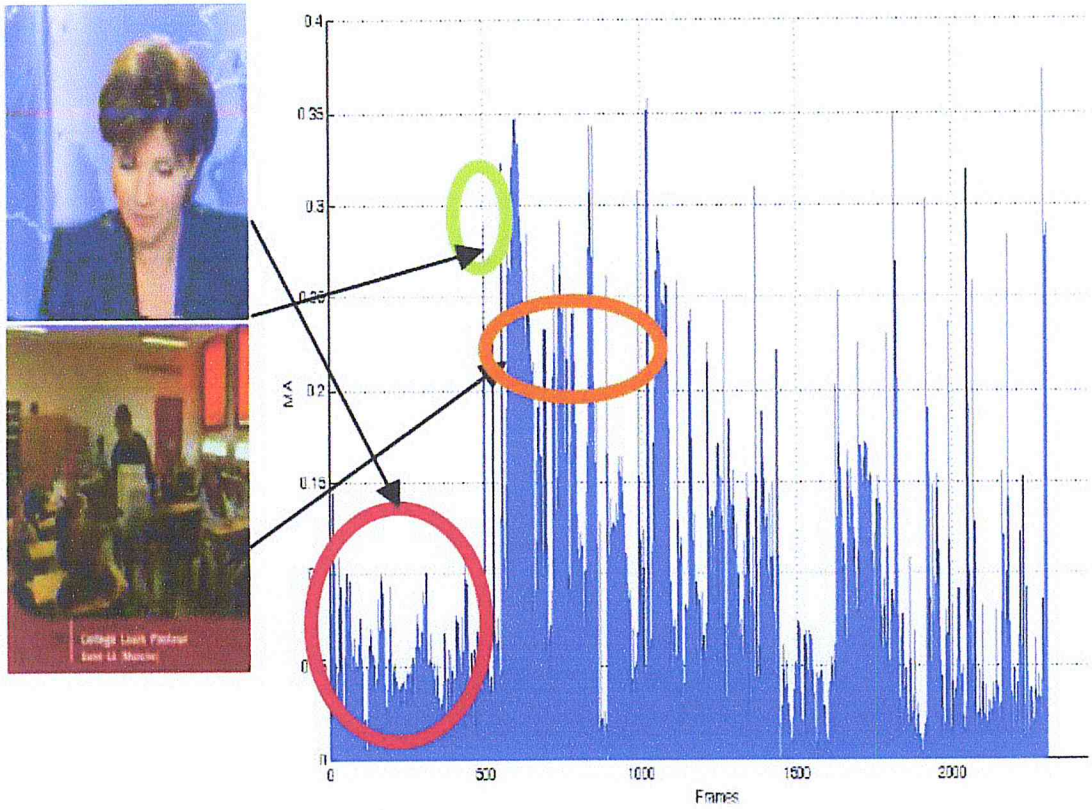


Figure 1.3 : Exemple de résultat de séquence vidéo en utilisent le descripteur d'activité de mouvement

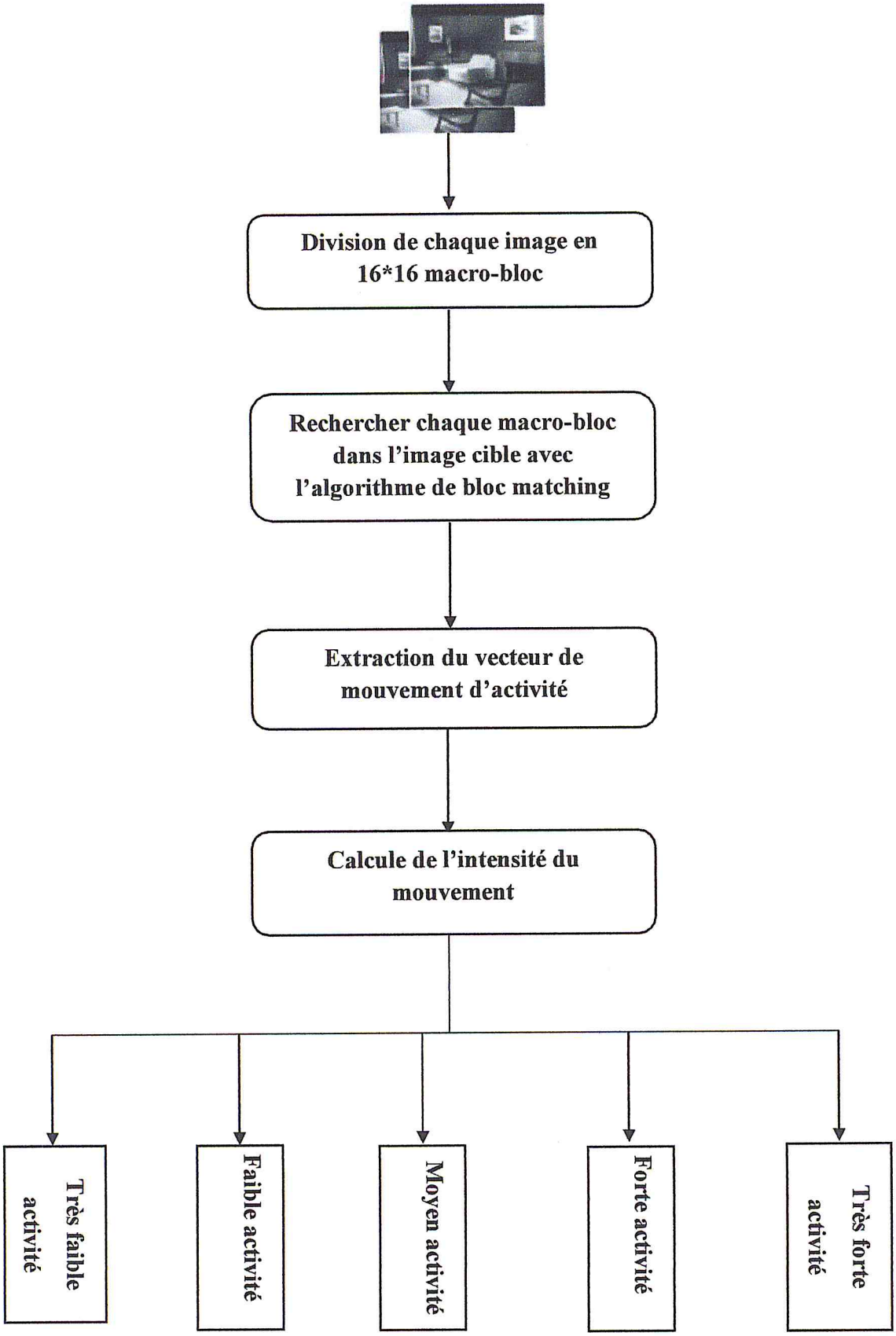


Figure 1.4 : schéma de fonctionnement de la motion activité descripteur

En conclusion si le résultat obtenu est une très forte activité de mouvement on conclut forcément qu'il y'a un changement de plans et aucun changement pour les autres résultats.

Dans le même contexte mais en utilisent les couleurs (histogramme est une technique qui permet de comparer les images (frames) d'une vidéo avec des algorithmes de comparaison d'histogramme

1.4 Histogramme comparaison

La comparaison d'histogrammes est l'une des techniques les plus utilisé dans le domaine de l'indexation de vidéo et d'images, utilisée principalement pour la comparaison d'images elle utilise pour cela la répartition des pixels selon leur valeur et compare la distance suivant différents métriques mais avant de commencer nous devons comprendre qu'est-ce qu'un histogramme?

1.4.1Histogramme d'une image

Un histogramme est une courbe statique indiquant la répartition des pixels selon leur valeur, ainsi que le ton d'une image, la figure montre le principe d'un histogramme d'une image.

L'axe horizontal indique les différents tons (du sombre au lumineux) figure 1.5

L'axe vertical Y indique le nombre de pixels (ou le pourcentage de pixel) figure 1.5 pour chaque ton

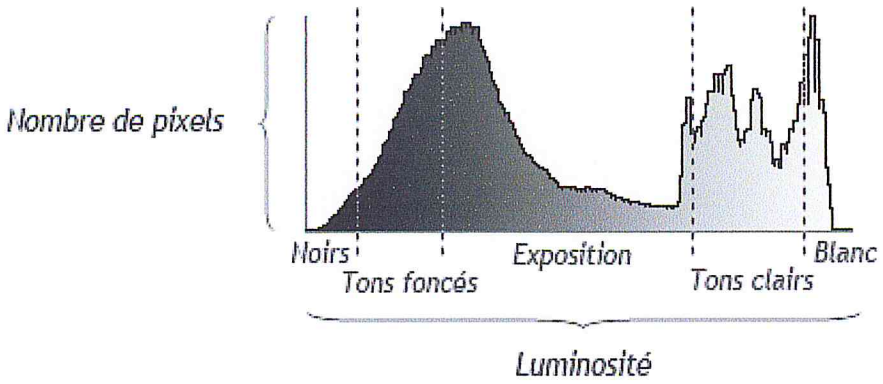


Figure 1.5 : Principe d'un Histogramme

Dans la vraie vie le nombre de tons ou nuances est infini, mais en imagerie numérique il est limité par le capteur optique. Alors peu importe si la photo utilise un très grand nombre de bits, l'histogramme est généralement calculer sur 8 bits, 256 valeurs de ton différents (de 0 à 255), cela signifie que pour chaque couleur i comprise entre la valeur de 0 à 255 en va calculer le nombre de pixels qui portent la valeur i on notera ce nombre H(i).

La représentation de l'histogramme est représentée par des bâtons qui prennent en abscisse les valeurs de i de 0 à 255 ainsi que les H(i) pour l'ordonnée.

La valeur de i pour les différents tons de luminosité pour la figure 1.5 et comme suite :

- **Noir**: de 0 à 25
- **Tons foncés**: de 26 à 63
- **Exposition** (sic): de 64 à 178
- **Tons clairs**: de 179 à 29
- **Blanc**: de 230 à 255

1.4.2 Extraction des caractéristiques de l'image

1.4.2.1 Attribut couleur

Les techniques les plus utilisées pour la représentation de la couleur est celle des histogrammes couleur. Ce sont les techniques les plus utilisées en comparaison d'images. Ils représentent la densité de chaque couleur présente dans l'image.

L'attribut couleur ou plus communément appelé le signal couleur peut être décomposé de diverses manières, en trois composants d'où l'appellation espace de couleur.

Les plus connus sont l'espace de couleur RVB qui utilise les trois couleurs primaires (rouge, vert, bleu) ou bien l'espace HSV, TSL en français pour (teinte saturation lumière).

1.4.2.2 Attribut de distance

Il existe plusieurs traitements sur l'histogramme qui varie selon le contexte utilisé mais celui qui nous intéresse s'appelle mesures de similarité afin de comparer deux images suivant une distance de similarité ou disimilarité ou bien le changement temporelles de contenu visuel dans une vidéo et ceci suivant plusieurs méthodes.

1.4.2.2.1 Méthode basée histogramme:

C'est une méthode basée sur le principe d'histogrammes en suivant des métriques bien précis, voici quelque exemple de métriques connus :

Corrélation : calcule la corrélation entre les deux histogrammes Figure 1.6

$$D(H1, H2)=\frac{\sum i(H1(I)-H1)(H2(I)-H2)}{\sqrt{\sum i(H1(i)-H1)^2 \sum i(H2(i)-H2)^2}}$$

Équation 1. 1 : formule de calcul par la distance de corrélation

Chi-Square : calcule le Chi-Square distance entre deux histogrammes Figure 1.7

$$D(H1, H2) = \sum i \frac{(H1(i)-H2(i))^2}{H1(I)}$$

Équation 1. 2 : formule de calcul par Chi-Square distance

Intersection : calcule l'intersection entre deux histogrammes Figure 1.8

$$D(H1, H2) = \sum i \min(H1(i), H2(i))$$

Équation 1. 3 : formule de calcul par intersection

Bhattacharyya distance : permet lamesure de chevauchement entre deux histogrammes figure 1.4.2.2.1.4

$$D(H1, H2) = \sqrt{1 - \frac{1}{\sqrt{H1H2N^2}} \sum \sqrt{H1(i).H2(i)}}$$

Équation 1. 4 : formule de calcul par la Bhattacharyya distance

1.4.2.2.2 Méthode basée Scipy :

C'est une méthode qui utilise la bibliothèque SciPy (bibliothèque présente dans python) qui permet d'utiliser plusieurs métriques de distance

Distance Euclidienne: permet le calcul des distances d'histogramme suivant la technique euclidienne figure 1.10

$$D(H1, H2) = \sqrt[2]{\sum j(H1, j - H2, j)^2}$$

Équation 1. 5 : Formule de calcul par la distance euclidienne

Manhattan aussi appelé (City block) : permet le calcul des distances d'histogramme suivant la technique euclidienne figure 1.4.2.2.2.2

$$D(H1, H2) = \sum j \mid H1,j-H2,j \mid$$

Équation 1. 6 : formule de calcul par la distance de Manhattan

Ou N est le nombre total d'histogrammes et H1, H2 deux histogrammes de même taille

Ces différentes méthodes couvrent quasi l'ensemble des exigences recherchées par les utilisateurs. Néanmoins il existe toujours la possibilité de modifier les méthodes ci-dessus ou bien de créer notre propre meure de similarité

1.4.3Etapes de comparaison

Comme nous l'avons expliqué, l'histogramme est unique pour chaque image donc une comparaison d'image se traduit par la comparaison de leur histogrammes respectifs, nous montrons dans la figure 1.12 les étapes de comparaison de deux images.

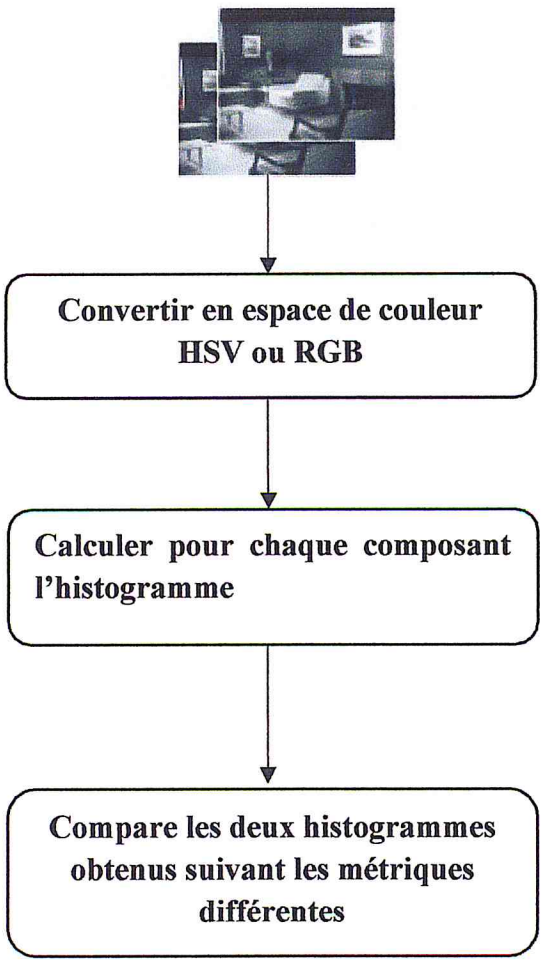


Figure 1.6 : Etapes de comparaison de deux images avec histogramme comparaison

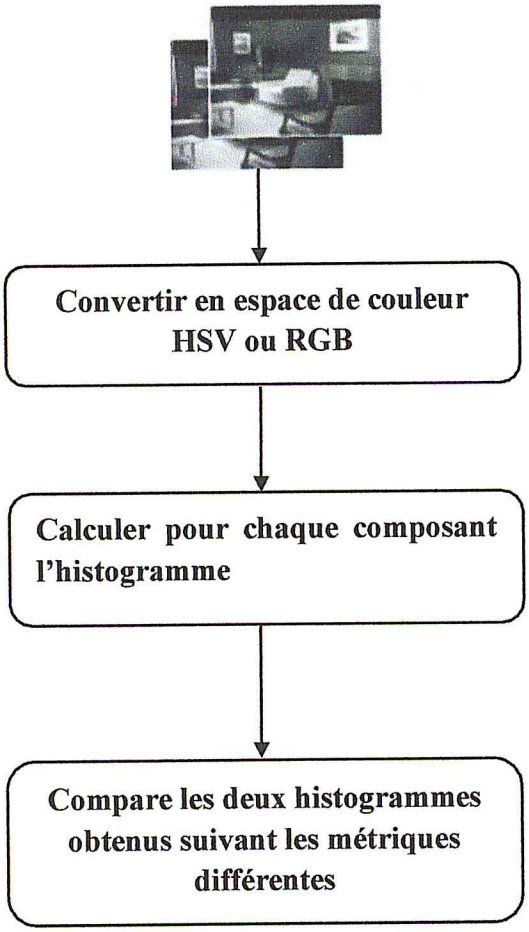


Figure 1.6 : Etapes de comparaison de deux images avec histogramme comparaison

1.5 Travaux spécifiques sur les méthodes de détections de scènes particulières

L'histogramme comparaison et la motion activité ne sont pas les deux seules techniques utilisées dans le Domaine de l'indexation, voici quelques travaux existants qui utilisent d'autres techniques d'indexation.

1.5.1 Reconnaissance des Scènes Vidéo Adultes

Thèse effectué par Bouirouga, et All université de rabat Maroc **[8].**

La détection de la couleur de la peau est un moyen efficace souvent utilisé pour définir une série de zone candidates. De plus, celles-ci peuvent être effectuées de manière assez simple grâce à des seuillages dans des espaces de couleur approprié.

Cette détection qui est peu couteuse en temps de calcule .puisqu'elle classifie automatiquement pixels par pixels de peau ou non-peau dans des images à caractère obscène

En premier, lieu l'activité est structurée sur l'analyse et la modélisation de séquences vidéo. Dès que des fonctionnalités d'interaction avec le contenu vidéo sont envisagées une phase préalable d'analyse de la séquence est nécessaire. Ce filtrage permet un gain de performance évitant les données de régions sans intérêt qui sont analysées par le module de détection de peau.

Puis à partir des images en niveau de gris obtenues à la fin du processus de détection de la peau neufs différents descripteurs sont calculés qui constituent l'entrée du dernier bloc de la chaine de blocage des images indésirables. Cette phase est la prise de décision qui peut être calculé par plusieurs modèles (arbres, réseaux de neurones). Pour leur thèse ils ont opté pour l'utilisation des réseaux de neurones comme méthode de décision de la nature de la vidéo analysée.

La figure 1.13 montre La structure du filtre des scènes vidéo adulte.

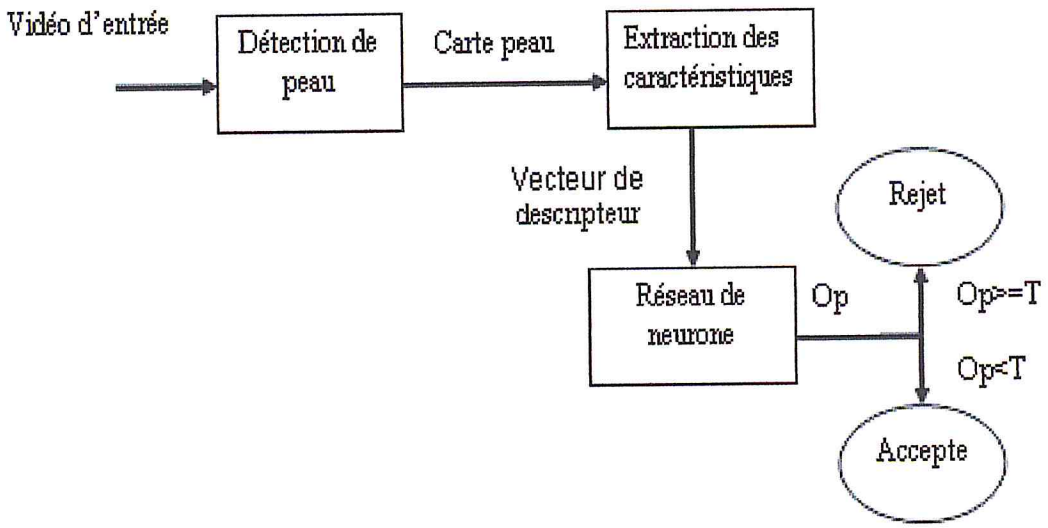


Figure 1.7 : Schéma montrant La structure du filtre des scènes vidéo adulte

1.6 Détection automatique de logos dans les vidéos

Thèse écrite par Dong-Kyu Lee, et all université de Sogang, Seoul, Corée du sud **[9]**.

Dans la majorité des cas les chaines de télévision doivent enlever des logos publicitaire pour plusieurs raison personnelles ou publiques, les chercheurs utilisent pour cela une méthode manuelle afin d'enlever le logos ou de flouter ce qui demandait beaucoup de temps.

Pour résoudre ce problème plus facilement et rapidement, Ils ont proposé des solutions qui sont décrite dans cette thèse ou ils ont proposé un nouveau algorithme qui détecte, cible et enlève les logos en mouvement dans une vidéo de manière automatique.

Ils ont utilisé pour cela la technique de saliency map **[10]** et l'algorithme de SIFT (scale invariant feature transform) **[11, 12]** pour la détection de logo. Alors que la méthode de changement de vitesse moyenne pour le ciblage du logo **[13, 14]** . En dernier pour la suppression c'est la méthode de inpainting **[15]** [16] qui a étais utilisé.
Dans la figure 1.16 un exemple de test montrant la suppression de logo sur une tasse de café et un bloc note

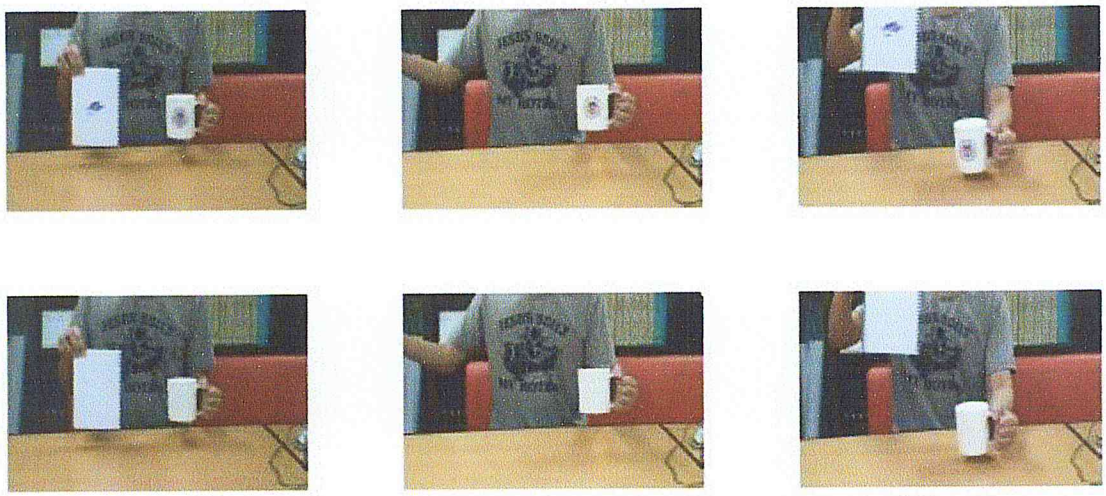


Figure 1.8 : Exemple montrant la suppression de logo sur la tasse de café et le cahier

1.7 Conclusion

Dans ce chapitre ont a vu quelques techniques d'indexation vidéo qui ont montré que le standard mpeg-7 ainsi que l'histogramme comparaison étaient deux méthodes très fiables et rapides pour la détection de scène temporelles dans une vidéo ainsi que certaines techniques de détection spécifique, utilisées dans le Domaine de l'indexation et de la vision par ordinateur.

Dans le cadre de notre travail, Nous avons opté pour la méthode de l'histogramme comparaison grâce à sa facilité d'implémentation ainsi qu'au temps de réponse relativement rapide qu'elle propose. En effet elle n'est qu'un prétraitement d'un long processus afin d'arriver au but souhaité car cette méthode nous permet l'indexation de manière temporelle d'une vidéo. Pour la suite de la méthode on va utiliser d'autres méthodes afin de pouvoir détecter la violence. Ceci va être détaillé dans le prochaine chapitre .

Chapitre 2 : conceptions de notre application

2.1 Introduction :

Détecter automatiquement une scène violente n'est pas chose facile, en effet, il existe dans l'état de l'art, plusieurs approches et différentes techniques de détection de la violence. Cependant, le caractère subjectif de la violence, c'est-à-dire, les nombreuses variables et les différents paramètres qui la caractérisent ainsi que la perception et la sensibilité variable du public vis-à-vis de la violence, nous a obligé à établir un choix et à cibler une caractéristique particulière, qui est le sang. Une fois notre choix effectué, nous avons essayé de trouver comment détecter le sang dans une séquence vidéo ? Pour ce faire, nous avons assimilé le sang à un objet afin d'appliquer la méthode de Viola et Jones bien connue et très utilisée dans le domaine de la détection des objets. Cependant, le sang n'est pas tout à fait un objet mais plus un contenu (Ensemble de plages, de points dans un système d'imagerie donnant une image). Néanmoins l'application de certains effets spécifiques nous permet de le filtrer. Ainsi, avec cette méthode nous octroyant une certaine forme au sang qui le rapproche de celle d'un objet. Cela en rajoutant un classificateur que nous avons conçu et qui nous permet de détecter les scènes sanglantes. Mais tout d'abord voyons un peu plus en détails le fonctionnement de la méthode de Viola et Jones

2.2 Méthode de Viola et Jones

La méthode de Viola et Jones est une méthode de détection d'objet dans une image
Numérique **[17]** , elle fait partie des toutes premières méthodes capables de détecter efficacement
Et en temps réel des objets dans une image.
Inventée à l'origine pour détecter des visages, elle peut également être utilisée pour détecter d'autres types d'objets comme des voitures ou des avions.
La méthode de Viola et Jones est l'une des méthodes les plus connues et les plus utilisées, en particulier pour la détection de visages et la détection de personnes.
En tant que procédé d'apprentissage supervisé, la méthode de Viola et Jones nécessite
De quelques centaines à plusieurs milliers d'exemples de l'objet que l'on souhaite détecter,
Pour entraîner un classifieur.
Une fois son apprentissage réalisé, ce classifieur est utilisé pour détecter la présence éventuelle de l'objet dans une image en parcourant celle-ci de manière exhaustive, à toutes les positions et dans toutes les tailles possibles.

2.2.1 Elément de la méthode :

La méthode de Viola et Jones consiste à balayer une image à l'aide d'une fenêtre de Détection de taille initiale 24px par 24px (dans l'algorithme original) et de déterminer si l'objet y est présent. Lorsque l'image a été parcourue entièrement, la taille de la Fenêtre est augmentée et le balayage recommence, jusqu'à ce que la fenêtre fasse la taille de l'image. L'augmentation de la taille de la fenêtre se fait par un facteur multiplicatif de 1.25. Le balayage, quant à lui, consiste simplement à décaler la fenêtre d'un pixel. Ce décalage peut être changé afin d'accélérer le processus, mais un décalage d'un pixel assure une précision maximale.
Cette méthode est une approche basée sur l'apparence, qui consiste à parcourir l'ensemble de l'image en calculant un certain nombre de caractéristiques dans des zones rectangulaires qui se chevauchent. Elle a la particularité d'utiliser des caractéristiques très simples mais très nombreuses.
Il existe d'autres méthodes mais celle de Viola et Jones est la plus performante à l'heure actuelle **[18]**. Ce qui la différencie des autres est notamment :

-l'utilisation **d'images intégrales** qui permettent de calculer plus rapidement les caractéristiques.

-la sélection par **boosting** des caractéristiques.

-la combinaison en **cascade de classifieurs boostés**, apportant un net gain de temps d'exécution.

2.2.2 Caractéristique de Haar :

La procédure de détection d'objet classe des images basées sur la valeur des caractéristiques simples. Il y a beaucoup de motivation pour utiliser les caractéristiques plutôt que les pixels directement. La raison la plus courante est que le système basé sur les caractéristiques fonctionne bien plus rapidement qu'un système à base de pixels.

Les caractéristiques simples utilisées rappellent les fonctions de base de Haar qui ont été utilisés par Papageorgiou et all. **[19]**

Plus précisément la méthode de viola et Jones utilisent plusieurs caractéristiques appeler caractéristiques pseudo-haar. Elles sont calculées par la différence des sommes de pixels de deux ou plusieurs zones rectangulaires adjacentes.

Figure 2.1 montre un Exemple de quatre caractéristiques de Haar. La somme des valeurs des pixels appartenant aux zones encadrées claires est soustraite à la somme des valeurs des pixels appartenant aux zones encadrées sombres pour obtenir la caractéristique de Haar. Chacune des quatre caractéristiques de Haar est représentée avec son cadre de détection respectif.

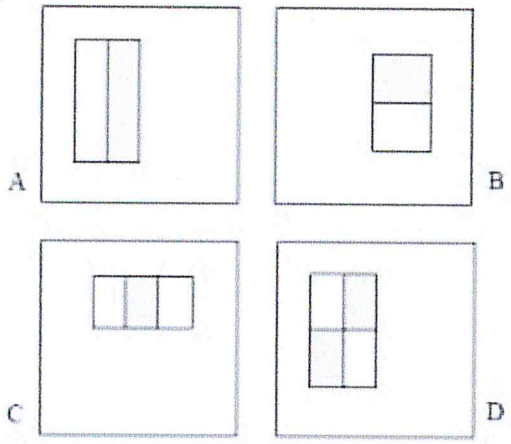


Figure 2.1 : quatre caractérisituqe de Haar

Etant donné que la résolution de base du détecteur est 24*24 pixel, l'ensemble de caractéristique a calculé pour cette résolution est très grande (45396 caractéristique).

Notant que contrairement à la base de Haar , l'ensemble des caractéristiques du rectangle est surcomplet, en effet d'autre caracthéristique ont etais ajouté depuis, la figure 2.2 montre tout les caracthéristique de haar qui ont etais rajouté.

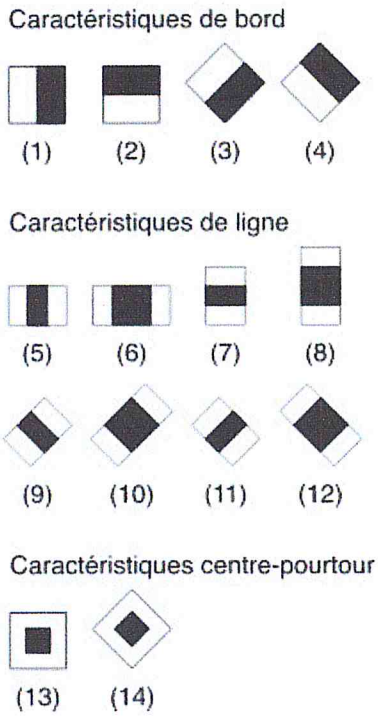


Figure 2.2 : caracthéristique de haar complet

Le calcul d'une caractéristique de Haar demande à chaque fois l'accès aux valeurs de tous les pixels contenus dans les zones rectangulaires considérées. Cela devient vite contraignant temporellement dès que les caractéristiques de Haar sont définies par des zones rectangulaires de grandes dimensions.

L'image intégrale permet de surmonter ce problème en rendant constant le temps de calcul d'une caractéristique de Haar à n'importe quelle échelle.

2.2.3 L'image intégrale :

L'algorithme se base sur les caractéristiques de haar (Haar Features) pour localiser les objets présents sur une image d'entrée. Dans le but d'extraire rapidement ces caractéristiques, l'image est représentée sous forme intégrale [Figure 2.3], [Figure 2.4]. En effet, sous cette forme l'extraction d'une caractéristique à n'importe quel endroit et à n'importe quelle échelle est effectuée en un temps constant tandis que le temps de conversion vers la représentation intégrale ne remet pas en cause ce gain de temps offert par l'utilisation de la représentation en image.

La définition des caractéristiques de Haar et la maniéré dont la représentation intégrale accélère considérablement leur extraction est présentée ci-après pour une image de niveau de gris

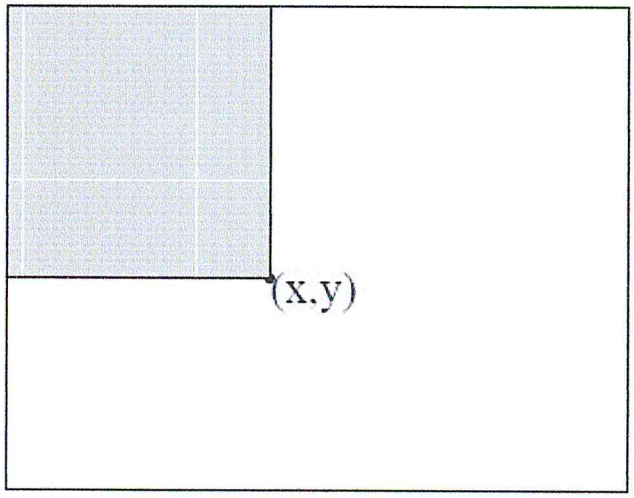


Figure 2.3 : la valeur intégrale de l'image au point (x, y) est la somme de tous les pixels en couleur, soient ceux à gauche et au-dessus

4	1	2	2
0	4	1	3
3	1	0	4
2	1	3	2

4	5	7	9
4	9	12	17
7	13	16	25
9	16	22	33

Figure 2.4 : exemple d'une matrice d'image intégrale (à droite) et de l'image d'origine (à gauche)

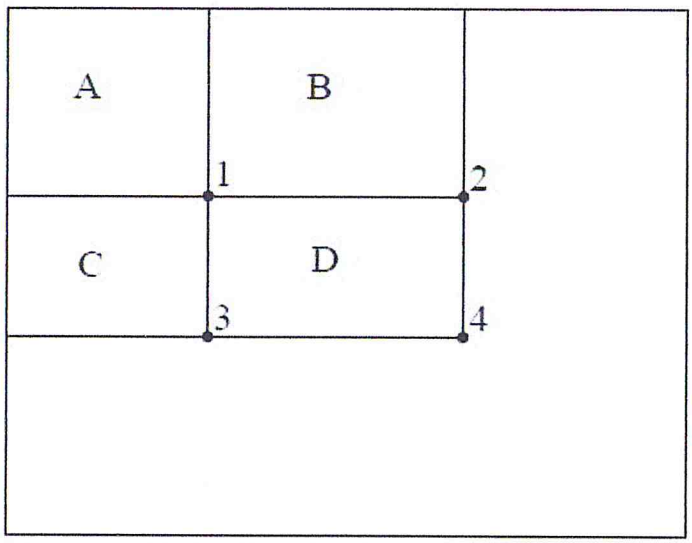


Figure 2.5 : exemple de calcule d'une image intégrale

La somme des pixels contenu dans le rectangle D peut être calculée avec quatre références du tableau. La valeur de l'image intégrale à l'emplacement 1de la figure [2.5] est la somme des pixels dans le rectangle A. La valeur à l'emplacement 2 est A+B, et à l'emplacement 3 elle est de A+C, et à l'emplacement 4 est de A+B+C+D La somme à l'intérieur de D peut être calculée ainsi

4+1-(2+3)

Avec une simple démonstration on peut démontrer cette formule par un simple changement de variable, si en change chaque numéro par le carrée adéquat cella nous donnerai

(A+B+C+D)+A-(A+B+A+C) =

2A+B+C+D-2A-B-C = D

Et grâce à cette méthode En seulement trois opérations la somme des pixels peut être calculée ainsi, on est en mesure de trouver la somme de pixels de n'importe quelle zone rectangulaire de l'image en seulement 3 opérations et 4 accès à l'image intégrale (un accès par point). Une caractéristique pseudo-Haar à deux rectangles peut alors être déterminée en seulement 6 accès (2 points sont partagés) à l'image figure [2.6], et une caractéristique à 3 rectangles en seulement 8 accès.

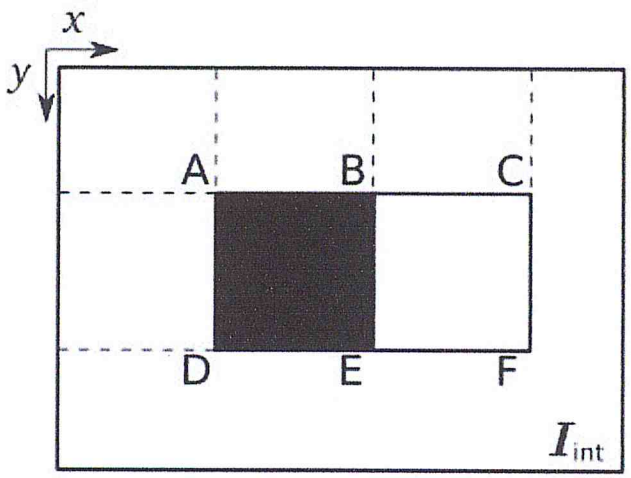


Figure 2.6 : caractéristique pseudo haar a deux rectangles déterminés en 6 accès

Une caractéristique de Haar étant une combinaison linéaire de tels rectangles ABCD, son calcul se fait alors en un temps indépendant de sa taille.

Pour localiser l'objet sur l'image d'entrée, cette dernière est scannée par une fenêtre de dimension déterminée. La fenêtre parcourt l'image et son contenu est analysé pour savoir s'il

s'agit de l'objet recherché ou non. Or comme dit plus haut pour une fenêtre de 24x24 px il y a 45396 caractéristiques de haar, les traiter toutes prendrait beaucoup trop de temps.

Pour surmonter ce problème, une variante de la méthode de boosting Adaboost est utilisée afin d'accélérer le temps de traitement.

2.2.4 Sélection par boosting :

Adaboost est une méthode d'apprentissage permettant de "booster" les performances d'un classifieur quelconque nommé classifieur fort. Et cella on utilisant plusieurs classifieurs "faibles" mis en cascade plutôt que d'utiliser un seul classifieur "fort". En effet, avec un seul classifieur dit "fort», figure [2.7] montre un schéma de ce classificateur

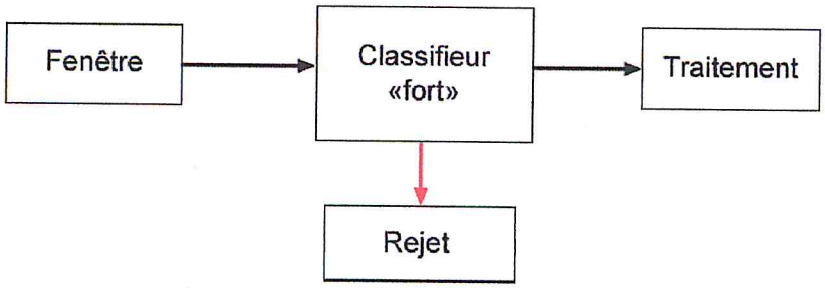


Figure 2.7 : schéma d'un classifieur fort

Il faudrait attendre que le classifieur est analysé toute la fenêtre afin de savoir si l'objet est présent dans l'image.et cella prendrai beaucoup trop de temps.

L'idée est de faire passer les candidats à classifier à travers plusieurs classifieurs faibles, chacun étant entraîné en portant plus d'attention sur les candidats mal classifiés par le classifieur précédent.

Pour arriver à ce résultat des poids sont associés aux échantillons du set d'entrainement de façon équilibrée ensuite le premier classifieur faible est entrainé avec le plus bas tôt d'erreur

Puis une nouvelle génération de poids sont créés tels qu'ils accordent plus d'importance aux échantillons mal classifiés par le premier classifieur qui permet d'entrainé un nouveau classifieur ensuite un nouveau classifieur est entrainé et un nouveau poids est générés et ainsi de suite et après T itération le classifieur fort H(X) est obtenu.

Au final Chaque classifieur fort est donc constitué d'un nombre T de classifieurs faibles.

Ces classificateurs seront mis en cascade de classifieur pour diminuer les critères de sélection.

plusieurs 'étages' de ce type, le processeur évite d'effectuer des analyses lourdes en temps de calcul sur des échantillons pour lesquels il est rapidement possible de se rendre compte qu'ils sont négatifs. Le processus de classification apparaît alors comme une cascade de classifieurs forts de plus en plus complexes où à chaque étage les échantillons classifiés négatifs sont sortis tandis que les échantillons classifiés positifs sont envoyés aux classifieurs suivants. Ceci est représenté à la figure [2.8]. Si le premier étage rejette un faux négatif, c'est un gros problème car il ne sera jamais récupéré par la cascade. Autrement dit c'est un objet qui ne sera pas détecté. Par contre, si le premier étage transmet un faux positif, il pourra toujours être éliminé aux étages suivants de la cascade. Ce petit raisonnement permet de mettre en évidence que les premiers nœuds constitutifs de la cascade peuvent se permettre d'avoir un taux de faux positifs élevés (de l'ordre de 40%) [20], mais doivent absolument assurer un taux de détection maximum

Il faudrait attendre que le classifieur est analysé toute la fenêtre afin de savoir si un objet est présent dans l'image ou non. Une mise en cascade de classifieurs dont le critère de sélection serait moins sévère se présenterait de la sorte :

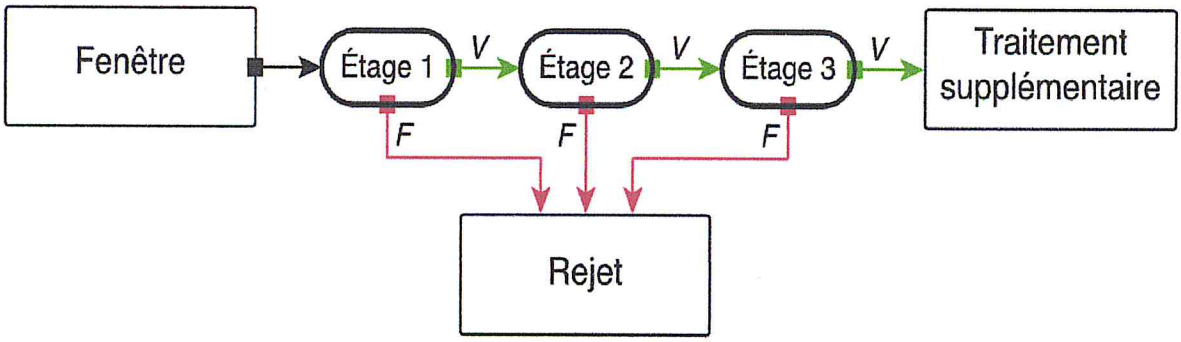


Figure 2.8 : schéma montrant le fonctionnement d'une cascade de classificateur

Ainsi dès que l'un des étages estime qu'il n'y a pas d'objet, la fenêtre est rejetée et l'algorithme passe à la suite ce qui permet un gain de temps considérable.[20]

La figure [2.9] nous montre un Schéma fonctionnel de l'algorithme de viola et Jones

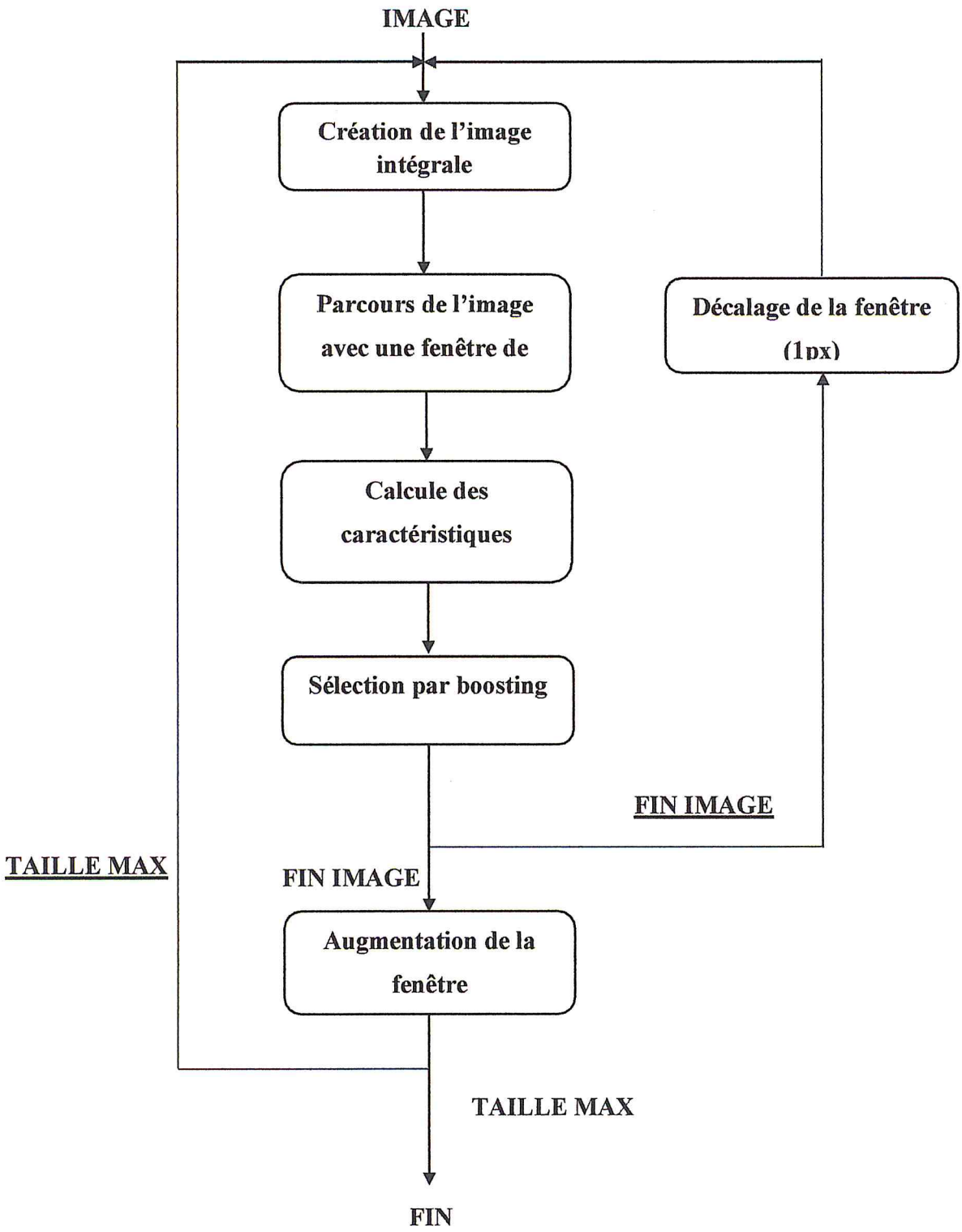


Figure 2.9 : Schéma fonctionnel de l'algorithme de viola et Jones

Pour l'étape de calcule des caractéristiques (pseudo-haar) elle est réalisée selon le besoin souhaité dans notre cas ont a créé notre propre classificateur de haar dédier uniquement à la détection de la forme spécifique du sang que nous recherchons. Les étapes de création de ce classifieur seront détails dans le chapitre d'implémentation et réalisation.

La figure [2.10] est un schéma montrant les étapes de création d'un classificateur

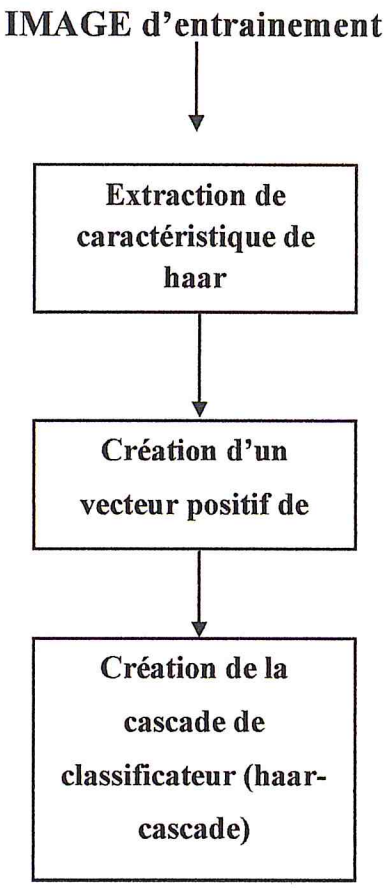


Figure 2.10 : Schéma fonctionnel de l'algorithme de viola et Jones

Pour ceux qui est de l'outil final il va utiliser plusieurs technique décrite dans les chapitre précédent ceci dans un ordre bien précise.

En premier il détectera les diffèrent scène contenu dans la vidéo en utilisant la méthode de l'histogramme comparaison, en deuxième lieu un filtre nuance rouge sera appliqué à la vidéo afin de pouvoir mieux détectant les scènes contenant potentiellement du sang et pour finir à ces même scène filtré on appliquera l'algorithme de haar cascade suivant un cascade de classifieur qu'on aura déjà implémenté, l'organigramme de la figure [2.11] montre le fonctionnement de notre outils de détection de scène violente.

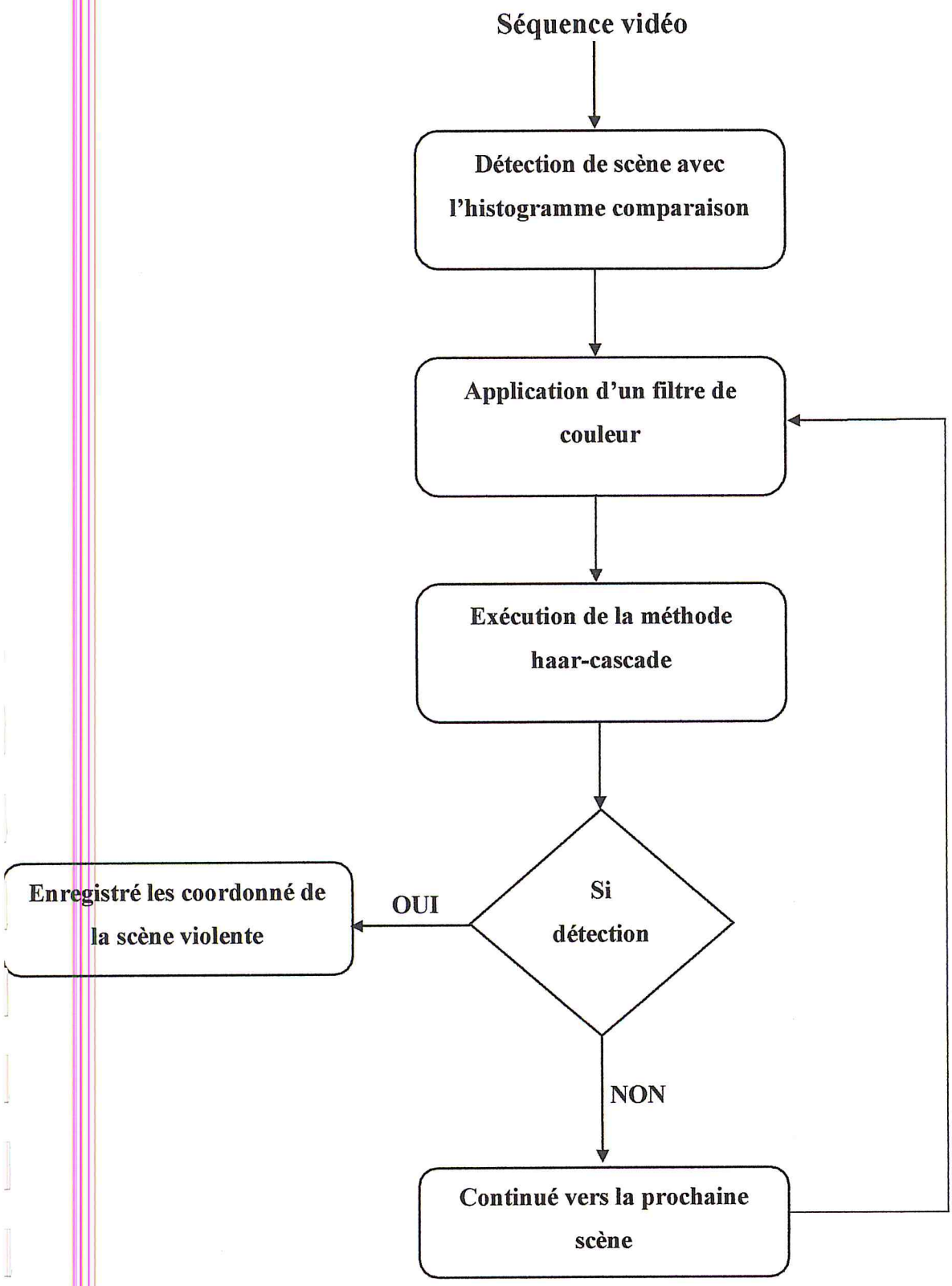


Figure 2.11 : Organigramme de fonctionnement de l'outil de détection de scène violente

Chapitre 3 : Implémentation et test

3.1 Introduction

Pour pouvoir utiliser l'algorithme de Haar cascade il faut crée ce qu'on appelle une cascade de classificateur qui est basé sur les caractéristique de Haar qu'on a vue précédemment il y a plusieurs étapes successive dans la création qui sont :

-Collection d'images d'apprentissage négatives

-Collection ou création d'image positive

-création d'un .vec (vecteur) fichier basé sur les images positive

-formation du classificateur en utilisent les outils creatsamples et traincascade

3.2 Étape 1 : collection d'images négatives

Nous avons aussi besoin de recueillir des images négatives qui ne contienne pas l'objet d'intérêt, par exemple des avions, voiture, fleurs … etc.

Il existe des base de donnée d'image déjà prête afin d'évité de récolté une à une les images rechercher, dans notre cas on a utilisé 800 images négatives pour réaliser notre cascade de classificateur.

Tout c'est images doivent avoir la même résolution pour notre cas elles sont de dimension 200*200 Figure [3.1].

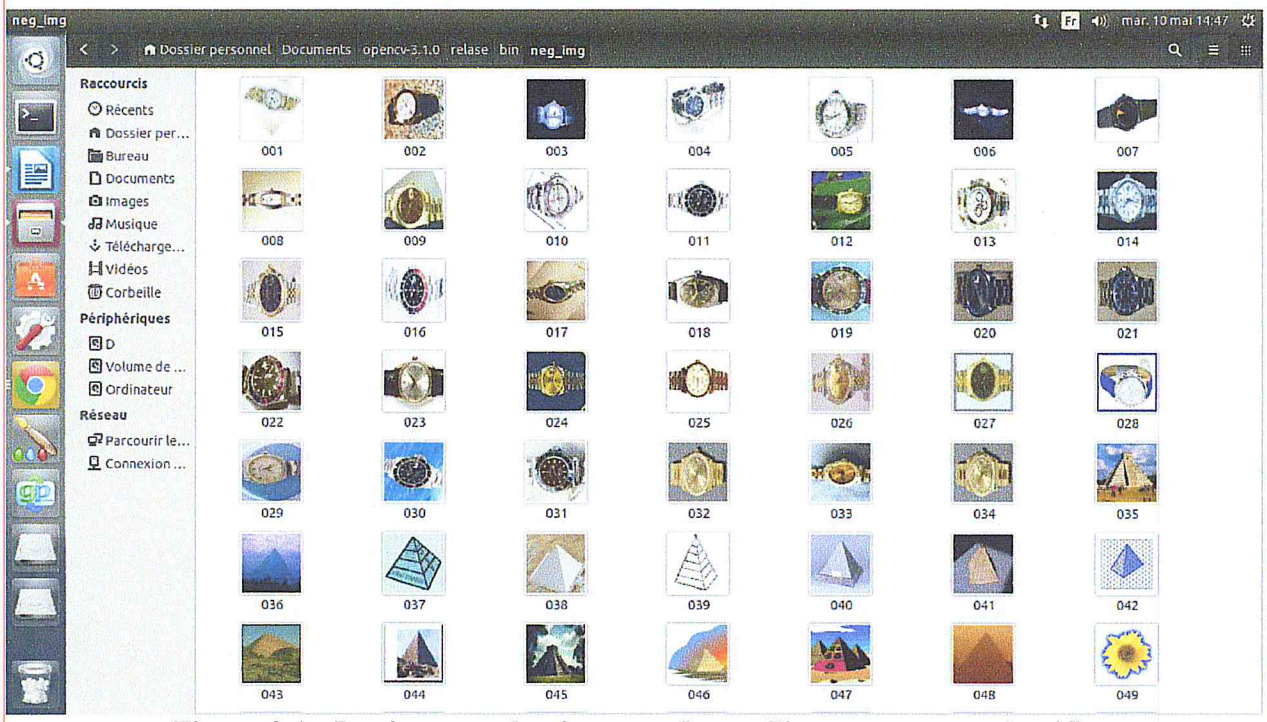


Figure 3.1 : Les images négative quand va utiliser pour notre classificateur

Toutes ces images négatives doivent être accompagnées d'un fichier de description généralement appelé bg.txt [figure 3.2] qui contient le chemin d'accès de toutes les images négatives ligne par ligne

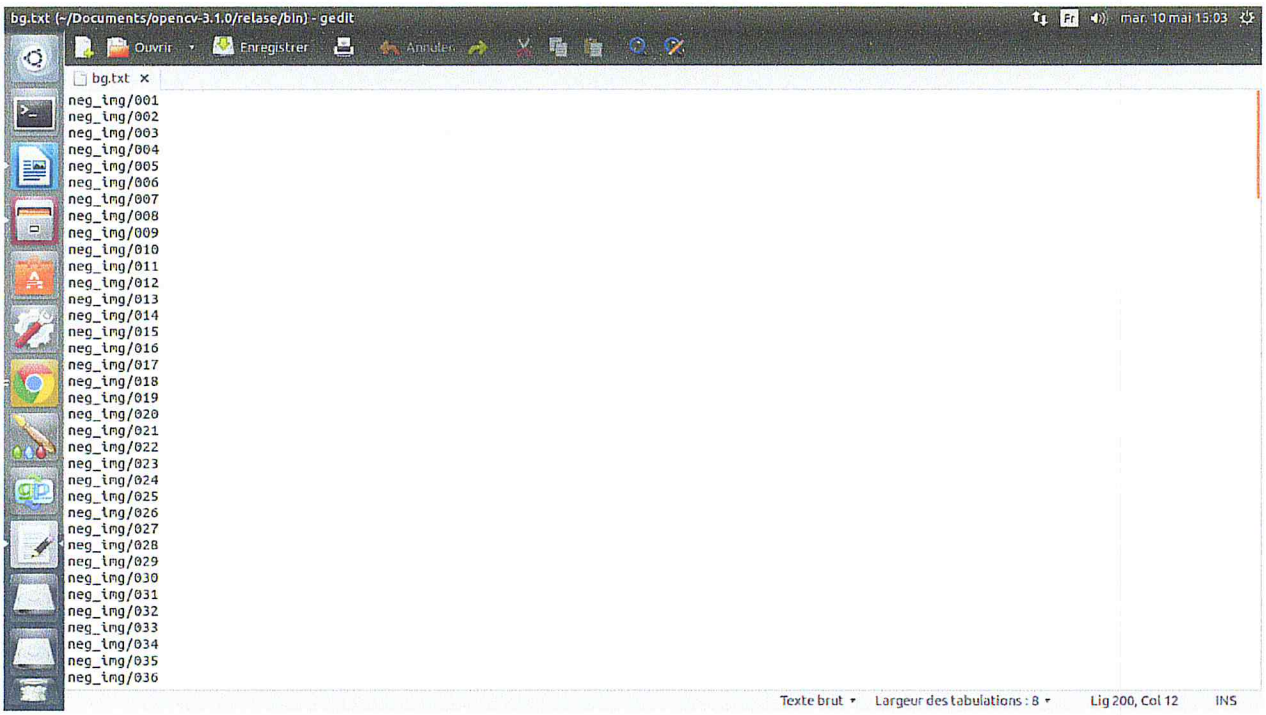


Figure 3.2 : Contenu du fichier BG.txt

3.3 Étapes 2 : Collection ou création d'image positive

Nous avons besoin de millier d'images de l'objet au quelle on s'intéresse et pour cela il existe deux méthodes,

3.3.1 Méthode 1 «Manuel »:

Où on doit récolter tous les images positive une à une puis spécifier dans un fichier texte généralement appeler info.txt, ou chaque ligne contient le chemin de chaque image positive ainsi que le nombre d'objet, les coordonné du point de départ X et les coordonnée du rectangle ou se situe l'objet

Exemple de ligne :

pos/1.jpg 1 51 476 171 478

pos/1.jpg: chemin d'accès de l'image

1: le nombre d'objet a détecté

51 476: coordonnée du point de départ X

171 478: résolution de carré ou se trouve l'objet

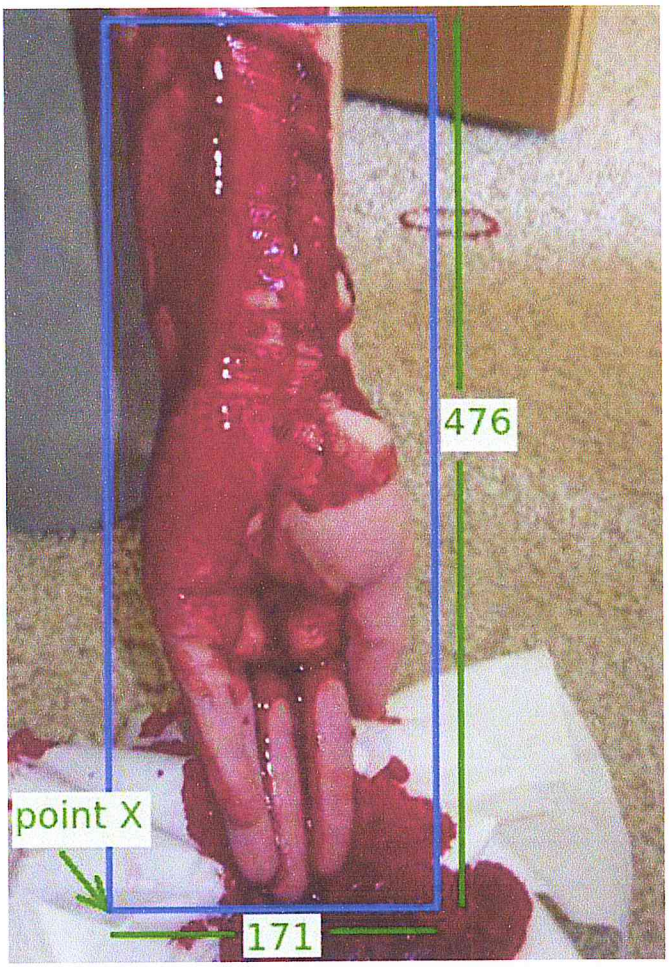


Figure 3.3 : Image positive 1.jpg

3.3.2 Méthode 2 « Automatique » :

Cela à partir d'une seul image positive ou elle va être superposé sur les images négative tout en subissant des rotations aléatoire sur les 3 axe (x, y, z) ainsi les images positive vont être crée.

Les images positive doivent avoir une résolution moins important que les images négative afin qu'elle puisse être superposé.

Pour cette méthode on utilise un utilitaire opencv_createsamples qui est présente l'or de l'installation de OpenCV.

3.3.3 OpenCV_CreateSamples :

Cette utilitaire est exécuté en forme de ligne de commande avec plusieurs argument d'entré.

Arguments de ligne de commande Figure [3.4] :

-vec <nom du fichier vec> :

Nom du fichier de sortie contenant les échantillons positifs pour la formation.

-img <nom du fichier image> :

Image de l'objet source (par exemple, un logo d'entreprise).

-bg <liste des images d'arrière-plan> :

Fichier de description d'arrière-plan; contient une liste des images qui sont utilisées comme un arrière-plan pour les versions déformées de manière aléatoire de l'objet.

-num <nombre d'échantillons> :

Nombre d'échantillons positifs à générer.

-bgcolor <background_color> :

La couleur de fond (actuellement images en niveaux de gris sont pris en charge); la couleur de fond représente la couleur transparente.

-inv :

Si spécifié, les couleurs seront inversées.

-randinv :

Si spécifié, les couleurs seront inversées au hasard.

-maxidev <déviation maximale d'intensité> :

Déviation d'intensité maximale dans les échantillons de pixels de premier plan.

-maxxangle <rotation max sur l'axe des x> :

Angles de rotation maximum sur l'axe des x doit être donnés en radians.

-maxyangle <rotation max sur l'axe des y> :

Angles de rotation maximum sur l'axe des x doit être donnés en radians.

-maxzangle <rotation max sur l'axe des z> :

Angles de rotation maximum sur l'axe des x doit être donnés en radians.

-show :

Option de débogage utile. Si spécifié, chaque échantillon sera affiché. En appuyant sur Echap continuera le processus de création d'échantillons.

-w <largeur d'échantillons> :

La largeur (en pixels) des échantillons de sortie.

-h <hauteur d'échantillons > :

La hauteur (en pixels) des échantillons de sortie.

-pngoutput :

Avec cette option activée opencv_createsamples outil génère une collection d'échantillons PNG et un certain nombre de fichiers d'annotation associés, au lieu d'un fichier unique vec.

L'utilitaire opencv_createsamples peut travailler dans un certain nombre de cas à savoir:

Création d'une formation mis à partir d'une seule image et une collection d'image négative:

avec un fichier unique en tant vec sortie (méthode automatique).

avec une collection d'images JPG et un fichier avec la liste des annotations en tant que sortie (méthode manuel).

Figure 3.4 : Liste des arguments de l'utilitaire opencv_creatsamples

-img 001.jpg: le nom de l'image positive d'où les échantillons vont être crée

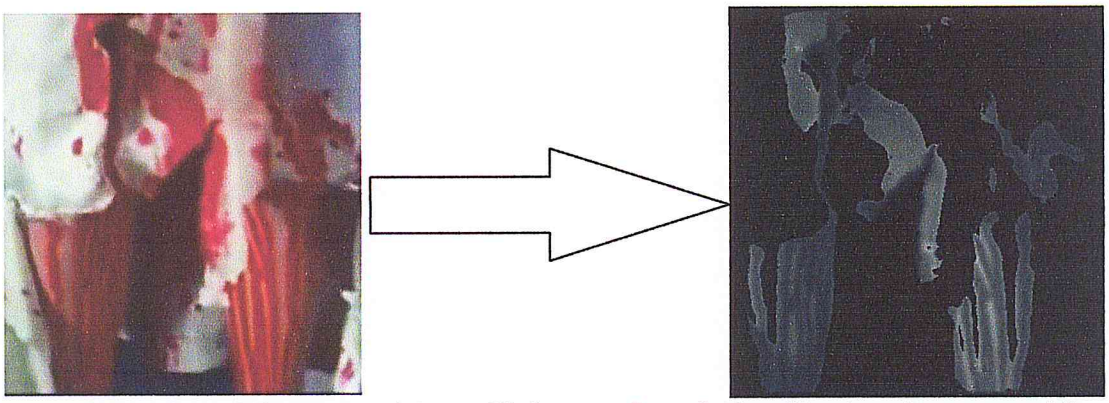


Figure 3.5 : L'image positive utilisée pour la création d'autres images positives après application du filtre de couleur rouge (à droite)

-bg bg.txt: est le fichier contant les arrière-plans (les images négatives)

-info info/info.lst: le fichier de description des images positives

-pngoutput info: ou les images positives vont être stocké ici dans le dossier info

-maxxangle,-maxyangle,-maxzangle: les valeurs des distorsions qui vont être appliqué a l'image 001.jpg

-num: le nombre d'image négative qui va être utilisé

Le procédé va superposer l'image négative (001.jpg) sur les 800 images négative afin de crée 800 image positive figure [3.6] ainsi qu'un fichier info.txt Figure [3.7] qui va être généré automatiquement contant les coordonné de chaque image positive.

Figure 3.6 : Exemple d'une image positifs générer avec opencv_createsamples

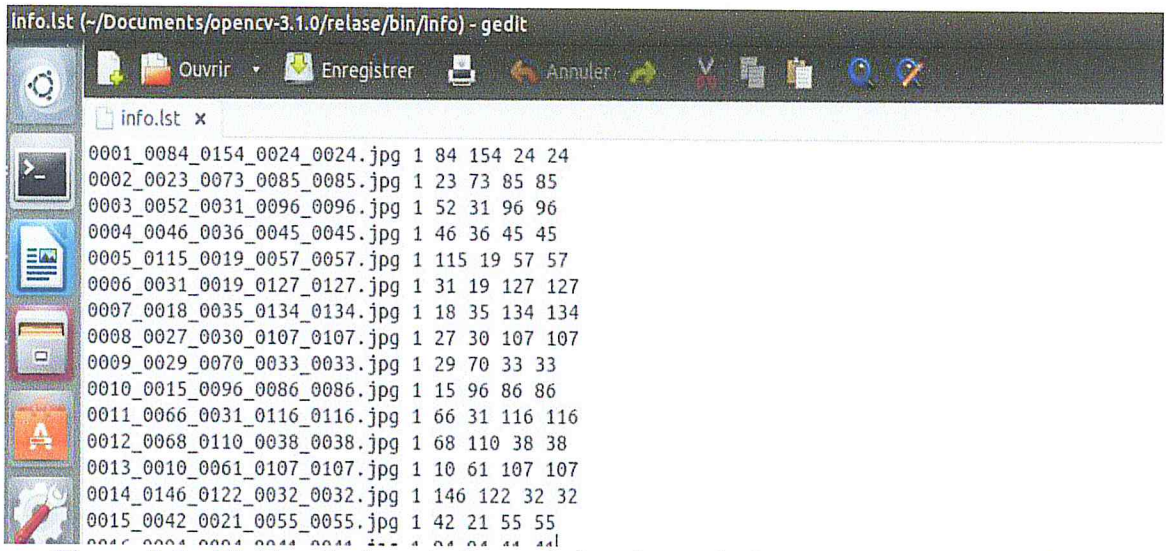


Figure 3.7 : Fichier de descriptions générer à partir de opencv_createsamples

Dans le cas de l'utilisation de la méthode manuel toute cette étape va être ignorée car on va créer les images positives manuellement avec le descripteur comme cité ci-dessous

3.4 Étape 3 : créations du .vec file

Pour cette étapes on va utiliser le même utilitaire opencv_createsamples afin de pouvoir crée le vec fille depuis les images positive généré dans l'étape précèdent,

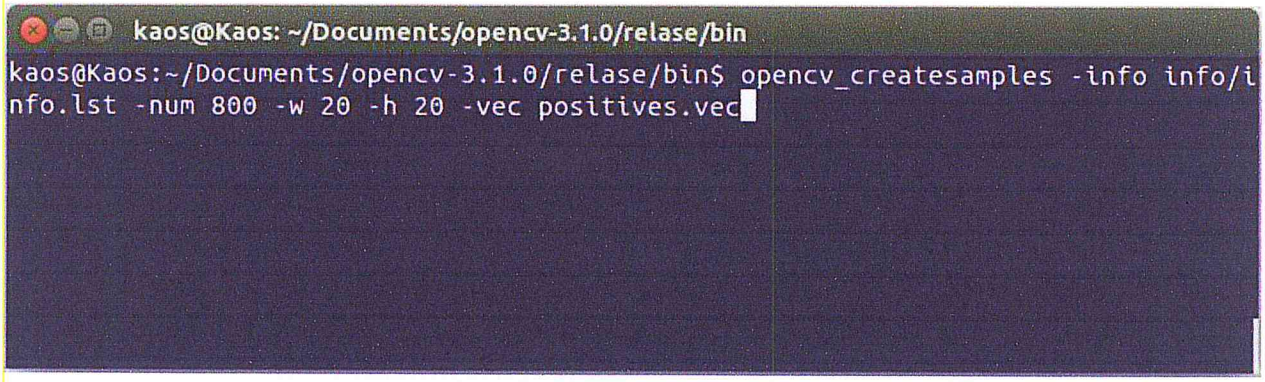


Figure 3.8 : Commande utilisé pour la création du .vec file

Argument de la commande Figure 3.8 :

-info :

Désigne l'emplacement du fichier info.lst vue ci-dessous.

-num :

Spécifier le nombre d'image qui va être utilisé pour la création du vec files.

-w,-h :

Est la hauteur et largeur du champ de caractéristique qu'il va être utilisé plus la valeur est grand plus sa prendre du temps a calculé mais généralement on utilise un carrée de 20 sur 20.

-vec :

Sous quel nom le vecteur va être enregistré.

Une fois les 3 étapes effectuer avec sucée la création des cascade de caractéristique peuvent être crée pour cela ont utilisent un autre utilitaire lui aussi fourni par OpenCv_Traincascade

3.4.1 OpenCv_Traincascade :

Arguments de ligne de commande figure [3.9] :

-data < emplacement de la sauvegarde> :

Ou la cascade de classificateur devrait être sauvegardée

-vec < nom du fichier vecteur> :

Nom du fichier vecteur crée précédemment par l'utilitaire OpenCV_Createsamples

-bg <nom du fichier de descriptions d'arriérer plans > :

Nom du fichier qui contient les descriptions des arriérer plans (les images négative)

-NumPos <nombre des échantillons positifs> :

Il doit être le double des négatives.

-NumNeg <nombre des échantillons négatifs> :

La moitié des positive

-NumStages <nombre de stages > :

Le nombre de cascade de classifieur qui va être générer

-w,-h :

Taille des échantillons de formation (en pixels). Elles doivent avoir exactement les mêmes valeurs que celles utilisées lors de l'étape de création des échantillons avec l'utilitaire opencv_Createsamples ici dans notre cas -w20,-h 20

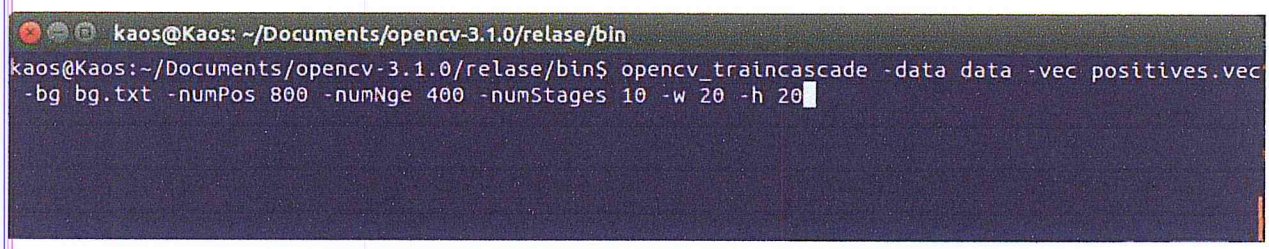


Figure 3.9 : Exemple de la commande opencv_traincascade utilisé dans notre cas

Une fois la commande exécuter un long processus d'apprentissage commence, suivant le nombre de stages a entraîné ainsi que le nombre d'images utiliser le temps peut varier de quelque minute a quelque heurs

À la fin du résultat ont obtiens un fichier nommé cascade.xml contenant le descripteur crée.

3.5 Test :

L'outil de détection a étais codé en utilisant le langage python ainsi que la bibliothèque openCv tout cela dans un environnement de travail dont les caractéristique sont les suivant :

Processeur : Intel(R) Core ™ i3-2375M CPU @ 1.50 GHZ

RAM : 4.00 GO

OS : Linux Ubuntu 14.04 LTS

Compilateur : python, Gnu gcc,

Il se présente sous forme d'exécutable, une fois exécuté une fenêtre apparaît figure [3.10] qui nous permet de choisi n'importe quelle contenu vidéo

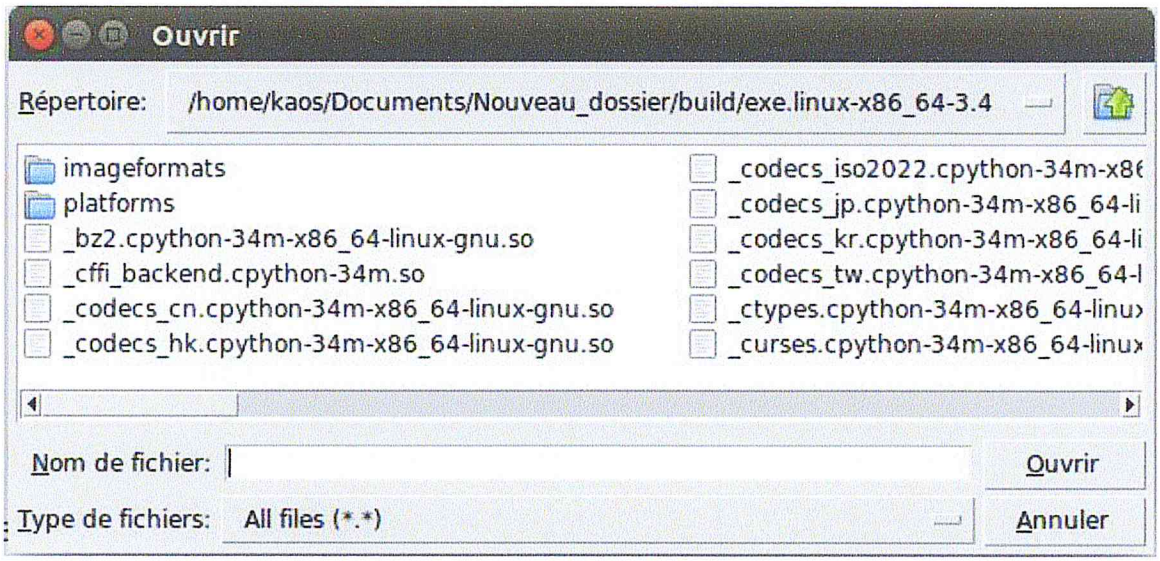


Figure 3.10 : fenêtre de sélection du fichier vidéo à analyser

Une fois le fichier sélectionné le processus de traitement commence, un dossier est créé portant le nom du fichier ajouté ou tous les résultats obtenus vont être enregistré sous forme d'image nommé par leur temps de détection de la scène dans la vidéo afin de pourvoir retrouvé la scène rapidement figure [3.11]

Figure 3.11 : résultat partiel du film SAW

Tous les résultats obtenu nous sont pas positifs une marge d'erreur existe et l'outil détecte aussi des faux positifs

Nous allons voir tout cela plus en détail dans le prochain chapitre ou en détaillera les résultats obtenu après plusieurs phase de tests.

Chapitre 4 : résultat et discussion

Introduction:

Ce dernier chapitre est consacré au résultat obtenu lors de la phase de test pour cela on a testé 9 film de différentes catégories qui varie des films les plus sanglant jusqu'au film pour enfant, les films testé sont :

Evil Dead

SAW VI

Destination Final

Gangster squad

Pirate des caraïbes

Harry Potter et le prince de sang mêlé

Mise à l'épreuve

Une nuit au musé

Un flic à la maternelle

L'outil de détection calcule en premier lieu, le nombre de scènes total détectées dans la vidéo. Sachant que le nombre d'images dans une vidéo est de 24 images par seconde, un choix sur les images à comparer s'impose afin de garantir un résultat meilleur. Il s'agit de trouver le meilleur intervalle d'images à comparer. Pour cela, nous avons fait une série de tests sur la même vidéo dont le nombre total de scènes était de 17. Le test est effectué en augmentant l'intervalle des images testées progressivement jusqu'à atteindre le résultat souhaité. Le tableau … résume les résultats obtenu.

4.1 Résultat :

Tableau 4.1 : nombre de scènes selon la fréquence d'intervalle d'images

Fréquence d'intervalle d'images	1/s	2/s	3/s	4/s	5/s	6/s	7/s	8/s	9/s	10/s
Nbr de scènes obtenues	1	5	7	8	8	9	10	13	15	17

D'après les résultats obtenus, on note qu'un intervalle de dix images par seconde s'avère être un bon choix.

La figure [4.1] montre le nombre de scène total associé à chaque film.

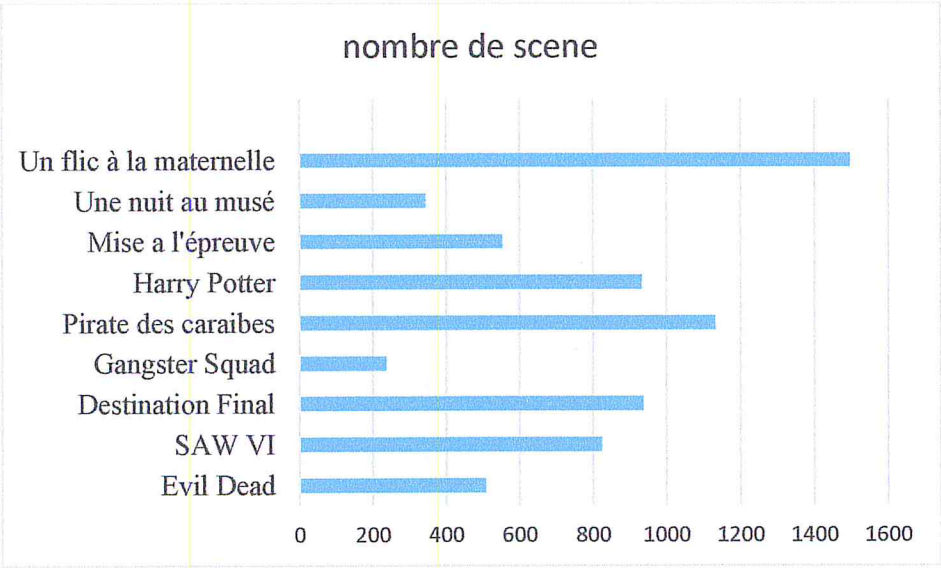


Figure 4.1 : Nombre de scène total pour chaque film testé

En ce qui concerne les scènes violentes figurant dans les films considérés nous avons d'abord eu recours au dénombrement direct sans utilisation de l'outil. C'est-à-dire toutes les scènes sanglantes ont été comptées en regardant tous les films en intégralité. Puis en second lieu, nous avons appliqué l'outil pour la détection automatique des scènes violentes.

L'histogramme représenté dans la figure [4.2] résume le nombre de scènes sanglantes obtenu avec et sans l'utilisation de l'outil.

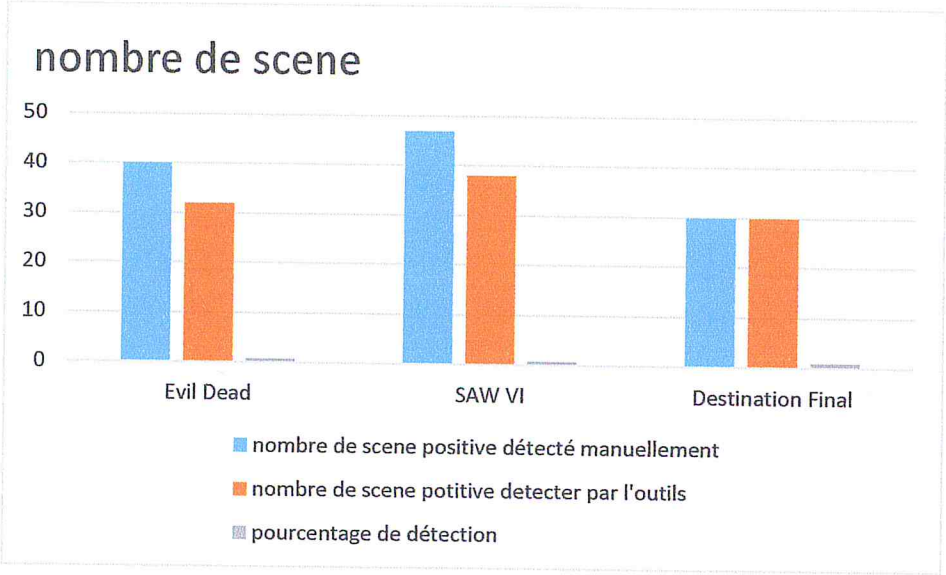


Figure 4.1 : Scène positive détecté de manier automatique et manuelle

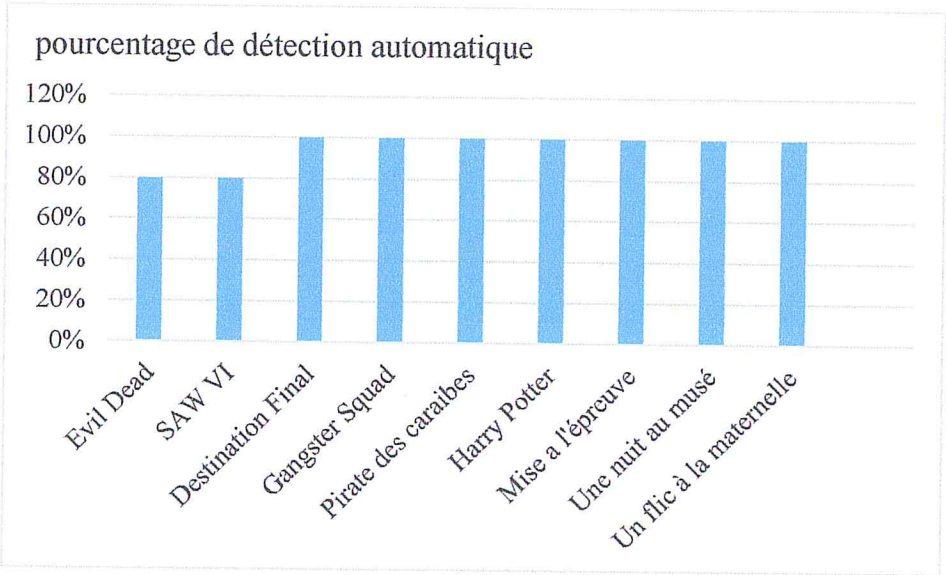


Figure 4.2 : Scène positive détecté de manier automatique et manuelle

Il est à noter que l'outil a généré, lors de la phase de test, des faux positifs, c'est-à-dire des scènes identifiées comme violentes ou sanglantes alors qu'elles ne le sont pas. La figure [4.4] représente le nombre de faux positifs obtenus pour chaque film.

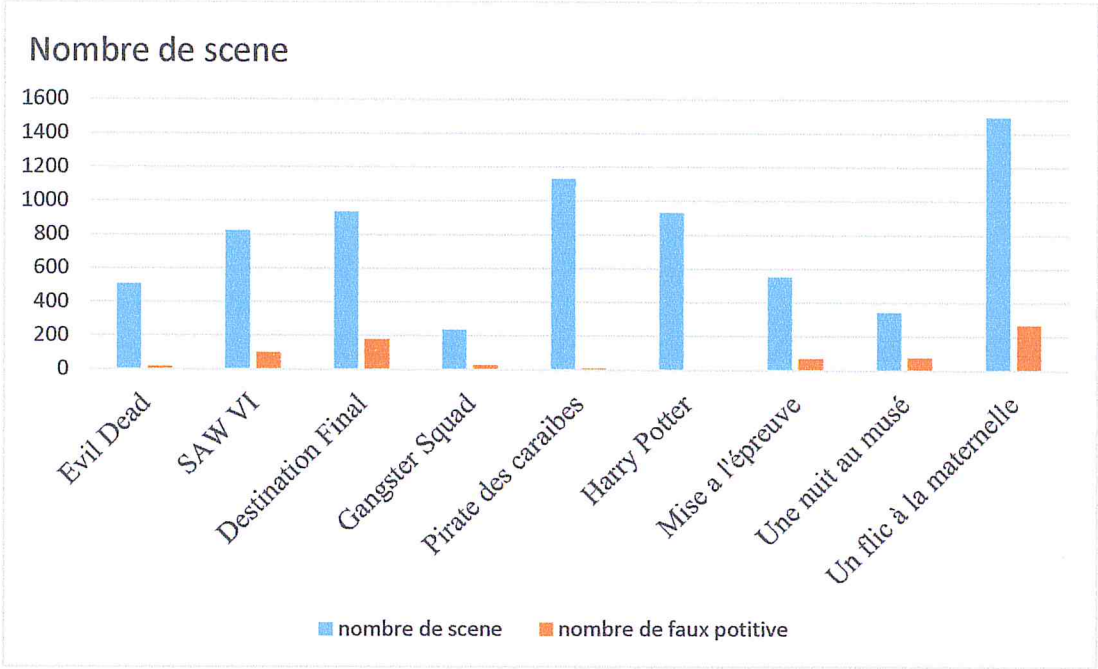


Figure 4.3 : Nombre de faux positifs obtenu pour chaque film

Le nombre de faux positifs rapporté au nombre total de scènes dans un film donne le taux de faux positifs pour chaque film testé. La figure [4.5] représente le taux de faux positifs pour chaque film testé

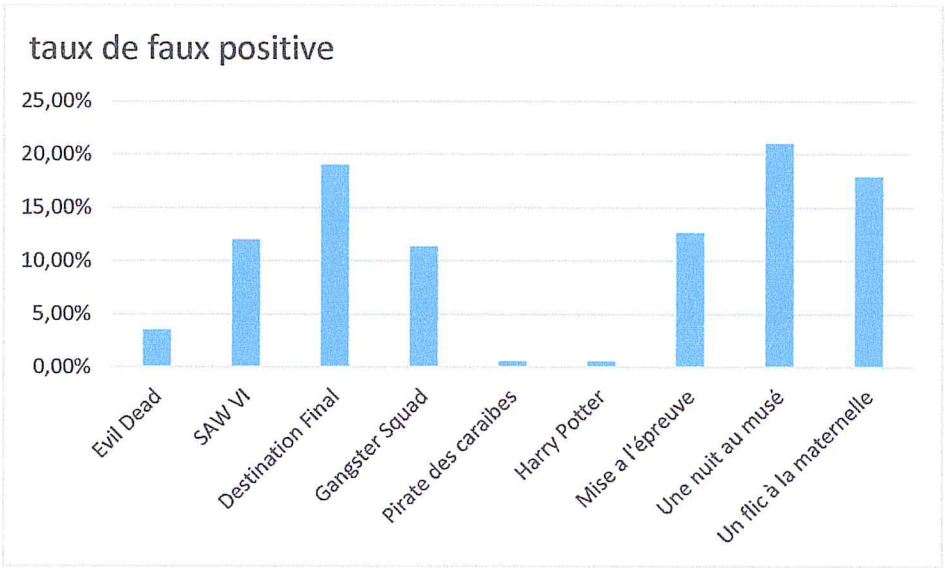


Figure 4.4 : Taux de faux positifs pour chaque film testé

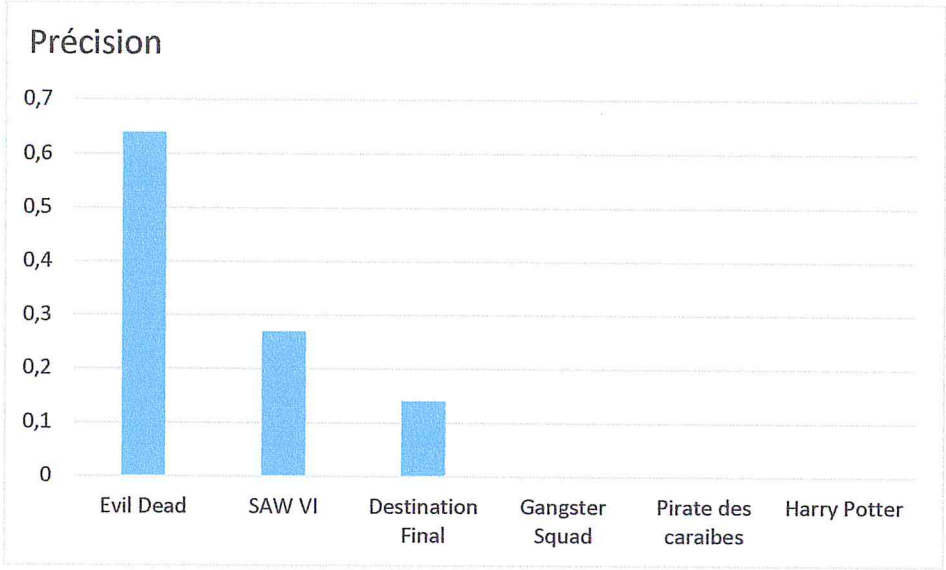


Figure 4.5 : taux de précision en suivant la F-Mesure

En ce qui concerne les résultats obtenu en suivant la F-Mesure cela nous donne un très bon taux pour le premier film (Evil Dead) en revanche la précision diminue considérablement pour les deux autre films (SAW VI Destination Final)

Tableau 4.2 Résume tous les résultats obtenus par les tests effectués

Film	Evil Dead	SAW VI	Destination Final	Gangster Squad	Pirate des caraïbes	Harry Potter	Mise à l'épreuve	Une nuit au musé	Un flic à la maternelle
Nbr scène	507	823	937	236	1132	932	552	345	1497
Nbr positifs détecté manuellement	40	47	30	0	0	0	0	0	0

Nbr positive détecté automatiquement	32	38	30	0	0	0	0	0	0
Nbr de faux positive détecté	18	99	178	27	6	5	70	73	269
Taux de faux positif détecté	3.5%	12.02%	19%	11.39%	0.53%	0.53 %	12.68%	21%	17.90%
Taux de positif détecté	80%	80%	100%	100%	100%	100%	100%	100%	100%
Taux de précision	0.64	0.27	0.14	0	0	0	0	0	0

Conclusion générale et perspective

Notre travail a porté sur la réalisation d'un outil de détection de scènes violentes dans une séquence vidéo afin de réduire leur impact négatif sur un publique spécifique ainsi que le temps de traitement et tout cela en rendant le processus automatique.

La solution proposée se présente sous la forme d'un outil écrit sous python et OpenCv qui détecte automatiquement et dans un premier temps toutes les scènes dans la séquence vidéo traitée. Puis, en deuxième lieu, il fait un tri parmi toutes ces scènes pour ne garder que celles qui sont potentiellement violentes. Les résultats obtenus après l'application de l'outil sur neuf films différents sont très satisfaisants. En effet un taux moyen de (86%) de résultats positifs a été obtenu. Néanmoins l'outil utilisé présente un inconvénient qui est la génération de résultats faux positifs .Un taux moyen de 11% de faux positif a été observé.

Les difficultés auxquelles nous étions confrontées nous ont permis de fournir d'avantage d'effort afin d'achever le travail dans le délai impartis et selon la qualité demandée. En effet il fallait à la fois :

Se familiariser avec le langage python ainsi que la très vaste bibliothèque OpenCv qui nous était totalement inconnue.

Comprendre le fonctionnement de base des traitements d'image.

Trouver une solution a une problématique majeure en termes de détection de scène violant dans des séquences vidéo qui reste à ce jour un sujet ouvert.

Implémenter et optimiser l'outil afin qu'il puisse être exploitable et surtout garantir l'aspect évolutif de l'outil.

Nous avons constaté que ce projet était une bonne occasion pour sortir du cadre théorique et appliquer les connaissances acquises lors de notre formation universitaire, tout en se penchant sur une des spécialités de l'informatique qui est la vision par ordinateur.

En perspective, nous pouvons élaborer manuellement un classificateur afin d'améliorer les résultats obtenus en terme de nombre de scènes violentes détectées et aussi d'autres types de classificateurs qui permettront un traitement plus complet des scènes violents et de ne pas se restreindre aux scènes sanglantes seulement.

13. D. Comaniciu, V.R., and P. Meer *Kernel-based object tracking Pattern*

Analysis and Machine Intelligence. May 2003. **vol. 25**, 564–575.

14. J. Jeyakar, R.V.B., and K. R. Ramakrichnan, *Robust object tracking using local kernels and*

background information, in *Image Processing*. Sep. 2007, IEEE: San Antonio, TX, USA. p. 49–52.

15. A. Criminisi, P.P., and K. Toyama, *Region filling and object removal by exemplar-based inpainting*, in *Image Processing*. Sep. 2004, IEEE. p. 1200-1212.

16. Ruan, J.W.a.Q., *Object removal by cross isophotes exemplar-based inpainting*, in *Pattern Recognition*. Aug. 2006, IEEE: Hong Kong, China. p. 810-813.

17. Jones, P.V.M., *Robust Real-time Object Detection*, in *SECOND INTERNATIONAL WORKSHOP ON STATISTICAL AND COMPUTATIONAL THEORIES OF VISION–MODELING, LEARNING, COMPUTING, AND SAMPLING*. JULY 13, 2001.: VANCOUVER, CANADA. p. 1-25.

18. ANDRIAN, I. *COMPARISON BETWEEN VIOLA JONES AND ROBERT CROSS METHODE IN THE FACE RECOGNITION SYSTEM*.

19. Poggio, C.P.P.M.O.T. *A General Framework for Object Detection*. 1-8.

20. PAUL VIOLA, M.J.J. *Robust Real-Time Face Detection*. 2004. **57(2)**, 137–154.