وزارة التعليم العالي والبحث العلمي

BADJI MOKHTAR- ANNABA UNIVERSITY
UNIVERSITE BADJI MOKHTAR ANNABA

جامعة باجي مختار- عنابة

Année : 2019

Faculté: Sciences de l'Ingéniorat
Département: Electronique

MEMOIRE

Présenté, en vue de l'obtention du diplôme de : MASTER

Intitulé :

Etude des performances de l'outil FFmpeg dans le traitement de la vidéo

Domaine : Sciences et Technologie
Filière : Télécommunications
Spécialité: Réseaux et Télécommunications

Par :
Lydia Boutella

DEVANT Le JURY

Président : M. Boutalbi	**MCB**	**UBM Annaba**
Directeur de mémoire : S. Harize	**MCA**	**UBM Annaba**
Examinateur 1 : S. Affifi	**Prof.**	**UBM Annaba**
Examinateur 2: M. Benouaret	**Prof.**	**UBM Annaba**

Résumé :

Parfois, on est amené à utiliser un logiciel sans savoir que derrière celui-ci se cache un autre logiciel qui se charge de traitements très importants. Un parfait exemple est l'outil FFmpeg. FFmpeg est une collection de logiciels libres destinés au traitement de flux audio ou vidéo (enregistrement, lecture ou conversion d'un format à un autre tel que AVI, MP4, MKV, compression utilisant différents codecs et même les plus populaire tels que x.264 ou le tout dernier x.265, découpage de vidéos, ajout d'audio, de sous titres, … etc.). Ce software est utilisé par de nombreux autres logiciels bien connus ou services comme VLC, iTunes ou YouTube. C'est donc un logiciel très intéressant qui mérite une étude assez approfondie.

Dans ce mémoire, les performances du FFmpeg dans le traitement de la vidéo sont présentées, analysées et discutées. La qualité est estimée en termes de PSNR.

Mots clés : FFmpeg, PSNR, AVI, MP4, MKV, x.264, x.265

Abstract:

Sometimes, we are in situation where we are using a software without realizing that behind, lies a particular piece of software in charge of important processing. A perfect example is FFmepg. FFmpeg is a collection of open source utilities intended to audio and video processing (recording, playing or conversion from one format to another such as AVI, MP4, MKV, compression using different codecs and even the most popular ones such as x.264 or the latest x.265, cutting/trimming videos, adding audio, subtitles, … etc). This software is used by several others well known software or services such as VLC, iTunes or YouTube. It is therefore an interesting piece of software which deserves to be deeply investigated.

In this dissertation, FFmpeg performances in video processing are presented, analyzed and discussed. The quality is estimated in terms of PSNR.

Key words: FFmpeg, PSNR, AVI, MP4, MKV, x.264, x.265

ملخص:

في بعض الأحيان ، يتم استخدام أحد البرامج دون معرفة أن وراء هذا يخفي برنامج آخر مسؤول عن العلاجات المهمة للغاية. مثال مثالي هو أداة FFmpeg .FFmpeg عبارة عن مجموعة من البرامج المجانية لمعالجة تدفقات الصوت أو الفيديو (التسجيل أو التشغيل أو التحويل من تنسيق إلى آخر مثل AVI و MP4 و MKV والضغط باستخدام برامج الترميز المختلفة وحتى الأكثر شعبية مثل x.264 أو أحدث x.265 ، قص مقاطع الفيديو ، إضافة الصوت ، الترجمة ، ... إلخ). يتم استخدام هذا البرنامج بواسطة العديد من البرامج أو الخدمات المعروفة الأخرى مثل VLC أو iTunes أو YouTube. لذلك فهو برنامج مثير للاهتمام ويستحق دراسة شاملة إلى حد ما.

في هذه الأطروحة ، يتم تقديم عروض FFmpeg في معالجة الفيديو وتحليلها ومناقشتها. يتم تقدير الجودة من حيث PSNR.

الكلمات الرئيسية: FFmpeg، PSNR، AVI، MP4، MKV، x.264، x.265

Dédicaces

A ma Chère Mère Hafidha
A mon Chère Père Azzedine
dont le mérite, les sacrifices et les qualités
humaines m'ont permis de vivre ce jour.

A mes sœurs Sandra et Nour.

A tous les gens qui m'aiment
Aicha, Nafissa, Safia, Fatima, Alla, Fodil,
Benyoucef……

(Lydia)

REMERCIEMENTS

A Mon Encadreur, Mme Harize Saliha,

J´ai eu l'honneur de faire partie de vos étudiants, et de bénéficier de votre riche enseignement.
Vos qualités pédagogiques et humaines sont pour moi, un modèle.
Votre gentillesse, et votre disponibilité permanente ont toujours suscité mon admiration.

Aussi, veuillez recevoir Madame, tous mes sincères remerciements pour le grand honneur que vous m´avez fait en acceptant l'encadrement de ce travail.

Aux membres du jury,

-Président : Mr. Boutalbi
-Examinateurs: Mr S.Affifi et Mr M.Benouaret,

Messieurs les jurys, vous nous faites un grand honneur en acceptant d'évaluer notre travail. Soyez convaincus de notre profond respect.

A mes enseignants du Département électronique,

Je vous dois un vif remerciement pour vos qualités scientifiques et pédagogiques des enseignements que vous m'avez inculqués.

A mes proches et amis,

Je tiens à vous exprimer ma gratitude, pour toute votre aide et soutien, de prés ou de loin, qui m'ont permis d'accomplir ce Travail.

LISTE DES ABREVIATIONS

A:

AVI: Audio Video Interleave.

APT: Advanced Packaging Tool.

C:

CRF: Constant Rate Factor.

F:

FLV: Flash Video.

FFmpeg: Fast Forward MPEG.

H:

HEVC: **H**igh **E**fficiency **V**ideo **C**oding.

HM: HEVC Test Model.

J:

JPEG: Joint Photographic Experts Group.

M:

MJPEG: Motion JPEG.

MKV: Matroska.

MP4: Moving Picture 4.

P:

PSNR: sigle de Peak Signal to Noise Ratio.

PPA: Personal Package Archives.

Q:

QT: QuickTime.

LISTE DES TABLES

LISTE DES FIGURES

TABLE DES MATIERES

Table des matières.

Introduction générale:

Depuis ces deux derniers siècles, l'humanité a fait des pas énormes à ce qui a trait à l'innovation et aux progrès technologiques. Toute cette grande évolution a grandement servi à l'homme dans son ensemble et aussi à améliorer de façon considérable son niveau de vie.
Chaque année, nous voyons une diminution du texte, la vidéo a pris le dessus et est devenue la meilleure façon de raconter des histoires dans ce monde, où tant d'information vient à nous.

La vidéo a une importance capitale dans le déroulement d'une bonne communication digitale et elle est incontournable dans cette période où c'est la bonne image qui fait le succès. C'est parce que tout simplement, la vidéo est un support très attractif qui ne passe jamais inaperçu.
L'homme est naturellement animé par la curiosité et il lui suffit de voir une image illustrative d'une vidéo pour tenter de découvrir ce qui se trouve dans le contenu.

La vidéo se présente sous forme de fichiers de très grandes tailles qui nécessitent souvent des traitements : Edition, stockage ou transmission. Les utilisateurs de vidéos sont de plus en plus exigeants en termes de qualité. Mais cette dernière ne peut être assurée puisque les bandes passantes des supports de transmission sont limitées.

Cette contradiction est derrière le développement d'un grand nombre de codecs pour la compression/décompression et d'outils pour le traitement des vidéos FFmpeg est l'un de ces outils.C'est un ensemble de logiciels libres destinés au traitement de flux audio ou vidéo. Cette bibliothèque est utilisée par de nombreux autres logiciels ou services comme VLC, iTunes ou YouTube. Développé sur GNU/Linux, FFmpeg peut être compilé sur la plupart des systèmes d'exploitation. Ce logiciel a suscité un grand intérêt, notamment dans le traitement de la vidéo (sans l'audio) et a donc fait l'objet de l'étude menée et présentée dans ce mémoire.

Ce mémoire comporte trois chapitres comme suit: Le premier chapitre intitulé " Généralités sur la vidéo " présente les aspects de base de la vidéo. Dans le deuxième chapitre, nous décrivons "L'outil FFmpeg" et sa gestion. " Etude des performances du FFmpeg dans le traitement de la vidéo " est l'objet du troisième chapitre.

Chapitre I
Généralités sur la vidéo

I.1 INTRODUCTION :

Une vidéo permet de mieux comprendre et mieux retenir : 65% des gens retiennent plus facilement des visuels que toute autre forme de communication.

C'est un outil plus facile d'accès et plus ludique qu'un texte à lire. C'est un support digital qui permet une diffusion multi canaux : sur un ordinateur, un téléphone portable, une télé, un écran d'affichage…Dans ce chapitre, la vidéo sera le centre d'intérêt.

I.2 Notions de base :

I.2.1 Définition de la vidéo :

La vidéo est une succession d'images animées défilant à une certaine vitesse. Elle est basée sur le principe fondamental de la rétention de l'œil humain, pendant un certain temps (de l'ordre du dixième de seconde), de toute l'image imprimée sur la rétine. On distingue deux grandes familles de systèmes vidéo : les systèmes vidéo analogiques et les systèmes vidéo numériques.

I.2.1.1 La vidéo analogique : [1]

La vidéo analogique est un signal vidéo pouvant prendre n'importe quelle valeur située entre deux extrémités. Le principe de l'analogique est de reproduire le signal à enregistrer sous forme similaire sur un support magnétique (en général). Le signal est en forme d'onde et est très sensible aux perturbations extérieures. Ces perturbations peuvent entraîner une modification importante du signal.

I.2.1.2 La vidéo numérique :

La vidéo numérique consiste en une succession d'images numériques.

Puisqu'il s'agit d'images numériques affichées à une certaine cadence, il est possible de connaître le débit nécessaire pour l'affichage d'une vidéo, c'est-à-dire le nombre d'octets affichés (ou transférés) par unité de temps. Ainsi le débit nécessaire pour afficher une vidéo (en octets par seconde) est égal à la taille d'une image que multiplie le nombre d'images par seconde.

I.2.1.3 Principe de la conversion analogique/numérique :

La première étape de la conversion analogique/numérique est l'échantillonnage : l'amplitude du signal analogique est prélevée ponctuellement à des instants réguliers et suffisamment rapprochés.

Il faut ensuite remplacer leurs valeurs par un nombre entier de longueur fixe, codé en base 2 c'est la seconde étape de la numérisation du signal, appelée quantification.

Puis vient l'étape du codage, au cours de laquelle le flux de données est mis en forme en vue de son stockage ou de sa transmission. Par ailleurs, pour conférer aux informations une

bonne immunité face aux perturbations amenées par le support d'enregistrement ou de transmission, il est nécessaire de leur adjoindre un certain nombre de données supplémentaires qui permettront, lors du décodage, de détecter et corriger les erreurs introduites.

Ces données redondantes ajoutées aux données utiles ne font cependant qu'accroître la quantité déjà énorme d'informations à débiter par unité de temps.

I.2.2 Caractéristiques d'une vidéo numérique :

I.2.2.1 Constitution d'une image numérique :[2]

Une image numérique contient un nombre fini de points appelés pixel**.** Les pixels sont situés sur une grille régulière. A chaque pixel de la grille est associée une couleur ou une nuance de gris. Le passage d'une image continue à une grille de pixels est réalisé par l'opération d'échantillonnage : on ne conserve que quelques points d'une image continue. La taille en pixel définit la résolution par rapport à l'image analogique originale, c'est-à-dire la finesse de la grille. Plus la résolution baisse, plus le nombre de pixels dans l'image diminue, et plus la qualité de l'image numérique se dégrade.

I.2.2.2 Définition des pixels :

Les **pixels** sont les plus petits éléments constitutifs d'une image numérique. Le nom de "pixel", abrégé **px**, provient de l'expression anglaise « **picture element »**, qui signifie "élément d'image" ou "point élémentaire".

Un pixel est généralement rectangulaire ou presque carré et présente une taille comprise entre 0,18 mm et 0,66 mm de côté.

La quantité de pixels composant un écran détermine sa résolution, exprimée en points par pouce (un pouce représentant 2,54 cm). Cette unité de mesure est abrégée PPP en français ou DPI en anglais pour Dots Per Inch.

I.2.2.3 Définition de la résolution :

La résolution définit la netteté et la qualité d'une image. Plus la résolution est grande (c'est-à-dire plus il y a de pixels dans une longueur de 1 pouce), plus l'image est précise dans les détails. Pour une vidéo, elle est exprimée en largeur x hauteur et où la largeur et la hauteur sont exprimées en nombre de pixels.

I.2.2.4 La taille d'un fichier vidéo:

La définition d'un fichier vidéo est le nombre total de pixels qui le compose. Plus elle est grande, plus la qualité de la vidéo est meilleure, cependant, plus le fichier devient volumineux. La taille d'un fichier vidéo se calcule de la façon suivante :

$$Taille = (Nbr\_frames) \times (résolution) \times (frame\_rate) \times 24 \quad (1)$$

Note : 24 =8x3 (si un pixel est codé sur 8 bits et que l'image est en couleur RGB)

I.2.2.5 Cadence d'une vidéo :

Une vidéo est une succession d'images à une certaine cadence. L'œil humain est capable de distinguer environ 20 images par seconde. Avec la cadence (20 images par seconde), il est possible de tromper l'œil et de lui faire croire à une image animée. On caractérise la fluidité (vitesse) d'une vidéo par le nombre d'images par secondes (en anglais frame rate), exprimé en FPS (Frames per second, en français trames par seconde).

I.2.2.6 les codecs vidéo: [3]

Contraction des mots **CO**mpression / **DEC**ompression, il désigne un algorithme capable de compresser/décompresser et décrypter des informations.

Il s'agit d'applications qui interviennent dans le traitement de la vidéo numérique afin de réduire le volume de mémoire occupé par celle-ci suivant la qualité d'image que l'on veut obtenir. Ces applications sont basées sur des algorithmes et des méthodes de codage adaptées à la vidéo pour la compresser de façon optimale.

On peut en citer quelques-uns : MJPEG, MPEG-1, MPEG-2, MPEG-4, HEVC, H.261, H.262, H.263, H.264 et le tout dernier H.265/HEVC.

I.2.2.7 les Formats vidéo: [4]

De nos jours, les nouvelles technologies font de grands pas, avec elles naissent aussi de nouveaux supports mais aussi de nouveaux formats de vidéos. Selon les constructeurs des différents périphériques les utilisant, les formats peuvent être propriétaire ou libre.

Tout d'abord définissons le terme « format » : Il est utilisé pour désigner un fichier de données aussi bien que son contenu.

Une vidéo est une succession d'images (idéalement entre 24 et 30 images par seconde) à laquelle s'ajoute du son, des titres et des sous-titres, qu'on appelle « métadonnées ».

L'ensemble de ces éléments est ensuite placé et compressé dans un fichier appelé « **conteneur** » identifié par son nom comme AVI, MOV, FLV, … Plus la qualité d'une vidéo est grande, plus le fichier créé est volumineux, c'est pour cela qu'on utilise un CODEC pour la compression et décompression.

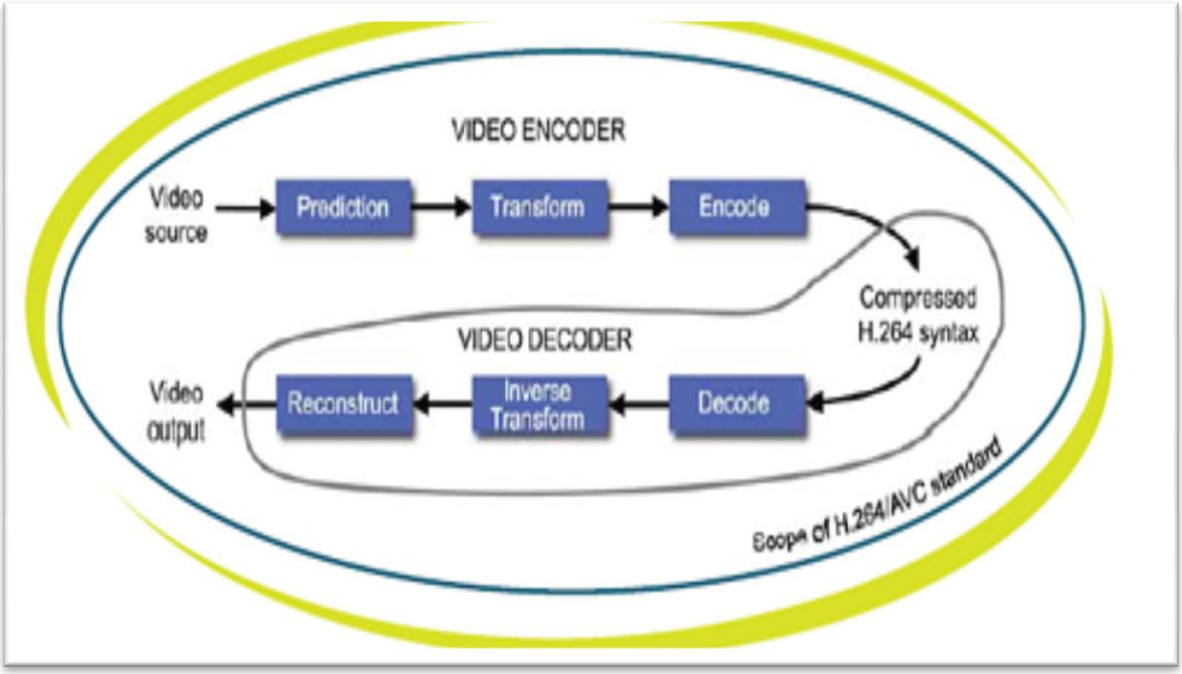


Figure 01 : Etapes principales de compression et décompression d'une vidéo (cas du H.264/AVC)

Il existe différentes catégories de format vidéo :

Les fichiers vidéo utilisent des codecs différents pour encoder et compresser les données. Ils ne sont pas tous des formats de données, par exemple .avi, .mpg, .mov sont des conteneurs, également appelés « encapsuleurs », ils peuvent tout de même indiquer à quel type de format appartient le fichier en question.

Deux catégories de formats vidéo se distinguent réellement :

-Les formats de fichiers dits de « projet » ou « conteneurs », qui sont conçus pour fonctionner à l'intérieur d'un programme d'édition, (ce sont des fichiers non finalisés qui peuvent encore être retouchés, ils sont conçus pour stocker des données audio et vidéo.)

-Les formats vidéo normaux, qui peuvent être créés grâce à des logiciels, à des fins de stockage, de visualisation, de diffusion ou même pour être partagés.

On peut citer quelques formats de vidéo:

Tableau 01: Quelques formats de vidéo

Formats	Extensions	Avantages	Inconvénients
AVI 	.avi	Dans un fichier AVI, chacune des composantes peut être compressée par n'importe quel codec. Il supporte la plupart des formats d'audio et vidéo. De plus, il est supporté par une grande majorité, si ce n'est pas tous, des logiciels de lecture vidéo.	Les fichiers utilisant le AVI restent relativement lourds. Il peut contenir tout type de format audio, vidéo mais pas de texte. Il n'est pas du tout recommandé pour le streaming.
MOV ou QT 	.mov .qt .qtx .qtr	Les fichiers dans ce format sont légers.	Il nécessite l'utilisation du lecteur QuickTime. Le format MOV supporte la haute définition mais il n'offre pas toujours une très bonne qualité d'image.
RM 	.rm .ram .rpm.	Il supporte de nombreux formats mais plus spécialement RealAudio et RealMovie.	Il ne supporte pas le texte, se lit uniquement avec le lecteur RealPlayer ou RealAlternative.et possède donc une mauvaise compression des données.
MKV 	.Mkv .mka .mks .mk3d	capacité d'adaptation avec la majorité des lecteurs. En effet, il est compatible avec des lecteurs matériels, mais aussi avec des lecteurs logiciels	il n'est pas compatible avec tous les logiciels.
YUV 	.Yuv	Il permet d'avoir des vidéos brut sans aucun traitement.	Il nécessite l'utilisation du lecteur YUVviewer.
MP4 	.mp4 ..mp4a .mp4v .m4P.	C'est un format de vidéo utilisé par un nombre croissant de caméras mais c'est aussi un format très recommandé. Il se retrouve aussi sur les Blu-ray et se lit avec la plupart des appareils et des logiciels de lecture vidéo à jour. La vidéo est compressée et prévue pour être lue en streaming. Son système de compression permet d'obtenir des fichiers plus légers en supprimant les images fixes.	Pas d'inconvénients majeurs trouvés
M4v 	.m4v	Les fichiers possédant ce format peuvent être protégés de la copie.	Pas d'inconvénients majeurs trouvés
MPEG2-TS 	.ts, .tsv, .tsa.	Ce format comprend des fonctions de correction d'erreurs.	Pas d'inconvénients majeurs trouvés

I.2.3 Compression vidéo : [5]

Aujourd'hui, La compression de donnée est un outil indispensable pour réduire la taille des fichiers. Lorsque les images, les sons ou les vidéos sont compressés, les données redondantes sont supprimées pour réduire la taille du fichier. Ceci est très utile lors du stockage, de la diffusion et du téléchargement de fichiers.

I.2.3.1 Concepts de compression :

Il y a deux concepts de compression importants qui sont une compression avec pertes et sans pertes :

I.2.3.1.1 Compression avec pertes :

Les techniques de compression avec pertes impliquent une certaine perte d'information, et les données ainsi compressées ne peuvent généralement pas être récupérées ou reconstruites exactement. En contrepartie de l'acceptation de cette distorsion dans la reconstruction, nous pouvons généralement obtenir des rapports de compression beaucoup plus élevés que ce qui est possible avec une compression sans pertes.

I.2.3.1.2 Compression sans pertes:

Les techniques de compression sans pertes, comme leur nom l'indique, ne comportent aucune perte d'information. Si les données ont été compressées sans pertes, les données d'origine peuvent être récupérées exactement à partir des données compressées.

I.2.4 Métriques d'évaluation de la qualité d'une vidéo :

La mesures objective la plus utilisée est :

- Le rapport crête signal sur bruit (Peak Signal to Noise Ratio ″PSNR″).

I.2.4.1 Le rapport crête signal sur bruit, PSNR : [6]

Au lieu de mesurer la distorsion, cette valeur PSNR (Peak Signal to Noise Ratio) mesure la fidélité, puisqu'elle est proportionnelle à la qualité. C'est une fonction du MSE (Mean Square Error ou Erreur quadratique moyenne) ; sa définition et son utilisation proviennent du domaine du traitement de signal:

$$PSNR = 10log_{10}\frac{Imax^2}{MSE} \qquad (2)$$

Pour une image en niveau de gris, Imax désigne la luminance maximale possible. Une valeur de PSNR infinie correspond à une image non dégradée. Et cette valeur décroît en fonction de la dégradation. L'erreur quadratique moyenne est calculée entre les pixels originaux et dégradés:

$$\text{MSE}=\frac{1}{\text{M} \times \text{N}}\sum_{m=1}^{M}\sum_{n=1}^{N}\big(I(m,n)-\hat{I}(m,n)\big)^2 \qquad (3)$$

Où (M × N) est la taille de l'image, et I p et Î p sont respectivement les amplitudes des pixels sur les images originale et dégradée. Une valeur de PSNR inférieure à **30 dB** traduit généralement une image présentant des dégradations perceptibles ; le PSNR est l'évaluation la plus couramment utilisée.

I.2.5 Quelques logiciels de traitement de la vidéo : [7]

Il existe un grand nombre de logiciels qui permettent d'apporter des retouches et/ou des modifications aux vidéos, image et son, dont on peut citer :

- FFmpeg.
- Shotcut .
- VSDC Video Editor.
- Kdenlive.
- Lightworks.

I.3 Conclusion :

Ce chapitre a permis de présenter la vidéo en générale. Des notions de base sur la vidéo ont été présentées et ont été suivies par des détails sur les caractéristiques d'une vidéo numérique. Ensuite une partie de ce chapitre a été consacrée aux notions de compression vidéo, les formats et les codecs. Pour finir, les métriques d'évaluation de la qualité d'une vidéo les plus utilisées on été résumées.

Dans le chapitre suivant on va s'intéresser principalement a l'outil FFmpeg.

Chapitre II
Présentation de l'outil FFmpeg

II.1 Introduction :

Le traitement de la vidéo a pour but l'amélioration la qualité, l'édition, la compression/décompression pour stockage ou transmission, l'extraction d'information … etc. Il s'agit d'une branche dérivée du traitement d'images, qui réutilise beaucoup de ses concepts. Bien qu'il soit courant de « traiter » une vidéo en traitant en réalité chacune de ses images (ou frames) indépendamment, le traitement vidéo consiste à priori à tirer également parti de la relation qui lie chacune des images à ses précédentes et ses suivantes, par évolution de mouvements ou autres modifications graduelles dans le temps. Ce traitement nécessite des outils performants, fiables et aussi rapides.

Dans ce chapitre, nous allons présenter un des ces outils qui est le FFmpeg.

II.2 Présentation de l'outil FFmpeg :

II.2.1 Définition : [8]

Fast Forward MPEG (FFmpeg) est une application informatique gratuite qui permet le traitement des fichiers audio et vidéo. Le traitement est basé sur des lignes de commande et est particulièrement apprécié pour sa rapidité lors du codage et décodage des fichiers audio et vidéo.

L'utilisation de cette application peut être basée sur cmd (l'invite de commandes est un logiciel d'interprétation des commandes , affiche une interface utilisateur en ligne de commande, ou sur l'introduction manuelle des commandes dans l'invite de commande).

Afin de comprendre les principes fondamentaux du logiciel et de se familiariser avec son utilisation sans l'interface graphique, l'utilisateur peut ajuster les commandes à ses préférences.

II.2.2 Installation du FFmpeg : [9]

FFmpeg est une application très puissante qui peut être installée non seulement sous Windows, Linux mais aussi sous d'autres systèmes d'exploitation. Dans ce qui suit, nous allons décrire les méthodes d'installation sous Windows ainsi que sous linux.

II.2.2.1 Installation sous Windows :

Pour l'installation de FFmpeg sous Windows il suffit de suivre les étapes suivantes :

1. Télécharger FFmpeg : Il suffit de consulter le lien suivant http://ffmpeg.zeranoe.com/builds/ et de télécharger la version statique 32 ou 64 bits (selon votre système).
2. Décompresser le fichier: Pour rendre la taille du téléchargement agréable et petite, il est compressé dans un fichier .7z, qui ressemble à un fichier .zip mais plus petit.
3. Décompressez-le dans un dossier facile à trouver, comme directement sur votre lecteur C: \. Il devrait créer un dossier du type ffmpeg-20140530-git-98a6806-win64-static , mais renommez-le simplement en ffmpeg pour simplifier.
4. Ajouter au chemin :Enfin, nous devons ajouter le dossier bin , qui contient le fichier ffmpeg.exe , à notre chemin système pour nous permettre d'exécuter facilement les commandes.
5. Dans le menu **Démarrer**, cliquez avec le bouton droit de la souris sur Ordinateur et choisissez **Propriétés.** Ensuite, sélectionnez **Paramètres système avancés**. Ouvrez **les variables d'environnement** et éditez ensuite la variable **Path.**
6. Le chemin est juste une liste de dossiers contenant des commandes que vous êtes autorisés à utiliser sans taper le chemin complet des fichiers exe.
7. Ajoutez alors C:\ffmpeg\bin à la fin de la ligne, en vous assurant qu'il y a un point-virgule (;) après le dossier précédent:
8. 5: Utilisez-le! C'est tout!

II.2.2.2 Installation Sous linux :

1. Il faut installer FFmpeg via le PPA ou Personal Package Archives (proposer facilement et rapidement les versions récentes de leurs logiciels aux utilisateurs d'Ubuntu.) recommandé dans le blog officiel.

2. Ouvrez un nouveau terminal (CTRL + ALT + T).

3. Puis exécutez les commandes suivantes.

```
$ sudo add-apt-repository ppa: mc3man / trusty-media
$ sudo apt-get update
$ sudo apt-get install ffmpeg
$ ffmpeg –version
```

Lorsque vous exécutez la commande sudo apt update, votre système utilise l'outil APT ou Advanced Packaging Tool qui est un système complet et avancé de gestion de paquets pour effectuer une vérification par rapport au référentiel et stocke les informations relatives au

logiciel et à leur version dans une mémoire cache. Lorsque vous utilisez la commande sudo apt install nom_package, elle utilise ces informations pour extraire ce package à partir de l'URL où le logiciel réel est stocké.

II.2.3 Principe de fonctionnement : [10]

ffmpeg repose sur un principe très simple. Chaque fichier d'entrée sera démultiplexé grâce à la bibliothèque libavformat. Les paquets ainsi encodés sont passés au décodeur qui va produire des frames non compressées. Une fois cette étape terminée, les paquets sont réencodés puis multiplexés pour devenir les fichiers de sortie.

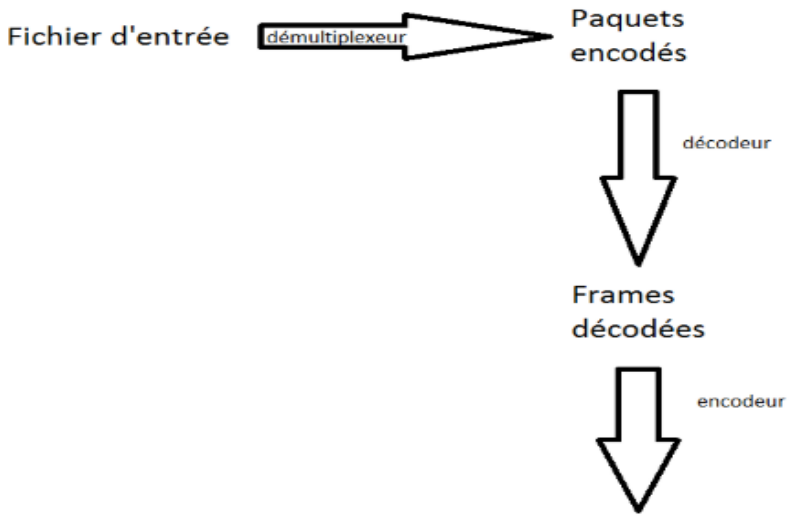


Figure 02 : Principe de fonctionnement du FFmpeg

II.2.4 Les bibliothèques: [11]

II.2.4.1 Définition :

Une bibliothèque logicielle est une collection de routines, qui peuvent être déjà compilées et prêtes à être utilisées par des programmes. Les bibliothèques sont enregistrées dans des fichiers semblables, voire identiques aux fichiers de programmes, sous la forme d'une collection de fichiers de code objet rassemblés accompagnés d'un index permettant de retrouver facilement chaque routine.

II.2.4.2 Bibliothèques utilisées par FFmpeg : [12]

Libavutil : La bibliothèque libavutil est une bibliothèque d'utilitaires facilitant la programmation multimédia portable. Elle contient des fonctions, des générateurs de nombres aléatoires, des structures de données, des fonctions mathématiques supplémentaires, des fonctions de cryptographie et multimédias (telles que des énumérations pour les formats de pixels et d'échantillons).

Libswscale : La bibliothèque libswscale effectue une mise à l'échelle hautement optimisée des opérations ainsi que des opérations de conversion d'espaces colorimétriques et de formats de pixels.

Libswresample : La bibliothèque libswresample effectue des opérations de ré-échantillonnage audio, de rematriçage et de conversion de format d'échantillon hautement optimisées.

Libavcodec : La bibliothèque libavcodec fournit un cadre générique de codage / décodage et contient plusieurs codeurs et décodeurs pour les flux audio, vidéo ainsi que plusieurs filtres de flux de données.

Libavformat : La bibliothèque libavformat fournit un cadre générique pour le multiplexage et le démultiplexage (multiplexage et démultiplexage) des flux audio, vidéo et de sous-titres. Elle englobe plusieurs multiplexeurs et démultiplexeurs pour les formats de conteneur multimédia.

Libavdevice : La bibliothèque libavdevice fournit un cadre générique pour la saisie et le rendu sur de nombreux périphériques d'entrée/sortie multimédia courants, et prend en charge plusieurs périphériques d'entrée et de sortie.

Libavfilter : La bibliothèque libavfilter fournit un cadre de filtrage audio / vidéo générique contenant plusieurs filtres.

II.2.5 Les formats: [13]

Le format est le container qui permet le transport de la vidéo, du son et des sous-titres soit sous forme de fichier (mkv, mov…) soit sous forme de flux (MPEG TS). A l'intérieur d'un container on peut insérer (muxer) ou extraire (demuxer):

- Un ou plusieurs flux vidéo (un film, ou des chaînes de télévision….)
- Un ou plusieurs flux audio (la version originale, la version française, …)
- Un ou plusieurs flux de sous-titres. (français, sourd et malentendant, …)
- Plus des métadonnées (titre, nom de l'artiste par exemple)

On parle de multiplexer les différentes pistes (flux ou stream) dans un format. FFmpeg fournit une liste des formats qu'il supporte: **ffmpeg -formats**

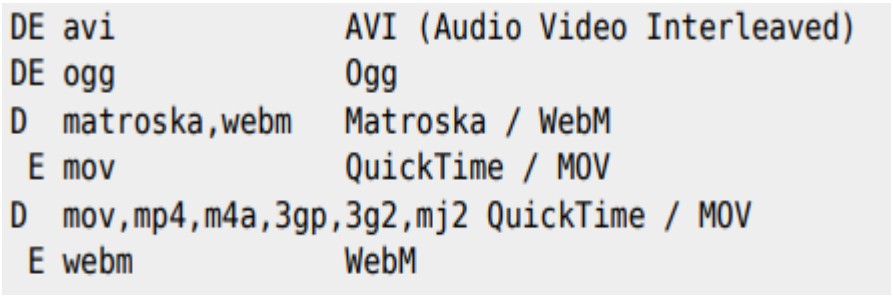


Figure 03 : Résultat de la commande ffmpeg -formats

Le D signifie la capacité à le lire, et E la possibilité d'encapsuler dans le format.

II.2.6 Les codecs : [14]

Un codec est un algorithme qui permet d'encoder la vidéo ou le son afin de l'adapter au protocole de transport (IP, DVB, fichier…) notamment en réduisant le débit (Kbits/s). Selon les codecs, la compression peut s'accompagner d'une perte de qualité dans l'image ou le son plus ou moins importante.

De la même manière que pour les formats, ffmpeg liste les codecs qu'il est capable de gérer: **ffmpeg –codecs .**

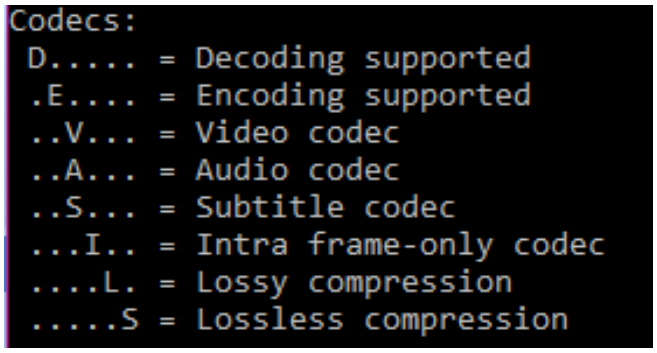


Figure 04 : Principaux codecs du ffmpeg

Puisqu'il y en a beaucoup, on va citer les plus connus :

DEV.L. h261 H.261

DEV.L. h263 H.263 / H.263-1996, H.263+ / H.263-1998 / H.263 version 2

D.V.L. h263i Intel H.263

DEV.L. h263p H.263+ / H.263-1998 / H.263 version 2

DEV.LS h264 H.264 / AVC / MPEG-4 AVC / MPEG-4 part 10 (decoders: h264 h264_qsv h264_cuvid) (encoders: libx264 libx264rgb h264_amf h264_nvenc h264_qsv nvenc nvenc_h264)

DEV.L. hev H.265 / HEVC (High Efficiency Video Coding) (decoders: hevc hevc_qsv hevc_cuvid) (encoders: libx265 nvenc_hevc hevc_amf hevc_nvenc hevc_qsv)

II.2.7 Les filtres : [15]

ffmpeg dispose aussi d'une base importante de filtres qui permettent de modifier le contenu de chaque flux, comme changer la résolution, modifier le volume d'une piste, incruster un logo etc…. ffmpeg liste les filtres qu'il est capable de gérer: **ffmpeg -filters**
Si on résume, ffmpeg permet de multiplexer ou de-multiplexer dans différents formats:

- Des flux vidéos compressés (ou pas),
- Des flux audio compressés (ou pas),
- Des sous-titres dans différents formats.

II.2.8 Principales Commandes FFmpeg : [16]

-L : Afficher la licence.
-h, - ?, -help, --help: Afficher l'aide.
-formats : Pour obtenir une liste de tous les multiplexeurs et démultiplicateurs.
-filters : Pour obtenir une liste de tous les filtres.
-version : Montrer la version.
-demuxers : Afficher les démultiplicateurs disponibles.
-muxers : Afficher les multiplexeurs disponibles.
-dispositifs : Afficher les appareils disponibles.
-codecs : Affiche tous les codecs connus de libavcodec.
-décodeurs : Afficher les décodeurs disponibles.
-codeurs : Afficher tous les encodeurs disponibles.
-bsfs : Afficher les filtres de flux binaires disponibles.
-protocoles : Afficher les protocoles disponibles.
-filtres : Afficher les filtres libavfilter disponibles.
-pix_fmts : Afficher les formats de pixels disponibles.
-sample_fmts : Afficher les formats d'échantillons disponibles.
-layouts : Afficher les noms de canaux et les dispositions de canaux standards.
-couleurs : Afficher les noms de couleurs reconnues
Ffplay : visualisation de la vidéo.
Ffprobe : Avoir des informations sur la vidéo.

II.2.9 Liste des traitements dont le FFmpeg est capable : [17]

- Obtenir des infos sur un fichier vidéo **: ffmpeg -i video.avi.**
- Transformer une série d'images en vidéo **: ffmpeg -f image2 -i image%d.jpg video.mpg**
- Extraire une série images d'une vidéo **:ffmpeg -i video.mpg image%d.jpg**
- Extraire le son d'une vidéo et l'enregistrer en mp3 **: ffmpeg -i video_origine.avi son_final.mp3**
- Convertir un fichier avi en mpeg **:ffmpeg -i video_origine.avi video_finale.mpg**
- Convertir un mpeg en avi **:ffmpeg -i video_origine.mpg video_finale.avi**
- Associer une vidéo et un son pour créer une vidéo sonorisée **:ffmpeg -i son.wav -i video_origine.avi video_finale.mpg**
- **Mixer un son et une vidéo :ffmpeg -i son.wav -i video_origine.avi video_finale.mpg**
- Convertir une vidéo mp4 en avi en conservant la même qualité **: ffmpeg -i video.mp4 -qscale 0 video.avi**
- Supprimer le son d'une vidéo **: ffmpeg -i video.mp4 -c copy -an video-sansson.mp4**
- Couper une vidéo **: ffmpeg -i sequenceacouper.avi -ss 00:39:45.00 -t 00:00:30.00 -c:v copy -c:a copy nouvellesequence.avi**
- Changer de Format **: ffmpeg -i input.mkv -c copy output.mov**

Tableau 02 : Signification des commandes

Commandes	Actions
-i	spécifie le fichier d'entrée
-c copy	copie à l'identique la totalité des flux
-c :v copy	copie à l'identique les pistes vidéos
-c :a copy	copie à l'identique les pistes audio
-c :s copy	copie à l'identique les pistes sous-titres

- **Encoder la vidéo :**

Table 03 : Options d'encodage de la vidéo

option	explication de l'option
-r	définit le nombre d'images par seconde
-s	configuration de la taille du cadre d'affichage
-aspect	configuration du format d'affichage
-vcodec ou -c:v	décision du choix du codec
-pass	nombre de passage à l'encodage
-crf	permet de définir un niveau de qualité entre 0 et 51 (petit nombre = meilleure qualité mais plus de temps de calcul) (défaut 23)

- **Encoder le son :**

Table 04 : Options d'encodage de l'audio

Option	explication de l'option
-acodec ou -c:a	détermine le choix du codec
-ar	configuration de la fréquence d'échantillonnage (44100 Hz)
-ab	configuration du débit binaire par défaut 64 kbps
-ac	configure le nombre de canaux (mono-stéréo)

II.2.10 CRF (Constant Rate Factor) :

Il faut utiliser le CRF pour définir une qualité d'image constante. L'échelle crf est logarithmique entre 0 et 51. Une différence de 6 points double ou divise par 2 environ la taille du fichier final. Un petit nombre égal une meilleure qualité mais plus de temps de calcul, la valeur par défaut est souvent 23. Le choix du CRF dépend du type d'image à encoder, de la résolution de l'image, de la qualité souhaitée ou encore de la taille du fichier désirée.

II.2.11 QP (quantization parameter) :[18]

La quantification dans un codeur est contrôlée par un paramètre de quantification, QP, compris entre 0 et 51. Lorsque le QP est définie directement, il reste constant tout au long du codage et chaque image est compressée en fonction de la valeur définie.

II.3 Conclusion :

Dans ce chapitre, le principe de fonctionnement du ffmpeg, son installation et son utilisation ont été présentés. Néanmoins, la découverte de toute la puissance de cet outil se fera en l'utilisant de manière plus approfondie et fréquente.

Dans le chapitre suivant, on s'intéressera principalement à l'utilisation du FFmpeg dans traitement de la vidéo.

Chapitre III Etude des performances de l'outil FFmpeg dans le traitement de la vidéo

III.1 Introduction:

Dans les deux précédents chapitres, la vidéo et l'outil FFmpeg ont été présentés. Mais pour mener à bien ce travail, des simulations, des tests ont été menés afin de démystifier l'FFmpeg et évaluer ses performances. Les simulations portent essentiellement sur des traitements de vidéos.

III.2 Problématique:

Le traitement de la vidéo en utilisant les codecs standards sont très lourds et prennent énormément de temps. C'est pour cela que l'outil FFmpeg a été utilisé dans ce projet afin de découvrir ses capacités et ses performances.

III.3 Méthodologie :

Ce travail a été entamé par le téléchargement de vidéos en format brut (yuv), puis les points cités ci-dessous ont été étudiés.

1. Investigation des filtres du FFmpeg : Scale et rotate. Ces filtres permettent de modifier la résolution des vidéos et de les faire pivoter selon des angles choisis respectivement.
2. Conversion de vidéos brutes (format yuv) en d'autres formats : MP4, AVI, MKV.
3. Compression des vidéos yuv.
4. Evaluation de la qualité des vidéos compressées en calculant le PSNR.
5. Faire une comparaison entre les codecs x264 et x265.
6. Comparaison des résultats de compression entre la plateforme HM (HEVC model) et FFmpeg en termes de bitrate et qualité.
7. Ajout de l'audio (son) à une vidéo.
8. Extraction de sous-séquences de vidéo.
9. Conversion d'une vidéo en images.
10. Création de vidéos à partir d'ensembles d'images.

Après chaque étape, les résultats ont été analysés et des conclusions tirées.

III.4 Simulation et analyses des résultats :

III.4.1 Caractéristiques des vidéos utilisées:

Le besoin d'utilisation des vidéos avec une extension yuv est d'avoir un format natif : il s'agit des vidéos brutes qui n'ont subi aucun traitement. Les caractéristiques des vidéos utilisées sont récapitulées dans le tableau ci-dessous :

Tableau 05 : Caractéristiques des vidéos utilisées

Nom de la vidéo	Taille (Mo)	Résolution (pixel)	FPS
BlowingBubbles	71.5	416*240	50
RaceHorses	171	832*480	30
BQMall	343	832*480	60
Kimono1	711	1920*1080	24
ParkScene	711 Mo	1920*1080	24

III.4.2 Utilisation des filtres du FFmpeg:

III.4.2.1 Modification de la résolution(filtre scale) :

Pour un fichier d*'entrée* YUV, il faut spécifier la fréquence d'images et la taille, car le ffmpeg ne dispose pas de ces informations.

Syntaxe:

ffmpeg -s:v (résolution initiale de la vidéo) -r 24 –i inputvideo.yuv -vf **scale**=(résolution désirée) -c:v rawvideo -pix_fmt yuv420p outputvideo.yuv

Tableau 06 : Résultats de changement de la résolution

Séquence vidéo	Taille Initiale (Mo)	Résolution initiale	Résolution souhaitée	Taille de la nouvelle vidéo (Mo)
BlowingBubble	71.7 Mo	416*240	300*200	43
			640*350	160
RaceHorses	177	832*480	640*350	96,1
			960*540	222
BQMall	343	832*480	640*350	192
			960*540	445
Kimono1	711 Mo	1920*1080	720*480	118
			768*576	151
			960*540	177
ParkScene	711	1920*1080	720*480	118
			768*576	151
			960*540	177

Analyse :

Les résultats montrent bien que la taille de la vidéo diminue ou augmente en rabaissant et en augmentant la résolution respectivement.

Lorsqu' on redimensionne une image, on modifie la quantité de données qu'elle contient.

Le redimensionnement change le nombre total de pixels contenus dans l'image, lequel est indiqué par Largeur et Hauteur en pixels. Lorsqu'on augmente le nombre de pixels (sur-échantillonnage), des données seront rajoutées à l'image. Lorsqu'on réduit le nombre de pixels (sous-échantillonnage), des données seront supprimées. Chaque fois que des données sont supprimées ou ajoutées dans l'image, la qualité de l'image se détériore dans une certaine mesure. En général, il vaut mieux supprimer des données d'une image qu'en rajouter.

Mais, visualisées à l'aide de VLC player , les vidéos créées apparaissent avec la même qualité que celle de la vidéo originale.

III.4.2.2 Pivoter une vidéo de 90° :

Syntaxe :

ffmpeg -i BlowingBubbles1.mp4 -vf "transpose=1" ret.mp4

Valeurs possibles de transpose:

0 = 90º : sens contraire au aiguilles d'une montre et vertical Flip (par défaut)

1 = 90º : sens des aiguilles d'une montre

2 = 90º : sens contraire au aiguilles d'une montre

3 = 90º : Flip dans le sens horaire et vertical

Figure 05 : Vidéo originale BlowingBubbles

Figure 06 : Rotation dans le sens des aiguilles d'une montre

Figure 07 : Rotation contre le sens des aiguilles d'une montre et Vertical Flip

Figure 08 : Rotation contre le sens des aiguilles d'une montre

Figure 09: Flip dans le sens horaire et vertical

III.4.3 Conversion de formats des vidéos:

L'objectif est de créer, à partir d'une vidéo originale ayant un format donné, d'autres vidéos ayant d'autres formats afin de pouvoir, par exemple, les visualiser en utilisant un lecteur donné.

Pour le format AVI : Dans un fichier AVI, chacune des composantes peut être compressée par n'importe quel codec. Il supporte la plupart des formats d'audio et vidéo. De plus, il est supporté par une grande majorité, si ce n'est pas tous, des logiciels de lecture vidéo.

Pour le format MP4 : C'est un format de vidéo utilisé c'est aussi un format très recommandé. Il se lit avec la plupart des appareils et des logiciels de lecture vidéo à jour. La vidéo est compressée et prévue pour être lue en streaming. Son système de compression permet d'obtenir des fichiers plus légers en supprimant les images fixes

Pour le format MKV : capacité d'adaptation avec la majorité des lecteurs.

En effet, il est compatible avec des lecteurs matériels, mais aussi avec des lecteurs logiciels

III.4.3.1 Conversion YUV en formats : MP4 , AVI, MKV en utilisant libx264 :

- **Conversion YUV en format MP4 :**
- **Syntaxe codage :**

ffmpeg -f rawvideo -vcodec rawvideo -s résolution de la vidéo -r fps -pix_fmt yuv420p -i inputfile.yuv -c:v libx264 –qp 0 output.mp4 -psnr

- Avec -f rawvideo pour la définition du format d'entrée sur un conteneur vidéo brut.
- Avec -vcodec rawvideo pour la définition du fichier d'entrée comme non compressé.
- Avec -i inputfile.yuv pour la définition du fichier d'entrée
- Avec -c:v pour la définition de l'encodeur pour encoder la vidéo à libx264.
- Ensuite, output.mp4 le nouveau conteneur dans lequel les données seront stockées.
- rawvideo -pix_fmt yuv420p : sélectionner le format de pixel pour une compatibilité maximale
- -vcodec libx264 utilise le codec h264

- **Syntaxe décodage :**

ffmpeg -i input.mp4 -f rawvideo -vcodec rawvideo -pix_fmt yuv420p -s résolution –r FPS rawvideo.yuv

- **Conversion MP4 en formats YUV :**

Dans les deux cas, une fois le fichier .MP4 ou .yuv a été obtenu, une vérification de la qualité s'impose même si dans le premier cas, on a essayé de compresser le fichier.yuv sans pertes (qp=0). Pour cela, le PSNR à été utilisé.

Résultats:

Tableau 07 : Résultats de la conversion YUV/MP4 (libx264)

Vidéo	Taille .yuv (MO)	Taille .MP4 (Mo)	Taille .yuv' (Mo)	PSNR (dB) (yuv et yuv')
BlowingBubbles	71.5	28.3	71.5	Inf
RaceHorses	171	74,9	171	Inf
BQmall	240	135	240	Inf
Kimono1	711	274	711	Inf
ParkScene	711	284	711	Inf

Analyse :

Une compression s'opère automatiquement l'or d'une conversion en format mp4.

Les résultats ainsi que la visualisation en utilisant le logiciel de lecture VLC montrent aussi un match entre les vidéos décodé et les vidéos originales.

L'ors du décodage on obtient la même vidéo et la preuve est que le PSNR tend vers l'infinie.

- **Conversion de vidéos YUV en format AVI et MKV :**

  La raison pour laquelle ces 2 conversions sont présentées en même temps est que, après un grand nombre de tests, on a déduit que les résultats sont identiques.

- **Syntaxe :**

Ffmpeg –s résolution -i inputfile.yuv -c:v rawvideo -pix_fmt yuv420p output.avi/mkv

Tableau 08 : Résultats de la conversion YUV/AVI.MKV

Vidéos	Taille (MO)	YUV to mkv/avi (Taille Mo)	mkv/avi to YUV' (Taille Mo)	PSNR (yuv et yuv')
BlowingBubbles	71.5	71.5	71.5	inf
RaceHorses	171	171	171	inf
BQmall	240	240	240	inf
Kimono1	711	711	711	inf
ParkScene	711	711	711	inf

yuv' : simple notation pour dire qu'il s'agit de la reconstruction de la vidéo en format yuv

Analyse :

La conversion en format MKV et AVI ne montrent aucun changement par rapport aux vidéos en formats YUV.

III.4.3.2 Conversion YUV en formats MP4 , MKV, AVI en utilisant libx265:

Syntaxe :

- **Codage ;**

Ffmpeg –s resolution -i input -c:v libx265 output.mp4

- **Décodage ;**

ffmpeg -i output.mp4 -s résolution essaye1.yuv

Tableau 09 : Résultats de la conversion YUV/MP4 (libx265)

Vidéo	Taille .yuv (MO)	Taille .MP4	Taille .yuv' (Mo)	PSNR (dB) (yuv et yuv')
BlowingBubbles	71.5	857 Ko	71.5	Inf
RaceHorses	171	2,27 Mo	171	Inf
BQmall	240	12,7 Mo	240	Inf
Kimono1	711	2,84 Mo	711	Inf
ParkScene	711	3,75 Mo	711	Inf

Tableau 10 : Résultats de la conversion YUV/MKV (libx265)

Vidéo	Taille .yuv (MO)	Taille .MKV	Taille .yuv' (Mo)	PSNR (dB) (yuv et yuv')
BlowingBubbles	71.5	854 Ko	71.5	Inf
RaceHorses	171	2,27 Mo	171	Inf
BQmall	240	2,23 Mo	171	Inf
Kimono1	711	2,84 Mo	711	Inf
ParkScene	711	3,75 Mo	711	Inf

Tableau 11 : Résultats de la conversion YUV/AVI (libx265)

Vidéo	Taille .yuv (MO)	Taille .AVI	Taille .yuv' (Mo)	PSNR (dB) (yuv et yuv')
BlowingBubbles	71.5	872 Ko	71.5	Inf
RaceHorses	171	2,28 Mo	171	Inf
BQmall	240	2,25 Mo	171	Inf
Kimono1	711	2,85 Mo	711	Inf
ParkScene	711	3,75 Mo	711	Inf

Analyse :

On remarque aussi une compression lors de la conversion mais dans l'utilisation du codec x265 résulte en une compression plus importante.

En termes de qualité de la vidéo, on a pu remarquer qu'il n'y avait aucune dégradation.

Un point très important est que l'opération se fait d'une façon très rapide, la seul chose qui va différer une vidéo d'une autre c'est sa taille .Quand la vidéo aura une taille grande bien

évidement elle va prendre plus de temps qu'une vidéo de petite taille, mais la durée du traitement reste courte (quelque secondes).

III.4.5 Préréglages :

Les différents préréglages de la commande 'ffmpeg -f rawvideo -vcodec rawvideo -r FPS -pix_fmt yuv420p –s Résolution -i input.yuv -c:v libx264 **-preset medium** -qp 22 output.yuv' on était utiliser avec la vidéo « BlowingBubbles » et les résultats se résument dans le tableau suivant :

Tableau 12 : Tableau récapitulatif des différents préréglages

Types	Temps de compression (S)	Qp	Taille du fichier
Very slow	6016.5	4.85	27.8 MO
Placebo	342.39	32.95	1.06 MO
Ultrafast	2.44	34.73	712 KO
Super fast	3.17	33.69	809 KO
Very fast	4.54	33.94	812 KO
Faster	4.29	33.95	813 KO
Medium	7.78	33.06	985 KO
Slow	17.52	33.14	985 KO
Slower	107.51	32.99	0.99 MO

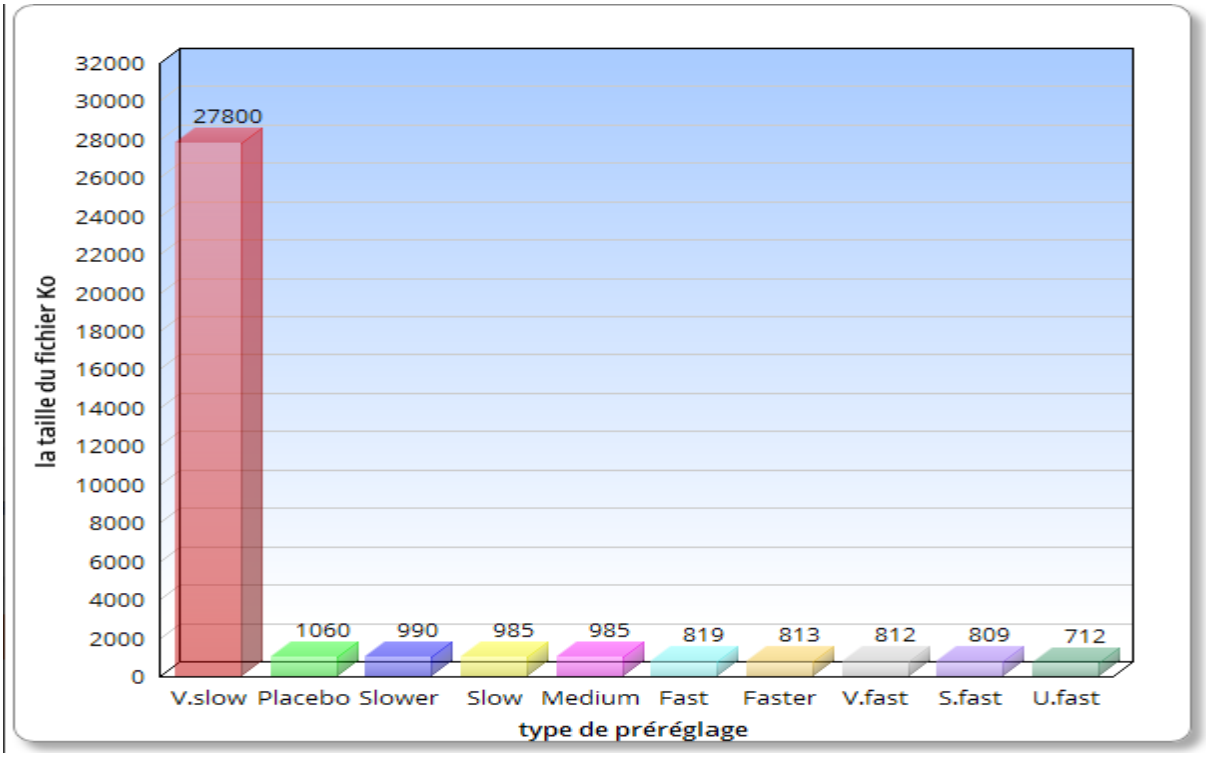


Figure 10 : Représentation de la taille du fichier en fonction du type de préréglage pour la vidéo BlowingBubbles

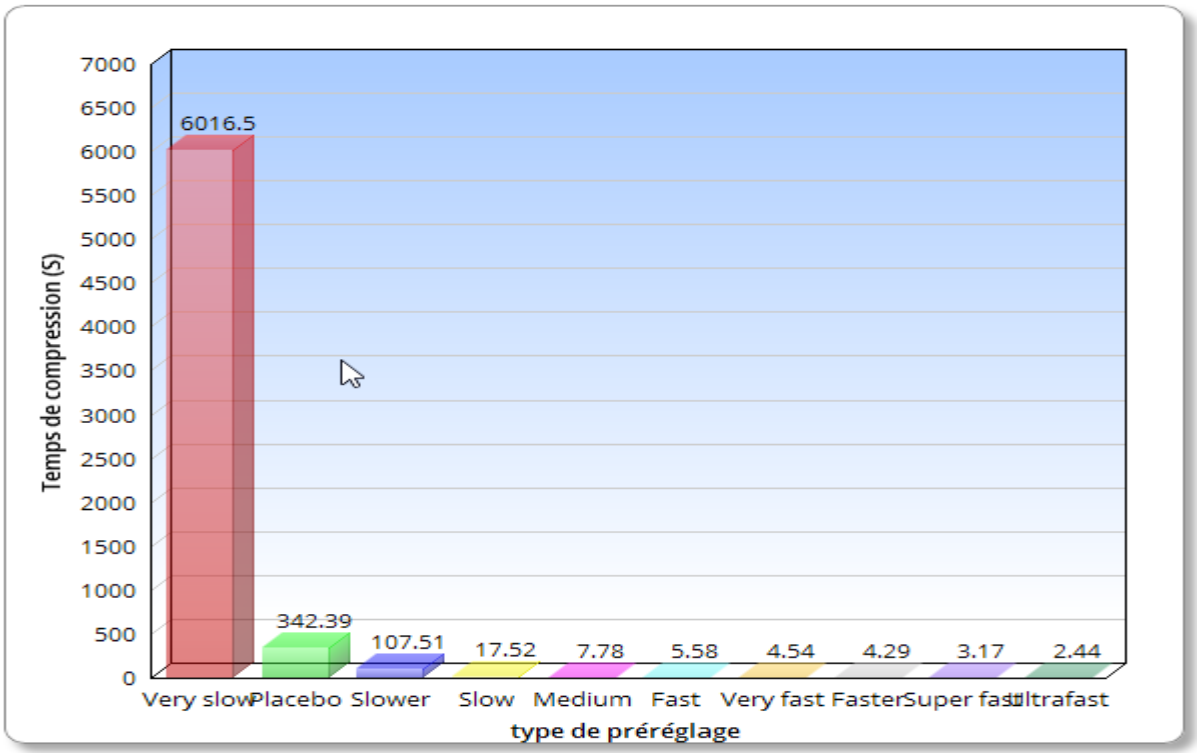


Figure 11 : Représentation du temps de traitement du fichier en fonction du type de préréglage pour la vidéo BlowingBubbles

Analyse :

Les « presets » gérent le temps nécessaire à l'encodage :
On peut conclure du tableau et des deux graphes ainsi qu'a la visualisation des vidéos que :

- ultrafast : c'est le codage le plus rapide mais qui donne la qualité d'image la moins bonne.
- veryslow : permet d'obtenir la meilleure qualité au détriment d'un temps de codage élevé.
- placebo : le plus lent et pas la meilleure qualité. C'est un preset d'aucune utilité.

Le préréglage medium est celui qui est utilisé par défaut.

III.4.6 Compression de vidéo en utilisant la bibliothèque libx264 :

Syntaxe :

ffmpeg -f rawvideo -vcodec rawvideo -s résolution -r fps -pix_fmt yuv420p -i input.yuv -c:v libx264 -preset medium -qp output.264 -psnr

Le choix de l'extension de la vidéo de sortie reste libre. Dans ce cas, on a opté pour une sortie en .264 afin d'avoir des vidéos en bitstream.
Une variation de Qp (paramètre de quantification) permet d'obtenir des vidéos compressées à différents taux et ainsi une conclusion peut être tirée sur l'influence de ce taux de compression sur la qualité de la vidéo.

Tableau 13 : Résultats de la compression en utilisant le lib264 et différents Qp (libx264)

YUV	QP=22			QP=27			QP=32			QP=37		
BlowinBubbles (taille 71.7 Mo)	Bitrate kb/s	Taille	PSNR dB	Bitrate kb/s	Taille	PSNR dB	Bitrate kb/s	Taille	PSNR dB	Bitrate kb/s	Taille	PSNR dB
	2003	2,38	41.17	1037	1,23 Mo	37.37	523.8	639 Ko	33.79	267.7	326 Ko	30.71
RaceHorses (Taille 171 Mo)	17091.6	10,1	40.92	8117.1	4,80 Mo	37.71	3891	2,30 Mo	34.74	1985.7	1,17 Mo	32.26
BQMall (343 Mo)	6559.1	7,80	41.23	3255.5	3,87 Mo	38.71	1746.2	2,07 Mo	36	979.1	1,16 Mo	33.29
Kimono1 (taille 771 MO)	9489.4	11,2	42.30	4665.4	5,51 Mo	40.70	2542.4	3,00 Mo	38.75	1433	1,69 Mo	36.56
ParkScene (711 Mo)	12220.5	14,4	40.82	5405.9	6,38 Mo	38.29	2679.7	3,16 Mo	35.77	1417.7	1,67 Mo	33.39

Analyse :

La première remarque qu'on peut tirer sur ces résultats c'est que le bitrate et par conséquent la taille du fichier diminue aussi en augmentant le taux de compression.

Logiquement la qualité de la vidéo se dégrade à chaque fois qu'on compresse plus. Ce jugement est valable aussi bien subjectivement qu'objectivement à travers la mesure du PSNR qui diminue.

III.4.7 Compression de vidéos en utilisant la bibliothèque libx265 :

Syntaxe :

Ffmpeg –s résolution -i input .yuv -c:v libx265 -crf 17 output.265 –psnr

Tableau 14 : Résultats de la compression en utilisant le lib265 avec différents Qp.

YUV	CRF=17			CRF=22			CRF=27			CRF=32		
BlowinBubbles (taille 71.7 Mo)	**Bitrate kb/s**	**Taille**	**PSNR dB**	**Bitrat kb/s**	**Taille**	**PSNR dB**	**Bitrate kb/s**	**Taille**	**PSNR dB**	**Bitrate kb/s**	**Taille**	**PSNR dB**
	718.2	**1,71 Mo**	**46.30**	**529.8**	**1,26 Mo**	**43.48**	**385.5**	**941 Ko**	**39.91**	**197.4**	**482 Ko**	**34.48**
RaceHorses (Taille 171 Mo)	**7953.5**	**11,3 Mo**	**41.43**	**3850.8**	**5,47 Mo**	**38.59**	**1849.9**	**2,62 Mo**	**35.98**	**891.1**	**1,26 Mo**	**33.38**
BQMall (343 Mo)	**4128.9**	**11,7 Mo**	**42.63**	**1795.5**	**5,12 Mo**	**40.56**	**892.1**	**2,54 Mo**	**38.22**	**463.7**	**1,32 Mo**	**35.70**
Kimono1 (taille 771 MO)	**16167.4**	**18,3 Mo**	**43.40**	**5889.4**	**6,68 Mo**	**41.92**	**2862.9**	**3,24 Mo**	**40.25**	**1481.2**	**1,68 Mo**	**38.28**
ParkScene (711 Mo)	**21963.5**	**24,9 Mo**	**42.55**	**8750.8**	**9,93 Mo**	**40.42**	**3868.5**	**4,38 Mo**	**38.15**	**1771.9**	**2,01 Mo**	**35.82**

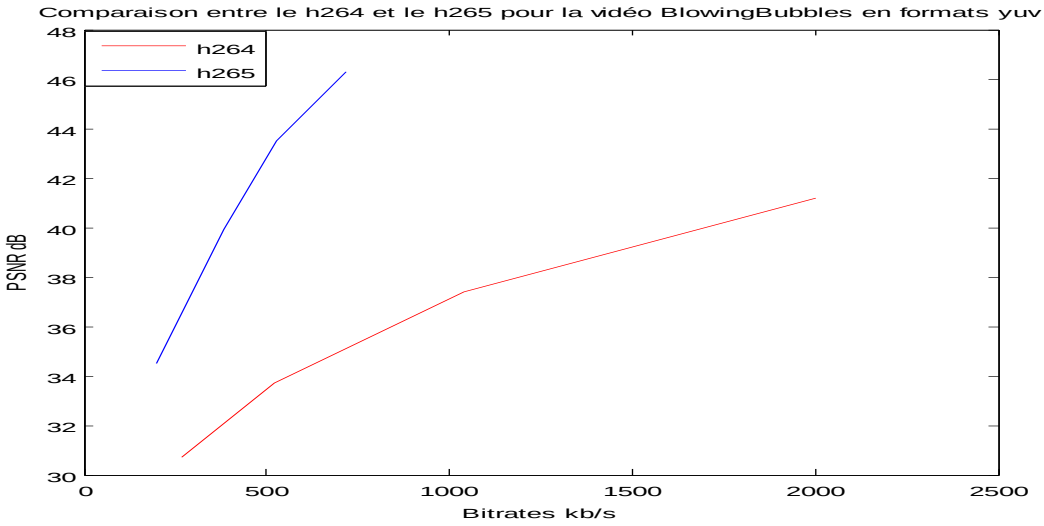
Analyse :

Les simulations précédentes menées avec la libx264 ont été refaites mais en utilisant, cette fois ci, le facteur CRF (Constant Rate Frame) plutôt que le QP. Une première conclusion tirée empiriquement est que:

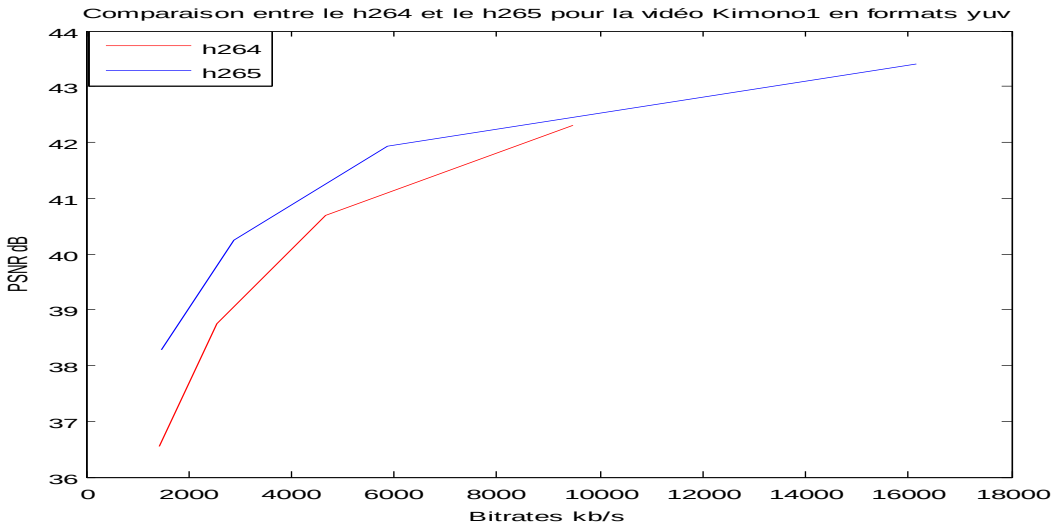
$$QP = CRF + 5 \qquad (3)$$

On peut aussi remarquer que les bitrates ainsi que les tailles ont vraiment diminué par rapport au libx264 et le PSNR confirme que la qualité est bonne.

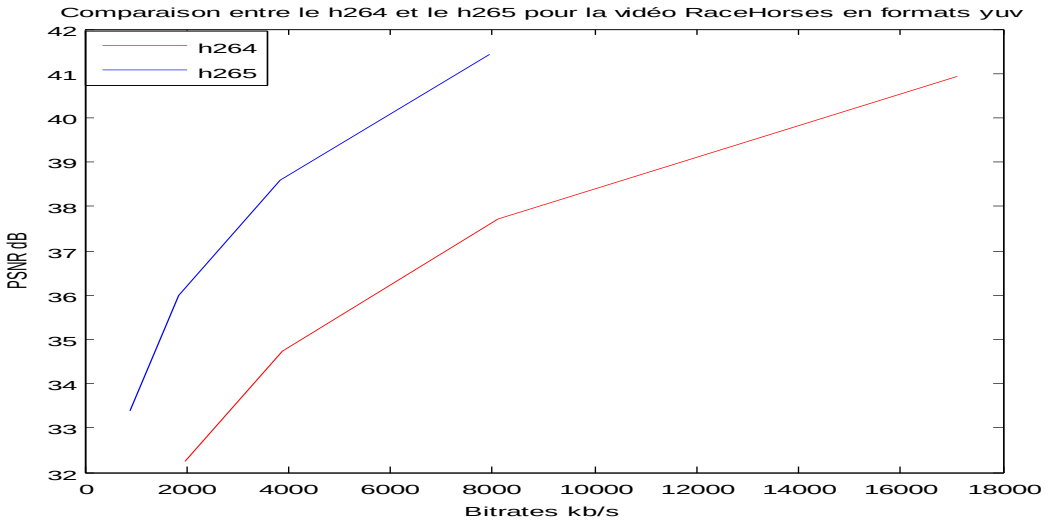
Pour tirer tout cela au clair et pour pouvoir tirer une conclusion, on a fait une comparaison entre les performances en termes de bitrate et qualité de la libx264 et la libx265 implémentant respectivement les deux normes H.264 et le H.265. Les résultats sont exposés dans les figures suivantes :


Comparaison entre le h264 et le h265 pour la vidéo BlowingBubbles en formats yuv
h264
h265
PSNR dB
Bitrates kb/s
48
46
44
42
40
38
36
34
32
30
0
500
1000
1500
2000
2500

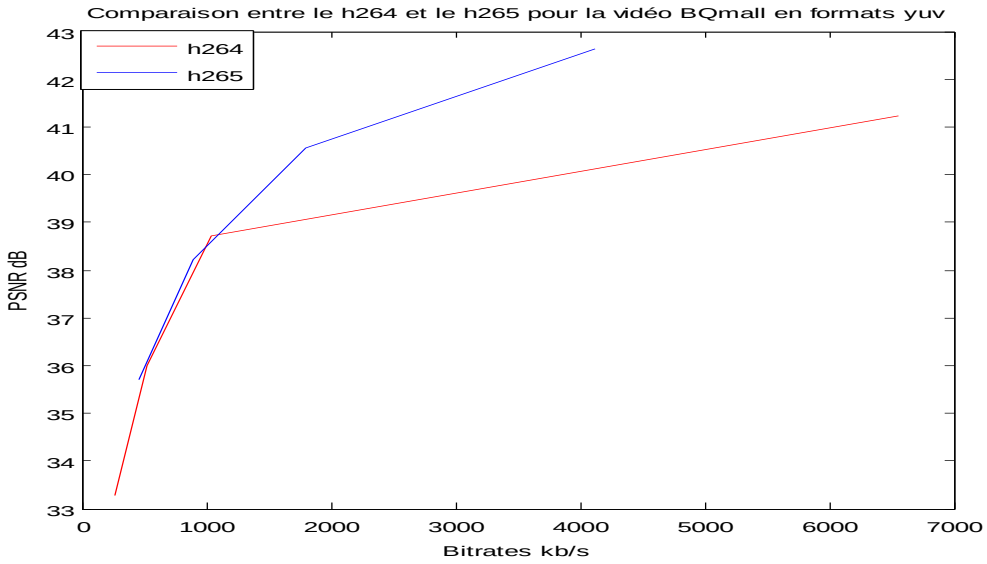
(a)


Comparaison entre le h264 et le h265 pour la vidéo Kimono1 en formats yuv
h264
h265
PSNR dB
Bitrates kb/s
44
43
42
41
40
39
38
37
36
0
2000
4000
6000
8000
10000
12000
14000
16000
18000

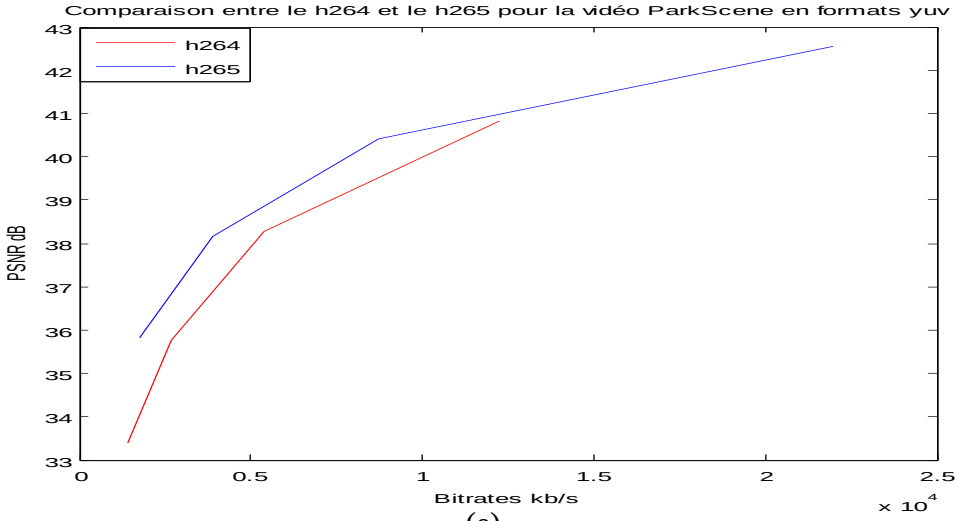
(b)


Comparaison entre le h264 et le h265 pour la vidéo RaceHorses en formats yuv
h264
h265
PSNR dB
Bitrates kb/s
42
41
40
39
38
37
36
35
34
33
32
0
2000
4000
6000
8000
10000
12000
14000
16000
18000

(c)



(d)



(e)

Figure 12 : Résultats de comparaison entre les normes H264 et H265

(a) BlowingBubbles
(b) Kimono1
(c) RaceHorses
(d) BQmall
(e) ParkScene

Ces résultats confirment bien que le bitrate généré par le codec x264 est à peu prés égal au double du bitrate du codec x265.

Analyse :

La différence est importante pour la majorité des vidéos comme on a pu le remarqué dans les figures ci-dessus.

Pour conclure cette partie, on peut dire que le codec x265 est nettement meilleur que le codec x264 en termes de bitrate , de taille de fichier résultant et surtout en termes de qualité.

III.4.8 Résultats de compression de vidéos en utilisant le HM :

Les résultats de compression de vidéos en utilisant le codec libx265 de FFmpeg et ceux obtenus avec la plateforme HM (qui implémente la norme H.265/HEVC ont été comparés. A noter que les résultats donnés par la HM ont été fournis par une doctorante.

Pour cette confrontation, les résultats du FFmpeg ont été comparés à ceux de la plateforme HM dans le cas d'une configuration random access. La raison est que, d'après l'analyse détaillée du traitement du FFmpeg, il a été noté qu'il code un certain nombre de frames en I et tous les autres en P ou B. En plus, le nombre de frames codés I est inférieur à celui codé P ou B. Ceci est donc beaucoup plus proche d'une configuration random access que des configurations All Intra ou lowDelay P/B.

Tableau 15 : Résultats de la compression en utilisant le HM avec différents Qp pour la vidéo BlowingBubbles.

QP	22	27	32	37
Bitrate (kb/s)	1649.38	754.75	352.72	163.38
PSNR (dB)	39.43	36.27	33.46	31.03

Tableau 16 : Résultats de la compression en utilisant le HM avec différents Qp pour la vidéo RaceHorses.

QP	22	27	32	37
Bitrate (kb/s)	4793.23	2027.26	945.83	463.02
PSNR (dB)	40.06	37.19	34.67	32.48-

Tableau 17 : Résultats de la compression en utilisant le HM avec différents Qp pour la vidéo BQmall

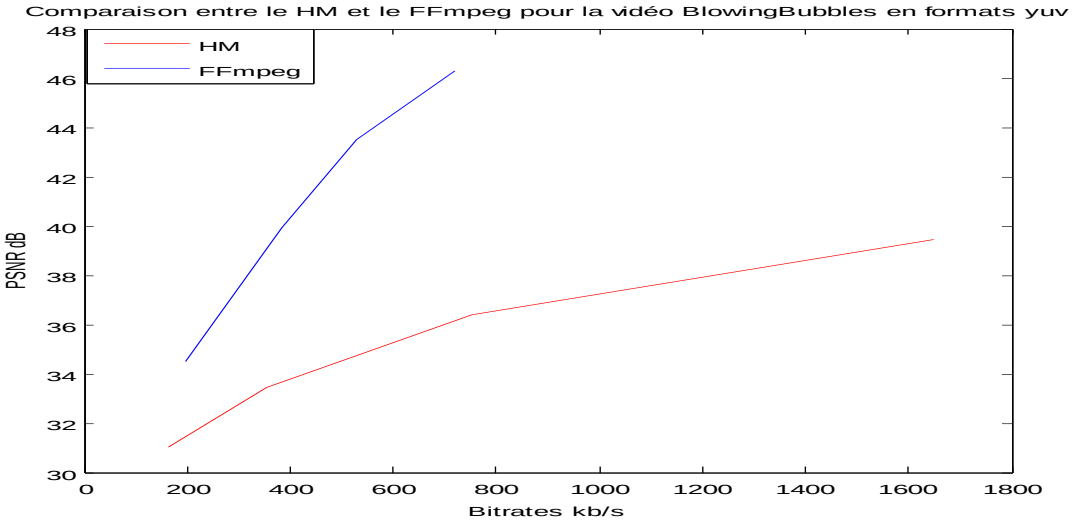
QP	22	27	32	37
Bitrate (kb/s)	3646.94	2027.26	945.83	463.02
PSNR (dB)	41.55	39.18	36.74	34.46

Tableau 18 : Résultats de la compression en utilisant le HM avec différents Qp pour la vidéo Kimono1

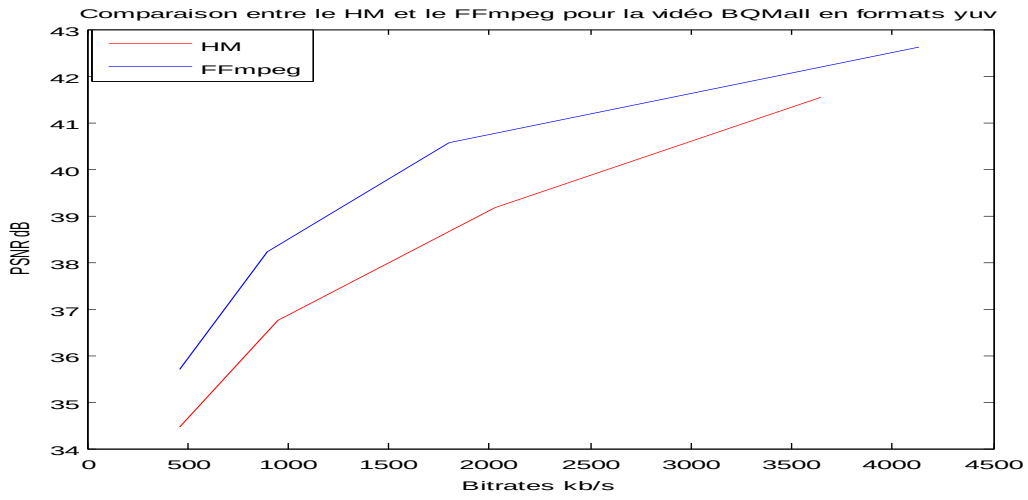
QP	22	27	32	37
Bitrate (kb/s)	4782.25	2185.16	1068.65	542.68
PSNR (dB)	42.53	40.75	38.79	36.93

Tableau 19 : Résultats de la compression en utilisant le HM avec différents Qp pour la vidéoParkScene

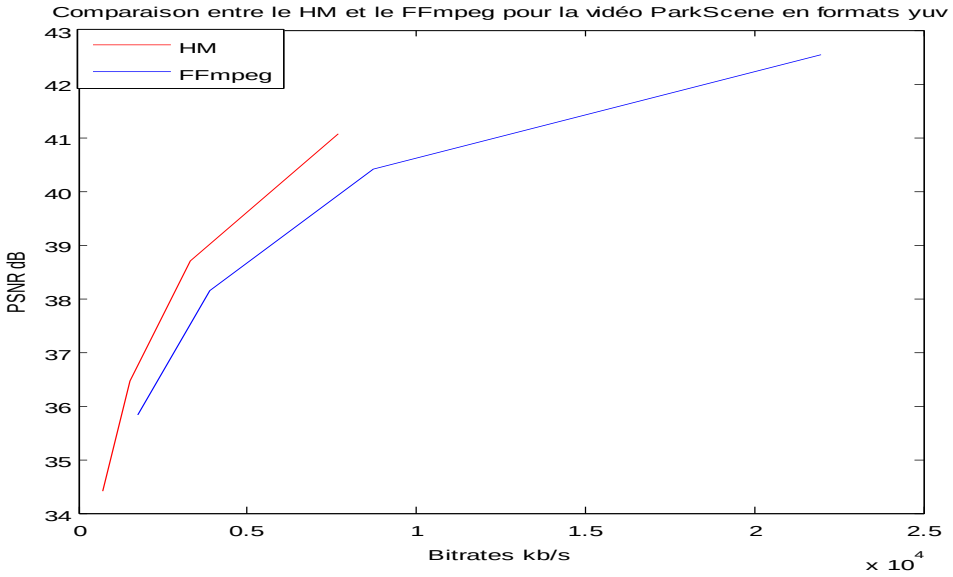
QP	22	27	32	37
Bitrate (kb/s)	7671.83	3333.06	1536.87	716.93
PSNR (dB)	41.07	38.70	36.45	34.42



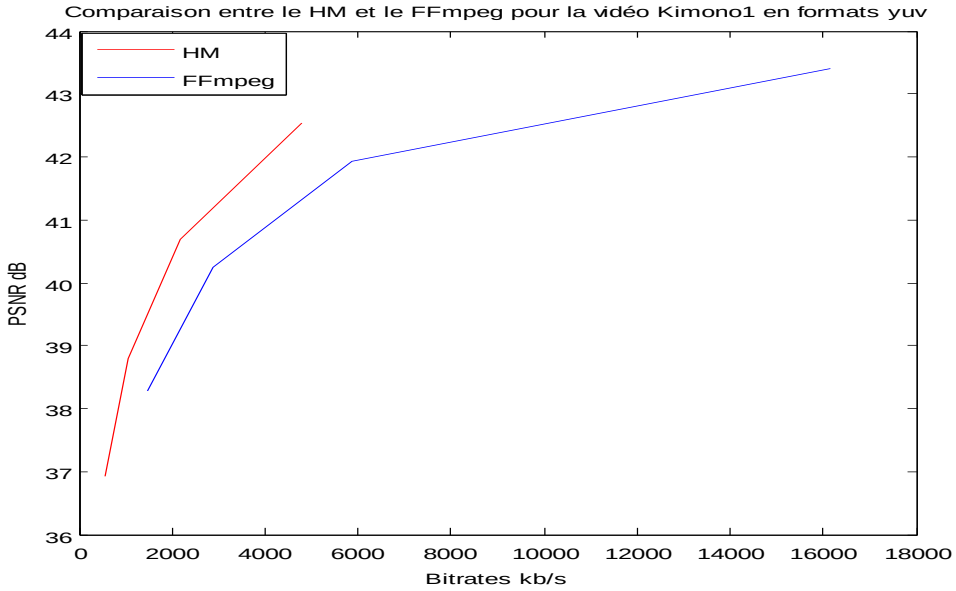
(a)



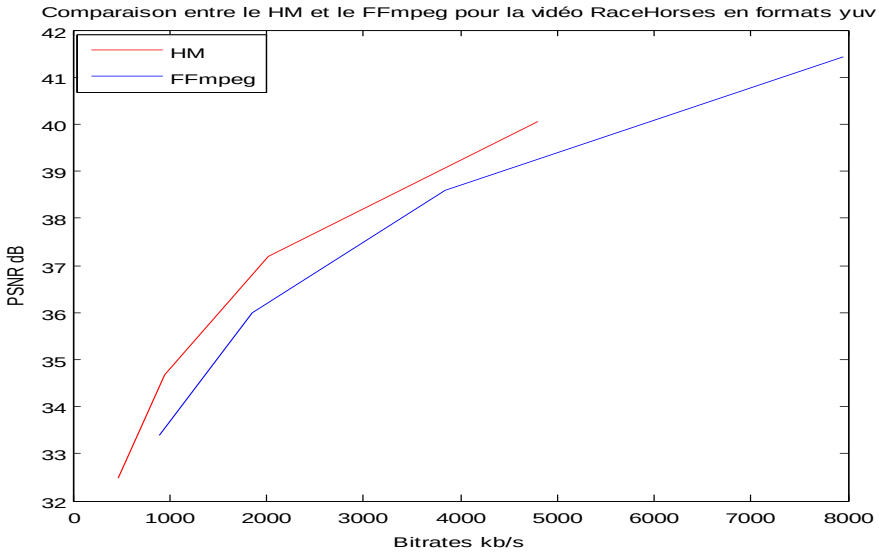
(b)


Comparaison entre le HM et le FFmpeg pour la vidéo ParkScene en formats yuv
HM
FFmpeg
PSNR dB
Bitrates kb/s
x 10^4
43
42
41
40
39
38
37
36
35
34
0
0.5
1
1.5
2
2.5

(c)


Comparaison entre le HM et le FFmpeg pour la vidéo Kimono1 en formats yuv
HM
FFmpeg
PSNR dB
Bitrates kb/s
44
43
42
41
40
39
38
37
36
0
2000
4000
6000
8000
10000
12000
14000
16000
18000

(d)



(e)

Figure 13 : Comparaison des résultats de compression obtenus par HM et FFmpeg

(a) Blowingbubbles
(b) Bqmall
(c) Parkscene
(d) Kimono1
(e) RaceHorses

Analyse :

Vue que le logiciel HM traite aussi bien les vidéos mais a ce qui parait le traitement peut prendre des journées entière contrairement a l'outil FFmpeg qui lui traite les vidéos en quelques secondes nous avons voulu faire une comparaison entre ces deux dernières en utilisant Matlab.

D'après les résultats on ne peut pas tirer une conclusion pour dire qui est le meilleur, chaque vidéo diffère d'une autre.

III.4.9 Ajouter un son :

Syntaxe :

ffmpeg -i input.mp4 -i son.mp3 -c copy output.mp4

Analyse :

Le son a été ajouté à la vidéo mais on remarque une légère augmentation de la taille du fichier.

III.4.10 Extraction de sous séquences vidéo :

Pour extraire une scène de la vidéo, on peut utiliser **-ss** pour le début de la séquence et

-t pour sa durée.

Syntaxe :

ffmpeg -i LeRoiLion.mp4 -ss 00:01:00.00 -t 00:02:00.00 -c:v copy -c:a copy LeRoiLion2.mp4

Les options **-c:v copy -c:a copy** permettent d'utiliser les mêmes codecs.

Figure 14 : Vidéo originale avant découpage

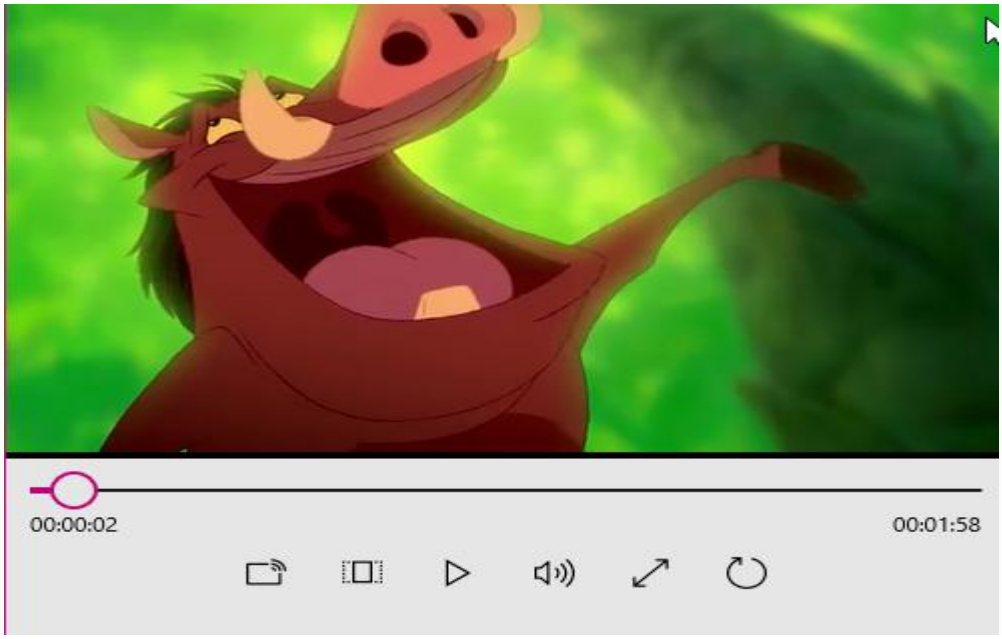


Figure 15 : Vidéo après découpage

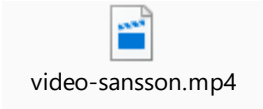
Analyse :

La séquence originale était de 3mn et 17s, celle extraite est de 2mn. La qualité reste cependant inchangée.

III.4.11 Supprimer le son d'une vidéo :

Syntaxe :

ffmpeg –i input.mp4 –c copy output.mp4



III.4.12 Extraction d'image à partir d'une vidéo :

Syntaxe :

ffmpeg -i RaceHorses1H265.mp4 image%djpg

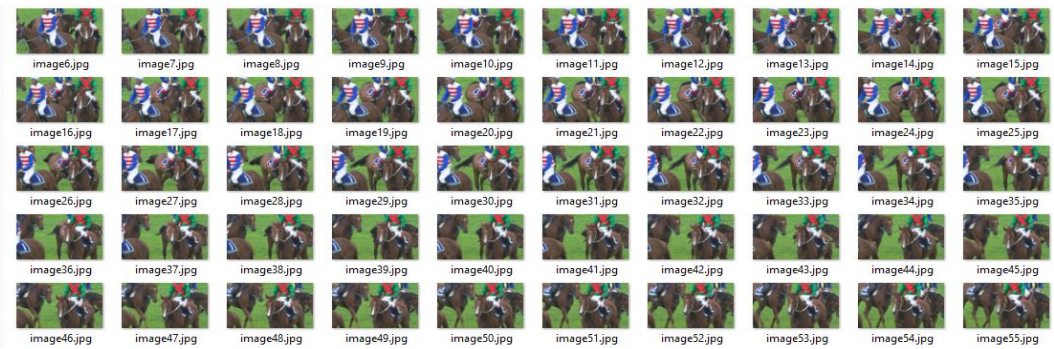


Figure 16 : un aperçu des images qui composent la vidéo (300 images)

III.4.13 Constitution d'une vidéo à partir d'un ensemble d'images :

Syntaxe :

ffmpeg -f image2 -i photo%03d.jpg -r 60 -vcodec mpeg4 -b 15000k nom_de_votre_video.mp4

nouvellevidéo.mp4

Analyse :

Pour vérifier les caractéristiques de la vidéo créé et si elle est conforme aux paramètres de la commande utilisée, une autre commande FFmpeg a été utilisée : ffprobe.

Le résultat est comme suit :

```
Duration: 00:00:12.00, start: 0.000000, bitrate: 10361 kb/s
  Stream #0:0(und): Video: mpeg4 (Simple Profile), yuv420p, 832x480, 10358 kb/s, 60 fps
```

On constate que le frame rate a été respecté (60fps) ainsi que la résolution qui est celle de l'image utilisée et le bitrate n'a pas dépassé celui exigé (15000Kb/s). Ce bitrate est un choix dont dépend directement la qualité souhaitée.

III.5 Conclusion :

Dans ce chapitre, les résultats des différentes simulations on été présentés. Ils ont été analysés et discutés. Ceci s nous a permis de conclure que ce logiciel caché derrière d'autres outils renommés, est très performant et surtout très rapide. Néanmoins, des tests sur d'autres aspects restent nécessaires pour tirer une conclusion finale et objective.

Conclusion et prescriptives :

Le monde d'aujourd'hui connait une évolution fulgurante des télécommunications et du multimédia dominé à plus de 75% par la vidéo.
Le traitement de la vidéo dans le but d'en améliorer la qualité, de la compresser pour le stockage ou la transmission ou d'en extraire de l'information nécessite la maitrise d'outils de plus en plus performants mais aussi de plus en plus complexes. L'un de ces outils très présent dans les logiciels les plus populaires mais méconnu est le FFmpeg.
L'objectif de mon travail était donc d'étudier les performances du FFmpeg pour le traitement de la vidéo.

L'outil FFmpeg s'est avéré très complexe et est capable d'un grand nombre de traitements. Ses filtres permettent des manipulations aussi variées qu'intéressantes des vidéos. L'autre aspect découvert est la performance du FFmpeg dans la compression Vidéo. En effet, FFmpeg renferme un nombre important de codecs dont les tout puissants et populaires x.264 et x.265. Ces codecs sont des implémentations des normes de compression vidéo H.264/AVC et H.265/HEVC respectivement. Le x265 offre une meilleure compression avec la même qualité vidéo que le x264 mais il est encore trop tôt pour le garantir pour toutes les scènes et les situations. Ce point nécessite la conduction d'autres simulations sur une grande variété de vidéos avec diverses résolutions, frame rates et durées.
Le FFmpeg est très puissant dans tout ce qui est multiplexage et démultiplexage de l'audio et la vidéo. Il permet aussi facilement l'extraction de scènes de durées données et leur reconstruction.

Comparé aux résultats de la plateforme HM, FFmpeg donne pratiquement les mêmes performances. Cependant, des difficultés ont été rencontrées pour réaliser les mêmes configurations que celles appliquées avec la HM.

.

Les problèmes rencontrés dans ce projet sont :

- L'évaluation du SSIM avec FFmepg. Bien que la commande apparait simple, les résultats obtenus sont illogiques. Les tests menés sur des vidéos et leurs copies donnent des valeurs inacceptables du SSIM qui devrait être égal, dans ce cas, à 1.
- La modification du frame rate de la vidéo.
- Certaines commandes dépendent de la version de l'outil FFmpeg utilisée. Leurs syntaxes sont différentes entrainant des résultats différents.

Pour conclure, ce travail a donc permis:

- De se familiariser avec l'outil FFmpeg.
- D'acquérir de nouvelles connaissances dans le domaine de la vidéo et particulièrement dans les techniques de compression de la vidéo.
- De découvrir certains nombre d'outils tels que les codecs et les lecteurs de vidéo, et les outils de mesure de la qualité des vidéos tels que le VQMT.
- Enfin, j'espère que ce travail puisse être un guide efficace pour les nouvelles promotions et suscitera leur intérêt pour le compléter et/ou l'améliorer.

Bibliographie

[1] https://www.commentcamarche.net/contents/1493-introduction-a-la-video-numerique.

[2] https://www.sites.univ-rennes2.fr/arts-spectacle/cian/image_numFlash/pdf/chap3_tout.pdf

[3] http://www.clashinfo.com/dico/definition-c/art44-codec.html

[4] https://www.supinfo.com/articles/single/2564-mieux-connaitre-formats-video

[5] http://dspace.univ-tlemcen.dz/bitstream/112/12592/1/

[6] https://fr.wikipedia.org/wiki/Peak_Signal_to_Noise_Ratio

[7] https://itsfoss.com/best-video-editing-software-linux/

[8] https://fr.wikipedia.org/wiki/FFmpeg

[9] https://fr.wikihow.com/installer-FFmpeg-sur-Windows

[10] https://tel.archives-ouvertes.fr/tel-01943877/document

[11] https://fr.wikipedia.org/wiki/Biblioth%C3%A8que_logicielle

[12] https://www.projet-plume.org/fiche/ffmpeg

[13] https://debian-facile.org/doc:media:ffmpeg

[14] https://wiki.debian.org/fr/ffmpeg

[15] https://debian-facile.org/doc:media:ffmpeg

[16] https://ffmpeg.org/ffmpeg.html

[17] https://doc.ubuntu-fr.org/ffmpeg

[18] https://www.vcodex.com/news/h264-quantization-parameter/