République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Sadik BENYAHIA - JIJEL

Faculté des Sciences Exactes et Informatique

Département d'Informatique

Mémoire

de fin d'études pour l'obtention du diplôme

Master de Recherche en Informatique

Option : Réseaux et Sécurité

Thème

Evaluation de la qualité d'une transmission vidéo utilisant le simulateur NS2 et le Framework Evalvid

Encadré par :

M^me^. BOUDJADJA Rima

Réalisé par :

M^lle^. SAIDI Nora

M^lle^. MOHDEB Messaouda

Promotion - 2015 /2016

M. inf. RS 09/16

République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mohamed Sadik BENYAHIA - JIJEL

Faculté des Sciences Exactes et Informatique

Département d'Informatique

09/02

Mémoire

de fin d'études pour l'obtention du diplôme

Master de Recherche en Informatique

Option : Réseaux et Sécurité

Thème

Evaluation de la qualité d'une transmission vidéo utilisant le simulateur NS2 et le Framework Evalvid

Encadré par :

M^me^. BOUDJADJA Rima

Réalisé par :

M^lle^. SAIDI Nora

M^lle^. MOHDEB Messaouda

Promotion - 2015 /2016

Remerciements

Nous tenons en premier lieu à remercier ALLAH le tout puissant qui nous a aidé et nous a donné la patience et le courage durant ces longues années d'études.

Nous profitons de cette occasion pour adresser nos très sincères remerciements à notre encadreur Mme R.BOUDJADJA, d'avoir dirigé notre projet de master avec beaucoup de patience.

Nos vifs remerciements vont également aux membres du jury pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Enfin, nous tenons également à remercier toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

DEDICACE

A mes parents

Dont le mérite, les sacrifices et les qualités humaines m'ont permis de vivre ce jour

A mon Frère et mes sœurs

Ahmed, Kamilia, Rima, Chahla, Karima, Samira, A qui je souhaite un avenir radieux plein de réussite

A mes nièces Basmala, Israa, Darine et mon neveu Abdarrahmane

A toute ma famille ainsi qu'à mes amis

Messaouda

DEDICACE

A mes parents pour leur amour inestimable, leur confiance, leur soutien, leurs sacrifices et toutes les valeurs qu'ils ont su m'inculquer.

A mes beaux frères et sœurs Kamel, Hocine, Nadjet, Houriya qui n'ont cessé d'être pour moi des exemples de persévérance, de courage et de générosité.

A mon neveu Yaakoub, ma plus grande source de bonheur, j'espère que la vie lui réserve le meilleur.

A toute ma famille ainsi qu'à mes amis.

Nora

Résumé

L'Internet a été utilisé pour les applications de données comme le courrier électronique, le chat, la navigation sur le Web, etc. La combinaison de la technologie multimédia et l'Internet attire actuellement un nombre considérable de l'attention de chercheurs, de développeurs et d'utilisateurs finaux.

La transmission vidéo est devenue aujourd'hui une nécessité pour presque tous les utilisateurs d'Internet, quel que soit le dispositif utilisé pour accéder à Internet. Généralement, la transmission vidéo exige une bande passante élevée. Sans compression, il est très difficile de transmettre la vidéo sur des réseaux filaires ou sans fil.

L'évaluation de la qualité de la vidéo est l'une des étapes les plus importantes dans le domaine de la transmission vidéo. Evalvid est un outil d'évaluation vidéo très utilisé dans les réseaux. La simulation de réseau est confiée à un simulateur réseau, nommé NS-2 (Network Simulator version 2). L'outil Evalvid devait donc s'intégrer à ce simulateur, de façon à pouvoir réaliser l'évaluation de la vidéo transmise.

Mots-clefs : NS-2, Evalvid, qualité vidéo, transmission vidéo.

Abstract

Conventionally, the Internet has been used for data applications like email, chat, Web surfing, etc. The combination of multimedia technology and the Internet is currently attracting a considerable amount of attention from researchers, developers and end users.

Video transmission nowadays has become a necessity for almost every Internet user, irrespective of the device used to access the Internet. Typically, video transmission demands high bandwidth. Without compression, it is very difficult to transmit video over wired or wireless networks.

The evaluation of the video quality is one of the most important steps in the field of video transmission, evalvid is a framework and a toolkit for unified assessment of the quality of video transmission. Simulation of our network is assigned to a network simulator, named NS-2(Network Simulator version 2).Our tool had to be integrated in this simulator, so as to realize the evaluation of the transmitted video.

Keywords: NS-2, Evalvid, video quality, transmission video.

LISTE DES ACRONYMES

AWGN	Addidive White Gaussian Noise
Awk	Awkward
AVC	Advanced Video Coding
ARQ	Automatic Repeat Request
BBAG	Bruit Blanc Additif Gaussien
BER	Bit Error Rate
CCE	Codes Correcteurs d'Erreurs
CODEC	Compression / DECompression
CIF	Common Intermediate Format
CBR	Constant Bit Rate
DCT	Discrete Cosine Transform
DSCQS	Double Stimulus Continuous Quality Scale
EvalVid	Evaluation Video
ET	Evaluate Trace
FEC	Forward Error Correction
FV	Fix Video
FTP	File Transfer Protocol
GOP	Group Of Pictures
HTTP	Hyper Text Transport Protocol
HD	High Definition

I	Intra
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IETF	Internet Engineering Task Force
ITU-T	International Telecommunications Union – Telecommuniction
ISO	International Organization for Standardization
JM	Joint Model
JVT	Joint Video Team
LAN	Local area network
MB	Macro Block
MPEG	Moving Picture Experts Groups
MTU	Maximum Transmission Unit
MOS	Mean Opinion Score
MSE	Mean Squared Error
NAL	Network Abstraction Layer
NS2	Network Simulator Version 2
NAM	Network AniMator
OTCL	Object Tools Command Language
QoS	Quality of Service
PSNR	Peak Signal-to-Noise Ratio
QP	Quantification Parameter
QCIF	Quarter CIF
RTP	Real-time transport Protocol

RTCP	Real-timeTransport Control Protocol
RTSP	Real Time Streaming Protocol
RGB	Red Green Blue
RVB	Rouge Vert Bleu
RGB	Red Green Blue
RAM	Random Acces Memory
SMTP	Simple Mail Transport Protocol
SIP	Session Initiation Protocol
SSIM	Structural SIMilarity
TCP/IP	Transmission Control Protocol / Internet Protocol
TCL	Tools Command Language
TPP	Taux de perte de paquet
UDP	User Datagram Packet
URL	Uniform Resource Locator
VCEG	Video Coding Experts Group
VS	Video Sender
VBR	Variable Bit Rate
YCbCr	Luminance, Chrominance blue, Chrominance rouge
YUV	Espace de couleur (idem que YCbCr)

Liste des Figures

Liste des Tables

Table de matières

Chapitre 2 : L'évaluation de la qualité vidéo

Chapitre 3 : Le Simulateur NS-2

Introduction générale

Les domaines des télécommunications, du multimédia et de la transmission vidéo ont connu une évolution fulgurante. Dans de nombreuses applications multimédia, il devient indispensable de juger la qualité de ces images traitées. La compression notamment, en fort développement avec l'explosion du multimédia, impose une garantie d'interprétation et de rendu des images après transfert ou stockage. Au cours des dernières décennies, les méthodes et les algorithmes de traitement des images et de la vidéo (le multimédia) ont eu des avancées importantes. Parmi les applications possibles, on trouve la vidéoconférence, la vidéo sur Internet, la téléprésence, le stockage de données multimédia, etc.

Ceci a donné cependant lieu à quelques problèmes importants qui doivent être abordés. Un des premiers problèmes à considérer est l'évaluation de la qualité de la vidéo, c'est un problème puisqu'aucune solution proposée jusqu'à présent n'est satisfaisante. Alors que de nos jours, les images et les vidéos numériques sont omniprésentes et la quantité de données associées est gigantesque. Ce qui rend l'évaluation de la qualité finale de la vidéo essentielle au vu des dégradations que subit la vidéo au cours de la compression nécessaire à sa transmission et au cours de sa transmission.

On admet que la qualité finale est perçue par des spectateurs humains et que c'est la qualité visuelle de ce qu'ils perçoivent qui est en jeu. Il est donc nécessaire de mettre en œuvre des évaluations subjectives de qualité effectuées par des observateurs, qui de préférence n'ont pas de lien avec l'étude de la qualité vidéo. Il serait évidemment souhaitable de disposer d'une méthode objective pour mesurer cette qualité, utilisant un modèle mathématique pour stimuler le système humain. Cette méthode serait relativement plus rapide et moins chère que l'évaluation subjective.

Dans le but de changer le jugement de la perception humaine par l'évaluation de la machine, plusieurs recherches ont été réalisées au cours des deux dernières décennies. Parmi les méthodes usuelles, l'erreur quadratique moyenne (Mean Squared Error (MSE)), le rapport signal sur bruit de crête (Peak Signal to Noise Ratio (PSNR)), qui mesure la qualité en faisant une différence entre les images originales et dégradée. Un autre axe de recherche dans ce

domaine se base sur les caractéristiques du système visuel humain, telle que la fonction de sensibilité au contraste. Il faut observer que pour vérifier l'exactitude de ces mesures, elles doivent habituellement être corrélées avec les résultats obtenus en utilisant des évaluations subjectives de la qualité.

Le plan de travail que nous avons suivi est :

Chapitre 1 : « Généralités sur la vidéo sans fil»

Au début de ce chapitre nous avons présenté des généralités sur les réseaux sans fils. Nous avons décrit brièvement les couches de transport du modèle TCP/IP ainsi que les différents protocoles et les canaux (stationnaires et non stationnaires) de transmission. Ensuite la chaine de transmission numérique et le processus de la transmission de la vidéo sans fil ont été détaillés. Nous avons fini par une présentation de standard de codage vidéo H.264.

Chapitre 2 : « Evaluation de la qualité de la vidéo »

Dans ce chapitre nous avons cité les méthodes subjectives et objectives les plus connues dans le domaine de traitement d'image et de vidéo. Nous avons présenté aussi Evalvid l'outil le plus connu d'évaluation de qualité vidéo.

Chapitre 3 : « Le simulateur NS2 »

Ce chapitre est consacré à la présentation du simulateur NS-2. Nous avons présenté les différentes étapes nécessaires pour l'installation de ce simulateur.

Chapitre 4: « Simulation et Résultats »

Dans le quatrième chapitre nous avons simulé puis évalué la transmission d'un ensemble de séquences vidéo sur des réseaux filaires et sans fil en utilisant NS-2 et Evalvid. Les expérimentations ont été réalisées par rapport à un certain nombre de paramètres d'évaluation.

Chapitre 1 : Généralités sur la Vidéo sans Fil

1.1 Introduction

Les réseaux sans fil (Wireless Networks) [1] constituent de plus en plus une technologie émergente permettant à ses utilisateurs un accès à l'information et aux services électroniques indépendamment de leurs positions géographiques. Le succès de ce type de réseaux ces dernières années est suscité par un grand intérêt de la part des particuliers, des entreprises et du milieu industriel. En effet, ce type de réseaux est perçu comme une nouvelle alternative complémentaire aux réseaux filaires traditionnels, car ils sont autant utilisés dans le cadre des réseaux locaux d'entreprise pour une utilisation purement professionnelle, que dans le cadre des réseaux locaux personnels à domicile.

Aujourd'hui les débits atteints avec les réseaux sans fil rendent possible le transfert de flux multimédia soumis à de fortes contraintes. Le respect de certaines contraintes telles que : la bande passante, le délai ou encore le taux de pertes de paquets devient primordial. Cependant les solutions qui ont été introduites dans le monde des réseaux filaires deviennent inadaptées pour des réseaux dont l'administration n'est pas centralisée et qui utilisent un médium radio partagé.

1.2 Généralités

Un réseau sans fil peut être considéré comme un système de transmission de données, dont le but est d'assurer une liaison indépendante de l'emplacement des entités informatiques qui composent le réseau. Plutôt que d'utiliser une infrastructure câblée, dans un réseau sans fil, l'ensemble des nœuds communiquent via des interfaces radios sous forme de cartes réseaux PCI, PCMCIA, etc. Ces cartes sans fil peuvent se connecter directement aux machines des utilisateurs et interagir avec toute la pile de protocoles de communication.

D'un autre côté, les réseaux locaux sans fil sont généralement utilisés pour faire le lien final entre le réseau câblé existant et des machines clientes, offrant aux utilisateurs un accès sans fil à l'ensemble des ressources et des services offerts par le réseau de l'entreprise, sur un

ou plusieurs bâtiments. Pour cela, ce type de réseaux est en passe de devenir l'une des principales solutions de connexion pour de nombreuses entreprises. Ainsi, le marché du sans fil s'est développé rapidement dès lors que les entreprises constatent les gains de productivité qui découlent de la disparition des câbles [1].

L'avantage essentiel d'un réseau sans fil est la mobilité. Dans Un réseau sans fil, un utilisateur peut rester connecté tout en se déplaçant dans un périmètre géographique plus ou moins étendu. Les réseaux sans fil sont basés sur une liaison utilisant des ondes radio ou infrarouges au lieu des câbles habituels.

Il existe plusieurs technologies sans fil se distinguant les unes des autres sur plusieurs points en particulier :

- La fréquence d'émission utilisée
- Le type de modulations (OFDM et autres)
- Le type de codage (source et/ou canal)
- Les correcteurs et contrôleurs d'erreurs
- Le type de multiplexage (TDMA, FDMA, CDMA ...etc)
- Le débit
- La portée des transmissions
- ...etc

Selon la technologie utilisée, les réseaux sans fil permettent de relier très facilement des équipements distants d'une dizaine de mètres à quelques kilomètres. De plus l'installation de tels réseaux ne demande pas de lourds aménagements.

Il existe plusieurs catégories de réseaux sans fil, selon l'étendue géographique de la connectivité offerte (zone de couverture).

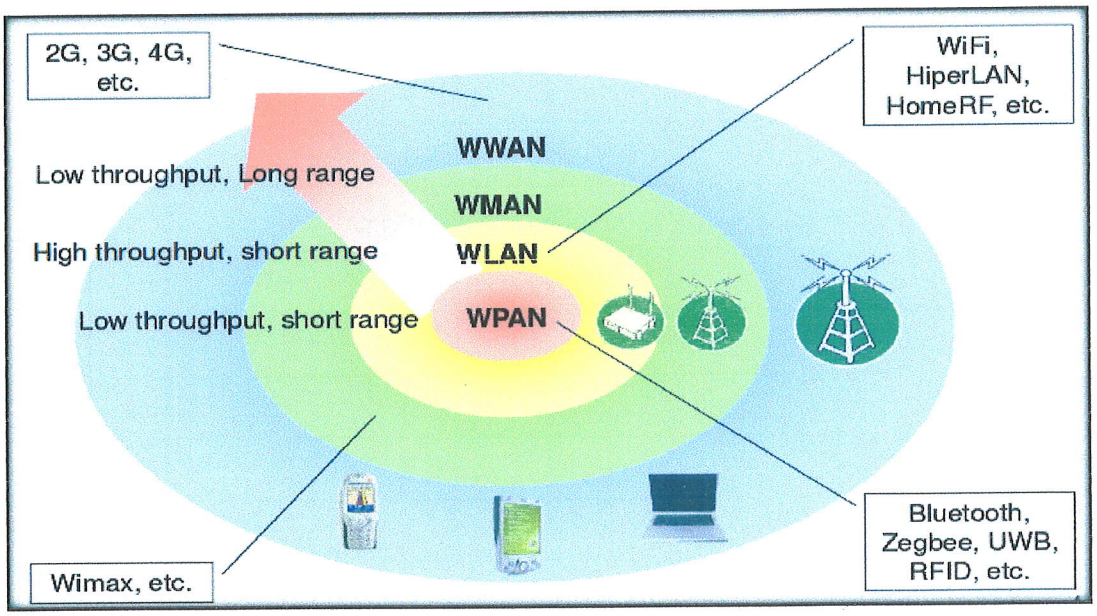


Figure 1.1 : Les différents types des réseaux sans fil.

1.2.1 Vue d'ensemble sur la communication vidéo

Nous vivons aujourd'hui un véritable technologique dans les systèmes de communications essentiellement ceux utilisant la vidéo. Les applications de communications sont très variées et dont les principes de fonctionnement sont relativement différents les uns par rapport aux autres. A titres d'exemples nous pouvons citer quelques exemples de techniques de communications vidéo utilisées :

- **Communication Unicast**

C'est l'envoie d'un message d'un émetteur vers un seul récepteur. On parle aussi de communication point à point.

- **Communication Broadcast**

Encore appelée diffusion large, le broadcast est l'envoi de messages à tous les récepteurs d'un même réseau ou d'un sous réseau donné sans que ces derniers ne le sollicitent.

- **Communication Multicast**

C'est une diffusion restreinte, elle consiste dans l'envoi de messages (données) d'un émetteur à un groupe de récepteurs ou de plusieurs émetteurs vers plusieurs récepteurs appartenant à un ou plusieurs groupes.

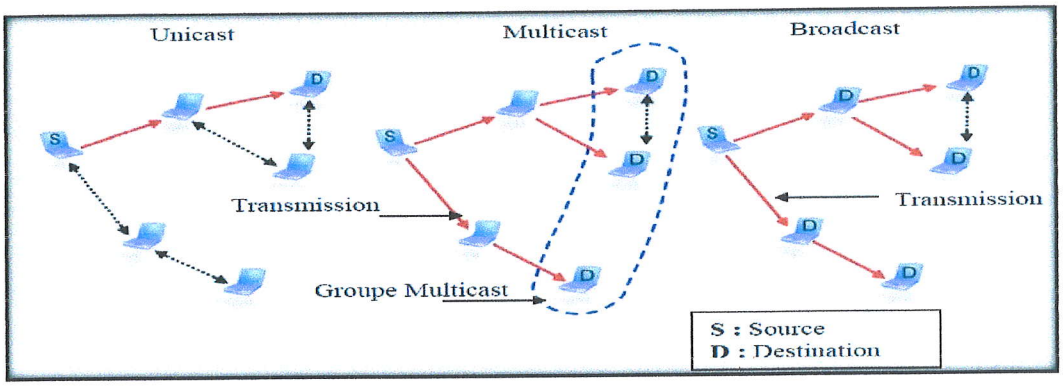


Figure 1.2 : Les différents modes de communication

1.3 Canal de transmission

Le canal de transmission est un support physique permettant de transmettre l'information entre un émetteur et un récepteur. Dans le cas d'une communication sans fil, c'est l'onde électromagnétique qui assure l'acheminement de l'information vers le récepteur. Le canal de transmission est caractérisé par sa capacité et par sa bande passante.

On distingue deux groupes de canaux : Les canaux stationnaires et Les canaux non stationnaires. La figure suivante représente la modélisation de l'action du canal

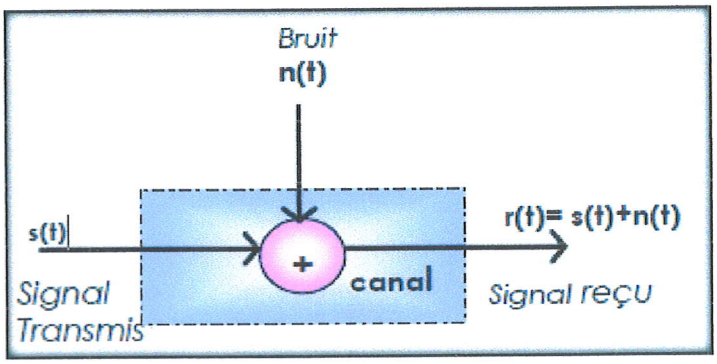


Figure 1.3 : La modélisation de l'action du canal

1.3.1 Les canaux stationnaires

Ce sont des canaux dont les paramètres sont fixes au cours du temps : fibres optiques, câbles métalliques …

Le canal Gaussien

Parmi les canaux stationnaires [2] les plus utilisés, le canal BBAG pour Bruit Blanc Additif Gaussien ou AWGN est celui sur lequel l'évaluation des performances de systèmes de communications est la plus simple à déterminer. Les résultats obtenus avec ce canal servent de référence pour les autres systèmes. Il représente assez fidèlement un canal de transmission radio électrique, lorsque les antennes d'émission et de réception sont en vue directe (simple trajet), la diffusion de la télévision par satellite en est un exemple courant.

Mentionnons que l'adjectif « additif » dans BBAG signifie que l'impact du bruit sur le signal transmis peut être modélisé comme une variable aléatoire N qui s'additionne au signal modulé, cette variable est supposée gaussienne de moyenne nulle et de variance σ et donc de densité de probabilité :

$$p(n) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-n^2}{2\sigma^2}}$$

La densité spectrale de puissance bilatérale du bruit présent lors de la transmission sur fréquence porteuse est égale à $N_0/2$. Si C_k est la variable qui représente le signal modulé correspondant à un symbole à l'instant k, alors à la sortie du canal nous recueillons :

R_k=C_k+n

Ou $R_k$est la version bruitée du signal $C_k$en sortie

n : vecteur de bruit

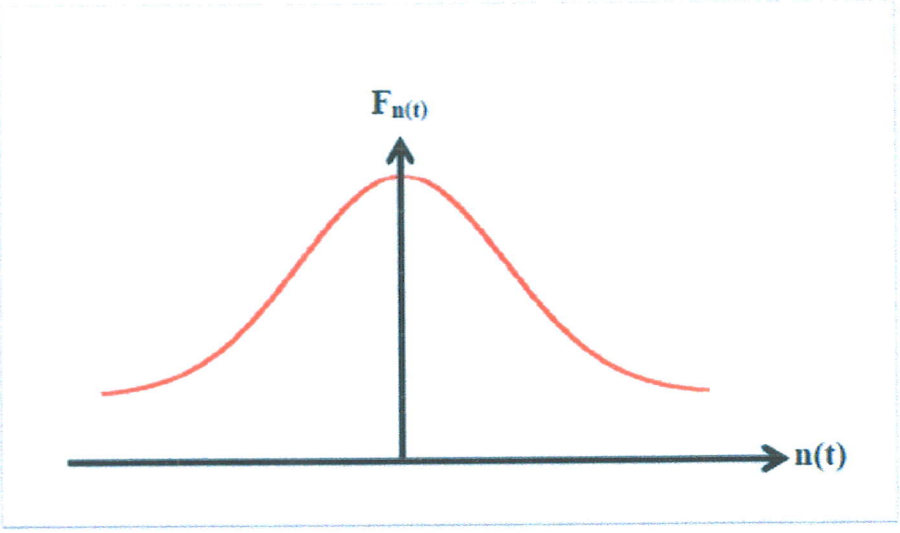


Figure 1.4 : Fonction de densité de probabilité gaussienne.

1.3.2 Les canaux non stationnaires

Ce sont des canaux dont les paramètres évoluent au cours du temps : canal radio mobile, canal ionosphérique. La modélisation du canal est une phase importante est très étudiée dans le domaine de la transmission sans fil, le modèle du canal dépend essentiellement de sa capacité.

Les canaux de Rice et de Rayleigh

Le modèle de Rice [2] est utilisé lorsque l'un des trajets reçus est prédominant ; cela est généralement le cas lorsque l'émetteur et le récepteur sont en visibilité directe. L'amplitude N= n du signal reçu suit alors la loi de probabilité suivante :

$$p(n) = \frac{s}{\sigma^2} e^{\left(\frac{-(n^2+A^2)}{2\sigma^2}\right)} I_0\left(\frac{A_n}{\sigma^2}\right)$$

- I_0 est la fonction de Bessel de première espèce à l'ordre 0.
- A est l'amplitude de l'onde prédominante.
- σ^2est la puissance totale du signal.

Signalons que l'on peut considérer la phase comme constante ou variant selon une loi de probabilité uniforme. Dans le cas où aucun trajet ne prédomine, le modèle de Rice se transforme en un modèle de Rayleigh défini par la loi de probabilité suivante :

$$p(n) = \frac{s}{\sigma} e^{\frac{-n^2}{2\sigma^2}}$$

En outre, il faut aussi noter l'existence possible du phénomène Doppler existant lorsque l'émetteur et/ou le récepteur se déplacent. Ce phénomène induit un décalage en fréquence des signaux transmis directement lié à la vitesse de déplacement, à la fréquence porteuse et aux directions d'arrivée des ondes par rapport à la direction de déplacement.

Ainsi, d'une façon générale en transmission numérique, l'étude du canal conduit à modéliser ses caractéristiques, permettant alors de mettre en place des techniques appropriées de mise en forme et de restitution de l'information. À titre d'exemple, citons la dispersion des retards des trajets propagés dont la valeur donne une indication sur le risque d'interférences entre symboles.

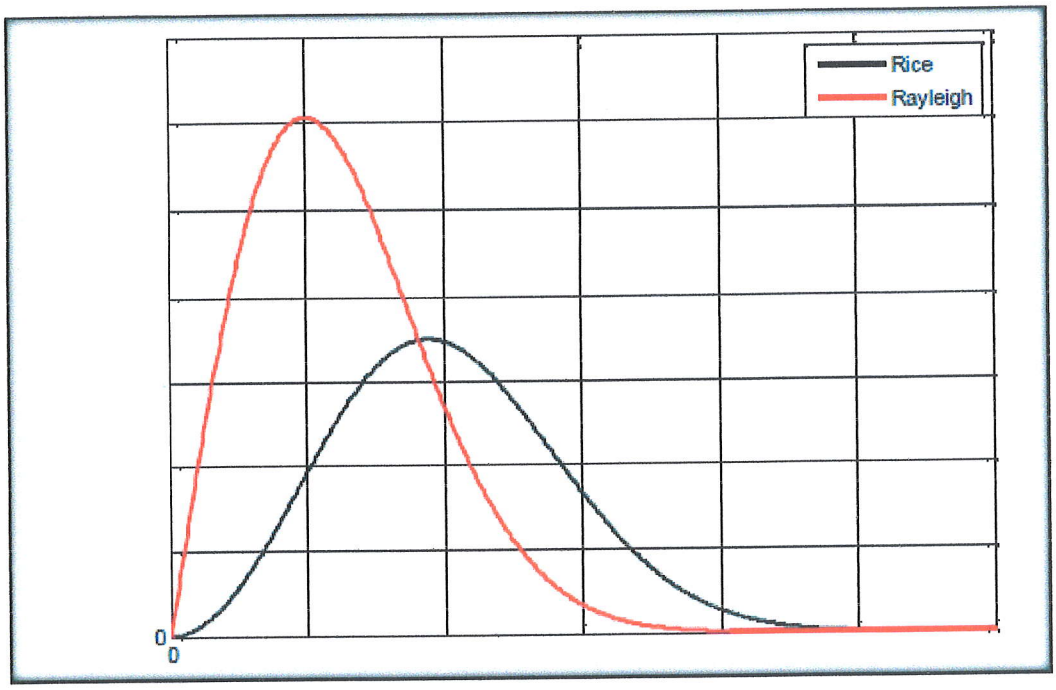


Figure 1.5 : Fonction de densité de probabilité de Rayleigh et de Rice

1.3.3 Bruit

Le bruit est un signal perturbateur provenant du canal lui même ou de son environnement externe. Il est de comportement aléatoire est vient s'ajouter au signal véhiculant les informations et provoquer ainsi les erreurs de transmission. On distingue généralement trois types de bruit : le bruit blanc, le bruit impulsif, et le bruit gaussien.

1.3.3.1 Bruit blanc

Le bruit blanc est un bruit dont la puissance est uniformément repartie dans toute la bande passante du canal, il s'agit essentiellement d'un bruit provoqué par l'agitation thermique des électrons dans le conducteur électrique.

1.3.3.2 Bruit impulsif

Comme son nom l'indique ce type de bruit est à caractère impulsif, il se présente sous forme de tensions perturbatrices de valeur élevée mais de durée brève. Ces bruits sont très gênants pour la transmission de données, car le signal perturbateur modifie la forme du signal reçu à des instants quelconques (aléatoires) telles qu'il se produit des erreurs à la réception.

1.3.3.3 Bruit gaussien

Soit x_n un signal discret. Le bruit est une perturbation, que l'on notera b_n. Le signal résultant est :

$$y_n = x_n + b_n$$

b_n est une variable aléatoire continue. On parle de bruit gaussien lorsque la densité de probabilité de cette variable est la loi gaussienne (ou loi normale). On suppose que la loi gaussienne est centrée. La densité de probabilité est alors :

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(\frac{x^2}{2\sigma^2}\right)$$

La probabilité pour que le bruit b soit compris entre x et x+dx est p(x)dx. Les signaux délivrés par les instruments de mesure comportent un bruit qui peut être généralement considéré comme gaussien. Cette hypothèse permet de définir l'écart type σ comme l'incertitude type due aux erreurs aléatoires.

1.3.3.4 Notion de rapport signal sur bruit

La quantité de bruit présente sur un canal de transmission, est exprimée par le rapport de la puissance du signal transmis sur la puissance de bruit et prend le nom de rapport S/B signal sur bruit. Ce rapport varie dans le temps, puisque le bruit n'est pas uniforme, toutefois on peut en estimer une valeur moyenne sur un intervalle de temps. Le rapport signal sur bruit est aussi une caractéristique d'un canal de transmission.

1.4 Le modèle TCP/IP

L'architecture en couches TCP/IP pour la gestion du réseau est utilisée par l'architecture courante de l'Internet. Cette architecture a permis aux différents réseaux d'être reliés ensemble d'une manière efficace. La hiérarchie des couches fournit des abstractions naturelles faites à la hiérarchie actuelle dans les réseaux.

Typiquement le modèle TCP/IP est présenté en quatre couches comme illustré dans la figure suivante (la couche Accès réseau regroupe deux couches de modèle OSI : La couche Physique et la couche liaison de données):

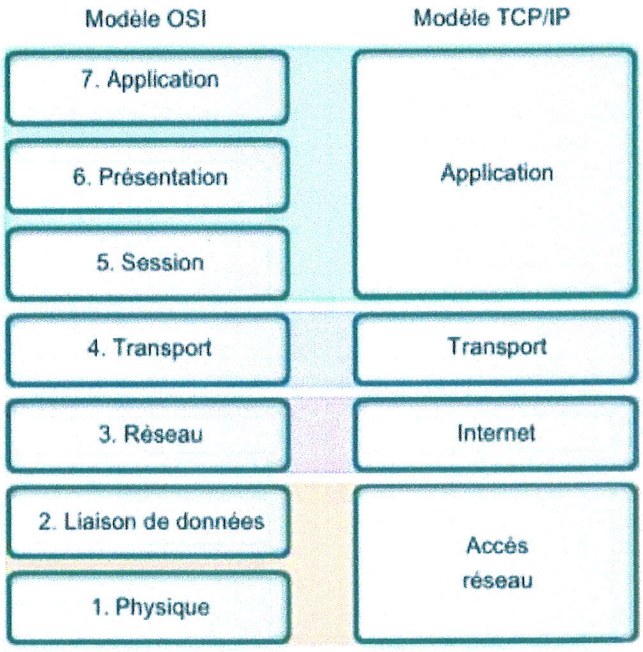


Figure 1.6 : Comparaison des modèles OSI et TCP_IP

1.4.1 La couche physique

Le niveau physique regroupe [3] les règles et procédures qui permettent d'acheminer des éléments binaires non structurés sur un support physique entre l'émetteur et le récepteur. Ce niveau prend en charge les aspects de codage et de modulation de signaux numériques électriques et optiques. La couche physique des réseaux sans fil est définie par la famille des normes IEEE 802.11, 802.11a, 802.11b, 802.11g et 802.11n.

Elle a pour rôle principal d'établir et de maintenir le lien radio ou infrarouge pour permettre la transmission de données sans fil entre les stations composant le réseau. Pour ce faire, elle propose certaines primitives à la couche supérieure.

1.4.2 La couche d'accès

La couche liaison (liaison de données : datalink (eng)) s'occupe de la bonne transmission de l'information entre les nœuds via le support, en assurant la gestion des erreurs de transmission et la synchronisation des données. Elle va transformer la couche physique en une liaison à priori exempte d'erreurs de transmission pour la couche réseau. Elle fractionne

les données d'entrée de l'émetteur en trames, transmet ces trames en séquence et gère les trames d'acquittement renvoyées par le récepteur.

La couche liaison de données doit être capable de renvoyer une trame lorsqu'il y a eu un problème sur la ligne de transmission. De manière générale, un rôle important de cette couche est la détection et la correction d'erreurs intervenues sur la couche physique. Cette couche intègre également une fonction de contrôle de flux pour éviter l'engorgement du récepteur.

1.4.3 La couche réseau

Elle s'occupe de l'acheminement, à bonne destination, des paquets de données indépendamment les uns des autres, soit donc de leur routage à travers les différents nœuds par rapport au trafic et à la congestion du réseau. Il n'est en revanche pas du ressort de cette couche de vérifier le bon acheminement.

Le protocole IP offre un acheminement au mieux (Best-Effort) [3], non fiable et non orienté connexion. La version largement déployée actuellement est la version IPv4 antérieure à la version IPv6 qui a été conçue principalement pour élargir la plage d'adressage IP. Le service Best-Effort fourni par le protocole IP n'offre aucun contrôle, ni garantie, sur les conditions de transport de bout-en-bout des paquets IP.

1.4.4 La couche transport

Cette couche est responsable du bon acheminement des messages complets au destinataire. Le rôle principal de la couche transport est de prendre les messages de la couche session, de les découper s'il le faut en unités plus petites et de les passer à la couche réseau, tout en s'assurant que les morceaux arrivent correctement de l'autre côté. Cette couche effectue donc aussi le réassemblage du message à la réception des morceaux.

1.4.5 La couche application

Cette couche est le point de contact entre l'utilisateur et le réseau. C'est est le point d'accès des applications aux services réseaux. On y retrouve toutes les applications de communication via le réseau communément utilisées sur un LAN ou sur internet. Un grand nombre de protocoles divers de haut niveau permettent d'assurer les services de cette couche :
Telnet : ouverture de session à distance ;
FTP (File Transfer Protocol) : protocole de transfert de fichiers ;

HTTP (HyperText Transfer Protocol) : protocole de transfert de l'hypertexte
SMTP (Simple Mail Transfer Protocol) : protocole simple de transfert de courrier
DNS (Domain Name System) : système de nom de domaine ; etc.

1.5 Chaîne de transmission numérique

Les systèmes de transmission numérique véhiculent l'information sous forme numérique entre une source et un ou plusieurs destinataires (Figure1.7) en utilisant un support physique comme le câble, la fibre optique ou encore la propagation sur un canal radioélectrique. Les signaux transportés peuvent être soit directement d'origine numérique, comme dans les réseaux de données, soit d'origine analogique (parole, image...) mais convertis sous une forme numérique. La tâche du système de transmission est d'acheminer l'information de la source vers le destinataire avec le plus de fiabilité possible.

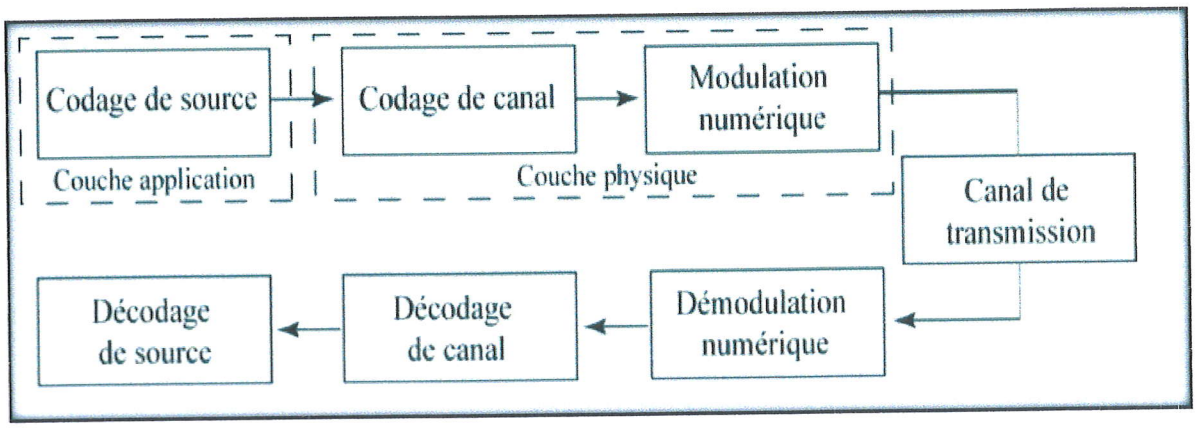


Figure 1.7 : Chaîne de transmission numérique

Codage de source : Le codage de source consiste à représenter l'information de la source sous forme binaire pour qu'elle puisse être transmise par un système de transmission numérique De plus, le codage de source cherche à représenter l'information de la source par le moins de bits possibles, ce qui permet d'améliorer le débit utile de la transmission. Il permet de compresser les données à transmettre en éliminant la plus grande partie de la redondance (spatiale et temporelle) du flux vidéo.

Codage de canal : L'objectif du codage de canal est de protéger l'information issue du codeur de source contre les perturbations du canal. Il s'agit donc de rajouter de la redondance de manière à détecter et éventuellement corriger les erreurs lors de la réception si la stratégie adoptée le permet. La protection est réalisée à l'aide des Codes Correcteurs d'Erreurs (CCE) qui consistent à rajouter de la redondance à l'information d'une manière contrôlée. Cette

redondance est exploitée par le décodeur pour détecter voir corriger certaines erreurs de transmission.

Modulation : Permet de transformer en signal physique les données numériques issues du codage de canal. La modulation numérique permet d'adapter l'information codée au support de transmission. La modulation, quelle soit dite analogique ou numérique, consiste à faire varier un paramètre de l'onde porteuse en fonction du signal qui constitue l'information à transmettre, appelé signal modulant.

Canal de transmission : Le canal de transmission est un support physique permettant de transmettre l'information entre un émetteur et un récepteur. Dans le cas d'une communication sans fil, c'est l'onde électromagnétique qui assure l'acheminement de l'information vers le récepteur. Cette étape regroupe donc toutes les composantes qui perturbent la transmission des informations dans la chaîne de base. Il existe plusieurs modèles théoriques du canal de transmission en fonction des types d'erreurs les plus fréquents.

Démodulation : Permet de traiter les formes d'onde en provenance du canal par des processus d'estimation et de quantification et les réduits à des séquences de nombres, qui représentent des estimations des symboles émis. Ces séquences sont ensuite décodées selon les opérations inverses de celles employées à l'émission, ce qui permet au destinataire de retrouver l'information binaire initiale.

Décodage de canal : Plusieurs stratégies différentes peuvent être utilisées par le décodeur de canal.

- La première est la détection d'erreurs. Le décodeur observe la séquence reçue (ferme ou souple) et détecte la présence éventuelle d'erreur. Cette détection peut servir à contrôler le taux d'erreur (Error Monitoring) ou à mettre en œuvre des techniques de retransmission (ARQ : Automatic Repeat Request) : le décodeur demande à l'émetteur de retransmettre la séquence dans laquelle une erreur a été détectée. Il est évident que ce type de procédé nécessite une voie de retour. Cette stratégie de détection est surtout utilisée par les couches transport et supérieures du modèle OSI.
- La deuxième est la correction d'erreurs (FEC : Forward Error Correction). Elle nécessite des algorithmes beaucoup plus complexes que la simple détection, et plus de redondance dans la séquence émise. Toutefois, le milieu de transmission est utilisé de manière plus efficace.

Décodage de la source : Permet de reconstruire, par la connaissance du processus de compression employé à l'émission, les données transmises en les interprétants pour générer un flux vidéo interprétable. Dans les solutions les plus évoluées, des techniques permettant de

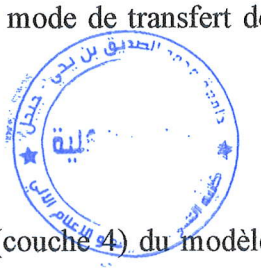
pallier aux erreurs résiduelles présentes dans le flux remonté au niveau applicatif peuvent être utilisées.

1.6 Protocoles de communication

1.6.1 Le protocole TCP

TCP « Transport Control Protocol » est un protocole de la couche transport (couche 4) du modèle TCP/IP. Il est orienté connexion. Il permet le transfert des données avec un contrôle et une reprise sur erreur. La taille minimum de son en-tête est de 20 octets qui contiennent en plus un nombre d'option. TCP est incompatible avec le multimédia temps réel et les jeux en réseau. Plusieurs champs dans l'en-tête TCP sont indispensables (dont numéro de port et numéro de séquence), et devront se retrouver dans les protocoles de substitution UDP et RTP.

Le protocole TCP n'est que peu utilisé dans le cadre de la transmission de données audio de type « stream ». En effet, il n'a que peu d'avantages pour ce mode de transfert de données, et déborde d'inconvénients.



1.6.2 Le protocole UDP

UDP « User Datagram Packet » est un protocole de transport (couche 4) du modèle TCP/IP. Contrairement à TCP, UDP a très peu d'options à gérer et l'en-tête des paquets UDP est beaucoup plus petit, ce qui contribue à rendre son traitement plus rapide. Le transport est en mode non connecté est donc non fiable et sans garantie de séquencement des datagrammes. Il n'y a pas de contrôle de reprise sur erreur, mais UDP indique qu'il y a eu erreur. Il apporte la possibilité de distinguer plusieurs destinations sur un même ordinateur. Un programme d'application qui utilise UDP doit gérer les problèmes de fiabilité, pertes de messages, duplications, retards, déséquencement et perte de connectivité. Chaque message UDP est appelé datagramme (datagramme UDP). Du point de vue conceptuel, le datagramme comporte deux parties ; un en-tête fixe (8 octets) et une zone de données.

1.6.3 Le protocole RTP

RTP « real-time transport Protocol » [4] est un protocole qui a été développé au début des années 90. Le but de ce protocole est de transporter des données avec une garantie temporelle sur l'arrivée de ces données et d'organiser les paquets à l'entrée du réseau et de les

contrôler à la sortie (de bout en bout). RTP est un protocole qui a été développé par l'IETF afin de faciliter le transport temps réel de bout en bout des flots données audio et vidéo sur les réseaux IP.

RTP est un protocole qui se situe au niveau de la couche application et qui utilise les protocoles sous-jacents de transport TCP ou UDP. Mais l'utilisation de RTP se fait généralement au-dessus d'UDP ce qui permet d'atteindre plus facilement le temps réel. Les applications temps réels comme la parole numérique ou la visio-conférence constitue un véritable problème pour Internet. Qui dit application temps réel, dit présence d'une certaine QoS que RTP ne garantit pas du fait qu'il fonctionne au niveau Applicatif. De plus RTP est un protocole qui se trouve dans un environnement multipoint, donc on peut dire que RTP possède à sa charge, la gestion du temps réel, mais aussi l'administration de la session multipoint.

1.6.4 Le protocole RTCP

RTCP [4], un protocole fonctionnant avec RTP au service des applications de réseau multimédias. Les émetteurs et les récepteurs envoient périodiquement des paquets RTCP aux groupes Multicast. Ces groupes reçoivent les paquets RTP (sur des ports UDP différents de des données). Chaque paquet RTCP contient généralement un rapport d'émission ou de réception suivi d'une description de la source.

RTCP transmet périodiquement des paquets de contrôle aux participants, utilisant les mêmes moyens de diffusion que RTP, simplement avec un numéro de port différent. RTCP permet de recevoir des informations de retour des participants, grâce aux messages "sender report" et "receiver report".

1.6.5 Le protocole IPV6

Le protocole IP version 6, élaboré par l'IETF au milieu des années 90. Il est conçu pour permettre le retour à un environnement mondial où les règles d'adressage du réseau seront de nouveau transparentes pour les applications. On peut donc se demander, quelles seront les potentialités de l'IPv6 et si, à court terme, ce protocole remplacera définitivement son prédécesseur.

La version 6 du protocole Internet comble beaucoup de lacunes de la version précédente, IPv4. Outre la plage d'adresses largement étendue, la mise en place des sous

réseaux et tables de routage est bien plus aisée, les paquets peuvent être plus petits et il est parfaitement adapté aux postes mobiles.

1.6.6 Le protocole RTSP

Le protocole RTSP « Real Time Streaming Protocol» [4] est un protocole de niveau applicatif, au-dessus du protocole TCP. Il utilise les protocoles RTP et RTCP. Parmi les fonctionnalités de protocole RTSP, nous citons :

- La recherche des médias sur des serveurs multimédia
- Faire des invitations à des serveurs médias pour rejoindre des conférences, ...
- Utilise dans des applications unicast et multicast
- Contrôler et synchroniser la distribution de flux audio et vidéo (streaming) sur un réseau IP

1.6.7 Le protocole SIP

SIP « Session Initiation Protocol »est un protocole de signalisation défini par l'IETF permettant l'établissement, la libération et la modification de sessions multimédias .Il hérite de certaines fonctionnalités des protocoles HTTP (Hyper Text Transport Protocol) utilisé pour naviguer sur le WEB, et SMTP (Simple Mail Transport Protocol) utilisé pour transmettre des messages électroniques (E-mails).

SIP s'appuie sur un modèle transactionnel client/serveur comme HTTP. L'adressage utilise le concept d'URL SIP (Uniform Resource Locator) qui ressemble à une adresse E-mail. Chaque participant dans un réseau SIP est donc adressable par une URL SIP. Par ailleurs, les requêtes SIP sont acquittées par des réponses identifiées par un code numérique.

1.7 La transmission de la vidéo sur les réseaux sans fil

1.7.1 Définition de la vidéo

La vidéo est une succession d'images animées. Le principe fondamental de la vidéo est que l'œil humain a la possibilité de retenir pendant un certain temps (de l'ordre du dixième de seconde) toute image imprimée sur la rétine. Il suffit donc de faire défiler un nombre suffisant d'images par seconde, pour que l'œil ne se rende pas compte qu'il s'agit d'images distinctes.

1.7.2 Les données vidéo

Dans le flux vidéo, les données sont hiérarchisées selon une manière bien précise. Une image est constituée de trois matrices U, V et Y où chaque élément de la matrice représente un pixel. A noter que les matrices U et V sont de dimensions plus petites que la matrice Y suivant le format utilisé. L'essentiel de l'information de l'image est stocké dans la matrice Y.

En effet, l'œil est plus sensible à la luminance que la chrominance. Plusieurs formats d'images existent et on les note selon les caractéristiques de leurs matrices YUV.
Chaque image est découpée en tranche (*slice*). Une tranche est constituée d'un ou plusieurs macroblocs (MBs) adjacents, ordonnés de gauche à droite. Ce sont des éléments importants pour la gestion des erreurs. Si une tranche contient une erreur, le décodeur passe alors immédiatement à la tranche suivante. Plus il y a de tranches dans une image, meilleur est le traitement des erreurs mais plus l'image prend de place également. Enfin, un MB est une matrice constituée de 4 blocs, chacun de taille 8 x 8 pixels.

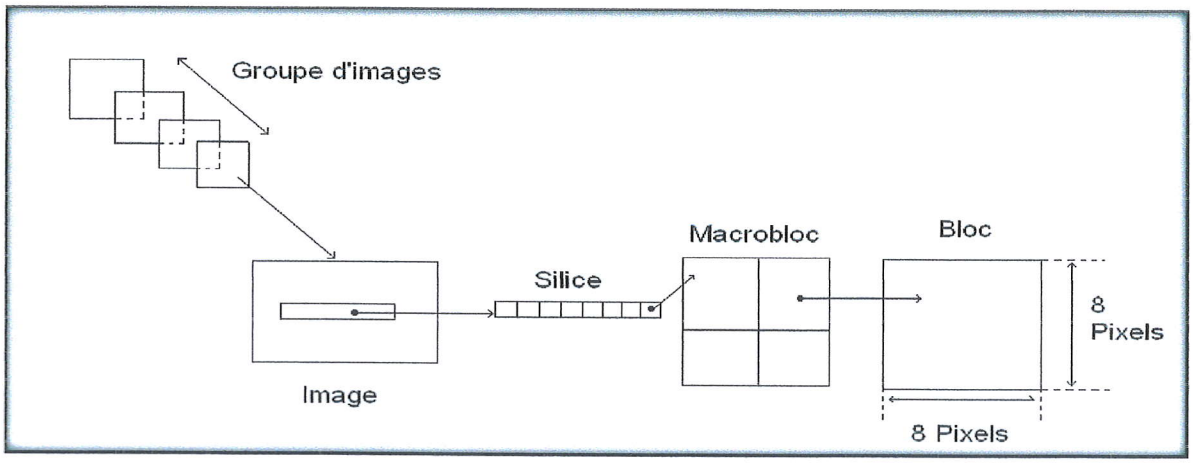


Figure 1.8 : Hiérarchie des données dans le flux vidéo

1.7.3 Transmission de la vidéo par paquets

Les flux vidéo ne sont pas transmis sur le réseau de distribution dans leurs formats bruts, tels qu'ils apparaissent au niveau de la tête de réseau après la phase de compression et d'encodage. Ils sont au contraire découpés en paquets, multiplexés entre eux et encapsulés avec des systèmes de correction d'erreurs en un flux spécifique. Cet unique flux découpé en

paquets de 188 octets intègre tous les éléments constitutifs nécessaires à la reconstruction de la vidéo.

Le traitement de l'information est hiérarchisé sous la forme de couches fonctionnelles, chacun d'entre elles correspondant à une fonctionnalité précise (couche application, couche transport, couche réseau…) et donc à un ou plusieurs protocoles spécifiques.
La figure Suivante montre un schéma générique d'un système de transmission de la vidéo par paquets [5]. Le transmetteur et le récepteur sont reliés à un réseau de transport, établissant la liaison entre eux.

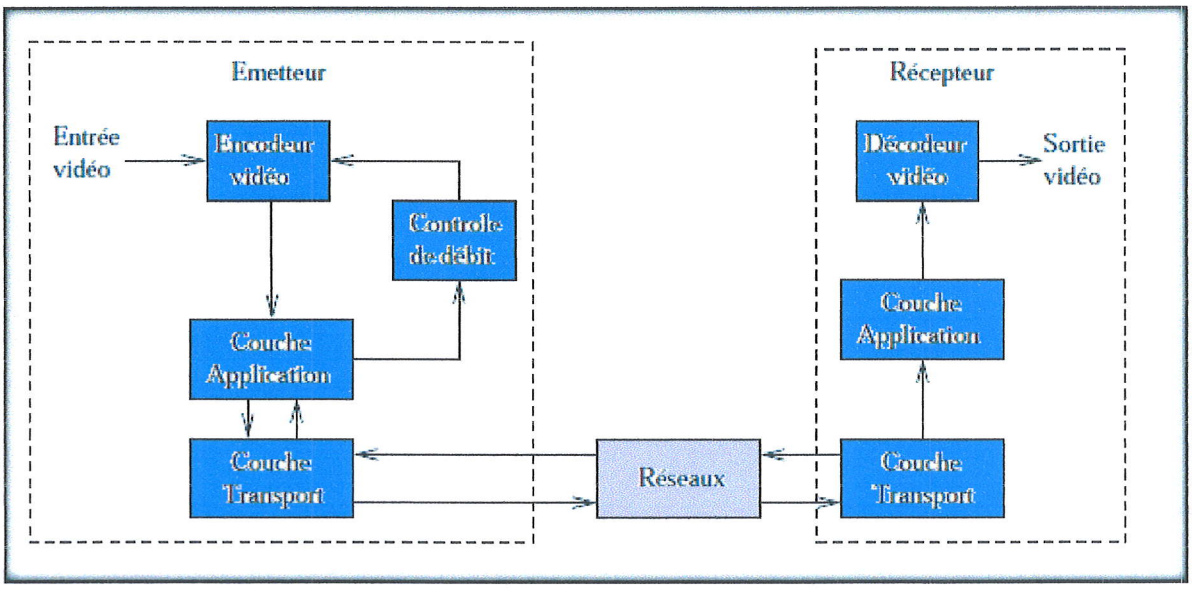


Figure 1.9 : Système de transmission de la vidéo par paquets.

Du côté du transmetteur, le codeur vidéo a la charge de compresser le signal vidéo, de formater les données codées en paquets et de les expédier à la couche transport, pour transmission. La figure 1.9 fait aussi abstraction des détails de l'infrastructure réseau puisque, en général, le transmetteur ne possède pas de tels détails. En effet, le réseau peut être hétérogène, comportant, par exemple, à la fois des liaisons filaires et des liaisons sans fil. En revanche, la couche application dispose en général d'un modèle intégrant les informations relatives au comportement du réseau vis-à-vis des données qui sont transmises. L'information la plus pertinente relative au canal de transmission est la probabilité de perte de paquets. Dans les applications à temps réel, le retard dans l'acheminement des paquets à destination peut aussi engendrer une perte de paquets.

Au niveau du récepteur, les paquets reçus sont traités au niveau de chaque couche, où les en-têtes propres au protocole de communication sont ôtés et les informations utiles envoyées vers le décodeur vidéo qui se charge de décompresser la vidéo codée et de l'afficher dans l'ordre et la synchronisation requis.

1.7.4 La compression vidéo

La compression vidéo est nécessaire pour la transmission des données vidéo numériques dans les réseaux à bande passante limitée d'aujourd'hui ainsi que pour les applications où le stockage constitue une limite.

Le principe fondamental de la compression vidéo est de réduire autant que possible les redondances d'informations dans les données sans que cela soit perceptible par l'œil humain. Il y a donc un compromis entre le taux de compression et la qualité de l'image. En fait, celle-ci devient de plus en plus médiocre avec l'augmentation du taux de compression.

Deux grandes méthodes de compression existent : la compression sans perte et la compression avec perte

- Dans le cas de la compression sans perte, les données décodées coté récepteur sont strictement identiques aux données codées au départ par l'émetteur
- Dans le cas de la compression avec perte, les données à la sortie du décodeur sont différentes par rapport à celles à l'entrée du codeur. C'est ce type de compression qui est utilisé en vidéo car on peut accepter des pertes d'informations qui ne sont pas toujours visibles à l'œil et qui se traduisent par de nets gains de compression.

1.7.5 Distorsion

La distorsion [6] désigne dans un appareil ou un canal de transmission, l'ensemble des modifications indésirables d'un signal qui ne soient ni un gain, ni une atténuation, ni un retard. D'une manière générale, on pourra dire que la distorsion désigne toute déformation du signal de sortie par rapport au signal d'entrée.

Différents types de distorsions peuvent être identifiés dans une vidéo décompressée :

- L'effet de bloc (blocking effect), comme son nom l'indique, est l'apparition de motifs en forme de blocs dans la séquence décompressée. Cet effet est du à la différence de quantification d'un bloc à l'autre (en général 8x8 pixels) dans les schémas de compression utilisant une DCT par bloc, qui introduit des discontinuités à la frontière

des blocs adjacents. L'effet de bloc est souvent la distorsion la plus frappante dans une vidéo décompressée à cause de la régularité et de l'étendu des motifs insérés.

- Le flou (blurring) se manifeste comme une perte des détails spatiaux, et comme une réduction de la finesse des contours. Il est du à la suppression des coefficients hautes fréquences, via une quantification grossière.
- Le color bleeding est une bavure des couleurs entre deux zones de chrominance fortement différentes. Il est le résultat de la suppression des coefficients hautes fréquences dans les composantes chromatiques. A cause du sous-échantillonnage de la chrominance, le « color bleeding » s'étend sur la totalité du macrobloc.
- L'« effet des fonctions de base DCT », est important lorsqu'un seul coefficient DCT est dominant sur un bloc. A des niveaux de quantifications élevés, le résultat est l'accentuation de l'image de la fonction de base DCT dominante, et la réduction de toutes les autres fonctions de base.
- Un effet « escalier » peut apparaitre sur les contours obliques. Cet effet est du au fait que les fonctions de base de la DCT sont plus adaptées à la représentation des contours horizontaux et verticaux. Les contours, dont l'orientation n'est ni horizontale ni verticale, nécessitent davantage de coefficients hautes fréquences pour obtenir une représentation précise. La quantification trop importante de ces coefficients introduit des irrégularités sur ces contours obliques (jagged).
- L'effet d'ondulation (ringing) est fondamentalement associé au phénomène de Gibbs (oscillation de reconstruction d'un signal discontinu avec une somme de signaux continus). Il est donc plus marqué sur les contours à fort contraste que sur les zones uniformes. Cet effet est un résultat direct de la quantification provoquant des irrégularités dans la reconstruction. L'effet de ringing se produit à la fois sur la luminance et la chrominance.
- Les contours fantomes sont une conséquence du transfert par compensation de mouvement de la discontinuité de la frontière d'un bloc, induite par un effet de bloc, depuis une image de référence vers une image prédite.
- Le mouvement saccadé (jagged motion) peut être du à une mauvaise estimation de mouvement. Les estimateurs de mouvement fonctionnant par bloc sont plus efficaces lorsque les mouvements de tous les pixels d'un macrobloc sont identiques. Lorsque l'erreur résiduelle d'estimation de mouvement est importante, elle peut être quantifiée grossièrement.

- La quantification de mouvement est souvent réalisée seulement sur la luminance et le même vecteur est utilisé pour les composantes de chrominance. Il peut en résulter une « chominance mismatch » sur un macrobloc.
- L'effet « mosquito », est une distorsion temporelle observée principalement sur les zones faiblement texturées comme des fluctuations de la luminance (ou de la chrominance) autour des contours à fort contraste ou des objets en mouvement. Il est la conséquence de la variation du choix des paramètres de codage d'une même zone d'une scène dans les images successives d'une séquence.
- Le papillotement (flickering) apparait principalement dans les zones texturées. La texture des blocs de ces zones est compressée avec des pas de quantifications variant au cours du temps, ce qui a pour effet de créer des papillotements dans ces zones

1.8 La norme de compression H.264/AVC

Le Video Coding Experts Group (VCEG) de l'ITU-T et le Moving Picture Experts Groups (MPEG) de l'ISO/IEC ont développé conjointement le standard de codage vidéo H.264, aussi appelé MPEG-4 Advanced Video Coding (AVC). Ce standard a été publié pour la première fois en 2003 en tant que ITU-T Recommandation H.264 et ISO/IEC MPEG-4 Part 10 (Rec, 2003).

Le standard H.264 a été conçu pour répondre aux besoins et convenir aux moyens de l'industrie de la télécommunication, et plus généralement, des industries concernées par la compression vidéo, actuellement et dans un avenir rapproché. De plus, il a été élaboré pour apporter des contributions significatives par rapport aux standards précédents (H.261, H.263, MPEG-1, MPEG-2, MPEG-4) (ISO/IEC 11172, 1993; ISO/IEC 13818, 1995; ISO/IEC 14496-2, 2001; ITU-T Recommandation H.261, 1993; ITU-T Recommandation H.263, 1998).

Plus spécifiquement, les concepteurs du standard H.264 avaient les principaux objectifs suivants : améliorer le taux de la compression vidéo par un facteur d'au moins de 2 sans affecter significativement la qualité; obtenir une complexité qui tienne compte de l'état de l'art de l'industrie des semi-conducteurs; créer une interface simple et efficace pour supporter l'intégration de différents types de réseaux (Ethernet, LAN, ISDN, réseaux mobiles, etc.) et supports de stockage; avoir un ensemble d'outils et de profiles suffisamment large pour supporter un vaste éventail d'applications (vidéophonie, mobile, télévision, HD, etc.) et de conditions réseaux (perte et corruption de paquets, taille des paquets, etc.)[7].

1.8.1 Structure du codeur

Cette section présente la structure de la norme H.264. Avant de détailler le codec et les profils de compression, il est nécessaire de définir les nouveaux termes introduits par ce standard.

1.8.1.1 Terminologie

- **Une trame** (ou une frame) est une image d'une séquence vidéo (une trame correspond à une vidéo entrelacée et un frame à une vidéo progressive). Á chacune de ces images est assigné un numéro (signalé dans le flux binaire), ne correspondant pas forcément à l'ordre du décodage.
- **Les images de référence** sont des images précédemment codées et décodées qui pourront être utilisées pour le codage d'images suivantes (prédiction inter-frame). Ces images de référence sont organisées en deux listes. Une image, avant d'être codée, est quadrillée en blocs de pixels (appelés macroblocs) contenant 16x16 échantillons de luminance (information sur la luminosité) et deux fois 8x8 échantillons de chrominance (information sur la couleur).
- Il existe trois types de codage pour un **macrobloc** : Les macroblocs **I** sont obtenus par prédiction intra (codage sans image de référence) à partir d'échantillons décodés dans la tranche courante. Une prédiction est formée soit pour un macrobloc complet, soit pour chaque bloc 4x4 de luminance (et les blocs de chrominance associés) du macrobloc.
- Les macroblocs **P** sont obtenus par prédiction inter, c'est-à-dire à partir d'images de référence.
- Un macrobloc codé inter peut être divisé en partitions de macroblocs : des blocs de taille 16x16, 16x8, 8x16 ou 8x8 de luminance (et les blocs de chrominance associés). En choisissant la partition 8 x 8, chaque sous macrobloc peut être de taille 8x8, 8x4, 4x8 ou 4x4. La prédiction se fait depuis une image de référence prise dans une des deux listes.
- Les macroblocs **B** sont obtenus par prédiction inter à partir de plusieurs images de référence. Chaque partition de macrobloc peut utiliser une ou deux images de référence.
- Les macroblocs sont arrangés en tranches (slice en anglais), chaque tranche contenant un nombre de macroblocs compris entre un et l'intégralité des macroblocs de l'image.

Une tranche **I** ne contient que des macroblocs **I**, une tranche **P** peut contenir des macroblocs **I** et **P** et une tranche **B** peut contenir des macroblocs **B** et I. Il existe également des tranches SI et SP.

- **Un GOP** (ou groupe d'images) est un ensemble d'images constituant une séquence. Il commence par une image I, suivie d'une image P et/ou B. Le prochain GOP commence à l'image I suivante (et ainsi de suite jusqu'à la fin de la séquence).

1.8.1.2 Le codec H.264

Tout comme les autres standards de codage vidéo [8], H.264 ne définit pas un codec mais la syntaxe d'un flux binaire codé de la vidéo et la procédure de décodage de ce flux. En pratique, un codeur/décodeur conforme à la norme inclut les fonctionnalités décrites sur les figures 1.10 et1.11, Le codeur (figure 1.10) inclut deux chemins pour le flux de données, le chemin "avant" (de gauche à droite) et le chemin de reconstruction (de droite à gauche).
Sur le chemin avant de la figure 1.10, une trame ou une image Fn est partitionnée en macroblocs, une prédiction P est formée en fonction des échantillons reconstruits. Dans le mode intra, P est obtenue à partir d'échantillons de la tranche courante F'_n ayant déjà été codée, décodée et reconstruite. En mode inter, P est obtenue par prédiction et compensation de mouvement à partir d'une ou deux images de référence F'_{n-1}.

La prédiction P est ensuite soustraite au bloc courant, générant un bloc résiduel Dn. Celui-ci est alors transformé (T) et quantifié (Q) afin de produire X, un ensemble de coefficients de transformation quantifiés. Ils sont réarrangés puis transmis au codeur entropique. Les coefficients obtenus et les informations nécessaires au décodage (mode de prédiction, table de quantification, vecteurs de mouvement, ...) sont codés en un flux binaire comprimé qui est passé au canal (NAL) pour transmission ou stockage. En plus du codage et de la transmission, le codeur exécute également un décodage suivant le chemin de reconstruction. Cette étape est nécessaire pour fournir une référence pour les futures prédictions. Les coefficients X subissent un rééchelonnement (Q−1) et une transformée inverse (T−1) pour produire un bloc D'n et créer le bloc reconstruit uF'n, c'est-à-dire, la version décodée du bloc original. Enfin, un filtre peut être appliqué afin de réduire les effets de bloc. L'image de référence reconstruite est ainsi créée par une série de blocs F'n.

Au niveau du décodage (figure 1.11), le même processus est appliqué. Les macroblocs reçus par le NAL son décodés, réarrangés pour obtenir les coefficients X. Rééchelonne et laissé transformée inverse sont appliqués pour obtenir D'n (identique au D'n du schéma

d'encodage). Grâce aux informations d'entête du flux binaire, le décodeur crée la même prédiction P que celle du codeur, ajoute D'n pour produire uF'n. Celle-ci est alors filtrée pour générer chaque bloc décodé F'n. Cette description du codage/décodage H.264 se veut générale. En effet, plusieurs possibilités existent dans l'utilisation de ces fonctions, ce sont les profils.

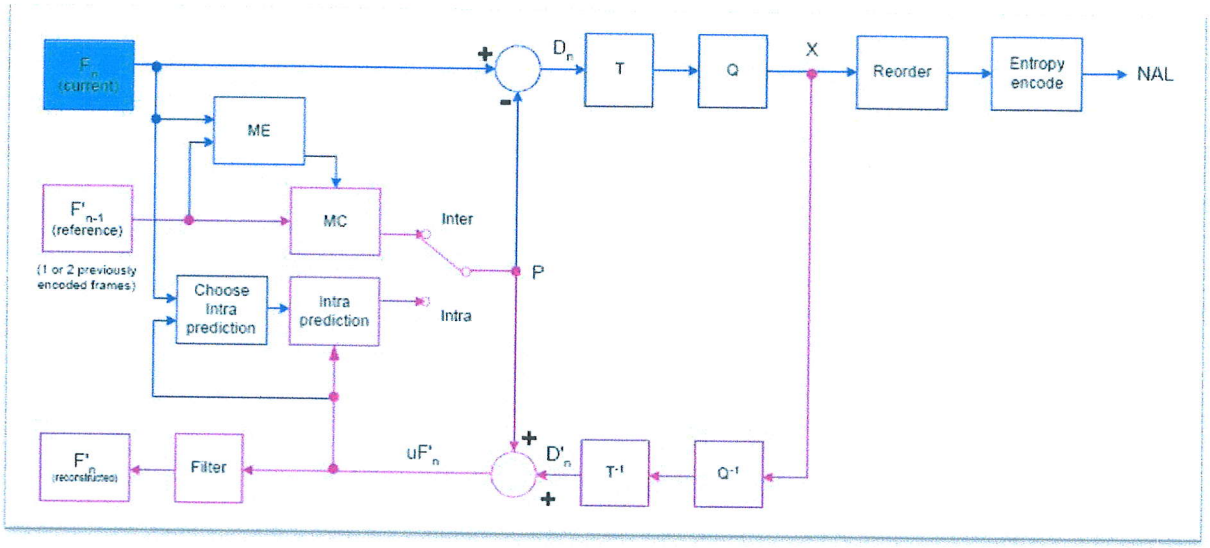


Figure 1.10 : Schéma du codeur H.264

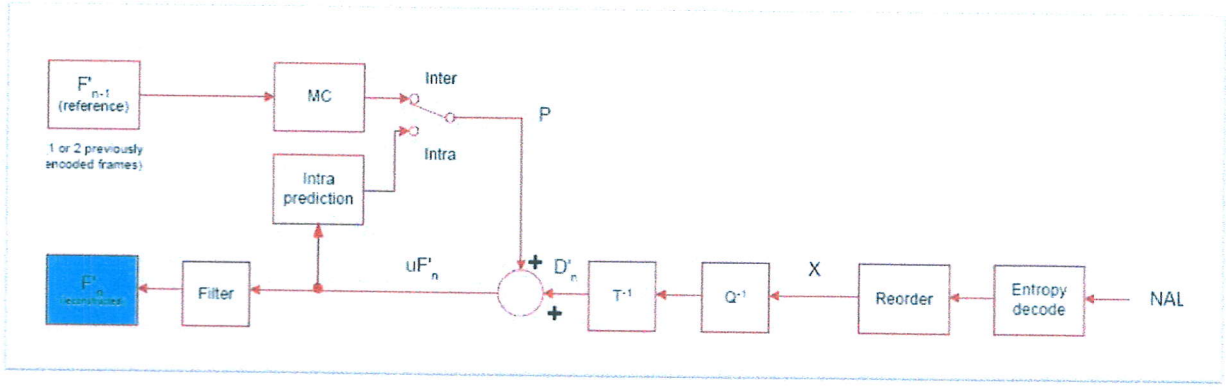


Figure 1.11 : Schéma du décodeur H.264

1.8.1.3 Les profils et niveaux

Le standard H.264 propose trois profils (baseline profile, main profile, extended profile) et onze niveaux afin d'établir les sites de conformité. Les premiers indiquent un ensemble de codage pouvant être utilisé pour générer un flux compatible. Les seconds imposent des contraintes à certains paramètres clés du flux (par exemple la taille maximale d'une image).

Ces différents profils permettent de produire un codage adapté à de multiples applications. Ainsi le profil de base est utilisé pour la vidéotéléphonie, la vidéoconférence et la communication sans fil. Le profil principal peut être utilisé pour la télévision et le stockage tandis que le profil étendu est destiné à la diffusion de vidéos en continu, au fur et à mesure du téléchargement. Néanmoins, chaque profil est suffisamment flexible pour supporter une large gamme d'applications.

1.8.2 Espaces de couleur

En réalité, le nombre de couleur est infini mais l'homme ne peut percevoir plus que 2 millions de couleurs différentes, il est donc essentiel de disposer d'un moyen de classer voire reproduire chaque couleur et ceci pour pouvoir se limiter au niveau de la vision humaine (et donc gagner en terme de compression) [9].

Les deux systèmes de représentation les plus fréquemment rencontrées dans les systèmes multimédia sont : RGB (Red, green, Blue) et YUV (Y Cb Cr).

1.8.2.1 Espace RGB

Le RGB est un codage destiné aux images couleur, c'est une représentation dans un espace tridimensionnel de la valeur d'intensité lumineuse du pixel. Un pixel sera donc codé sur 3 octets, 1pour le rouge (red), 1pour le vert (green), 1pour le bleu (blue) ; ce qui donne 256 niveaux pour chaque composante et donc 16 million de couleurs possibles.

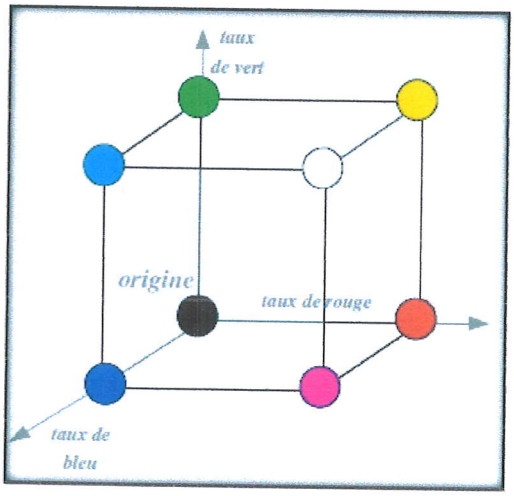


Figure 1.12 : Représentation graphique du codage RGB

1.8.2.2 Espace YUV

Historiquement et techniquement, les signaux vidéo couleurs sont représentés dans un autre mode de représentation, on utilise un system nommé YUV, ou Y représente la luminance et U, V les composantes rouge et bleue (chrominance). Ces 3 informations permettre de restituer au final les composantes RVB et apporte l'avantage de permettre une compression de couleurs et de fiabiliser le transport du signal vidéo(l'essentiel de l'information de l'image est stocké dans la matrice Y, car l'œil est plus sensible à la luminance), mais surtout la partie Y(luminance)permet une totale compatibilité avec les anciens équipements noir et blanc ,ce qui a permis le déploiement de la diffusion couleur alors que la totalité des foyers étaient alors équipées de téléviseurs noir et blanc.

- **Formats d'échantillonnage YUV**

Il exite 3 modèles d'échantillonnage de l'espace Y,Cb et Cr supportés par la norme MPEG-4 et H.264 [10] :

- ✓ **4 :4:4** chaque composante est codée de la même manière, il n'y a pas de sous échantillonnage. Ce format est utilisé lorsque toute la qualité de la vidéo doit être gardé », c'est-à-dire pour le stockage ou la poste-production en studio, le cinéma numérique.
- ✓ **4 :4:2** On ne garde qu'une ligne sur deux pour les composantes de couleur. Seulement la moitié de l'information de couleur est gardée. C'est un format de qualité studio.
- ✓ **4 :2:0** On ne garde qu'une ligne sur deux et une colonne sur deux pour les composantes de couleur. Un quart de l'information de couleur est conservé. C'est le format le plus utilisé pour la vidéo grand public.

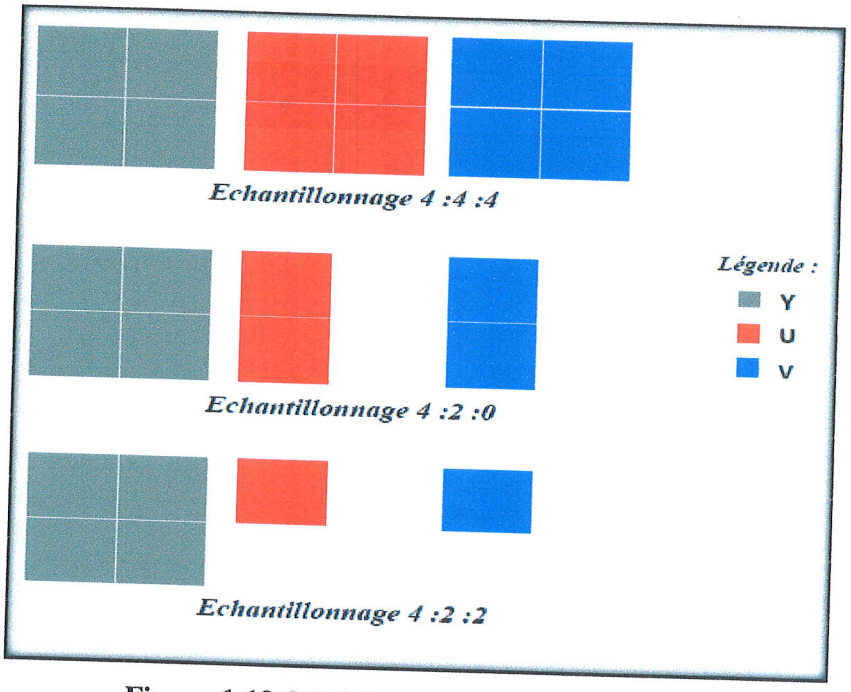


Figure 1.13 Modèles d'échantillonnage YUV

1.8.3 Mécanismes de prédiction

Dans le H.264, la prédiction est le cœur du processus d'encodage. Le codage prédictif est « une technique qui prévoit la valeur probable d'un élément en se basant sur les valeurs antérieures et qui code la différence entre les deux valeurs, passées et présentes pour permettre la compression effective ».

Cette technique permet donc de réduire la redondance d'information en encodant seulement la différence entre deux valeurs. On parle de redondance temporelle quand il s'agit d'information qui se répète d'une trame à l'autre et de redondance spatiale pour de l'information liée à l'intérieur d'une même trame. H.264 réduit ces deux types de redondance par l'emploi de différentes méthodes de prédiction intra et Inter [8].

1.8.3.1 Prédiction inter-image

La prédiction inter (type P) est utilisée pour réduire la corrélation temporelle avec l'aide de l'estimateur et le compensateur de mouvement. Dans H.264, l'image courante peut être subdivisée en macro-blocs ou en blocs plus petits. Pour le macro-bloc (16x16), il peut y avoir jusqu'à 7 modes : 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 et 4x4. Cependant, le choix de la taille des partitions dépend essentiellement des caractéristiques de la séquence d'entrée. En

général, les grandes partitions sont choisies pour couvrir les zones homogènes de la même image et les partitions de petites tailles peuvent être utilisées pour des zones avec beaucoup de détails.

Le processus de la prédiction inter, peut faire la segmentation de la représentation des mouvements aussi petite que la taille des blocs 4x4, en utilisant des vecteurs de mouvement avec une précision d'un quart de l'échantillon pour la luminance. La compensation en mouvement des sous-pixels peut donner de meilleures performances de compression que la compensation entière de pixel.

La compensation de mouvement utilise des similitudes qui existent entre l'image courante d'entrée et l'image encodée précédemment. Cette image déjà codée appelée image de référence qui est en fait une image déjà reconstruite.

1.8.3.2 Prédiction intra-image

Contrairement à la prédiction inter, la prédiction intra- image (type I) n'utilise pas d'images de référence mais uniquement l'image courante. Il s'agit donc d'exploiter la redondance spatiale présente dans toute image. En effet, le principe de base est que dans un certain voisinage, la structure de l'image varie peu globalement. Bien sûr, ceci dépend de la taille de l'échantillon considéré, c'est pourquoi chaque taille de bloc correspond à un traitement spécifique. De fait, pour la luminance, la prédiction peut être réalisée sur des blocs 4X4 ou des macro-blocs 16X16. De plus, bien que la luminance et la chrominance n'aient pas la même résolution, la chrominance n'est codée que par blocs 8X8. Un bloc prédit est donc construit à partir de ces voisins déjà codés et reconstruits. Il est ensuite soustrait au bloc original et le bloc différence est codé et transmis. Toute la procédure de prédiction doit ensuite être insérée au flux codé, ce qui nécessite de l'information supplémentaire dont l'importance ne doit pas être négligée.

1.8.4 Transformation et Quantification

Après prédiction et soustraction de la prédiction au bloc original, le résidu est transformé puis quantifié. Le codeur H.264 utilise trois transformées selon le type de résidu à coder [8]: une transformée pour les matrices 4x4 des coefficients des composantes continues de luminance des macroblocs intra (prédiction en mode 16x16), une transformée sur les matrices des coefficients des composantes continues de chrominance (pour tous les macroblocs) et une transformée en cosinus discrète (DCT) pour tous les autres macroblocs

4x4. La transformée inverse est réalisée par des opérations exactes sur des entiers pour éviter les discordances.

La quantification des coefficients de la transformée entraîne des pertes mais produit une compression importante des données. Le codeur H.264 utilise une quantification scalaire en calculant l'arrondi d'un nombre fractionnel entier. Pour chaque macrobloc, le paramètre de quantification (QP) sélectionne un des 52 quantificateurs. Les quantificateurs sont disposés de manière à obtenir une augmentation approximative de 12,5% de la taille du pas de quantification lorsque QP augmente de 1.

1.8.5 Codage Entropique

Les données arrivant à l'unité de codage entropique sont essentiellement les données résiduelles issues de la quantification mais peuvent être également des entêtes, des vecteurs de mouvement, des méthodes de prédiction, des index des images de référence, etc... Après une étape de réarrangement des données dans un tableau, le codeur H.264 dispose de plusieurs méthodes de codage entropique, pour les éléments syntaxiques et pour les coefficients de transformée quantifiés [8].

1.9 Conclusion

Le but de ce chapitre est de donner un aperçu sur la vidéo dans les réseaux sans fil de façon à comprendre les concepts de base et quelque explication sur les réseaux sans fil et les protocoles de communication. De toute évidence ce chapitre couvre l'ensemble des fonctions de la couche de modèle TCP/IP ainsi que les deux types de canaux, le canal gaussien et le canal rayleigh, aussi ce chapitre couvre la transmission vidéo sur les réseaux sans fils.

Dans la deuxième partie de ce chapitre, nous avons présenté le standard de codage vidéo H.264. Nous avons abordé la séquence d'encodage H.264 et discuté brièvement de ses principales étapes (méthode de prédiction, transformés, quantification, codage entropique). Par la suite, nous avons décrit les principaux mécanismes de prédiction, intra et inter, définis par la syntaxe de H.264.

Chapitre 2 : Evaluation de la Qualité Vidéo

2.1 Introduction

De nos jours, la vidéo numérique est présente pratiquement partout, c'est pourquoi la qualité visuelle est arrivée au centre des préoccupations des professionnels. L'évaluation de la qualité finale de la vidéo est un élément clé de la performance du codage et elle est essentielle vu les dégradations que subit la vidéo au cours de la compression nécessaire à sa transmission. Nous admettons que la qualité finale est perçue par des spectateurs humains et que c'est leur qualité de perception visuelle qui est en jeu. De ce fait les techniques de traitement d'images et de vidéos ont intérêt à prendre en compte et à exploiter la manière dont l'homme perçoit son environnement visuel. Il est donc nécessaire de mettre en œuvre des évaluations subjectives de qualité effectuées par des observateurs, qui de préférence n'ayant pas de lien avec l'étude de la qualité vidéo.

Dans ce domaine, nous distinguons deux catégories de méthodes : les méthodes subjectives et les méthodes objectives. Les méthodes subjectives sont les plus proches de la réalité, car elles consistent à faire évaluer la qualité par un groupe d'observateurs. Les méthodes objectives sont basées pour la plupart sur la mesure de différence entre la vidéo originale et la vidéo compressée. Cette mesure peut utiliser un simple calcul mathématique comme le PSNR.

Figure 2.1: Figure d'illustration d'EvalVid

2.2 L'outil Evalvid

Le Framework Evalvid [11] est un ensemble d'outils qui évaluent la qualité de la vidéo transmise sur un réseau de communication réel ou simulé. Il permet de mesurer des paramètres de la qualité de service du réseau. Actuellement H.264, MPEG-4 et H.263 sont supportés. Il est destiné aux chercheurs qui souhaitent évaluer leurs conceptions de réseau ou des configurations en termes de qualité vidéo perçue par l'utilisateur. Il est correctement utilisé avec le simulateur de réseau NS-2.

2.2.1 L'architecture d'Evalvid

Cette section présente les outils du framework Evalvid, décrit leurs objets et leurs utilisations et montre des exemples de résultats obtenus. Par ailleurs les sources de fichiers et codecs vidéo échantillons sont donnés.

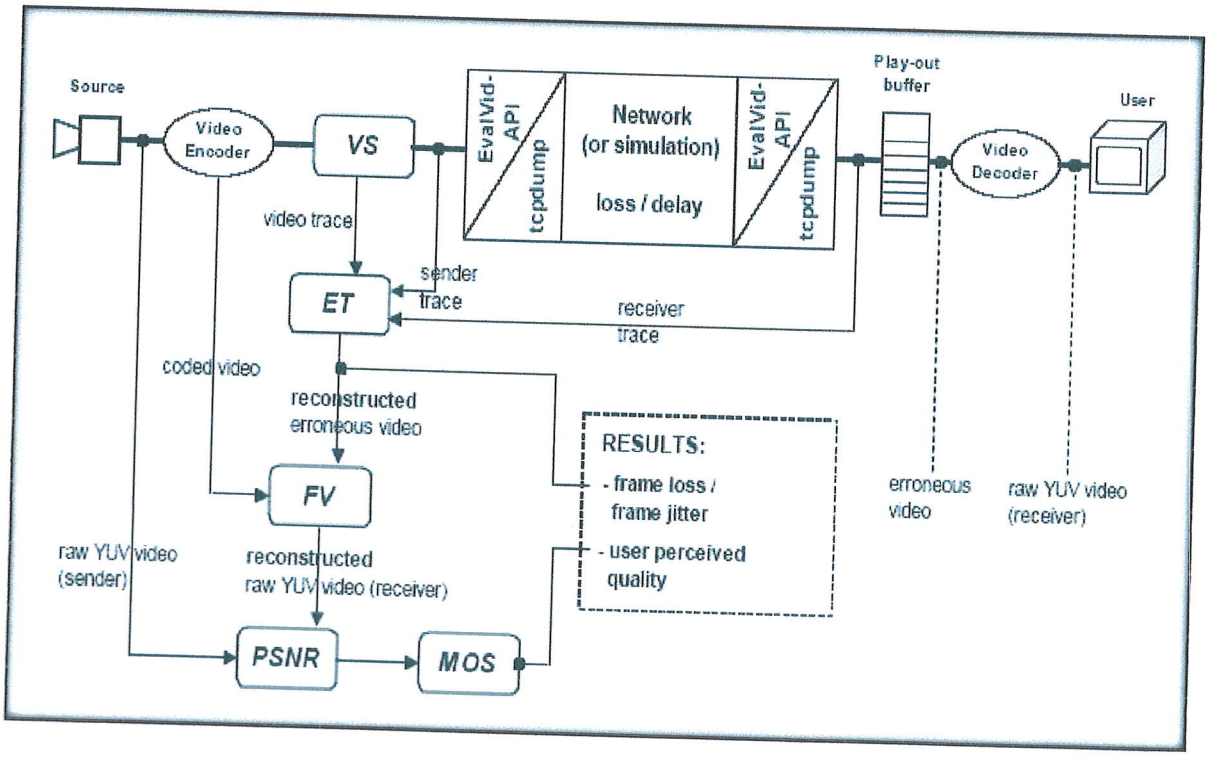


Figure 2.2 : L'architecture d'Evalvid

2.2.2 Fichiers et structures de données

Au début une source vidéo est nécessaire [11]. Les fichiers vidéo bruts (non codés) sont généralement stockés dans le format YUV, car ceci est le format d'entrée préféré de

plusieurs encodeurs vidéo disponibles. Ces fichiers peuvent être obtenus à partir de différentes sources, ainsi que les codecs MPEG-4 gratuits.

Une fois les fichiers vidéo codés (flux binaires) existent, les fichiers de trace sont produits à partir d'eux. Ces fichiers de trace contiennent toutes les informations pertinentes pour les outils d'Evalvid. Les outils d'évaluation fournissent des routines pour lire et écrire ces fichiers de trace et utilisent une structure de données centrale contenant toutes les informations nécessaires pour produire les résultats souhaités.

2.2.3 VS (Video Sender : Emetteur de la Vidéo)

Il génère deux fichiers de trace nécessaires pour l'évaluation ultérieure de la qualité de la vidéo.

- Fichier de trace de l'émetteur
- Fichier de trace de la vidéo

Pour les fichiers vidéo MPEG-4, un analyseur a été développé sur la base de la norme vidéo MPEG-4 part 10. Cela permet de lire toute la vidéo MPEG-4 produite par un encodeur compatible. Le but de VS est de générer un fichier de trace a partir du fichier vidéo encodé. Optionnellement, le fichier vidéo peut être transmis via UDP (si le système étudié est une configuration réseau). Les résultats produits par VS sont deux fichiers de trace qui contiennent des informations sur chaque frame du fichier vidéo et chaque paquet généré pour la transmission (Tableau 2.1 et 2.2).

Tableau 2.1 : Fichier de la trace de la vidéo.

Frame Number	Frame Type	Frame Size	Number of UDP-packets	Sender Time
0	H	29	1 segment at	33 ms
1	I	3036	4 segments at	67 ms
2	P	659	1 segment at	99 ms
3	B	357	1 segment at	132 ms
4	B	374	1 segment at	165 ms
...	...	...	...	...

Tableau 2.2 : Fichier de la trace de l'émetteur

Time stamp (sec)	Packet ID	Packet Type	Payload Size (bytes)
0.033333	0	udp	29
0.066666	1	udp	1000
0.066666	2	udp	1000
0.066666	3	udp	1000
0.066666	4	udp	36
0.099999	5	udp	659
0.133332	6	udp	357
0.166665	7	udp	374
...	...	...	...

Ces deux fichiers de trace représentent ensemble une transmission vidéo complète (du côté de l'expéditeur) et contiennent toutes les informations nécessaires pour d'autres évaluations par Evalvid. Avec VS, ces fichiers de trace peuvent être générés pour différents fichiers vidéo et avec différentes tailles de paquets, et peuvent ensuite être introduites dans la boîte noire du réseau (par exemple : une configuration réseau simulée).

2.2.4 ET (Evaluate Trace : Évaluer la Trace)

Le cœur du Framework d'évaluation est un programme appelé ET (évaluer les traces) [11]. Ici se produit le calcul des taux de perte de paquets et de trames et de délais/gigue. Pour le calcul de ces données seulement les trois fichiers de trace sont nécessaires tant que toutes les informations nécessaires sont incluses pour effectuer le calcul de la perte et de la gigue. Le calcul de la perte est assez facile, compte tenu de la disponibilité des identifiants uniques pour les paquets. Avec l'aide du fichier de trace de vidéo, on affecte à chaque paquet un type. Chaque paquet possédant un type qui ne figure pas dans la trace de récepteur est compté perdu.

Les pertes de frames sont calculés par rechercher n'importe qu'elle frame et verifier qui sont ses segments (paquets) perdus. Si le premier segment de frame est parmi les segments perdus, le frame est compté perdu parce que le décodeur vidéo ne peut pas décoder un frame dans lequel la première partie est manquante.

2.2.5 FV - Fix Video

L'évaluation de la qualité de la vidéo numérique est effectuée trame par trame. Cela signifie que le nombre de trames nécessaire de côté du récepteur est exactement le même du

côté d'émetteur. L'outil de FV est nécessaire uniquement si le codec utilisé ne peut pas produire les trames perdues.

2.3 Les fonctionnalités d'Evalvid

2.3.1 Détermination de paquet et frame perdus

2.3.1.1 Perte de paquet

Ils sont généralement calculés sur la base des identificateurs de paquets Ids [11]. Dans les mesures, les identifiants de paquets sont souvent prises à partir d'IP, qui fournit un identifiant de paquet unique. Le paquet unique ID est également utilisé pour annuler l'effet de la réorganisation. Dans le cadre de la transmission vidéo, il est non seulement intéressant de voir comment beaucoup des paquets se sont perdus, mais aussi le type de données dans les paquets. Par exemple, le codec MPEG-4 définit quatre types différents de frames (I, P, B, S) et également quelques en-têtes génériques. Puisqu'il est très important pour les transmissions visuelles dans lesquelles du genre de données obtient perdu (ou pas) il est nécessaire de distinguer les différents genres de paquets.

L'évaluation des pertes de paquet devrait être faite sur le type (type, en-tête frame) dépendant. La perte de paquet est définie dans l'équation (2.1). Elle est exprimée en pourcentage.

$$\textbf{Paquets Perdus } PL_T = 100\frac{nt_{recv}}{nt_{sent}} \quad \text{Ou:} \qquad (2.1)$$

T: le type de données dans le paquet (entête, P, B, S)

nt_{sent}: Nombre de paquets de type T envoyé

nt_{recv}: Nombre de paquets de type T reçu

2.3.1.2 Perte de frame

Une frame vidéo [11] (réellement étant une image codée simple) peut être relativement importante, non seulement dans le cas de vidéos de débit binaire variables, mais aussi dans les vidéos de débit binaire constant, puisque le terme constant applique à une moyenne courte de temps.

Il est possible et probable que certaines trames sont plus grandes que l'unité de transfert maximum (MTU) du réseau. Ceci est la taille maximum de paquet pris en charge par

le réseau (par exemple Ethernet = 1500 et 802.11n WLAN = 2312 octets). Ces trames doivent être segmentées en paquets plus petits pour adapter le réseau MTU. Cette segmentation possible des frames présente un problème pour le calcul des pertes de frames.

En principe, le taux de perte de frame peut être dérivé du taux de perte de paquets (paquets signifie toujours paquet IP ici). Mais ce processus dépend un peu sur les capacités du décodeur vidéo réel en cours d'utilisation, car certains décodeurs peuvent traiter une trame même si certaines parties sont manquantes et certains ne peuvent pas.

Si le premier paquet est manquant, le frame ne peut presque jamais être décodé. Ainsi, les capacités de certains décodeurs doivent être prises en compte pour calculer le taux de perte de frame. Elle est calculée séparément pour chaque type de frame.

$$\textbf{Frames perdus } FL_T = 100\frac{nt_{recv}}{nt_{sent}} \quad \text{Ou:} \qquad (2.2)$$

T: le type de données dans le paquet (entête, I, P, B, S)

nt_{sent}: Nombre de paquets de type T envoyé

nt_{recv}: Nombre de paquets de type T reçu

2.3.2 Détermination de délai et de la gigue

Dans les systèmes de transmission vidéo non seulement la perte réelle est importante pour la qualité visuelle perçue, mais aussi le retard de frame et la variation du retard, habituellement désigné sous le nom de frame gigue. Les vidéos numériques sont toujours constituées de frame qui doit être affichées à un taux constant. L'affichage d'une image avant ou après les résultats de temps définis dans "jerkiness" est un problème résolu par des tampons play-out. Ces tampons ont pour but d'absorber la gigue introduite par le réseau.

Il est évident qu'un tampon de play-out assez grand peut compenser tout montant de la gigue. Dans le cas extrême, le tampon est aussi grand que la totalité de la vidéo et l'affichage ne commence pas jusqu'à ce que le dernier frame soit reçu. Cela éliminerait toute gigue possible au prix d'un retard supplémentaire de la durée de transmission. L'autre extrême serait un tampon capable de contenir exactement un frame. Dans ce cas, aucune gigue ne peut être éliminée, mais pas de retard supplémentaire est introduit.

La définition formelle de la gigue telle qu'elle est utilisée dans ce Framework est donnée par les équations (2.3) à (2.6).

$$\textbf{Temps inter packet} \quad it_{po} = 0 \qquad (2.3)$$

$$it_{pn} = it_{p_n} - it_{p_{n-1}} \qquad \text{Ou :}$$

t_{pn}: Horodatage du dernier segment de la trame numéro n

$$\textbf{Temps inter frame} \quad it_{Fo} = 0$$

$$it_{Fn} = it_{F_n} - it_{F_{n-1}} \qquad \text{Ou :} \qquad (2.4)$$

t_{Fn} : Horodatage du dernier segment de la trame numéro n

$$\textbf{Gigue } j_p = \frac{1}{N}\sum_{i=1}^{N}(it_i - it_N)^2 \qquad \text{Ou:} \qquad (2.5)$$

N: nombre de paquet

it_N:Le temps moyenne inter-paquets

$$\textbf{Gigue } j_F = \frac{1}{M}\sum_{i=1}^{M}(it_i - it_M)^2 \qquad \text{Ou:} \qquad (2.6)$$

M: nombre de paquet

it_N:Le temps moyen inter-paquets

2.3.3 Évaluation de la qualité de la vidéo

Pour évaluer la qualité d'une vidéo [12], différentes approches ont été adoptées dans la littérature dont la majorité avait comme objectif l'évaluation d'une version dégradée de la vidéo originale plutôt que l'évaluation d'une version améliorée.

Il existe essentiellement deux approches pour mesurer la qualité de la vidéo, l'approche subjective ou des observateurs humains évaluent la qualité de la vidéo. Elle consiste à demander à un groupe de personnes d'attribuer une note de qualité de la vidéo qu'ils utilisent selon leur degré de satisfaction. Et l'approche objective ou un algorithme

appliqué à une vidéo donnée lui attribue une valeur ou une note. Les performances des mesures objectives sont mesurées par rapport aux résultats des tests subjectifs.

2.3.3.1 Méthodes subjectives

La mesure subjective qui fait intervenir des personnes pour juger la qualité des images, implique des observateurs humains qui ont comme tache de donner une note de qualité a une image ou une vidéo donnée lors d'une séance de test. La note attribuée à l'image ou à la vidéo et choisie parmi une échelle définie a priori.

Les observateurs subissant ce test peuvent être des « experts » c'est à dire des personnes travaillant dans le domaine du traitement d'images ou des observateurs « naïfs » c'est -à-dire des personnes n'ayant aucune expérience dans le domaine du traitement d'images ou vidéos.

Parmi les protocoles explicites de test vidéo, on distingue [13]:

- les méthodes à stimulus simple :
  - Discret : des échantillons de test, indépendants ou non, sont projetés et jugés par les observateurs.
  - Continu : une vidéo continue est projetée et les mesures sont prises à des instants donnés de cette vidéo.
- les méthodes à stimulus double :
  - Comparaison par rapport à une référence : deux images sont présentées à l'observateur. La première est considérée comme la référence, la seconde est jugée par rapport à la première.
  - Comparaison entre deux scènes de qualité différentes : la référence et l'image dégradée sont projetées dans un ordre aléatoire et les deux images sont jugées. L'analyse des résultats se fait par rapport à la différence de notation entre les images.
- les méthodes comparatives : l'image de référence et l'image dégradée sont présentées en même temps sur des écrans similaires et sont jugées en absolu (une note à chaque image) ou en relatif (une image est meilleure ou moins bonne que l'autre).

❖ **Avantages et Inconvénients :**

- Les résultats obtenus sont valides pour des vidéos compressées ou non.

- Une moyenne des mesures de plusieurs observateurs est un indicateur valide quant à la qualité globale des vidéos ou images.

Mais :

- Les méthodes et critères de tests sont nombreux.
- La configuration matérielle (conditions réelle) est lourde et difficilement reproductible
- de nombreux observateurs doivent être sélectionnés et surveillés.
- La complexité générale est énorme et pose surtout des problèmes de temps.

Dans la famille des méthodes subjectives, nous avons cités : le Mos et DSCQS:

➢ **MOS *(Mean Opinion Score)***

Une **Note d'opinion moyenne** (*Mean Opinion Score* (en) - MOS en abréviation) est une note donnée à un codec audio pour caractériser la qualité de la restitution sonore. La note peut varier entre 0 (très mauvais) et 5 (excellent, comparable à la version d'origine). Il est défini par l'UIT-T dans la norme « P.800 : Méthodes d'évaluation subjective de la qualité de transmission ».

Le MOS donne une indication numérique de la qualité perçue des médias reçus après avoir été transmis et par la suite comprimé en utilisant des codecs. Le MOS est exprimé en un nombre, de 1 à 5, 1 étant le plus mauvais et 5 le meilleur. Le MOS est tout à fait subjectif, car c'est des figures basées qui résultent de ce qui est perçu par des personnes pendant les essais.

Tableau 2.3 : Note d'opinion moyenne

Score	Qualité	Affaiblissement
5	Excellent	Imperceptible
4	Good	Perceptible, mais ne gênant pas
3	Faire	Légèrement gênant
2	Poor	Gêner
1	Bad	Très ennuyant

Tableau 2.4: La Conversion PSNR à MOS

PSNR	MOS
>37	5 (Excellent
31 - 37	4 (Good)
25 - 31	3 (Faire)
20 - 25	2 (poor)
<20	1 (Bad)

- **DSCQS (Double Stimulus Continuous Quality Scale)**

C'est la méthode la plus utilisée parmi toutes les méthodes subjectives de mesure de qualité. L'observateur, suivant un processus DSCQS, visualise des vidéos deux à deux. Il donne son jugement sur chaque vidéo à partir d'une échelle linéaire allant de mauvais à excellent.

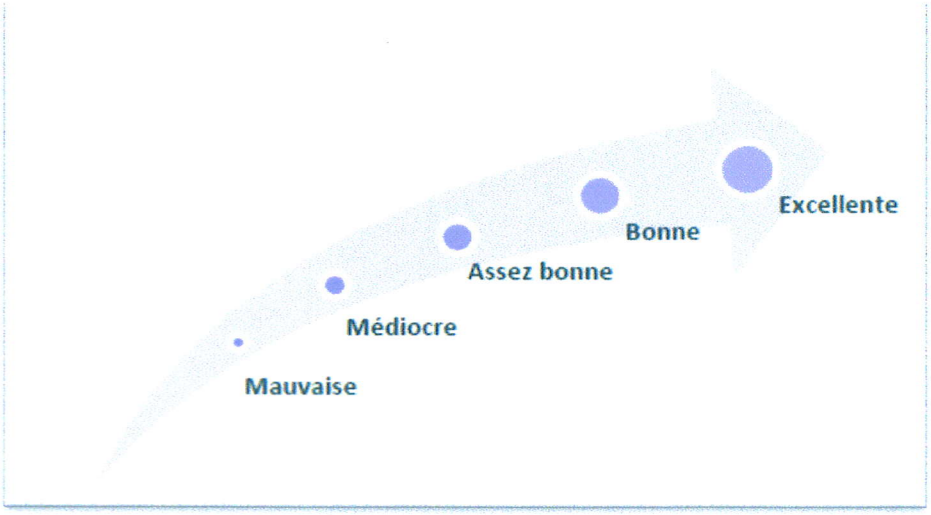


Figure 2.3 Echelle de catégories de qualité visuelle.

2.3.3.2 Méthodes objectives

Un critère objectif de qualité est un traitement particulier qui produit, à partir des données particulières d'une image ou d'une vidéo, une note de qualité de cette image ou de cette vidéo. Les données en entrée varient selon le type du critère de qualité qui peut être de types différents.

Les performances d'un critère objectif d'évaluation de qualité sont mesurées en comparant les notes de qualité données par ce critère, quand il est appliqué sur un group donné d'images ou de vidéos, aux notes fournies par des observateurs humains lors de tests subjectifs d'évaluation de qualité (sur ce même groupe d'images ou vidéos).
On peut diviser les indicateurs objectifs en deux catégories [13].

- ✓ La première catégorie part du principe que si le signal composant une image est différent / dégradé par rapport à celui d'une image de référence, alors l'image sera dégradée, avec une dépendance plus ou moins directe.
- ✓ La deuxième catégorie s'appuie sur un modèle de la perception humaine pour dégager des critères facilement distinguables par un ordinateur.

Les méthodes objectives sont assez largement utilisés car leur complexité algorithmique est très faible nous citons comme avantage majeur, la facilité de calcul. Et comme inconvénient, ils reflètent en général assez mal la qualité réelle de la vidéo ou de l'image c'est-à-dire qu'elles ne s'ajustent toujours pas bien avec la perception humaine.

Dans la famille des méthodes objectives, nous avons cités : le PSNR, BER, SSIM et TPP:

- **PSNR (Peak Signal-to-Noise Ratio)**

Le PSNR est une métrique utilisée pour évaluer la fidélité d'une image (par rapport à celle de référence) [12]. C'est un critère qui traite l'image purement comme un signal numérique. Il se base sur L'erreur quadratique moyenne MSE (Mean Squared Error) liée à la différence entre l'image dégradée et celle de référence :

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(x_i - y_i)2$$

Avec N le nombre de pixels dans l'image ou la vidéo, $x_i ety_i$ les ièmes pixels De l'image originale et dégradée, respectivement. Nous en, déduisons le rapport signal à bruit crête :

$$\mathbf{PSNR} = 10log_{10}\frac{L^2}{MSE}$$

Avec L la dynamique des valeurs des pixels (L = 255 pour un codage des pixels sur 8 bits). Une valeur de PSNR supérieure à 34 dB indique une assez bonne qualité d'image. MSE et PSNR sont largement utilisés car leur implémentation est facile et leur complexité est faible. Mais plusieurs tests ont démontré qu'ils ne sont pas corrélés suffisamment avec la perception humaine.

- **SSIM (The Structural SIMilarity)**

SSIM (The Structural SIMilarity) est une nouvelle méthode de mesure de la similitude entre deux images. L'indice SSIM peut être considéré comme une mesure de qualité de l'une des images comparées, à condition que l'image soit considérée comme d'une qualité parfaite. Il est conçu pour améliorer les paramètres traditionnels comme PSNR et MSE, qui se sont révélés incompatibles avec la perception de l'œil humain. La différence par rapport à d'autres techniques telles que mentionné précédemment MSE et PSNR est que ces approches estiment l'erreur absolue, d'autre part, SSIM est un modèle basé sur la perception qui considère la dégradation de l'image est le changement perçu dans l'information structurelle, tout en intégrant des phénomènes perceptifs importants, y compris les deux termes luminance de masquage et de contraste masquage.

Algorithme

L'indice SSIM est calculé sur les différentes fenêtres d'une image, la mesure entre deux fenêtres x et y de taille commune N × N est:

$$\mathbf{SSIM}(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}$$

μ_x: La moyenne de x

μ_y: La moyenne de y

σ_x^2 : La variance de x

σ_y^2: La variance de y

σ_{xy}: la convariance de x,y

c1=$(K_1L)^2$, c2=$(K_2L)^2$, deux variable destinées à stabiliser la division quand le dénominateur est très faible ;

L : la dynamique des valeurs des pixels, soit 255 pour des images codées sur 8 bits : K1=0.01 et k2=0.03 par défaut.

L'évaluation de la qualité d'image de cette formule est généralement appliquée sur la luminance, mais il peut aussi être appliqué sur la couleur (par exemple, RVB) ou (par exemple YCbCr). L'indice SSIM résultant est une valeur décimale comprise entre -1 et 1, et la valeur 1 est accessible uniquement dans le cas de deux ensembles de données identiques. En général, il est calculé sur de fenêtre de taille 8 × 8. La fenêtre peut être déplacée pixel par pixel sur l'image, mais les auteurs proposent d'utiliser uniquement un sous-groupe des fenêtres possibles pour réduire la complexité du calcul.

- **Taux d'Erreur Binaire(BER)**

Le taux d'erreur ou B.E.R., abréviation de l'expression anglaise *Bit Error Rate*, désigne une valeur relative au taux d'erreur mesuré à la réception d'une transmission numérique, relative au niveau d'atténuation et/ou de perturbation d'un signal transmis.
Ce phénomène survient également lors de l'échantillonnage (numérisation), de la lecture et de la sauvegarde des données (CD-R, DVD-R, disque dur, RAM...).

Ce taux détermine le nombre d'erreurs apparues avant la modulation et juste après la démodulation. Ce taux d'erreur ne tient généralement pas compte du codage des données sauf dans le cas d'une norme de transmission combinant modulation et correction d'erreur. La cause des perturbations, donc de l'augmentation du taux d'erreur, peut être multiple : équipement ou réseau défectueux, pointage incorrect d'une antenne, longueur des câbles, etc. Le taux d'erreur (BER) s'exprime en puissance négative. Par exemple, 10^{-3} signifie que l'on a en moyenne une erreur binaire pour mille bits transmis.

- **Taux de Perte de Paquet (TPP)**

La perte de paquets peut être causée par un certain nombre de facteurs, notamment la dégradation de signal sur le réseau en raison de multi-chemin, la perte de paquets à cause de la congestion du canal, les paquets corrompus rejetés en transit, matériel réseau défectueux. La formule :

Taux de perte = Nombre de paquet perdues/ Nombre total de paquets transmis

C'est le rapport entre le nombre d'octets émis perdus et la totalité des octets émis

2.4 Conclusion

Dans la première partie de ce chapitre nous avons présenté l'outil d'évaluation de la qualité vidéo Evalvid, en outre nous avons parlé aussi sur leurs composants et fonctionnalités. Dans la deuxième partie nous avons parlé sur les deux approches d'évaluation de la qualité vidéo (subjective et objective), où nous avons introduit quelque méthode subjective et objectif comme MOS et PSNR.

Chapitre 3 : Le Simulateur NS2

3.1 Introduction

Pour tester les performances d'une solution apportée à un problème de communication dans un réseau, il n'est pas toujours possible d'accéder aux infrastructures nécessaires en raison de leurs coûts élevés. De plus, les expérimentations réelles n'offre souvent pas une grande souplesse. Rappelons que les réseaux sans fils sont des réseaux qui englobent plusieurs unités mobiles qui se déplacent dans un territoire quelconque et dont le seul moyen de communication est l'utilisation des interfaces radio.

En effet, il serait très coûteux voire impossible de mettre en place un réseau à des fins de tests de certains critères. Pour remédier à ce problème, on a recours à la simulation qui met à la disposition de l'utilisateur un environnement d'expérimentation [14].

3.2 La simulation

3.2.1 Qu'est ce que la simulation ?

Simuler, c'est modéliser un système complexe, afin de prévoir son comportement dans le monde réel [14]. Il s'agit d'une approche permettant de représenter le fonctionnement d'un système réel constitué de plusieurs entités, de modéliser les différentes interactions entre elles, et enfin évaluer le comportement global du système et son évolution dans le temps.

Le recours à la simulation permet de contourner les limites de la complexité des modèles analytiques. Toutefois, il est nécessaire de bien identifier les caractéristiques du système afin de le représenter, le plus finement possible, par des modèles abstraits.

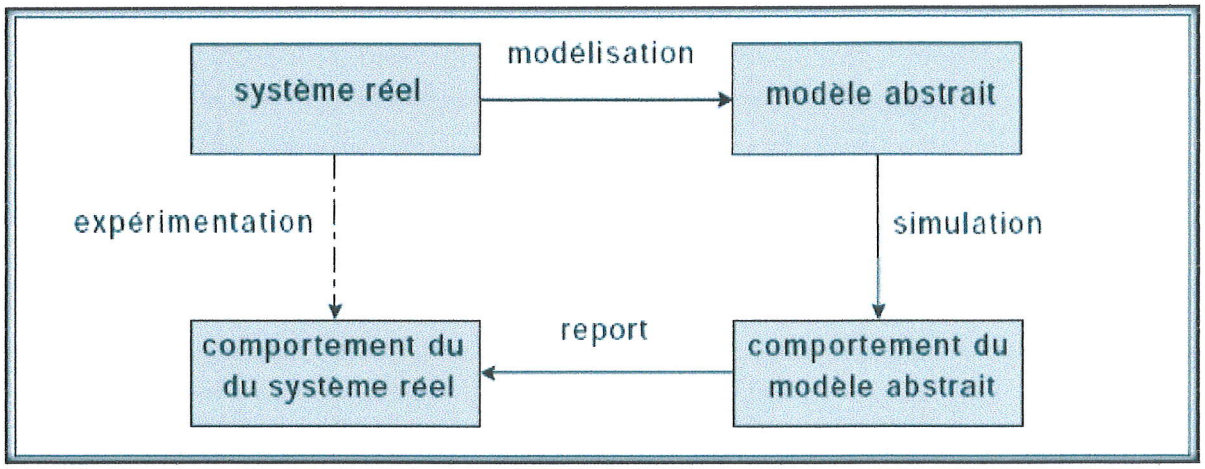


Figure 3.1 : cycle modélisation-simulation

La simulation d'un système réel devient nécessaire dès lors que les modèles analytiques deviennent, soit trop complexes en termes de calcul et de temps de résolution, soit trop simplifiés vis-à-vis de la réalité rendant, par ce fait, les résultats obtenus non représentatifs du comportement du système dans un environnement réel. Ainsi, la simulation peut s'avérer nécessaire dans les cas suivants :

- Les expériences sur système réel sont trop coûteuses en termes de ressources matérielles et Humaines
- Les expériences sur système réel ne sont pas reproductibles ni représentatives de tous les environnements possibles. Dans ce cas, la simulation permet de caractériser le comportement global du système pour différents environnements

3.2.2 Simulateur

Nous appelons simulateur [14] un programme qui met en œuvre un modèle de simulation par événements discrets. La tache première d'un simulateur est d'assurer que la chronologie des événements soit respectée. A chaque occurrence d'un événement, les actions qui sont associées à celui-ci sont exécutées. Les applications de la simulation sont innombrables. Parmi les domaines dans lesquels elle est le plus utilisée, on s'intéresse à la simulation des réseaux mobiles (réseaux sans fils). L'un des simulateurs de ce type est NS-2.

3.3 Généralités sur NS-2

3.3.1 Définition de NS-2

NS-2 signifie Network Simulator Version 2. NS est un outil logiciel de simulation de réseaux informatiques, il est parmi les simulateurs les plus utilisés dans les laboratoires de recherche, développé dans le cadre du projet VINT, ce dernier est un projet en cours de développement avec la collaboration de plusieurs acteurs (USC/ISI, Xerox parc, LBNL et UCB) dans l'objectif principal de construire un simulateur multiprotocole pour faciliter l'étude de l'interaction entre les protocoles et le comportement d'un réseau à différentes échelles.

Le projet contient des bibliothèques pour la génération de topologies réseau, des trafics ainsi que des outils de visualisation tels que l'animateur réseau NAM (network animator).
Le simulateur NS-2 actuel est particulièrement bien adapté aux réseaux à commutation de paquets et à la réalisation de simulations de grande taille (le test du passage à l'échelle). Il contient les fonctionnalités nécessaires à l'étude des algorithmes de routage unicast ou multicast, des protocoles de transport, de session, de réservation, des services intégrés, des protocoles d'application comme FTP. A titre d'exemple la liste des principaux composants actuellement disponibles dans NS-2 par catégorie est :

- application : Web, ftp, telnet, générateur de trafic (CBR...) ;
- transport : TCP, UDP, RTP, SRM ;
- routage unicast : Statique, dynamique (vecteur distance) ;
- routage multicast : DVMRP, PIM ;
- gestion de file d'attente : RED, DropTail, Token bucket.

3.3.2 L'outil de visualisation NAM

NS-2 ne permet pas de visualiser le résultat des expérimentations. Il permet uniquement de stocker une trace de la simulation, de sorte qu'elle puisse être exploitée par un autre logiciel, comme NAM.

NAM [15] est un outil de visualisation qui présente deux intérêts principaux : représenter la topologie d'un réseau décrite avec NS-2, et afficher temporellement les résultats d'une trace d'exécution NS-2. Par exemple, il est capable de représenter des paquets TCP ou UDP, la rupture

d'un lien entre nœuds, ou encore de représenter les paquets rejetés d'une file d'attente pleine. Ce logiciel est souvent appelé directement depuis les scripts TCL pour NS-2, pour visualiser directement le résultat de la simulation.

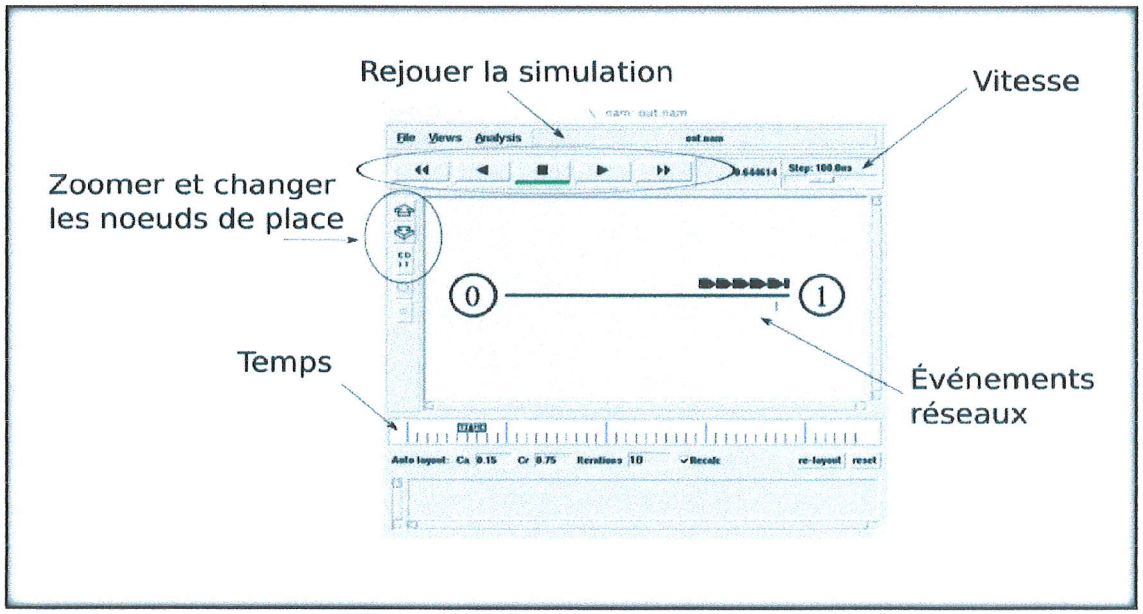


Figure 3.2 : L'outil de visualisation NAM

3.3.3 Outil graphique xgraph

Xgraph [15] est un autre utilitaire utilisé par NS-2 et qui permet lui aussi de fournir un compte rendu graphique, mais cette fois-ci sous forme de courbes statistiques (voir Figure 3.3), les formats de données acceptés en entrée sont de type deux dimensions (x et y). Les valeurs dans chaque ligne séparées par des espaces ou des colonnes, et les données peuvent être à deux ou à plusieurs colonnes.

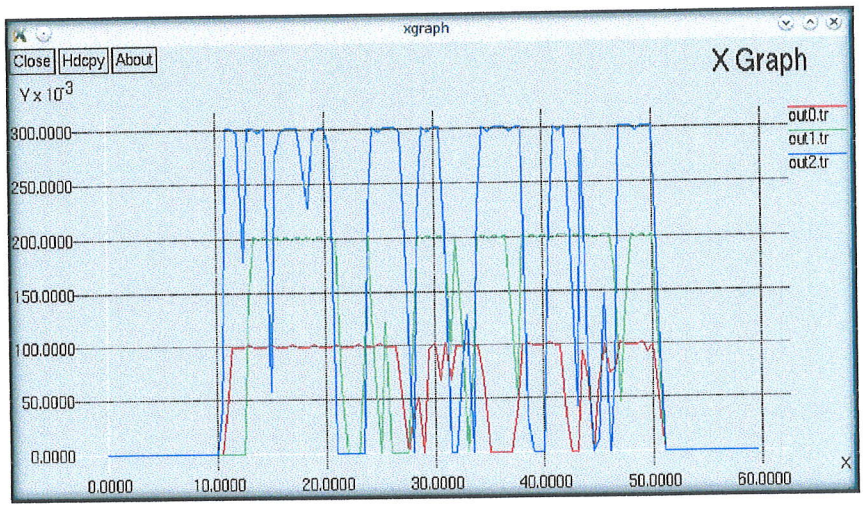


Figure 3.3 : Outil graphique xgraph

3.4 Installation de NS-2

➢ **Etape1 :**

- téléchargez ns-allinone-2.35.tar.gz
- lien de téléchargement :
  http://sourceforge.net/projects/nsnam/files/allinone/ns-allinone-2.35/

➢ **Etape2 :**

Le package téléchargé sera nommé "ns-allinone-2.35.tar.gz". Copiez-le dans le dossier home. Ensuite, dans un terminal utiliser les deux commandes suivantes pour extraire le contenu de l'emballage :

- **cd**
- **tar -xvzf ns-allinone-2.35.tar.gz**

- Tous les fichiers seront extraits dans un dossier appelé "ns-allinone-2.35".

➢ **Etape 3 :**

Construire les dépendances :

NS-2 nécessite quelques paquets pour être pré installés. Il exige également la version GCC-4.3 pour fonctionner correctement. Donc, on peut les installer en utilisant la commande suivante:

- **sudo apt-get install gcc-4.4**

Le résultat d'exécution de cette commande est donné par la figure :

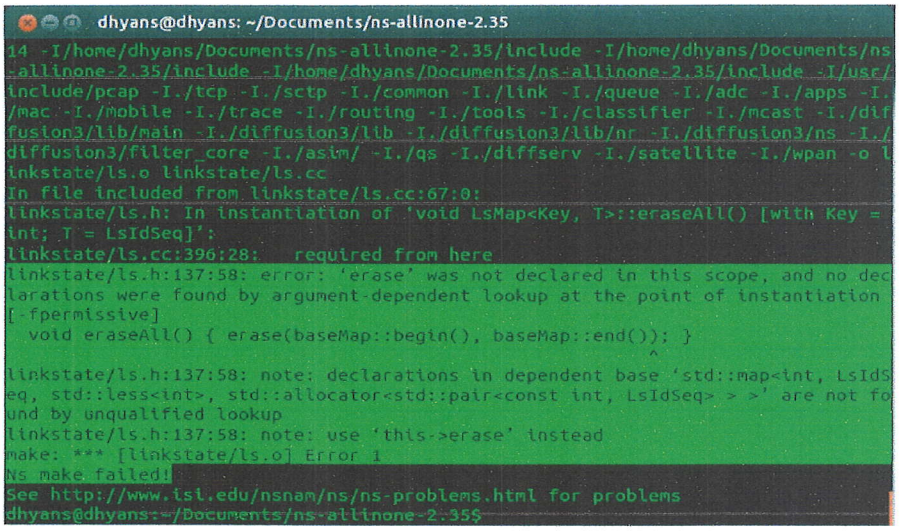


Figure 3.4 : résultat d'installation des paquets gcc-4.4

- Une fois que l'installation est terminée, nous devons faire un changement dans le fichier "ls.h".
- Utilisez les étapes suivantes pour effectuer les modifications: Accédez au dossier "linkstate", utilisez la commande suivante. Ici, il est supposé que le dossier ns extrait est dans le dossier home du système.

  cd ~/ns-allinone-2.35/ns-2.35/linkstate
- Maintenant, ouvrez le fichier nommé "ls.h" et faites défiler jusqu'à la ligne 137e. Remplacer le mot «error» par «this- error>". L'image ci-dessous montre la ligne 137, après avoir effectué les modifications du fichier ls.h. Pour ouvrir le fichier, utilisez la commande suivante:
  - **gedit ls.h**

Figure 3.5 : commande d'ouverture du fichier ls.h

- Sauvegardez ce fichier et fermez-le.

```
// this next typedef of iterator seems extraneous but is required by gcc-2.96
typedef typename map<Key, T, less<Key> >::iterator iterator;
typedef pair<iterator, bool> pair_iterator_bool;
iterator insert(const Key & key, const T & item) {
        typename baseMap::value_type v(key, item);
        pair_iterator_bool ib = baseMap::insert(v);
        return ib.second ? ib.first : baseMap::end();
}

void eraseAll() { this->erase(baseMap::begin(), baseMap::end()); }
T* findPtr(Key key) {
        iterator it = baseMap::find(key);
        return (it == baseMap::end()) ? (T *)NULL : &((*it).second);
}
```

Figure 3.6: Fichier ls.h après modification

- Maintenant, il y a une étape de plus qui doit être fait. Nous devons dire aux NS quelle version de GCC sera utilisée. Pour ce faire, allez dans votre dossier ns et tapez la commande suivante:
  - **sudo gedit ns-allinone-2.35/otcl-1.13/Makefile.in**

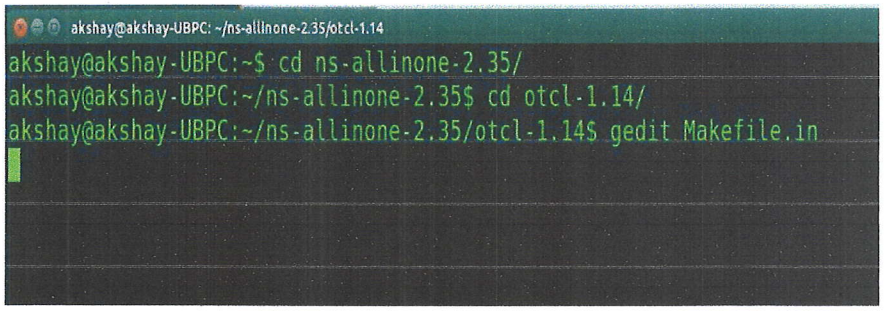


Figure 3.7 : La commande d'ouverture du fichier makefile.in

- Dans le fichier, changement de CC = @ CC par CC = gcc-4.4, comme le montre l'image ci-dessous.

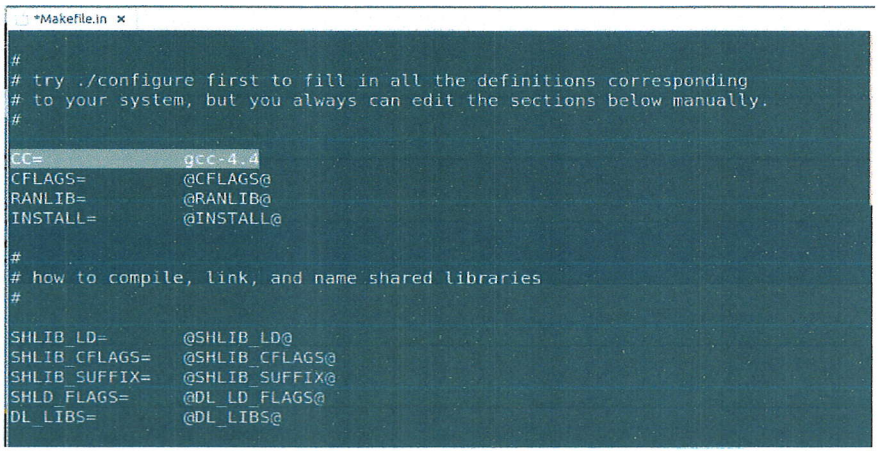


Figure 3.8 : Le fichier makefile.in

➢ **Etape 4: installation :**

Maintenant, nous sommes prêts à installer NS-2. Pour ce faire nous avons d'abord besoin de privilèges root, puis nous pouvons exécuter le script d'installation. Utilisez les deux commandes suivantes:

- **sudo su cd ~/ns-allinone-2.35/./install**

Ce qui suit est une rupture de ces commandes:

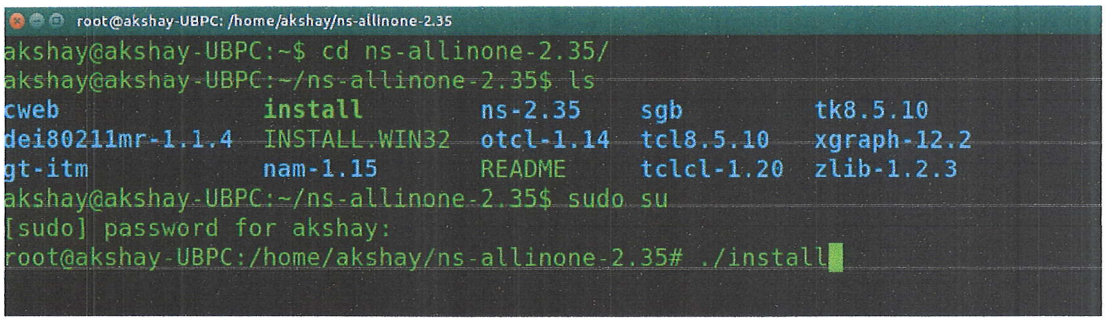


Figure 3.9 : Commande d'installation du NS-2

- L'image ci-dessous montre à quoi elle ressemble l'exécution réussie :

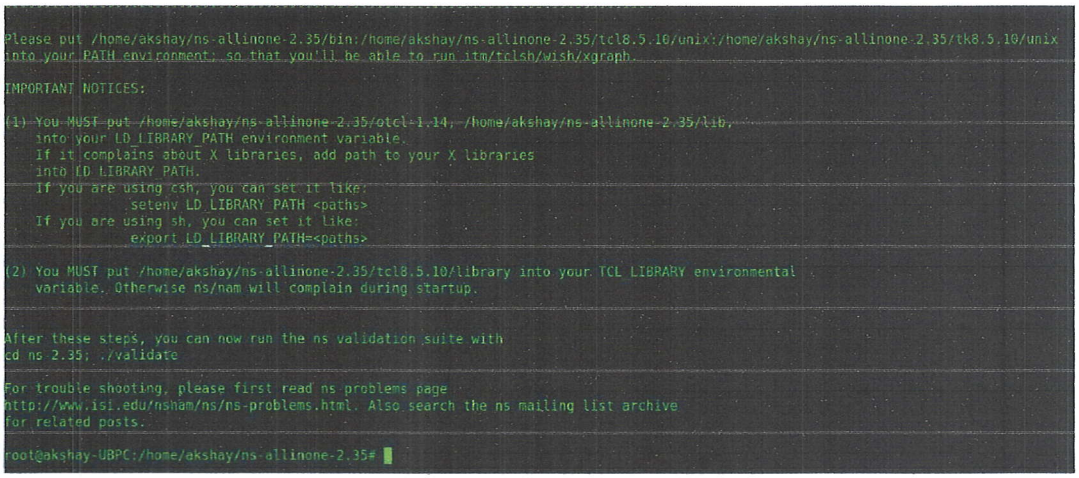


Figure 3.10 : installation du NS-2

L'installation prend 6 minutes, maintenant nous devons ajouter le chemin de construction sur le chemin de l'environnement.

➢ **Etape 5 : Définition du chemin d'environnement :**

La dernière étape consiste à indiquer au système, dans lequel les fichiers NS-2 sont installés ou présents. Pour ce faire, nous devons définir le chemin de l'environnement en utilisant le fichier «.bashrc». Dans ce fichier, nous avons besoin d'ajouter quelques lignes au fond. Les commandes à ajouter sont donnés ci-dessous. Mais pour le chemin indiqué dans ces commandes doit être remplacé par notre chemin d'installation.

Pour ce faire, il faut ouvrir le fichier .bashrc comme suit :

- **sudo gedit ~/.bashrc**

- Lignes à ajouter:

```
# LD_LIBRARY_PATH
OTCL_LIB=/home/akshay/ns-allinone-2.35/otcl-1.14
NS2_LIB=/home/akshay/ns-allinone-2.35/lib
X11_LIB=/usr/X11R6/lib
USR_LOCAL_LIB=/usr/local/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$OTCL_LIB:$NS2_LIB:$X11_LIB:$USR_LOCAL_LIB
# TCL_LIBRARY
TCL_LIB=/home/akshay/ns-allinone-2.35/tcl8.5.10/library
USR_LIB=/usr/lib
export TCL_LIBRARY=$TCL_LIB:$USR_LIB
# PATH
XGRAPH=/home/akshay/ns-allinone-2.35/bin:/home/akshay/ns-allinone-2.35/tcl8.5.10/unix:
/home/akshay/ns-allinone-2.35/tk8.5.10/unix
#the above two lines beginning from xgraph and ending with unix should come on the sam
e line
NS=/home/akshay/ns-allinone-2.35/ns-2.35/
NAM=/home/akshay/ns-allinone-2.35/nam-1.15/
PATH=$PATH:$XGRAPH:$NS:$NAM
```

Figure 3.11 : Le fichier bashrc

Une fois que les modifications ont été apportées, enregistrez le fichier et redémarrez le système.

➢ **Etape 6 : Exécuter NS-2 :**

Après l'initialisation du système, utiliser un terminal et lancer NS-2 en utilisant la commande suivante:

- **ns**

Si l'installation est correcte, alors le terminal ressemble à l'image ci-dessous :

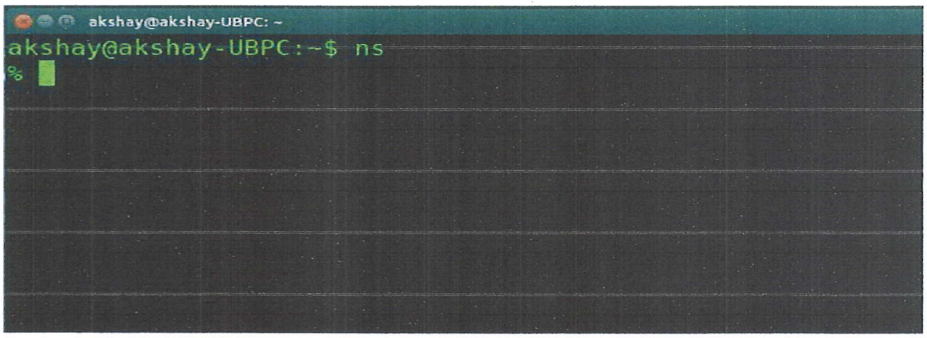


Figure 3.12 : Vérification du fonctionnement du NS-2

3.5 Le modèle de réseau sous NS-2

Un modèle de réseaux sous ns est constitué :

- ✓ De nœuds de réseau : endroits où est généré le trafic, ou nœuds de routage ;
- ✓ De liens de communication entre les nœuds.
- ✓ D'agents de communication, représentant les protocoles de niveau transport (TCP, UDP) ; ces agents sont attachés aux nœuds et connectés l'un à l'autre, ce qui représente un échange de données (connexion TCP, flux UDP).
- ✓ D'applications qui génèrent le trafic de données selon certaines lois (CBR, VBR), et se servent des agents de transport.

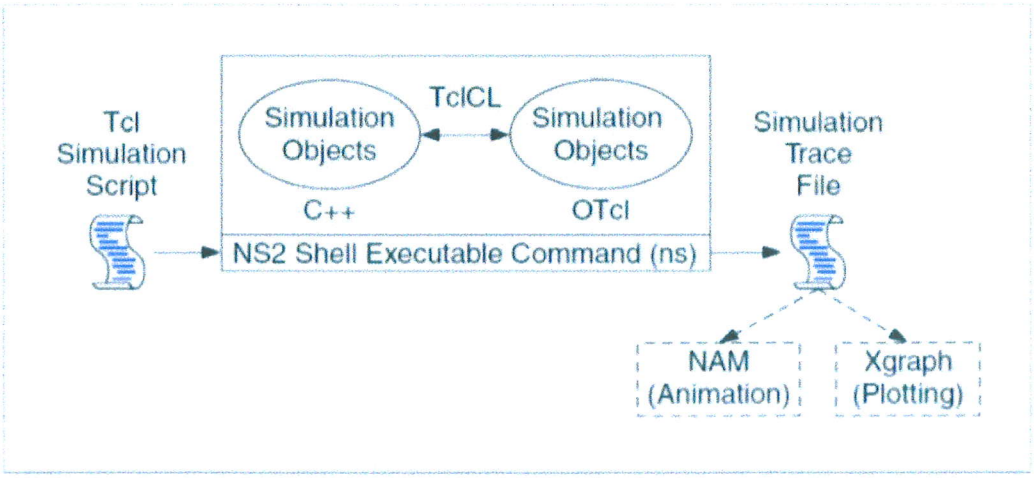


Figure 3.13: L'architecture de NS-2

L'architecture générale du NS2 est présentée sur la (figure 3.13), elle consiste en deux type de Langage de programmation: le C++ et l'OTCL (Object-Oriented Tool Command Language).

Le C++ est utilisé pour programmer les entités internes des systèmes simules, alors que l'OTCL est utilisé pour définir les scenarios des simulations et les paramètres de configuration. Ces deux types de langages sont, ensuite, lies via le TclCL qui permet le passage des codes C++ vers les codes OTCL et vice-versa.

Une fois la simulation terminée, NS produit deux fichiers de traces qui visualisent la dynamique des systèmes simules, et qui peuvent être interpréter en utilisant les outils: NAM

(Network AniMator) et Xgraph.

3.5.1 Définition de TCL

TCL est un langage de commande comme le shell UNIX mais qui sert à contrôler les applications. Son nom signifie Tool Command Language. Tcl offre des structures de programmation telles que les boucles, les procédures ou les notions de variables.

Il y a deux principales façons de se servir de TCL : comme un langage autonome interprété ou comme une interface applicative d'un programme classique écrit en C ou C++. En pratique, l'interpréteur TCL se présente sous la forme d'une bibliothèque de procédures C qui peut être facilement incorporée dans une application. Cette application peut alors utiliser les fonctions standards du langage TCL mais également ajouter des commandes à l'interpréteur.

3.5.2 OTCL

OTCL est une extension orientée objet de TCL. Les commandes TCL sont appelées pour un Objet. OTCL est une extension du langage de commande TCL, qui utilise une programmation Structuré (boucles, procédures, notions de variables. Le moteur OTCL transforme les instructions TCL en instructions C++. En OTCL, les classes sont également des objets avec des possibilités d'héritage. La définition d'une classe commence par la directive Class. Les fonctions et les attributs d'une classe s'installent par la suite par les commandes « instvar » et « insproc ». L'utilisation « instproc » définit les méthodes de la classe de manière assez semblable à C++. Lors d'une instanciation en OTCL.

3.5.3 Caractéristiques d'une entité communicante sous NS-2

Le nœud (entité communicante) constitue l'élément de base d'un modèle. Un nœud dans NS-2 est une classe définie dans OTCL, qui a une adresse et qui contient trois entités : le classifier, le lien et l'agent.

- **Le Classifier**

  La fonction d'un nœud est d'examiner des champs du paquet reçu, et plus précisément, l'adresse source et l'adresse destination. Selon ces valeurs, le nœud envoie ce paquet sur

ses interfaces de sortie (F). En NS-2, ceci est effectué par un objet qui s'appelle «classifier ». Il existe plusieurs types de classifier qui sont utilisés pour différents buts :

- « adresse classifier » : il est utilisé pour traiter les paquets unicast, son rôle est de sélectionner les paquets adressés directement au nœud, et de choisir le lien vers le prochain nœud.
- « port classifier » : son rôle est de sélectionner l'agent auquel le paquet est destiné.
- « multicast classifier » : il est utilisé pour classifier les paquets multicast.

➢ **Le lien**

Il est utilisé pour relier les nœuds (voir figure 3.14). Un lien est défini par plusieurs paramètres comme : sa bande passante, le point d'entrée, la durée de vie de chaque paquet, etc. NS2 présente plusieurs types de liens, ainsi on peut distinguer des liens unidirectionnels ou bidirectionnels, des liens filaires et des liens non filaires pour modéliser les réseaux sans fils.

➢ **L'agent**

Les agents représentent des points terminaux, là où des paquets de couche réseau sont Construits ou consommés. Ces agents constituent le troisième composant du nœud. Dans NS2, le rôle de l'agent est de fournir l'adresse de destination, les fonctions pour générer les paquets et l'interface à la classe application (voir figure 3.14).

Dans NS2 il existe plusieurs types d'agents, chacun a un rôle spécifique :

- ❖ agent TCP : pour émettre un trafic TCP ;
- ❖ agent UDP : pour émettre un trafic UDP ;
- ❖ agent TCPSink : pour la réception du trafic TCP ;
- ❖ agent NULL : pour la réception des paquets UDP.

La figure 3.14 représente les entités existantes dans un nœud et les liens entre ses entités.

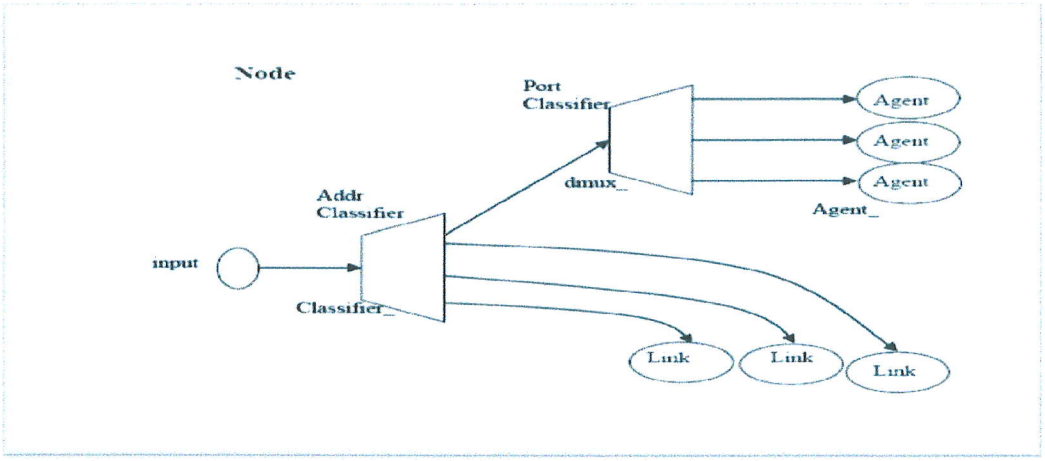


Figure 3.14 : Schéma d'un nœud dans NS-2

L'analyse des résultats est en général peu aisée, le résultat de la simulation étant essentiellement composé d'un fichier retraçant l'ensemble des envois, réceptions et suppressions de paquets. Un certain nombre de scripts ont été développés (ou sont en cours de développement) pour faciliter cette analyse.

3.6 Le processus de simulation

Le processus de simulation en utilisant NS-2 est composé de trois phases principales :

3.6.1 Phase de préparation

Elle s'occupe de la génération des fichiers d'entrées. A cette étape, on introduit des fichiers de scripts OTCL qui décrivent l'environnement avec tous ses nœuds, leurs déplacements et leur trafic de données. Ces fichiers sont classés en deux catégories :

- Fichiers de scénario qui décrivent les nœuds, leurs positions ainsi que leurs mouvements.
- Fichiers de communication qui décrivent le trafic dans le réseau.

3.6.2 Phase de simulation

Pour lancer les simulations et générer les traces. Les deux fichiers obtenus de la phase de préparation sont introduits dans un script de lancement OTCL. Le script de simulation consiste à indiquer la topologie du réseau, à activer des traces aux endroits pertinents et à engendrer des

événements particuliers à des instants donnés. A la fin de cette étape on obtient deux fichiers (journaux) appelé aussi «fichiers traces ». Le premier fichier sera traité par l'outil de visualisation NAM. Et le deuxième doit être filtré par un script awk afin d'afficher le résultat en utilisant l'outil Xgraph [14].

3.6.3 Phase d'analyse

Pour analyser [14] les traces et générer les courbes. L'outil de visualisation NAM s'occupe du premier fichier trace. Deux éléments intéressants sont proposés à la visualisation : un dessin de la topologie du réseau étudié, et une visualisation dynamique du déroulement du programme dans le temps. Le deuxième fichier de trace sauvegarde tous les échanges de paquets effectués. Afin de dessiner les courbes en utilisant Xgraph, le fichier doit être filtré par un script awk pour ne garder que les informations pertinentes.
Le schéma des étapes de simulation est illustré dans la figure suivante :

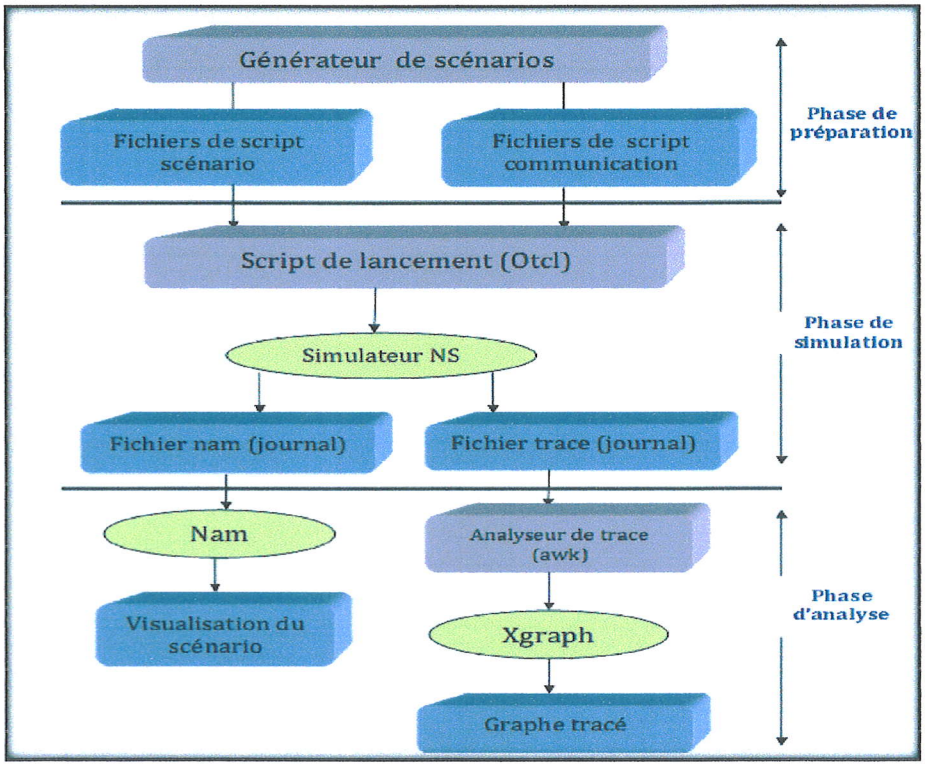


Figure 3.15 : le schéma de processus NS-2

3.7 Les avantage et les inconvénients du simulateur NS-2

Les avantages :

- ✓ Un logiciel de simulation multicouche.
- ✓ Un outil complètement libre pour plusieurs plateformes.
- ✓ Possibilité d'ajouter des composants à la demande.
- ✓ Développement orienté objet.
- ✓ Du fait de sa popularité, de nombreux protocoles sont à priori disponibles pour NS-2.

Les inconvénients :

- ✓ la modélisation dans NS-2 reste une tâche complexe : il n'y a pas d'interface graphique.
- ✓ Une forte technicité est requise pour utiliser ce simulateur.

3.8 Conclusion

Dans ce chapitre, nous avons bien décrit les outils et les composants de NS-2, et nous avons pu cerner la problématique d'installation du simulateur NS-2, aussi bien que nous avons présenté le processus de simulation de NS-2.

Chapitre 4 : Simulation et Résultats

4.1 Introduction

La transmission de la vidéo est devenue une nécessité pour tous les utilisateurs d'internet, quel que soit le dispositif utilisé pour accéder à l'internet. En générale la transmission vidéo exige une bande passante élevée. Sans compression, il est très difficile de transmettre la vidéo sur des réseaux filaires ou sans fils.

Dans ce chapitre nous allons simuler la transmission d'un fichier vidéo H.264/AVC et calculer la qualité de la vidéo transmise sur un réseau filaire et sans fils selon trois critères qui sont : le nombre de trames, le pas de quantification, la taille de la trame.

4.2 Environnement de simulation

Il faut noter que les résultats obtenus, dépendent des caractéristiques de l'ordinateur utilisé. Les résultats des simulations exposées dans ce chapitre ont été acquis en travaillant avec un microordinateur dont les paramètres du système sont les suivants :

Processeur : Intel ® Core TM i3.

Mémoire RAM : 4 Go.

Type du système : système d'exploitation : ubuntu 14.04.

4.2.1 Le simulateur NS-2

L'environnement de simulation que nous avons choisi était NS-2 (network simulator), la raison que nous avons fait ce choix est par ce que NS-2 est un simulateur dont le code source est libre et par conséquent extensible (possibilité de modifier la source et de la recompiler).

Le simulateur NS-2 se caractérise par :

- C'est un simulateur gratuit et open source.
- Il fonctionne sous Linux, Mac et Windows. Il a une grande bibliothèque et riche d'objets de réseau et de protocole.

- Il est basé sur un simulateur orienté objet écrit en C ++ et un interprète OTCL (un objet d'extension orientée de Tool Command Langage TCL).
- Il utilise TCL comme langage de script et Simule les réseaux filaires et sans fil.
- Il couvre une grande partie des applications (Web, FTP, CBR,...), protocoles (protocoles de transport et de routage), type de réseaux (Satellite liens, filaire et sans fil), des éléments de réseau (pas mobiles, sans fil modèles de canal, lien et modèles de files d'attente).
- Il permet également d'ajouter et de tester de nouveaux protocoles et applications et / ou de modifier celles qui existent déjà.
- Pour analyser les fichiers de trace, d'autres outils indépendants seront nécessaires pour filtrer, calculer et afficher les résultats (par exemple Awk, Matlab,gnuplot, etc ..).

4.2.2 Présentation du software de référence JM

Le JVT (Joint Vidéo Team) [16], le groupe responsable de développement et de maintenance de la norme H.264/AVC a publié une implémentation de référence de standard H.264 en langage C. Cette implémentation est connue sous le nom de JM (Joint Model). Une version ancienne a été publiée sous le nom ITU-T standard H.264. Dans notre application, nous avons utilisé la version (JM15.0), téléchargeable depuis le site :

http://iphome.hhi.de/suehring/tml/download/ jm15.0.

Le software de référence JM se compose de deux éléments principaux : un encodeur ou un compresseur (lencod) et un décodeur ou décompresseur (ldecod). L'encodeur encode la source vidéo pour produire une suite binaire codée (bitstream) de type h.264 (l'extension est donc .264). Le décodeur pour sa part, prend en entrée un flux de données codé en h.264, effectue des opérations de décodage pour obtenir à la fin une vidéo décodée.

Le travail de l'encodeur (resp. le décodeur) est contrôlé par un fichier de configuration qui inclut les définitions de tous les paramètres nécessaires à la compression (resp. décompression). Le nom de ce fichier est par défaut encoder.cfg (resp. decoder.cfg).

En plus de sa fonction principale, l'encodeur peut également reconstruire le fichier de la vidéo source pour générer un fichier d'une vidéo reconstruite (une copie d'un fichier vidéo décodé), et peut optionnellement générer un fichier de trace qui enregistre tous les éléments de syntaxe de la séquence codée

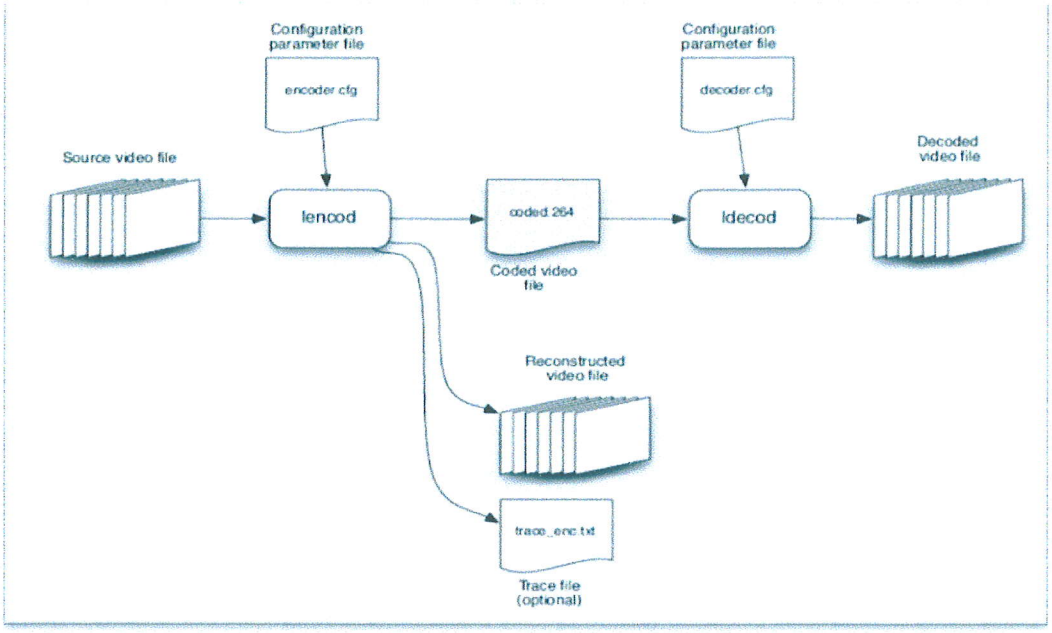


Figure 4.1 : Structure de JM software de référence

4.2.2.1 Installation et compilation

Après le téléchargement du paquet logiciel, copier et extraire « jm15.0.zip » dans le dossier /home. Ouvrir le terminal et exécute les étapes suivantes :

```
>>> cd JM
>>> . unixprep.sh
```

ou

```
>>> chmod u+x unixprep.sh
>>> ./unixprep.sh
```

- Pour compiler l'encodeur ou "lencod" tapez sur le terminal

```
>>> cd lencod
>>> make
```

- Pour compiler le décodeur ou "ldecod" tapez sur le terminal :

```
>>> cd ldecod
>>> make
```

Donc, deux fichier nommés "lencod.exe" et "ldecod.exe" sont créés dans le répertoire "bin".

4.2.2.2 Utilisation de l'encodeur

Pour exécuter l'encodeur JM, il faut taper sur le terminal une commande qui suit la syntaxe suivante :

```
>>> Lencod [-h] [-d defenc.cfg] {[-f curenc1.cfg]... [-f curencN.cfg]} {[-p EncParam1=EncValue1]... [-p EncParamM=EncValueM]}
```

Tel que :

Tableau 4.1 : Options de l'encodeur

Option	Description
-h	Utilisation spécifique des paramètres
-d	Utiliser « defenc.cfg » pour les initialisations des paramètres, comme un choix par défaut. S'il n'était pas utilisé, le fichier par défaut devient encoder.cfg dans le répertoire local
-f	Lire le fichier « curenc1.cfg » pour le reparamétrage, de l'encodeur.
-p	Mettre un paramètre quelconque «EncParamM» à une valeur «EncValueM»

Fichier de configuration de l'encodeur

Le grand nombre de paramètres disponibles dans le fichier de configuration encoder.cfg offre à l'utilisateur la possibilité de bien contrôler, et en détail aussi, l'opération de l'encodeur JM.

```
! New Input File Format is as follows
! <ParameterName> = <ParameterValue> # Comment
!
! See configfile.h for a list of supported ParameterNames
!
! For bug reporting and known issues see:
! https://ipbt.hhi.de


###############################################################################################
! Files
###############################################################################################
InputFile             = "carphone_qcif.yuv"       # Input sequence
InputHeaderLength     = 0      # If the inputfile has a header, state it's length in byte here
StartFrame            = 0      # Start frame for encoding. (0-N)
FramesToBeEncoded     = 300     # Number of frames to be coded
FrameRate             = 30.0   # Frame Rate per second (0.1-100.0)
SourceWidth           = 176   # Source frame width
SourceHeight          = 144   # Source frame height
SourceResize          = 0      # Resize source size for output
OutputWidth           = 176    # Output frame width
OutputHeight          = 144    # Output frame height
ProcessInput          = 0      # Filter Input Sequence

TraceFile             = "trace_enc.txt"      # Trace file
ReconFile             = "test_rec.yuv"       # Recontruction YUV file
OutputFile            = "carphone_qcif_Sqp.264"           # Bitstream
StatsFile             = "stats.dat"          # Coding statistics file

###############################################################################################
! Encoder Control
###############################################################################################
Grayscale             = 0   # Encode in grayscale (Currently only works for 8 bit YUV 420 input)
ProfileIDC            = 100 # Profile IDC (66=baseline, 77=main, 88=extended; FREXT Profiles: 100=High, 110=High 10, 122=High 4:2:2, 244=High
4:4:4, 44=CAVLC 4:4:4 Intra)
IntraProfile          = 0   # Activate Intra Profile for FREXT (0: false, 1: true)
```

Figure 4.2 : Fichier de configuration de l'encodeur

4.2.2.3 Utilisation du décodeur

Syntaxe du décodeur

```
>>> ldecod [-h] {[defenc.cfg] | {[-i bitstream.264]... [-o
output.yuv] [-r reference.yuv]}}
```

Tel que:

Tableau 4.2 : Options de décodeur

Option	Description
-h	Utilisation spécifique des paramètres.
[defdec.cfg]	[defdec.cfg] Fichier de configuration optionnel de décodeur (contient tout les informations qui concerne le décodeur).
-i	Décoder le fichier bitstream.264. cette option est mise à test.264 par défaut.
-o	Le nom de fichier reconstruit est mis à output.yuv. la valeur par défaut est test_dec.yuv.
-r	Le fichier de séquence de référence pour le calcul de PSNR est mis à reference.yuv. la valeur par défaut est test_rec.yuv.

```
test.264              .....H.264/AVC coded bitstream
test_dec.yuv          .....Output file, YUV/RGB
test_rec.yuv          .....Ref sequence (for SNR)
1                     .....Write 4:2:0 chroma components for monochrome streams
0                     .....NAL mode (0=Annex B, 1: RTP packets)
0                     .....SNR computation offset
2                     .....Poc Scale (1 or 2)
500000                .....Rate_Decoder
104000                .....B_decoder
73000                 .....F_decoder
leakybucketparam.cfg  .....LeakyBucket Params
0                     .....Err Concealment(0:Off,1:Frame Copy,2:Motion Copy)
2                     .....Reference POC gap (2: IPP (Default), 4: IbP / IpP)
2                     .....POC gap (2: IPP /IbP/IpP (Default), 4: IPP with frame skip = 1 etc.)
0                     .....Silent decode
1                     .....Enable Deblocking filter in intra only profiles (0=disable, 1=filter according to SPS parameters)

This is a file containing input parameters to the JVT H.264/AVC decoder.
The text line following each parameter is discarded by the decoder.

For bug reporting and known issues see:
https://ipbt.hhi.de
```

Figure 4.3 : Fichier de configuration du décodeur

Exemple d'utilisation de JM

Pour coder ou compresser 100 images vidéo (frames) d'une séquence YUV de résolution QCIF. Nous devons suivre les étapes suivantes :

- Copier un fichier source QCIF dans le répertoire bin de JM. par exemple la séquence akiyo_qcif.yuv.
- Créer un fichier de configuration. On prend par exemple le fichier encoder.cfg, on le copie puis on donne un nouveau nom à la copie, soit encoder_exemple.cfg.
- Modifier les valeurs de paramètres du nouveau fichier de configuration encoder_exemple.cfg : les noms des fichiers d'entrée et de sortie, le nombre des images vidéo à encoder et mettre la valeur de quantificateur à 32.
- Ouvrir le terminal dans le répertoire bin.
- Exécuter l'encodeur en tapant la commande suivante :

```
./lencod.exe -d encoder_exemple.cfg
```

L'exécution de l'encodeur va générer une liste des informations illustrée dans la figure suivante. Pour chaque image vidéo codée, le type d'image, IDR/I ou P, le nombre de bits codés, le paramètre de quantification (QP), le PSNR ou SNR des composants Y, U et V, et le temps de codage sont listés.

```
------------------------------ JM 15.0 (FREXt) ------------------------------
Input YUV file                   : akiyo_qcif.yuv
Output H.264 bitstream           : akiyo_qcif.264
Output YUV file                  : test_rec.yuv
YUV Format                       : YUV 4:2:0
Frames to be encoded I-P/B       : 1/0
Freq. for encoded bitstream      : 30.00
PicInterlace / MbInterlace       : 0/0
Transform8x8Mode                 : 1
ME Metric for Refinement Level 0 : SAD
ME Metric for Refinement Level 1 : Hadamard SAD
ME Metric for Refinement Level 2 : Hadamard SAD
Mode Decision Metric             : Hadamard SAD
Motion Estimation for components : Y
Image format                     : 176x144 (176x144)
Error robustness                 : Off
Search range                     : 32
Total number of references       : 5
References for P slices          : 5
References for B slices (L0, L1) : 5
List1 references for B slices    : 1
Sequence type                    : I-B-P-B-P (QP: I 5, P 5, B 7)
Entropy coding method            : CABAC
Profile/Level IDC                : (100,40)
Motion Estimation Scheme         : Fast Full Search
Search range restrictions        : none
RD-optimized mode decision       : used
Data Partitioning Mode           : 1 partition
Output File Format               : H.264/AVC Annex B Byte Stream Format
```

Figure 4.4 : Résultat de l'exécution de l'encodeur.

4.3 Outil Evalvid :

Afin de simuler la transmission d'une vidéo, nous avons intégré une application dans le simulateur NS-2. Cet outil simule la transmission d'un fichier vidéo d'un nœud source qui émit les paquets vers un nœud destination.

4.3.1 Les composants d'Evalvid

Les trois composants responsables de la transmission et la réception de la vidéo sont :

❖ **myEvalvid**

Il est utilisé pour extraire le type et la taille de la trame du fichier de trace de la vidéo. Ensuite, il fragmente les trames de la vidéo en segments plus petits et les envoie vers la couche UDP inférieur au moment approprié (voir Figure 4.5).

❖ **MyUDP**

C'est une extension de l'agent UDP. Ce nouvel agent permet aux utilisateurs de spécifier le nom du fichier de sortie et du fichier trace de l'émetteur et il enregistre l'horodatage de chaque paquet transmis, l'identifiant du paquet, et la taille des données utiles du paquet (voir Figure 4.5)

Les données nécessaires (côté émetteur) sont:

- La vidéo (par defaut brute et non compressée).

- Vidéo encodée.

- Horodatage (time-stamp) et le type de chaque paquet envoyé.

❖ **MyEvalvid_Sink**

C'est l'agent de réception des trames fragmentées de la vidéo envoyée par *MyUDP*. Cet agent enregistre également l'horodatage, l'ID de paquet, et la taille des données utiles de chaque paquet reçu dans un fichier spécifié par l'utilisateur (voir Figure 4.5). Les données nécessaires (de coté de récepteur) sont:

- Horodatage (time-stamp) et le type de chaque paquet reçu.

- La video réassemblée et encodée.

- La vidéo brute et non compressée à afficher

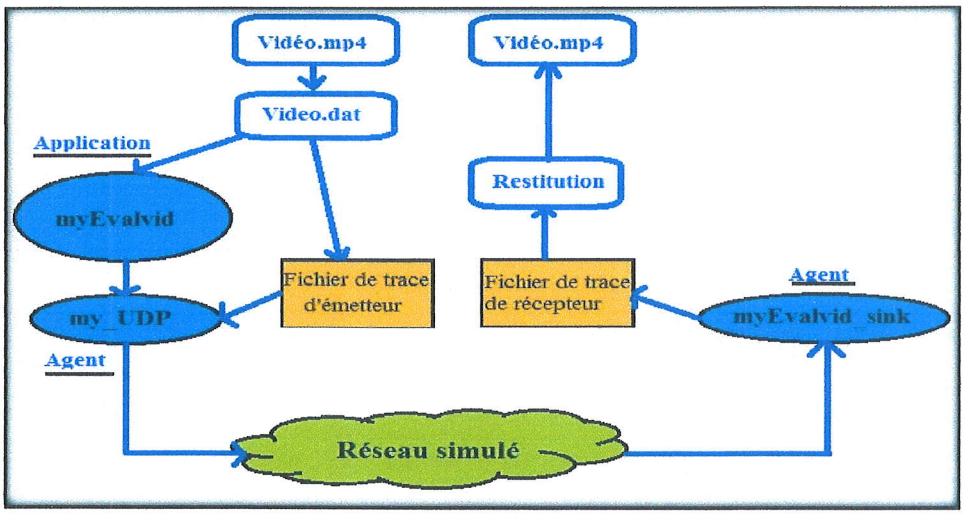


Figure 4.5 : Les composants d'Evalvid

4.3.2 L'installation d'Evalvid

Pour installer Evalvid il faut installer quelque paquet nécessaire pour notre application, donc on ouvre le terminal et on exécute les étapes suivantes :

- **Installer FFMPEG**
  - `sudo add-apt-repository ppa:kirillshkrogalev/ffmpeg-next`
  - `sudo apt-get update`
  - `sudo apt-get install ffmpeg`
- **Installer LIBSSL**
  - `sudo apt-get install libssl0.9.8`
- **Installer GPAC**
  - `sudo apt-get install subversion`
  - `sudo svn co https://svn.code.sf.net/p/gpac/code/trunk/gpac gpac`
- **Configurer la source de package GPAC**
  - `cd gpac`
  - `sudo ./configure --disable-opengl --use-js=no --use-ft=no --use-jpeg=no --use-png=no --use-faad=no --use-mad=no --use-xvid=no --use-ffmpeg=no --use-`

```
ogg=no --use-vorbis=no -use- theora=no --use-
openjpeg=no
```

- `sudo make`
- `sudo make install`

- **Copier le fichier libgpac.so dans le dossier usr**
  - `sudo cp bin/gcc/libgpac.so /usr/lib`
- **Télécharger Evalvid binaires depuis ce lien**
  http://www2.tkn.tu-berlin.de/research/evalvid/fw.html
  télécharger binary linux (EvalVid 2.7 binaries (Linux)) et le code source (EvalVid 2.7)
- **Copier et extraire** « evalvid-2.7-lin.tar.bz2 » dans le dossier /home

Exemple d'utilisation : pour vérifier les fonctions d'Evalvid par exemple le PSNR, on procède comme suit :

- cd evalvid-2.7-lin
- ls
- sudo ./psnr

4.3.3 Intégration d'Evalvid

- Mettre les champs *frametype_* et *sendtime_* dans l'entête *hdr_cmn.*

frametype_: ce champs est utilisé pour indiquer quel type de trame appartient-il le paquet.

La trame de type « **I** » a la valeur 1, La trame « **P** » a la valeur 2 et la trame « **B** » a la valeur 3

sendtime_ : ce champs a pour but d'enregistrer le temps de l'envoi de paquets, il peut être utilisé pour mesurer le délai.

Modifiez le fichier d'entête **packet.h** qui se trouve dans le répertoire :

/ns-all-in-one/ns2.27/common

- Modifier la structure « **hdr_cmn** » comme suivant :

```
struct hdr_cmn {
enum dir_t { DOWN= -1, NONE= 0, UP= 1 }; packet_t ptype_; //
packet type (see above)
int size_; // simulated packet size
```

```
int uid_; // unique id
int error_; // error flag
int errbitcnt_; // # of corrupted bits jahn
int fecsize_;
double ts_; // timestamp: for q-delay measurement
int iface_; // receiving interface (label)
dir_t direction_; // direction: 0=none, 1=up, -1=down
// source routing char src_rt_valid;
double ts_arr_; // Required by Marker of JOBS
//Ajoutez les lignes suivantes:
int frametype_; // frame type for MPEG video transmission (Henry)
double sendtime_; // send time (Henry)
unsigned long int frame_pkt_id_;
```

- Modifier le fichier « **agent.h** » dans le dossier qui se trouve dans le répertoire :

/ns-allinone-2.27/ns-2.27/common

```
class Agent : public Connector {
public:
Agent(packet_t pktType); virtual ~Agent();
void recv(Packet*, Handler*);
inline packet_t get_pkttype() { return type_; }
// Ajoutez les lines suivantes:
inline void set_frametype(int type) {frametype_ = type; }
inline void set_prio(int prio) { prio_ = prio; }
```

```
protected:

int command(int argc, const char*const* argv);

int defttl_; // default TTL for outgoing pkts

// Ajoutez la ligne suivante:

int frametype_; // Type de trame pour la transmission de la
vide MPEG

private:

void flushAVar(TracedVar *v);};
```

- Modifiez le fichier ***agent.cc*** le répertoire /ns-allinone-2.27/ns-2.27/common

```
Agent::Agent(packet_t pkttype) :

size_(0), type_(pkttype), frametype_(0),

channel_(0), traceName_(NULL),

oldValueList_(NULL), app_(0), et_(0)

{

}

Agent::initpkt(Packet* p) const

{ hdr_cmn* ch = hdr_cmn::access(p);

ch->uid() = uidcnt_++; ch->ptype() = type_; ch->size() =
size_;

ch->timestamp() = Scheduler::instance().clock();

ch->iface() = UNKN_IFACE.value(); // from packet.h (agent is
local) ch->direction() = hdr_cmn::NONE;

ch->error() = 0; /* pkt not corrupt to start with */

// Ajoutez la ligne suivante:
```

```
ch->frametype_= frametype_;
```

- Téléchargez et décompressez le fichier ***myevalvid2.rar*** de ce lien http://140.116.164.80/%7Esmallko/ns2/myevalvid2.rar

Copiez le dossier qui s'appelle « *myevalvid* » (il contient : *myevalvid.cc, myudp.cc, myudp.h, myevalvid_sink.cc, and myevalvid_sink.h), dans le répertoire suivant :*

ns-allinone-2.27/ns-2.27

- Modifiez le fichier *ns-allinone-2.28/ns-2.28/tcl/lib/**ns-default.tcl*** et ajoutez à la fin du fichier ***ns-default.tcl*** les deux lignes suivantes :

```
Agent/myUDP set packetSize_ 1000 Tracefile set debug_ 0
```

- Modifiez le fichier : *ns-allinone-2.28/ns-2.28/ **Makefile.in***

Mettre dans la partie ***OBJ_CC*** et avant le mot ***@V_STLOBJ@ ;*** les lignes ci-dessous et ajouter une tabulation (TAB) avant chaque commande :

```
myevalvid/myudp.o,

myevalvid/myevalvid_sink.o myevalvid/myevalvid.o
```

- Dans le terminal, allez au chemin */ns-allinone-2.27/ns-2.27* et tapez :

```
. /configure ; make clean ; make
```

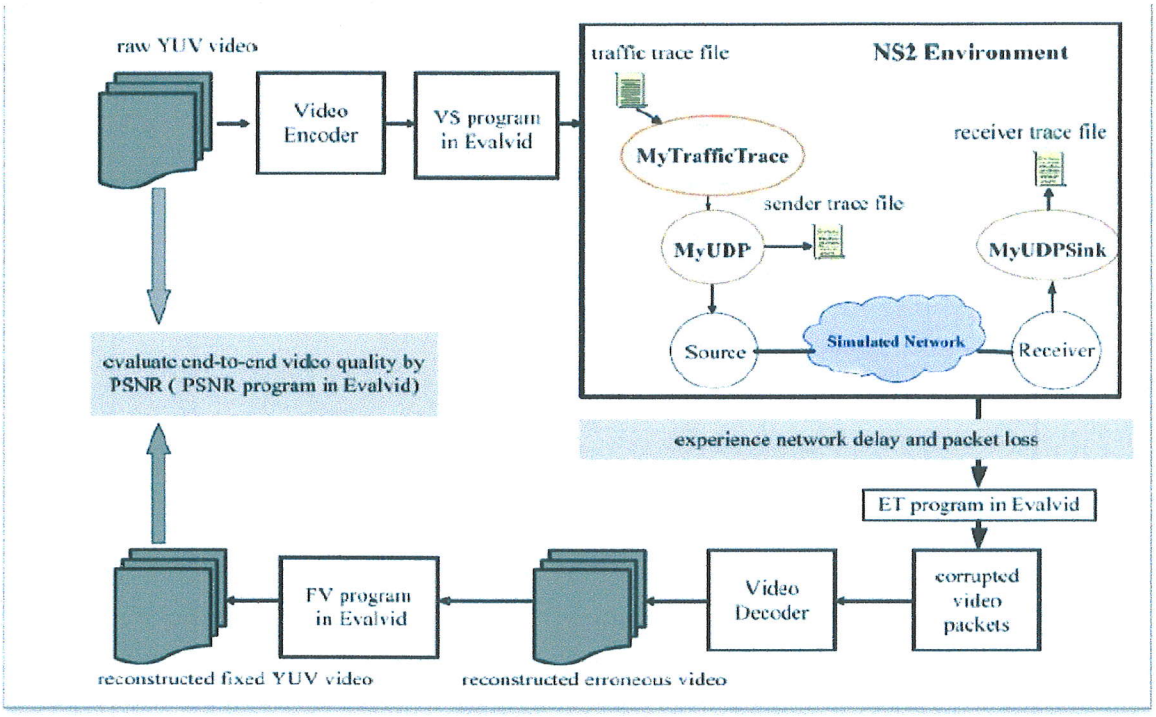


Figure 4.6 : Combinaison entre NS-2 et Evalvid

4.4 Scénario de la simulation

4.4.1 Création du scénario de l'application

Pour décrire un réseau et son trafic, il faut définir dans l'ordre :

- la topologie du réseau : les nœuds et les arcs
- la couche transport (UDP, TCP, ...) entre les pairs de nœuds
- la couche application qui va fournir les données
- des temporisateurs précisant les instants auxquels les transferts vont démarrer la couche réseau correspondent à IP et est gérée par ns-2.

❖ Dans un fichier tcl (par exemple : be_a01.tcl) on commence par créer un simulateur :

```
set ns [new Simulator]
```

❖ Pour conserver l'historique de la simulation (notamment pour la visualiser avec nam), on peut indiquer les deux fichiers vers lesquels seront dirigés les deux historiques :

```
set tracefd       [open simulation_tras.tr w]
set namtrace      [open simulation_tras.nam w]
```

❖ **Topologie**

Ensuite, on crée les nœuds du réseau (dans notre application 4 nœuds), puis les liens qui les relient avec leurs débits et leurs latence, et la politique de rejet des paquets (ici DropTail mais il y en a d'autres). On peut aussi préciser le nombre limite de paquets dans la file d'attente :

```
$ns duplex-link  $s1 $r1  10Mb   1ms DropTail
$ns simplex-link $r1 $r2  640kb  1ms DropTail
$ns simplex-link $r2 $r1  640Mb  1ms DropTail
$ns duplex-link  $r2 $d1  10Mb   1ms DropTail
```

❖ **Transport**

Il faut créer les agents de transport, les attacher aux nœuds et les connecter entre eux. Parmi les agents disponibles, on trouve Agent/Null, Agent/myUDP, Agent/myEvalvid_Sink . Le code est par exemple:

✓ pour créer une source myUDP :

set udp [new Agent/myUDP]

✓ pour attacher la source à un nœud :

$ns attach-agent $s1 $udp

```
set udp1 [new Agent/myUDP]
$ns attach-agent $s1 $udp1
set null1 [new Agent/myEvalvid_Sink] .
$ns attach-agent $d1 $null1
$ns connect $udp1 $null1
```

Noter qu'il n'est pas nécessaire de spécifier quelle est la route suivie par les paquets d'une paire d'agents : c'est le simulateur qui calcule les routes les plus courtes.

❖ Il faut enfin attacher l'application à l'agent de transport :

```
set video1 [new Application/Traffic/myEvalvid]
$video1 attach-agent $udp1
```

❖ **Temporisateurs**

Il est possible de programmer des évènements qui seront exécutés à des dates précises de la simulation. Par exemple:

```
$ns at 0.0 "$video1 start"
$ns at $end_sim_time "$video1 stop"
$ns at [expr $end_sim_time + 1.0] "$null1 closefile"
$ns at [expr $end_sim_time + 1.0] "finish"
```

Typiquement, la fin de la simulation se fera par l'invocation d'une procédure chargée du traitement final des statistiques. Exemple:

```
proc finish {} {
        global ns nd
        $ns flush-trace
        close $nd
        exit 0
}
```

4.4.2 Les étapes de la transmission de la vidéo

Pour effectuer la simulation, l'application doit tout d'abord effectuer une compression en utilisant le software de référence JM. Le JM a comme entrée une séquence vidéo brute au format YUV (.yuv). Il doit être configuré pour produire à la fin un flux binaire (.264) et une séquence vidéo reconstruite (.yuv), constitue par l'encodeur. On peut résumer le processus de notre simulation comme suit :

- **La conversion de format** :

  Afin que NS2 soit capable de transmettre la vidéo, il faut convertir la vidéo en format adéquate. La commande ci-dessous convertie le format 264 en un format de vidéo de type m4v.

  $ffmpeg -i news_cif.264 a01.m4v

- Ensuite, nous devons obtenir le fichier .mp4, la commande suivante convertir le fichier m4v en un fichier de type mp4

  $. /MP4Box -hint -mtu 1024 -fps 30 -add a01.m4v a01.mp4

- Maintenant que nous avons le fichier mp4, nous avons besoin de générer le fichier de trace nécessaire pour l'envoyer dans le réseau. L'outil de mp4trace de EvalVid est

capable d'envoyer le fichier mp4 par les protocoles RTP / UDP à un hôte de destination spécifié. La sortie de mp4trace sera nécessaire plus tard, il devrait donc être redirigé vers un fichier. Le fichier st_a01.st est le fichier trace de la vidéo qui a les informations sur chaque trame de la vidéo et qu'il sera utilisé par ns2 comme une source de trafic.

$./mp4trace -f -s 192.168.0.2 12346 a01.mp4 > st_a01.st

- Créer une topologie de réseaux à simuler (evalvid,tcl) et exécuter la simulation comme suit :

ns be_a01.tcl

- Après la simulation, NS2 va créer deux fichiers, *sd_a01* et *rd_a01*. Le fichier *sd_a01* consiste à enregistrer le moment d'envoi de chaque paquet pendant que le fichier *rd_a01* est utilisé pour enregistrer le moment de réception de chaque paquet.
- Pour reconstruire la vidéo transmis tel qu'il est vu par le récepteur L'outil *etmp4* génère un fichier vidéo du récepteur en se référant au fichier trace d'expéditeur (*sd_a01*), le fichier trace du récepteur (*rd_a01*) et le fichier trace original (*st_a01*). En comparant ces fichiers, nous saurons quelles sont les parties du fichier d'origine, sont perdus.

$./etmp4 -f -0 sd_a01 rd_a01 st_a01.st a01.mp4 a01out

- Maintenant, décoder la video reçu en format yuv

$ffmpeg -i a01out.mp4 a01out.yuv

- Enfin calculer le psnr :

$./psnr 352 288 420 news_cif.yuv a01out.yuv

- Tracer le graph de PSNR

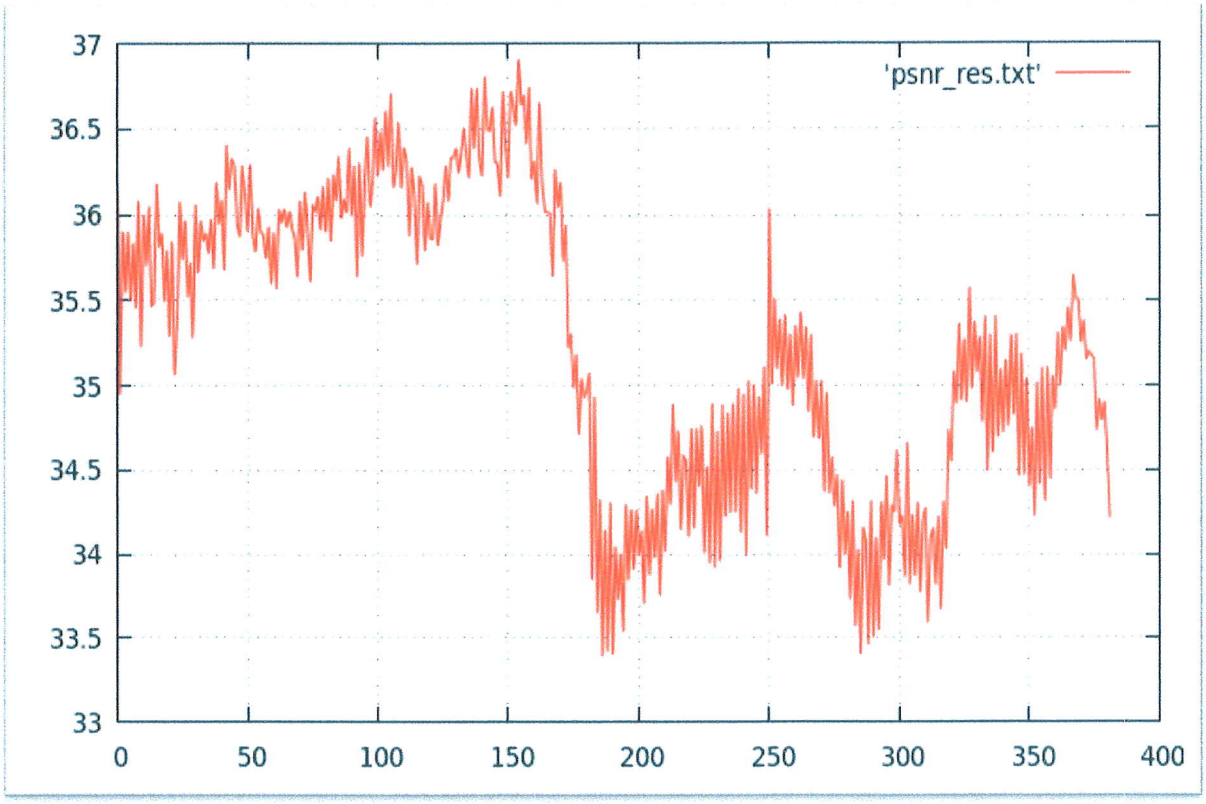


Figure 4.7 : Le graph de PSNR

La figure suivante résume toutes les étapes de la conversion de format ainsi que les étapes de la simulation.

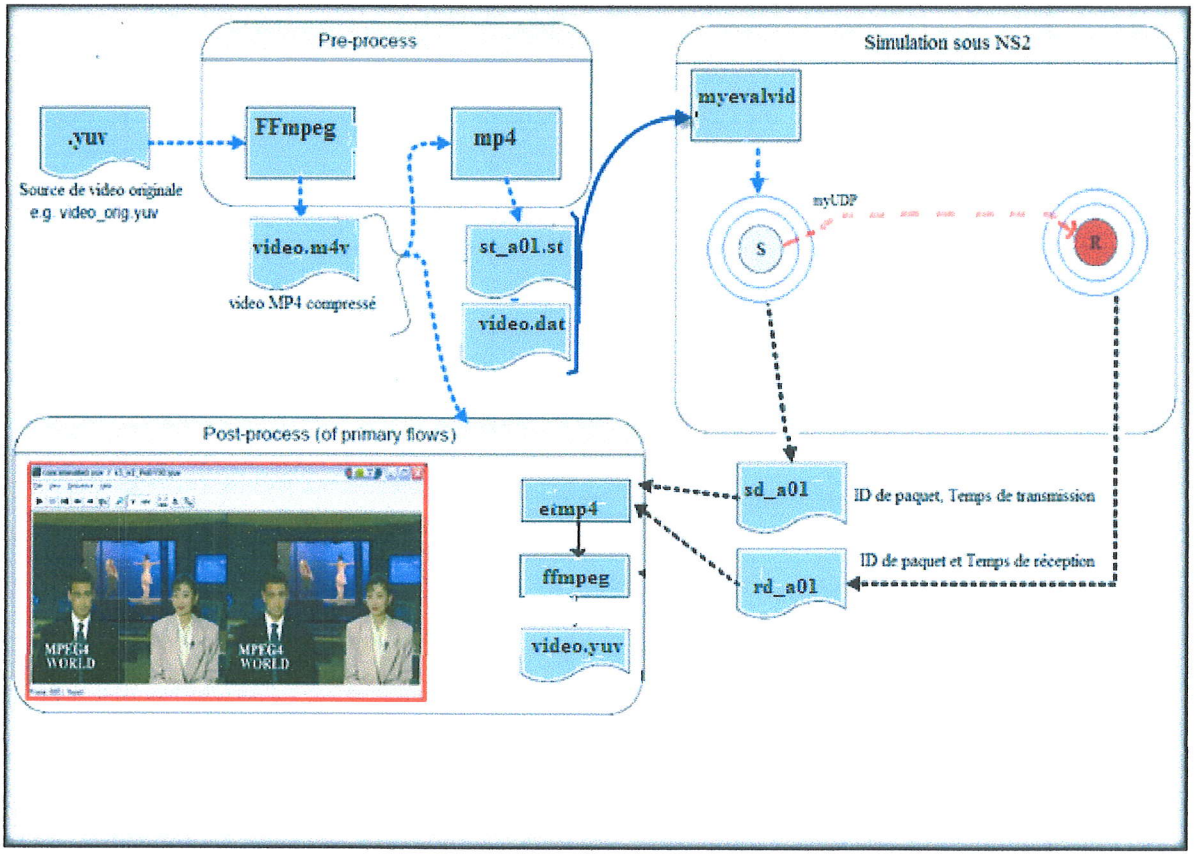


Figure 4.8 : Fonctionnement d'Evalvid

4.5 Simulation et analyse des résultats

Dans cette partie, nous allons simuler la transmission de la vidéo à travers un réseau sur NS2 puis évaluer la qualité des résultats de la transmission en utilisant l'outil Evalvid selon trois critères :

- ✓ Le pas de quantification.
- ✓ La taille des trames.
- ✓ Le nombre des trames.

Les tests sont effectués sur des séquences vidéos brutes (.yuv) mobile et non mobile échantillonnées en format 4:2:0. Les séquences utilisées sont : news, mobile, et mother-daugther chacune avec deux résolutions différentes QCIF (176 × 144) et CIF (352 × 288). La qualité de la vidéo reconstruite après la transmission est évaluée avec trois mesures : PSNR, SSIM, et MOS.

4.5.1 La qualité de la vidéo selon le pas de quantification

Le tableau ci-dessous montre l'effet du paramètre de quantification sur les vidéos reconstruites des séquences News et Daughter-Mother.

Tableau 4.3 : Résultat de la qualité selon le pas de quantification

		Pas de quantification					
Séquence	**Mesures de qualité**	5	10	15	20	30	48
News	PSNR	39,89	39,82	39,66	39,17	35,56	24,47
	SSIM	0,98	0,98	0,98	0,97	0,96	0,69
	MOS	Excellent	Excellent	Excellent	Excellent	Good	Poor
Daughter-Mother.	PSNR	40,38	40 ,49	40,39	39,89	35,88	26,06
	SSIM	0,98	0,98	0,98	0,98	0,94	0,66
	MOS	Excellent	Excellent	Excellent	Excellent	Good	Fair

Il faut noter que nous avons choisi dans cette première analyse, les paramètres de codage par défaut proposés par le software de référence JM (QP = 28, Nombre de trames = 300).

Les résultats montrent que la qualité de la vidéo, en fonction de PSNR, SSIM, et MOS est inversement proportionnelle au pas de quantification QP. La figure suivante montre qu' 'un pas de quantification supérieur a 38 cause une véritable dégradation de la qualité vidéo.

Discussion

Ces observations sont justifiées par la nature et la fonction du processus de quantification. En effet c'est le principal responsable de la dégradation de l'image. C'est lors de cette étape que l'image va être réellement compressée.

En fait, plus le pas de quantification est grand, plus le nombre de valeurs nulles augmente Dans ce cas, le codeur nécessite moins d'efforts pour coder l'image. D'autre part, dans le cas où la vidéo ne contient pas beaucoup de mouvements comme par exemple le cas des séquences statiques (Daughter-Mother.) où les images vidéo sont presque similaires,

l'image résiduelle (différence entre l'image courante et l'image prédite) obtenue par l'estimation de mouvement contient d'origine plusieurs coefficients nuls. Alors, l'application de quantification qui consiste à rendre les valeurs d'image nulles ou proches à zéro n'apporte pas beaucoup de choses à la séquence vidéo et donc, la qualité de l'image ne sera pas beaucoup affectée par rapport aux séquences plein de mouvement comme Mobile.

QP = 30

QP=48

Figure 4.9 : dégradation de la qualité de compression

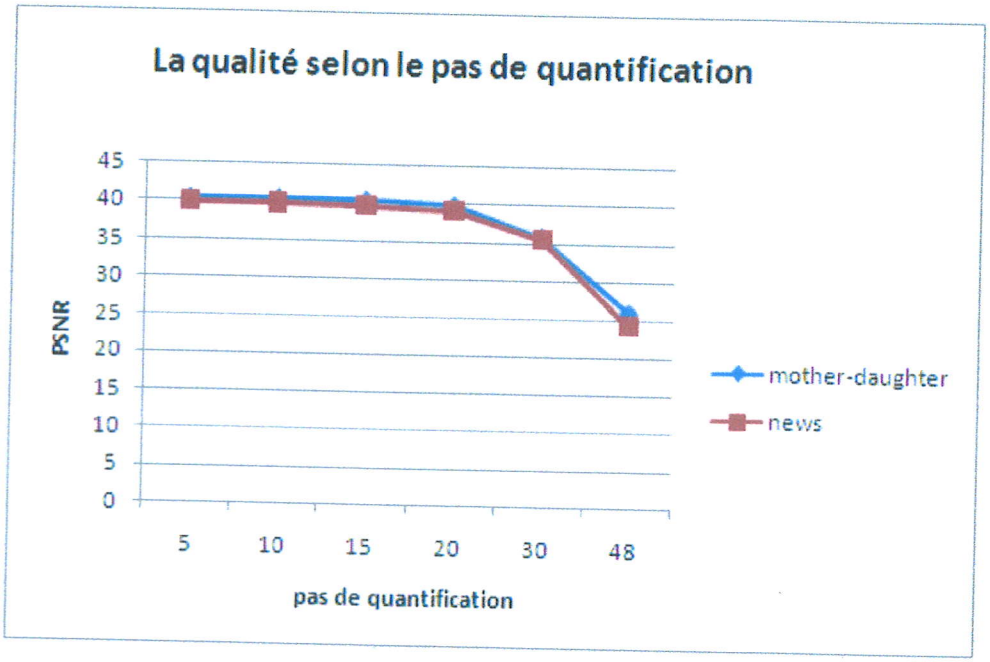


Figure 4.10 : le PSNR en fonction du pas de quantification

4.5.3 La qualité de la vidéo selon le nombre de trames

Tableau 4.5 : Résultat de la qualité selon le nombre de trame

		Nombre de trames							
Séquences		25	50	75	100	150	200	250	300
News	PSNR	38,62	39,19	39,34	39,14	39,04	38,45	33,52	39,18
	SSIM	0,98	0,98	0,98	0,98	0,98	0,98	0,91	0,98
	MOS	Excellent	Excellent	Excellent	Excellent	Excellent	Excellent	Good	Excellent
Mother_ Daughter	PSNR	35,75	37,32	37,02	36,92	36,85	36,81	36,91	37,01
	SSIM	0,96	0,96	0,95	0,95	0,95	0,95	0,95	0,95
	MOS	Good	Excellent	Excellent	Good	Good	Good	Good	Excellent

Le tableau 4.5 représente la variation de PSNR, SSIM, et MOS en termes de nombre de trames. Logiquement, plus le nombre de trames n'est grand, la qualité de la vidéo reconstruite se dégrade. Cependant, les résultats montrent que les valeurs de PSNR, SSIM, et MOS sont variables sans régularité. On remarque particulièrement que lorsque le changement se produit sur le fond de la séquence et il existe en plus du mouvement dans les objets, les valeurs de PSNR, SSIM, et MOS sont diminués. Donc, le nombre de trames n'a pas un grand effet sur la qualité de la vidéo transmise.

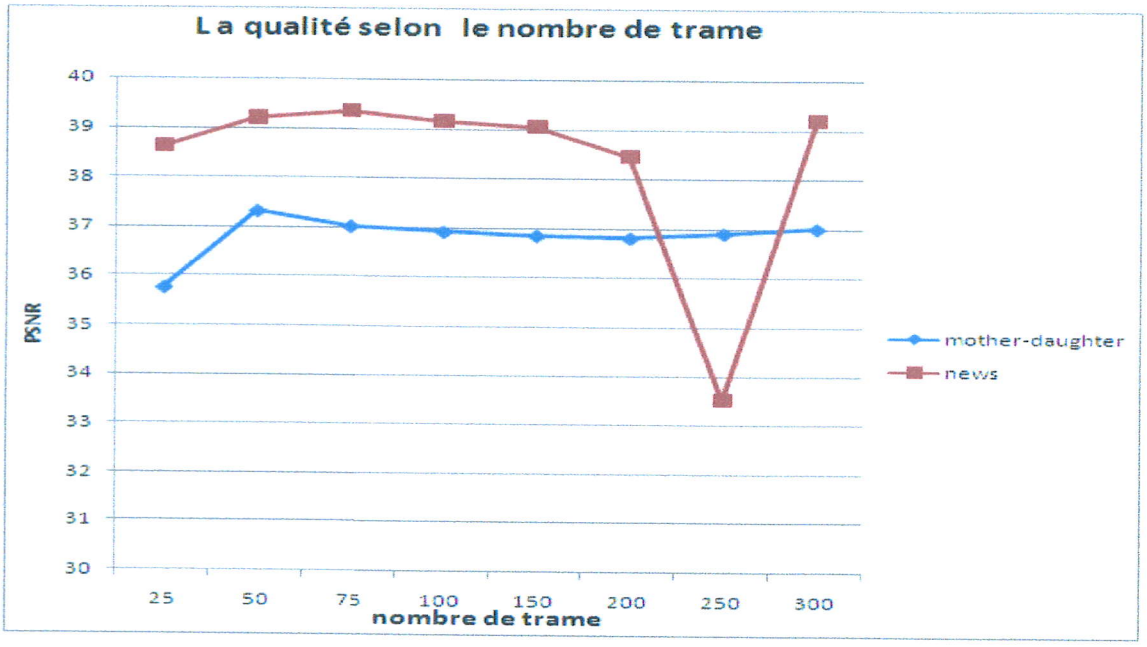


Figure 4.13 : le PSNR en fonction du nombre de trame

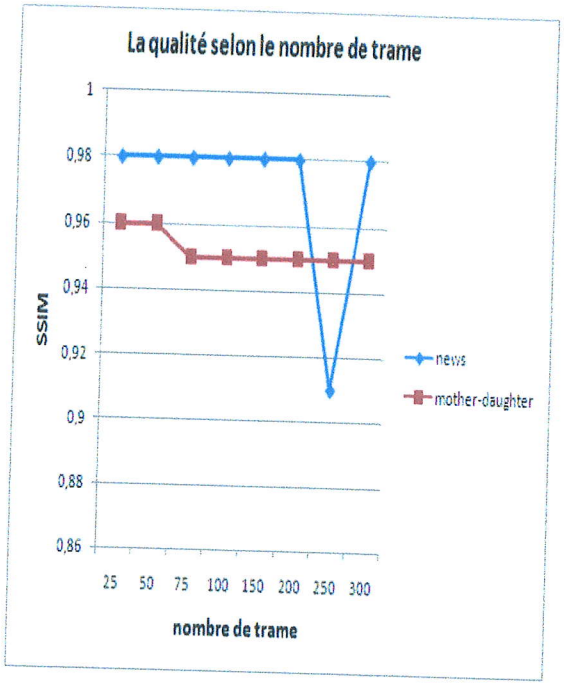


Figure 4.14 : le SSIM en fonction du nombre de trame

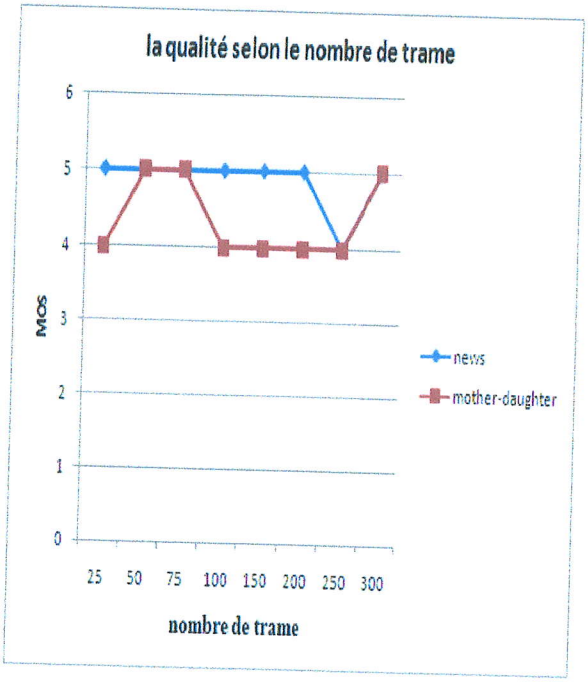


Figure 4.15 : le MOS en fonction du nombre de trame

4.6 Conclusion

Dans ce chapitre, nous avons expliqué l'opération de l'installation du software de référence JM et l'utilisation du fichier de configuration de l'encodeur et décodeur, Puis, nous avons installé l'outil d'évaluation Evalvid dans NS-2. Enfin, on a simulé la transmission d'un fichier vidéo de la norme H.264. L'évaluation que nous avons effectuée (simulation sous NS-2), nous a permis de voir l'impact du pas de quantification, le nombre de trame et la taille de trame sur la qualité de la vidéo transmise.

Conclusion générale

Le multimédia d'une manière générale et les services utilisant la vidéo d'une manière particulière ont évolué ces dernières années d'une manière très significative. En effet, l'utilisation des services multimédia s'est répandue dans la vie quotidienne du grand public mais aussi et surtout chez les professionnels. La transmission et l'acheminement des paquets multimédia, n'ont pas les mêmes contraintes que les autres types de données numériques que l'on utilise habituellement dans les réseaux informatiques. Sans compression il est très difficile de transmettre la vidéo sur un réseau filaire et sans fils. Les technologies de compression vidéo ont pour but de réduire et de supprimer les données vidéo redondantes de façon à permettre la transmission et le stockage efficaces d'un fichier vidéo numérique sur un réseau. H.264 est la norme de compression vidéo la plus récente et la plus performante.

L'évaluation de la qualité de l'image vidéo est l'une des étapes les plus importantes dans le domaine de transmission de la vidéo. Evalvid (Evaluation Vidéo), est un outil d'évaluation vidéo très utilisé dans les réseaux. Il consiste en un ensemble d'outils qui servent à l'évaluation de la qualité d'une vidéo transmise sur un réseau de communication réel ou simulé est ce du point de vue qualité de la vidéo perçue. Dans ce mémoire, nous avons effectué des simulations sur un réseau filaire et un autre sans fil en analysant l'impact des changements des principaux paramètres comme le pas de quantification, le nombre de trames et la taille de la trame sur la qualité de la vidéo. La simulation des réseaux est confiée au simulateur NS-2 (Network Simulator version 2). L'outil « Evalvid » devait donc s'intégrer à ce simulateur. Le principal objectif visé dans cette thèse a été de faire une simulation d'une transmission vidéo, et de faire une correcte combinaison entre NS-2 et Evalvid, de façon à pouvoir réaliser l'évaluation de la vidéo transmise selon plusieurs mesures comme PSNR, SSIM et MOS.

Bibliographie

[1] :M. BRAHMA, «*Étude de la QoS dans les Réseaux Ad hoc : Intégration du Concept de l'Ingénierie du Trafic*», Thèse de Doctorat, Université de Haute Alsace -Mulhouse, 13 décembre 2006.

[2] :http://nuxeo.edel.univ-poitiers.fr/nuxeo/site/esupversions/258b63e9-c522-407c-b73c-85f456a33ed0.

[3] :R. BOUDJADJA, « *Transmission par paquets de la vidéo en temps réel sur les réseaux sans fils* », Mémoire de Magister, Université Abderrahmane Mira de Bejaia ,2010.

[4] : H. SAADANE, « *La qualité de service d'un streaming vidéo Dans un réseau ad hoc (égale à égal)* », Mémoire de Magister, Université de Badji Mokhtar Annaba, 2012.

[5] :M. AZNI, «*Codage Conjoint Source Canal: Modèles et Application à la Transmission Vidéo*», thèse Pour l'obtention du grade de Docteur en sciences, Université Abderahmane mira de Bejaia, 01 juillet 2010.

[6] : A. NINASSI, « *de la perception locale des distorsions de codage à l'appréciation globale de la qualité visuelle des images et vidéos. apport de l'attention visuelle dans le jugement de qualité*», Thèse de Doctorat de l'Université de Nantes, Université de Nantes,17 mars 2009.

[7] : J. FRANÇOIS, « *optimisation d'algorithmes de codage vidéo sur des plateformes à plusieurs processeurs parallèles* », mémoire de l'obtention de la maîtrise en génie, école de technologie supérieure université du Québec, 10 janvier 2011.

[8] : J. GORIN, « *Pré-analyse de plans vidéos Haute Définition Application à l'optimisation du codage en flux H.264* », mémoire de master de recherche, institut de recherche en communications et en cybernétique de Nantes, 20 avril 2007.

[9] : N. ZEGLAM, « *Codec de la vidéo appliqué à la visiophonie sur IP* », Mémoire de Magister, Institut National de formation en Informatique (I.N.I), 15 Juillet 2008.

[10] : F. URBAN, « *Implantation optimisé d'estimateurs de mouvement pour la compression vidéo sous plates-formes hétérogène multi-composant* », Thèse de Doctorat, l'institut national des sciences appliquées de rennes, 6 Décembre 2007.

[11]:KLAUE, JIRKA, BERTHOLD RATHKE, and ADAM WOLISZ, « *A framework for video transmission and quality evaluation*»,Computer Performance Evaluation. Modelling Techniques and Tools. Springer Berlin Heidelberg, September 2003.

[12] : F.BOULOS, « *Etat de l'art des critères objectifs de qualité d'images et de vidéos* », Mémoire de Master de Recherche, Ecole Polytechnique de l'Université de Nantes, 21 Avril 2006.

[13] :*Optimisation de la bande passante pour le broadcast vidéo sur réseau IP Rapport bibliographique*, http://www.cri.ensmp.fr/~hurbain/biblio_DEA_hurbain.pdf.

[14] : M. BEN SALEM, O. BOUGOUFFA, « *Etude comparative de deux simulateurs pour les réseaux ad-hoc sans fil*», memoire de master, université KASDI MERBAH OUARGLA, 14 juin 2014.

[15] :T. HAFNAOUI, « *le streaming dans les reseaux ad hoc* »,mémoire de master, UNIVERSITÉ EL-HADJ LAKHDAR- BATNA, 24 juin 2013.

[16] : A. MEHIMEH, D. MOHDEB, « *Analyse de Compression Vidéo* », Mémoire de fin d'études pour l'obtention du diplôme Master de Recherche en Informatique, Université Mohamed Sadik BENYAHIA – JIJEL, Juillet 2012.

[17] :A. MICHAEL TOURAPIS, «*H.264/14496-10 AVC REFERENCE SOFTWARE MANUAL* », Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), Jan 2009.

[18] : Intégration NS2 et Evalvidhttp://csie.nqu.edu.tw/smallko/ns2_old/Evalvid_in_NS2.htm.

[19] : V.S UKANI , « Testing Video Transmission in NS2 usingthe Evalvid Framework», Thèse de Doctorat,Université de Nirma,Juillet 2015.