UNIVERSITAS INDONESIA

SIMULASI KOMPRESI SINYAL VIDEO DIGITAL KE SINYAL AUDIO

SKRIPSI

M RASYIDI FAKHRI

0906602912

FAKULTAS TEKNIK

PROGRAM STUDI TEKNIK ELEKTRO

DEPOK

JUNI 2012

UNIVERSITAS INDONESIA

SIMULASI KOMPRESI SINYAL VIDEO DIGITAL KE SINYAL AUDIO

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar sarjana

M RASYIDI FAKHRI

0906602912

DEPARTEMEN TEKNIK ELEKTRO

PROGRAM STUDI TEKNIK ELEKTRO EKSTENSI

DEPOK

JUNI 2012

HALAMAN PERNYATAAN ORISINALITAS

Tugas akhir ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama	**: M Rasyidi Fakhri**
NPM	**: 0906 602 912**
Tanda Tangan	**:** 
Tanggal	**: 5 Juli 2012**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh :

Nama	:	M Rasyidi Fakhri
NPM	:	0906 602 912
Program Studi	:	Ekstensi Teknik Elektro
Judul Tugas Akhir	:	Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia

DEWAN PENGUJI

Pemimbing : Dr. Ir. Arman D. Diponegoro ( )

Penguji : Prof. Dr. Ir. Harry Sudibyo DEA ( )

Penguji : Ir. Purnomo Sidi Priambodo M.Sc., Ph.D. ( )

Ditetapkan di : Depok
Tanggal : 5 Juli 2012

KATA PENGANTAR

Alhamdulillah, segala puji dan syukur penulis panjatkan kepada ALLAH SWT yang telah memberikan rahmat, hidayah, karunia, serta rezeki kepada penulis sehingga dapat menyelesaikan Tugas Akhir ini dengan baik.

Laporan Tugas Akhir dengan judul "**Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio**" ini disusun sebagai salah satu syarat untuk menyelesaikan masa studi dan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro, Fakultas Teknik, Universitas Indonesia. Selain itu, saya juga ingin mengucapkan banyak terima kasih pada beberapa pihak yang telah membantu penulis baik selama masa studi maupun dalam penyusunan Tugas Akhir, antara lain:

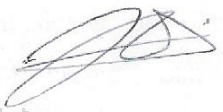
1. ALLAH SWT yang telah melimpahkan rahmat serta hidayah-Nya.
2. Kedua orang tua yang telah mendidik, membesarkan, merawat, dan membiayai pendidikan penulis hingga saat ini.
3. Dr. Ir. Arman D. Diponegoro, selaku dosen pembimbing yang telah memberikan petunjuk, kemudahan dalam berpikir dan bimbingan dalam penyelesaian tugas akhir ini.
4. Dosen – dosen pengajar FT UI.
5. Rekan-rekan Mahasiswa Ekstensi Elektro angkatan 2009.
6. Dan semua pihak yang tidak dapat saya sebutkan satu – per – satu.

Semoga penulisan ilmiah ini benar-benar dapat memberikan kontribusi positif dan menimbulkan sikap kritis kepada para pembaca khususnya dan masyarakat pada umumnya untuk senantiasa terus memperoleh wawasan dan ilmu pengetahuan di bidang teknologi dan sain.

Penulis menyadari keterbatasan pengalaman dan kemampuan yang tentu terdapat kekurangan serta kemungkinan jauh dari sempurna, untuk itu saya tidak menutup diri dan mengharapkan adanya saran serta kritik dari berbagai pihak yang sifatnya membangun guna menyempurnakan penulisan ilmiah ini.

Akhir kata semoga penulisan ilmiah ini dapat memberikan manfaat bagi semua pihak yang bersangkutan, khususnya bagi saya dan umumnya bagi para pembaca.

Depok, 5 Juli 2012



M Rasyidi Fakhri

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini:

Nama	: M Rasyidi Fakhri
NPM	: 0906 602 912
Program Studi	: Ekstensi Teknik Elektro
Departemen	: Elektro
Fakultas	: Teknik
Jenis karya	: Skripsi

demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia **Hak Bebas Royalti Noneksklusif (*Non-exclusive Royalty-Free Right*)** atas karya ilmiah saya yang berjudul :

"Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio"

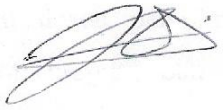
beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (*database*), merawat, dan memublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok

Pada tanggal : 5 Juli 2012

Yang menyatakan



(M Rasyidi Fakhri)

ABSTRAK

Nama : M Rasyidi Fakhri (0906602912)

Program Studi : Teknik Elektro

Judul : Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio

Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio merupakan suatu sistem kompresi data video digital dengan bitrate yang tinggi menjadi data video digital dengan bitrate yang rendah agar dapat dilewatkan pada transmisi sinyal audio yang bandwidthnya sebesar 64 Kbps. Sistem ini menggunakan kompresi Windows Media Video 9 karena dianggap mampu melakukan kompresi data video digital hingga 64 Kbps. Dengan mode encoding Constant Bit Rate (CBR), bitrate data video akan tetap konstan atau mendekati target bitrate yang sudah diatur sebelumnya.

Kata Kunci : sinyal, video, digital, kompresi, bitrate, bandwidth

ABSTRACT

Name : M Rasyidi Fakhri (0906602912)
Program of Study : Electrical Engineering
Title : Simulation of Digital Video Compression Signal to Audio Signal

Simulation of Digital Video Compression Signal to Audio Signal is a digital video data compression system with a high bitrate digital video data to a lower bitrate in order to pass the audio signal transmission bandwidth of 64 Kbps. The system uses Windows Media Video 9 compression because it is able to perform data compression digital video of up to 64 Kbps. With Constant Bit Rate encoding mode (CBR), bitrate video data will remain constant or close to the target bitrate is prearranged.

Keyword : signal, video, digital, compression, bitrate, bandwidth

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pengolahan sinyal data digital saat ini sudah sangat berkembang. Dengan adanya teknik kompresi, data digital yang pada dasarnya ukuran datanya cukup besar dapat diperkecil. Semakin kecil data hasil kompresinya, maka resiko penurunan kualitas data akan semakin tinggi.

Hal ini berbanding terbalik dengan sinyal audio atau analog dimana ukuran data *bandwidth*nya cukup kecil dan terbatas. Kompresi dalam pengolahan sinyal analog yaitu dengan mengurangi *power* sinyal (dB), berbeda dengan kompresi sinyal digital yaitu dengan merubah sinyal informasi secara matematis sehingga ukuran data menjadi lebih kecil.

Kompresi pada sistem ini menggunakan codec windows media video. Codec ini memiliki kemampuan kompresi pada Multiple bit rate (MBR) yaitu kemampuan kompresi pada berbagai bandwidth transmisi (Low, Medium dan High). Selain itu juga, codec ini memiliki teknik encoding Constant bit rate (CBR) yaitu mampu menjaga bit rate video tetap stabil atau selalu mendekati target bitrate sehingga kualitas video tetap terjaga.

Hal ini yang melatarbelakangi penulis untuk membuat skripsi yang berjudul **"Simulasi Kompresi Sinyal Video Digital ke Sinyal Audio"**. Dimana data video digital dikompresi sekecil mungkin hingga mampu dikirim melalui transmisi audio atau analog yang *bandwidth* transmisinya cukup kecil seperti jalur telepon, satelit dan lain-lain yaitu sebesar 64 Kbps.

1.2 Perumusan Masalah

Perumusan masalah dari penulisan skripsi ini yakni:

1. Bagaimana menentukan *encoder* dan *decoder* (*codec*) yang tepat untuk kompresi data video agar dapat dilewatkan pada *bandwidth* sebesar 64 kbps.

2. Bagaimana menentukan konfigurasi *encoder* (*bitrate*, *buffer size dan frame rate*) agar didapatkan kualitas hasil kompresi yang tidak terlalu buruk

1.3 Tujuan Penulisan

Adapun tujuan dari skripsi ini, yakni agar dapat mengirimkan data video digital melalui transmisi audio atau analog yang *bandwidth*nya relative kecil seperti jalur telepon, satelit dan lain-lain.

1.4 Pembatasan Masalah

Adapun yang menjadi batasan masalah pada pembuatan skripsi ini, yakni:

1. Bersifat *live streaming* dengan delay
2. Menggunakan video kompresi *Windows Media Video* 9 sehingga akan menghasilkan format file video .wmv
3. Parameter encoding seperti frame rate, bitrate dan buffer size sudah ditentukan sebelumnya.
4. Aplikasi *Encoder* dan *Decoder* yang dimodifikasi ialah *Windows Media Encoder* dan *Video Lan Converter*
5. Teknik Encoding yang digunakan adalah *Constant Bit Rate*
6. Protokol yang dipakai ialah HTTP
7. Tidak membahas ADC

1.5 Metodologi Penelitian

1. Studi literatur

   Mencari informasi dari berbagai sumber seperti: jurnal, buku, pencarian melalui internet, dan artikel-artikel mengenai beberapa topik seperti: *video streaming*, *codecs*, *Encoder*, *Decoder*, *HTTP*, teknik kompresi, dan lain-lain.

2. Perancangan dan Simulasi Sistem

   Merupakan suatu proses perancangan dan simulasi proses kompresi data video pada jaringan LAN

3. Pengukuran Sistem

   Proses pengujian dilakukan pada proses encoding dan decoding pada beberapa waktu yang berbeda, tujuannya untuk melakukan perbandingan dan mengetahui kinerjanya (QoS).

4. Analisa Pengukuran

   Dari pengukuran dan pengambilan data kemudian dilakukan suatu analisa sehingga dapat diambil suatu kesimpulan

BAB II
LANDASAN TEORI

2.1 *Video Streaming*

Video adalah teknologi pengiriman sinyal elektronik dari suatu gambar yang bergerak. *Video* digital adalah sistem perekaman digital yang bekerja dengan menggunakan sinyal digital. *Video* digital terdiri dari serangkaian gambar digital bitmap yang ortogonal yang ditampilkan dengan cepat pada kecepatan yang konstan. Dalam konteks *video*, gambar disebut dengan *frame*. *Frame* digunakan sebagai tingkat ukuran banyaknya *frame* yang ditampilkan setiap detiknya yang disebut *frame* per second (FPS).

Setiap *frame* terdiri dari susunan piksel berwarna yang membentuk gambar. Jika memiliki lebar piksel W dan ketinggian piksel H kita katakan bahwa ukuran *frame* WxH. Piksel hanya memiliki satu properti, yaitu warna. Warna pixel diwakili oleh sejumlah bit yang tetap. Semakin besar nilai Bitnya, maka warna yang dihasilkan akan lebih bervariasi dan lebih halus. Hal ini disebut *Color Depth* (CD) dari *video*.[1]

Video Streaming merupakan sebuah teknologi yang mampu mengkompresi atau menyusutkan ukuran *file video* agar mudah ditransfer melalui jaringan *Internet*. Pen*transfer*an *file video* tersebut dilakukan secara terus-menerus. Dari sudut pandang prosesnya, *streaming* berarti sebuah teknologi pengiriman *file* dari *server* ke *client* melalui jaringan *packet-based*. *streaming* merupakan sebuah metode untuk membuat *video* yang tersedia untuk *real-time* pada tipe jaringan yang berbeda. Data pada *file streaming* dibagi-bagi ke dalam beberapa paket kecil yang dikirim ke sebuah aliran secara terus menerus ke komputer *end-user* atau *mobile phone*.[2]

Aplikasi dalam layanan *streaming* dibagi menjadi dua, yaitu "*on-demand*" dan "*live*". Sebagai contoh yang termasuk dalam kategori "*on-demand*" yakni musik dan *video*, sedangkan yang termasuk "*live*" yakni acara radio atau televisi yang disiarkan secara *broadcasting* pada saat itu juga.

Ide dasar dari *video streaming* adalah untuk membagi-bagi *video* asli menjadi beberapa *packet* yang kemudian dikirim secara berurutan, dan

memungkinkan *receiver* melakukan *decode* dan *playback video* berdasarkan *packet* tersebut tanpa harus menunggu seluruh *video* terkirim. Ada beberapa masalah dasar dalam *video streaming*, yakni:

a. Bandwidth

Jika pengirim mengirimkan data lebih cepat dibandingkan dengan ketersediaan *bandwdith* yang ada, maka akan terjadi *congestion* (kemacetan) pada jaringan, *packet loss*, dan kualitas *video* yang jelek. Tapi apabila pengirim mengirimkan paket data lebih lambat dari *bandwidth* yang tersedia, maka kualitas *video* yang sampai ke penerima akan kurang optimal. Salah satu cara untuk mengatasinya adalah dengan melakukan estimasi terhadap ketersedian *bandwidth*, yang kemudian mencocokan dengan *bit rate video* yang akan ditransmisikan.

b. *Delay Jitter*

Delay jitter merupakan masalah yang menyebabkan penerima harus menerima/*decode*/menampilkan *frame* dengan *rate* yang konstan, dan setiap *frame* yang terlambat datang akan menyulitkan rekonstruksi *video* yang diterima.

c. Loss Rate

Pada jaringan *wireless*, *loss rate* dapat disebabkan oleh *bit error* dan *bursh error*

2.2 Hypertext Transfer Protocol

Hypertext Transfer Protocol (HTTP) adalah sebuah protokol jaringan lapisan aplikasi yang digunakan untuk sistem informasi yang terdistribusi, kolaboratif, dan menggunakan *hypermedia*. Penggunaannya banyak pada pengambilan sumber daya yang saling terhubung dengan jaringan, yang disebut dengan dokumen *hypertexts*, yang kemudian membentuk *World Wide Web* pada tahun 1990 oleh fisikawan Inggris, Tim Berners-Lee. Hingga kini, ada dua versi mayor dari protokol HTTP, yakni HTTP/1.0 yang menggunakan koneksi terpisah untuk setiap dokumen, dan HTTP/1.1 yang dapat menggunakan koneksi yang sama untuk melakukan transaksi. Dengan demikian, HTTP/1.1 bisa lebih cepat karena memang tidak perlu membuang waktu untuk pembuatan koneksi berulang-ulang.[3]

Pengembangan standar HTTP telah dilaksanakan oleh Konsorsium World Wide *Web* (World Wide *Web* Consortium/W3C) dan juga *Internet Engineering Task Force* (IETF), yang berujung pada publikasi beberapa dokumen *Request for Comments* (RFC), dan yang paling banyak dirujuk adalah RFC 2616 (yang dipublikasikan pada bulan Juni 1999), yang mendefinisikan HTTP/1.1.

HTTP adalah sebuah protokol meminta/menjawab antara *client* dan *server*. Sebuah *client* HTTP (seperti *web browser* atau robot dan lain sebagainya), biasanya memulai permintaan dengan membuat hubungan ke port tertentu di sebuah *server Webhosting* tertentu (biasanya *port* 80). *Client* yang mengirimkan permintaan HTTP juga dikenal dengan *user agent*. *Server* yang meresponsnya, yang menyimpan sumber daya seperti berkas HTML dan gambar, dikenal juga sebagai *origin server*. Di antara user agent dan juga *origin server*, bisa saja ada penghubung, seperti halnya *proxy*, *gateway*, dan juga *tunnel*.

HTTP tidaklah terbatas untuk penggunaan dengan TCP/IP, meskipun HTTP merupakan salah satu protokol aplikasi TCP/IP paling populer melalui *Internet*. Memang HTTP dapat diimplementasikan di atas protokol yang lain di atas *Internet* atau di atas jaringan lainnya. seperti disebutkan dalam "*implemented on top of any other protocol on the Internet, or on other networks.*", tapi HTTP membutuhkan sebuah protokol lapisan transport yang dapat diandalkan. Protokol lainnya yang menyediakan layanan dan jaminan seperti itu juga dapat digunakan.

Sumber daya yang hendak diakses dengan menggunakan HTTP diidentifikasi dengan menggunakan *Uniform Resource Identifier* (URI), atau lebih khusus melalui *Uniform Resource Locator* (URL), menggunakan skema URI http: atau https:.

2.2.1 HTTP *Live Streaming*

HTTP *Live Streaming* (juga dikenal sebagai HLS) adalah HTTP berbasis media komunikasi *streaming* protokol yang diimplementasikan oleh *Apple Inc* sebagai bagian dari *QuickTime X* dan sistem perangkat lunak *iPhone*. Protocol ini bekerja dengan memecah keseluruhan data *stream* menjadi urutan kecil dari HTTP berbasis download *file*, setiap download memuat salah satu short chunk dari keseluruhan transportasi data *stream* tak terbatas. Pada saat data *stream* dimainkan, *client* dapat memilih dari sejumlah alternatif data *stream* yang berbeda

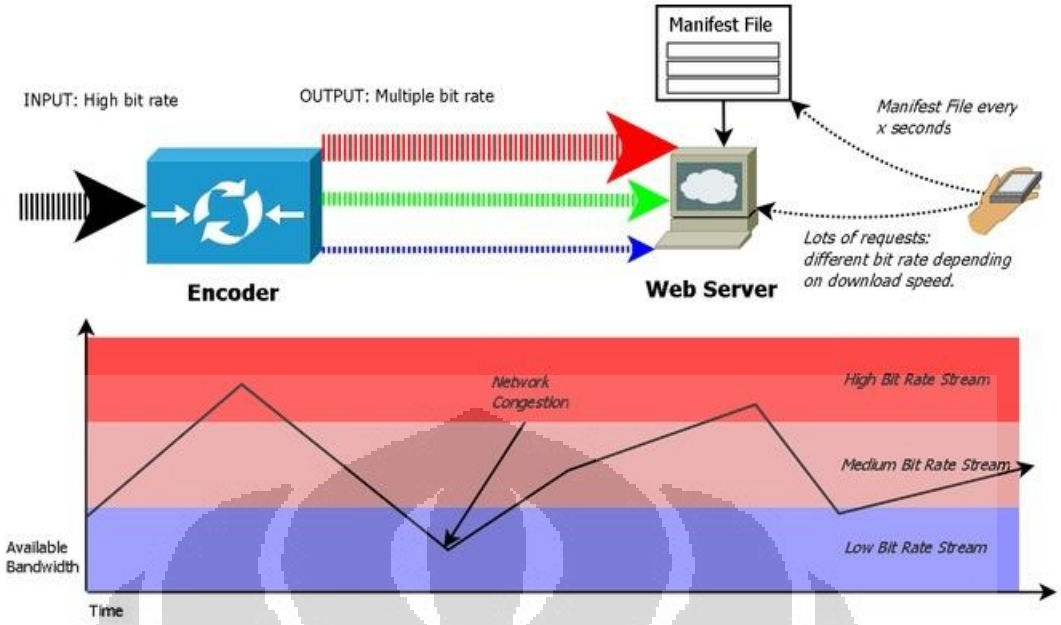
yang berisi data yang sama dikodekan pada berbagai kecepatan data, yang memungkinkan sesi *streaming* untuk beradaptasi dengan laju data yang tersedia. Pada awal sesi *streaming*, protocol ini akan mendownload *file* M3U (m3u8) *playlist* yang berisi metadata untuk sub-berbagai data *stream* yang tersedia[4].

Karena permintaan ini yang hanya menggunakan standar transaksi HTTP, HTTP *Live Streaming* mampu melintasi setiap *firewall* atau *server proxy* yang memungkinkan koneksi melalui standar HTTP *traffic*, tidak seperti UDP berbasis protokol seperti RTP. Hal ini juga memungkinkan jaringan pengiriman konten dengan mudah diimplementasikan untuk setiap data *stream* tertentu.

2.2.2 Adaptive *Bitrate Streaming*

Adaptive *bitrate streaming* adalah teknik yang digunakan dalam *streaming* multimedia melalui jaringan komputer. Sementara di sebagian besar teknologi *streaming video* lama menggunakan protokol *streaming* RTP dan RTSP, teknologi adaptif *streaming* saat ini hampir secara semuanya berbasis HTTP dan dirancang untuk bekerja secara efisien pada distribusi jaringan HTTP yang lebih besar seperti *Internet*.

Adaptive *bitrate streaming* bekerja dengan mendeteksi *bandwidth* pengguna dan kapasitas CPU secara *real* time dan menyesuaikan kualitas data *video streaming* dengan tepat. Hal ini membutuhkan penggunaan sebuah program *encoder* yang dapat mengkodekan single source *video* pada *multiple bit rate*s. Player *client* beralih ke pengkodean data *streaming* yang berbeda tergantung pada sumber daya yang tersedia. Hasilnya, *Buffer*ing sangat sedikit, waktu mulai lebih cepat dan bagus untuk koneksi *high-end* ataupun *low-end*.[5]



Gambar 2.1 Adaptive Bit Rate Overview

2.3 *Codec*

Codec adalah sebuah perangkat atau program komputer yang mampu melakukan *encoding* atau *decoding* sinyal data *stream* baik digital maupun analog. Kata *codec* berarti "*encoder-decoder*" atau umumnya dikenal dengan istilah "*compressor*-decompresssor". Sebuah *codec* (program) tidak harus memusingkan dengan format coding, kompresi atau standarnya. format adalah dokumen (standar), suatu cara untuk menyimpan data, sedangkan *codec* adalah sebuah program (implementasi) yang dapat membaca atau menulis *file* tersebut. Dalam prakteknya, bagaimanapun, "*codec*" kadang-kadang digunakan untuk merujuk pada format *file*nya.[6]

Codec mengkodekan data *stream* atau sinyal untuk transmisi, penyimpanan atau enkripsi, atau men-decode untuk pemutaran atau *editing*. *Codec* digunakan dalam konferensi *video*, *streaming* media dan aplikasi *video editing*. Sebuah kamera *video* konverter analog-to-digital (ADC) mengubah sinyal analog menjadi sinyal digital, yang kemudian melewati sebuah kompresor *video* untuk transmisi digital atau penyimpanan. Sebuah perangkat penerima kemudian menjalankan sinyal melalui *decompressor video*, kemudian konverter digital-ke-analog (DAC) untuk tampilan analog. Istilah *codec* ini juga digunakan sebagai nama generik untuk unit konferensi *video*.

2.3.1 Kualitas Kompresi

a. *Lossy codec*s:

Kebanyakan program *codec* yang ada didunia adalah *lossy*, *codec* tipe ini mengurangi kualitas data untuk kompresinya. Seringkali, jenis kompresi ini hampir tidak bisa dibedakan dari suara atau gambar asli yang terkompresi, tergantung pada *codec* dan pengaturan yang digunakan. Dengan besar data yang lebih kecil, mampu mengurangi penggunaan media penyimpanan besar yang relatif mahal seperti *memory card* dan *hard disk*, serta disc storage seperti CD-ROM, DVD dan *Blu-ray Disc*. Kecepatan data yang lebih rendah juga mengurangi biaya dan meningkatkan kinerja ketika data ditransmisikan.

b. *Lossless codec*s

Ada cukup banyak juga *codec lossless* yang biasanya digunakan untuk pengarsipan data dalam bentuk kompresi namun tetap menyimpan semua informasi data dalam original *stream*nya. *Codec* tipe ini memilih menjaga kualitas asli dari data *stream*nya daripada mengurangi ukuran datanya. Hal ini biasanya terjadi jika data adalah akan menjalani proses lebih lanjut (untuk mengedit misalnya) dalam hal aplikasi pengolahan berulang (*encoding* dan *decoding*) pada *codec lossy* akan menurunkan kualitas data yang dihasilkan sedemikian rupa sehingga tidak lagi diidentifikasi (visual, terdengar atau keduanya). Menggunakan lebih dari satu *codec* atau skema *encoding* secara berturut-turut juga dapat menurunkan kualitas secara signifikan, sehingga terjadi pengurangan kapasitas penyimpanan data dan *bandwidth* jaringan.

2.3.2 Media *Codec*

Codec sering dirancang untuk menekankan aspek-aspek tertentu dari media, atau penggunaannya, untuk dikodekan. Sebagai contoh, sebuah *video* digital (menggunakan *codec* DV) dari sebuah acara olahraga perlu untuk mengkodekan gerak juga tapi tidak perlu mengkodekan warna yang tepat, sementara *video* dari pameran seni perlu untuk mengkodekan warna dan tekstur permukaan juga. Ada ribuan *codec video,* ada yang gratis dan ada juga yang berbayar. variasi *codec* ini dapat membuat masalah kompatibilitas dan tren penggunaan.

Codec bitrate yang rendah memungkinkan lebih banyak pengguna, tetapi mereka juga memiliki distorsi lebih. Di luar peningkatan awal dalam distorsi, *codec bitrate* yang rendah juga dapat mencapai *bitrate* yang lebih rendah lagi dengan menggunakan algoritma yang lebih kompleks yang membuat asumsi tertentu, seperti tentang media dan tingkat packet loss. *Codec* lain mungkin tidak membuat asumsi-asumsi yang sama. Ketika seorang pengguna dengan *codec bitrate* rendah berhubungan dengan pengguna dengan *codec* yang lain, ada distorsi tambahan pada proses *transcoding* di masing-masing pengguna.

AVI kadang-kadang keliru digambarkan sebagai *codec*, tetapi AVI sebenarnya merupakan format kontainer, sedangkan *codec* adalah sebuah perangkat lunak atau perangkat keras yang *encoding decoding* audio atau *video* ke dalam beberapa format audio atau *video*. Audio dan *video* encode dengan banyak *codec* mungkin dimasukkan ke dalam format AVI, walaupun AVI bukan merupakan standar ISO. Ada juga format kontainer terkenal lainnya, seperti Ogg, ASF, *QuickTime*, *RealMedia*, *Matroska*, dan DivX *Media Format*. Beberapa format kontainer yang standar ISO adalah MPEG transport *stream*, MPEG program *stream*, MP4 dan ISO berbasis *format* media *file*.

2.3.3 *Windows Media Video*

Windows Media Video (WMV) adalah format *video* yang paling dikenal dalam keluarga WMV. Penggunaan dari istilah WMV sering mengacu pada *codec Microsoft Windows Media Video* saja. Pesaing utamanya adalah MPEG-4 AVC, AVS, *Realvideo*, dan MPEG-4 ASP. Versi pertama dari *codec* ini adalah WMV 7, diperkenalkan pada 1999, dan dibangun di atas implementasi *Microsoft* dari MPEG-4 Bagian 2. Pengembangan versi terbaru dari *codec* ini, tetapi sintaks bit *stream* tidak dibekukan sampai WMV 9. Sementara semua versi dari WMV menyediakan konten variable *bit rate*, average *bit rate*, dan constant *bit rate*,, WMV 9 diperkenalkan beberapa fitur penting termasuk dukungan asli untuk *interlaced video*, *non-square pixel*, dan *interpolasi frame*. WMV 9 juga memperkenalkan profil baru berjudul *Windows Media Video 9 Professional*, yang diaktifkan secara otomatis setiap kali resolusi *video* melebihi 300.000 piksel (misalnya, 528x576, 640 × 480 atau 768x432 dan seterusnya) dan *bitrate* 1000

kbps. Hal ini ditargetkan untuk konten *high-definition video*, dengan resolusi seperti 720p dan 1080p.[7]

Tingkat profil Sederhana dan Main di WMV 9 telah sesuai dengan tingkat profil yang sama dalam spesifikasi VC-1. Profil Lanjutan di VC-1 diimplementasikan dalam sebuah *codec* WMV baru yang disebut *Windows Media Video* 9 *Advanced Profil*. Ini meningkatkan efisiensi kompresi untuk konten interlaced dan dibuat transportasi-independen, sehingga dapat dirumuskan dalam MPEG transport *stream* atau format paket RTP. *Codec* ini tidak kompatibel dengan versi *codec* WMV 9 sebelumnya, namun WMV merupakan *video codec* wajib bagi PlaysForSure-bersertifikat toko perangkat online, serta perangkat *Portable Media Center*. *Zune Microsoft, Xbox* 360, perangkat *Windows Mobile* yang mendukung *Windows Media* Player, serta banyak perangkat uncertified lainnya yang mendukung *codec*. WMV HD mengatur penggunaan WMV 9 untuk program sertifikasinya, pada tingkat kualitas yang ditetapkan oleh *Microsoft*, WMV digunakan untuk menjadi *video codec* yang hanya didukung untuk *platform Microsoft* Silverlight, tetapi *codec* H.264 versi 3 sekarang juga didukung *platform* tersebut.

2.3.4 *Windows Media Encoder*

Windows Media Encoder adalah media *encoder* yang dikembangkan oleh *Microsoft* yang memungkinkan pengembang konten dapat mengubah atau menangkap audio, *video* dan gambar layar komputer baik secara *live* ataupun yang telah direkam sebelumnya ke format *Windows Media* yang dapat dikirim secara *live* ataupun on demand. *Windows Media Encoder* adalah penerus dari *Encoder Netshow*.[8]

Windows Media Encoder 9 Series memungkinkan two-pass *encoding* untuk mengoptimalkan kualitas untuk konten on-demand (*stream* atau download-and-play). *Encoding* ini juga mendukung variable *bitrate encoding* (VBR) skenario download-and-play. VBR dapat mengaplikasikan urutan high-motion di seluruh durasinya, untuk memastikan kualitas yang terbaik. Versi ini juga memungkinkan pengkodean ditulis dengan *file* VBScript wmcmd.vbs sehingga memungkinkan pengembang konten untuk mengkodekan sejumlah besar *file* media rekaman.

Windows Media Encoder dapat mengkodekan konten audio dan *video* dengan dua cara yaitu constant *bit rate* (CBR) dan variable *bit rate* (VBR)

a. CBR *encoding*

CBR *encoding* dirancang untuk bekerja paling efektif dalam skenario *streaming*. Dengan CBR *encoding*, *bit rate* tetap cukup secara konstan dan tetap dekat dengan target *bit rate* selama *streaming*, dengan *frame* per second yang diatur oleh *buffer size*-nya. Kerugian dari CBR *encoding* adalah bahwa kualitas konten yang di encode tidak konstan. Karena ada beberapa packet data dari konten sulit untuk dikompres daripada yang lain, beberapa bagian dari aliran CBR memiliki kualitas lebih rendah daripada yang lain. Selain itu, kualitas hasil *encoding* CBR tidak konsisten dari satu *streaming* ke yang *streaming* berikutnya. Secara umum, variasi kualitas lebih ditunjukkan pada *bit rate* yang rendah.

b. VBR *encoding*

VBR *encoding* dirancang untuk distribusi konten untuk men-download dan memainkan data baik secara lokal atau pada perangkat yang memiliki kecepatan membaca terbatas, seperti CD atau DVD player. (dapat menggunakan modus peak VBR *encoding* ketika melakukan *streaming* konten). VBR *encoding* paling menguntungkan saat *encoding* konten yang berisi campuran dari data yang sederhana dan kompleks, misalnya, sebuah *video* yang beralih antara gerakan lambat dan cepat. Dengan VBR *encoding*, bit yang lebih sedikit secara otomatis dialokasikan ke bagian konten yang kurang kompleks, menyediakan bit yang cukup untuk menghasilkan kualitas yang lebih baik dengan tingkat kompleksitas yang lebih. Ketika digunakan pada konten campuran, VBR *encoding* menghasilkan output *encoding* yang jauh lebih baik mengingat ukuran *file* yang sama bila dibandingkan dengan *encoding* CBR. Dalam beberapa kasus dapat dihasilkan dengan sebuah *file* VBR-encoded yang memiliki kualitas yang sama dengan *file* CBR-encoded yang ukuran *file*nya setengahnya VBR-encoded.

2.4 *Windows Media Server* dan *Web Server*

Untuk mengirim konten dari *server* ke *client*, dapat menggunakan *Windows Media Server* atau dari *WebServer* ke *client*. *Server* dan *client* dapat berada dalam jaringan *Internet* atau intranet, dan dapat dipisahkan oleh *firewall*.

Meskipun *Windows Media server* dirancang khusus untuk *streaming Windows Media* berbasis konten, sedangkan standar *Web server* tidak. Jika menggunakan *Web server*, akan ada perbedaan dalam cara konten dikirimkan, yang dapat mempengaruhi kualitas pemutaran.

2.4.1 *Web server*

Sebuah *Web server* ini dirancang untuk men-download data sebanyak dan secepat mungkin. Ini adalah metode yang dipilih untuk mengirimkan paket yang berisi gambar statis, teks, dan script halaman *Web*, tetapi bukan metode terbaik untuk mengirimkan paket yang berisi media *streaming*. Media *streaming* harus disampaikan secara *real* time, bukan dalam semburan besar, dan *client* harus menerima paket tepat setelah di-render.

Web server tidak mendukung konten MBR. Ketika *stream file* dari *Web server*, kualitas pengiriman tidak diawasi dan *bit rate*nya tidak dapat disesuaikan. *Web server* tidak dapat menggunakan protokol pengiriman yang dipilih, User Datagram Protocol (UDP), sehingga pengiriman data *stream* mungkin akan terganggu oleh waktu dari proses *buffer* data *client*. Selain itu, *Web server* tidak mendukung siaran langsung dan aliran *multicast*.

2.4.2 *Windows Media server*

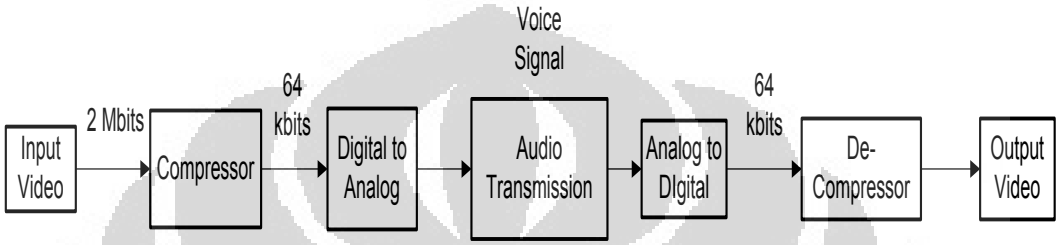
Sebuah *Windows Media Server* menghitung pengiriman paket sesuai dengan informasi feedback yang diterimanya saat mengirim data *stream* ke *client*. Ketika *client* menerima paket dengan cara ini, presentasinya mungkin menjadi jauh lebih lancar. Karena penggunaan *bandwidth* yang dikendalikan, sehingga lebih banyak *client* dapat terhubung secara bersamaan ke *server* dan menerima data *stream* yang bebas dari interupsi.

Untuk menyampaikan konten sebagai data *stream unicast* dari *Windows Media server*, data *stream* dapat dikodekan menjadi *multiple-bit-rate* (MBR). Konten ini memungkinkan *client* medapat konten yang lebih berkualitas pada saat kondisi jaringan penuh. Ketika konten MBR diterima oleh *client*, besar data *bit rate* yang di *streaming* disesuaikan dengan kondisi jaringan.

BAB III
PERANCANGAN SISTEM

3.1 Rancangan Simulasi Sistem Kompresi

Sistem Aplikasi "**Simulasi Kompresi Sinyal *Video* Digital ke Sinyal *Audio***" dapat dilihat pada gambar 3-1 dibawah ini.

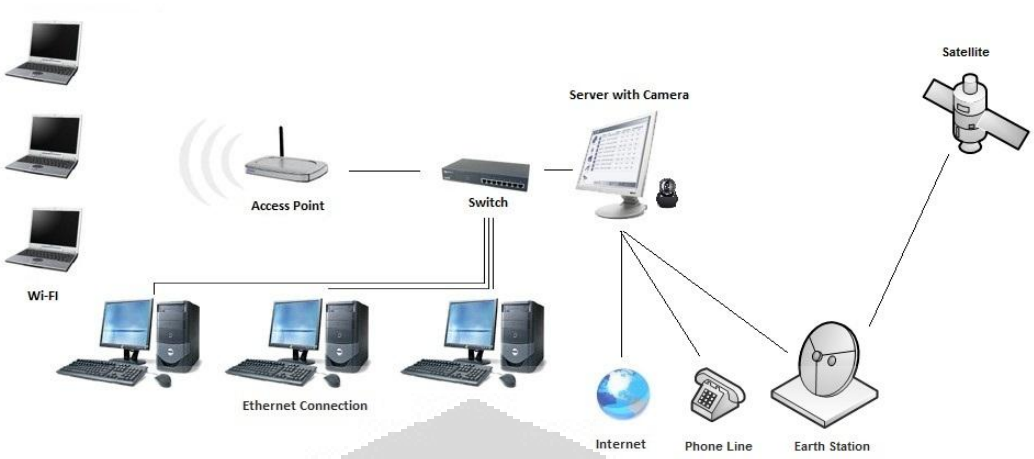


Gambar 3-1 Blok Diagram Sistem

Pada Gambar 3-1 diatas menjelaskan blok diagram sistem input data *video* yang ditangkap atau direkam oleh kamera. Data *video* digital tersebut lalu diolah oleh *Encoder* untuk dikompresi. Data *video* digital yang memiliki *bitrate* yang tinggi dikompresi hingga *bitrate*nya sebesar 64 Kbps. Kompresi ini dilakukan agar data *video* digital dapat dikirim melalui transmisi sinyal *audio* yang *bitrate*nya 64 Kbps, contohnya transmisi satelit, kabel telepon, dan lain-lain. Data *video* digital yang sudah dikompresi lalu diubah sinyalnya menjadi sinyal analog oleh Analog-Digital *Converter* (ADC) agar dapat dikirim melalui transmisi *audio* (analog).

Compressor dapat dikatakan sebagai perangkat computer yang memproses kompresi dengan menggunakan *codec video*. *Codec video* yang digunakan ialah *Windows Media Video* 9. Untuk perangkat Analog-Digital *Converter* sendiri tidak dibahas dalam skripsi ini.

Sistem ini disimulasikan pada jaringan LAN dengan *bitrate* yang diatur sesuai dengan bandwidth transmisi sinyal *audio* (64 Kbps). Protokol yang digunakan ialah HTTP Live *Streaming* dengan Teknik *Adaptive Bitrate Streaming* yang mampu menyesuaikan kualitas *video streaming* dengan kapasitas bandwidth jaringan dan kemampuan CPU dalam memproses kompresi.



Gambar 3-2 Simulasi Sistem

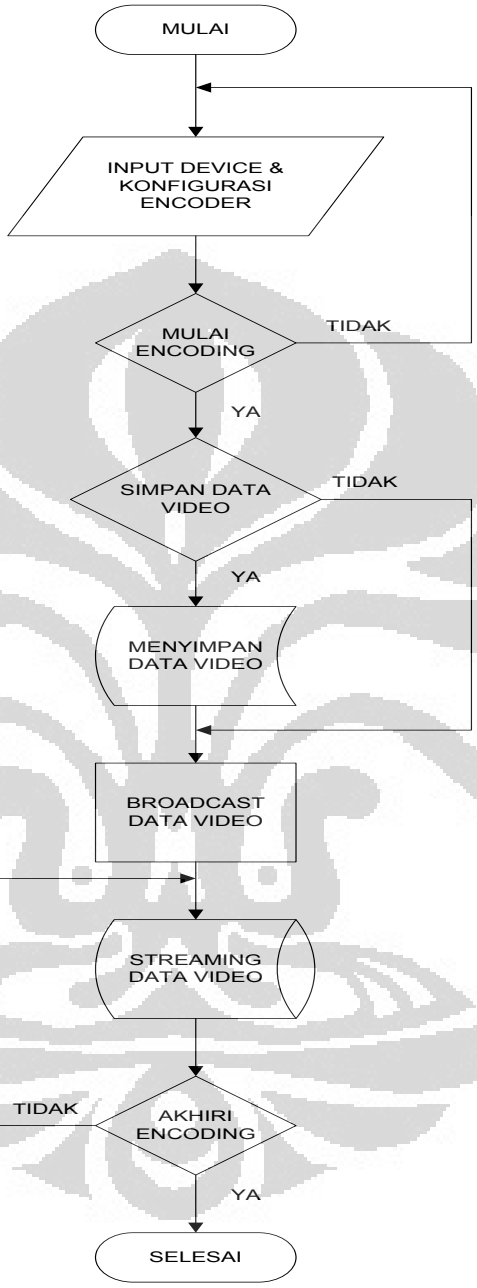
Gambar 3-2 diatas yaitu gambar simulasi sistem pada aplikasi jaringan antara *server* dengan *client*. Cara kerjanya yaitu objek yang ditangkap oleh Camera di sisi *server* lalu di encodingkan dengan menggunakan aplikasi *Windows Media Encoder* 9 dimana data *video* dikompres dengan konfigurasi yang sudah diatur sebelumnya yaitu *bitrate* 64 kbps, *frame rate* 12,5 fps, buffer size 5 second, key *frame* interval 8 second, dan *video* smoothness 70 (sharper).

Encoding yang digunakan adalah Constant *Bit Rate* (CBR) karena encoding tipe ini memang dirancang untuk bekerja paling efektif dalam skenario *streaming*. Dengan encoding CBR, *bit rate* akan tetap secara konstan dan mendekati target *bit rate* (64 kbps) selama proses encoding.

Data *video* yang telah di encoding kemudian di kirim ke *client* melalui jaringan LAN. *Client* menggunakan aplikasi *Video* Lan *Converter* (VLC) untuk menerima data *video streaming* yang dikirimkan. Data *video* yang telah diterima oleh *client* kemudian di decoding lalu di mainkan.

3.2 *Flowchart*

3.2.1 *Flowchart* Sistem *Streaming Server*

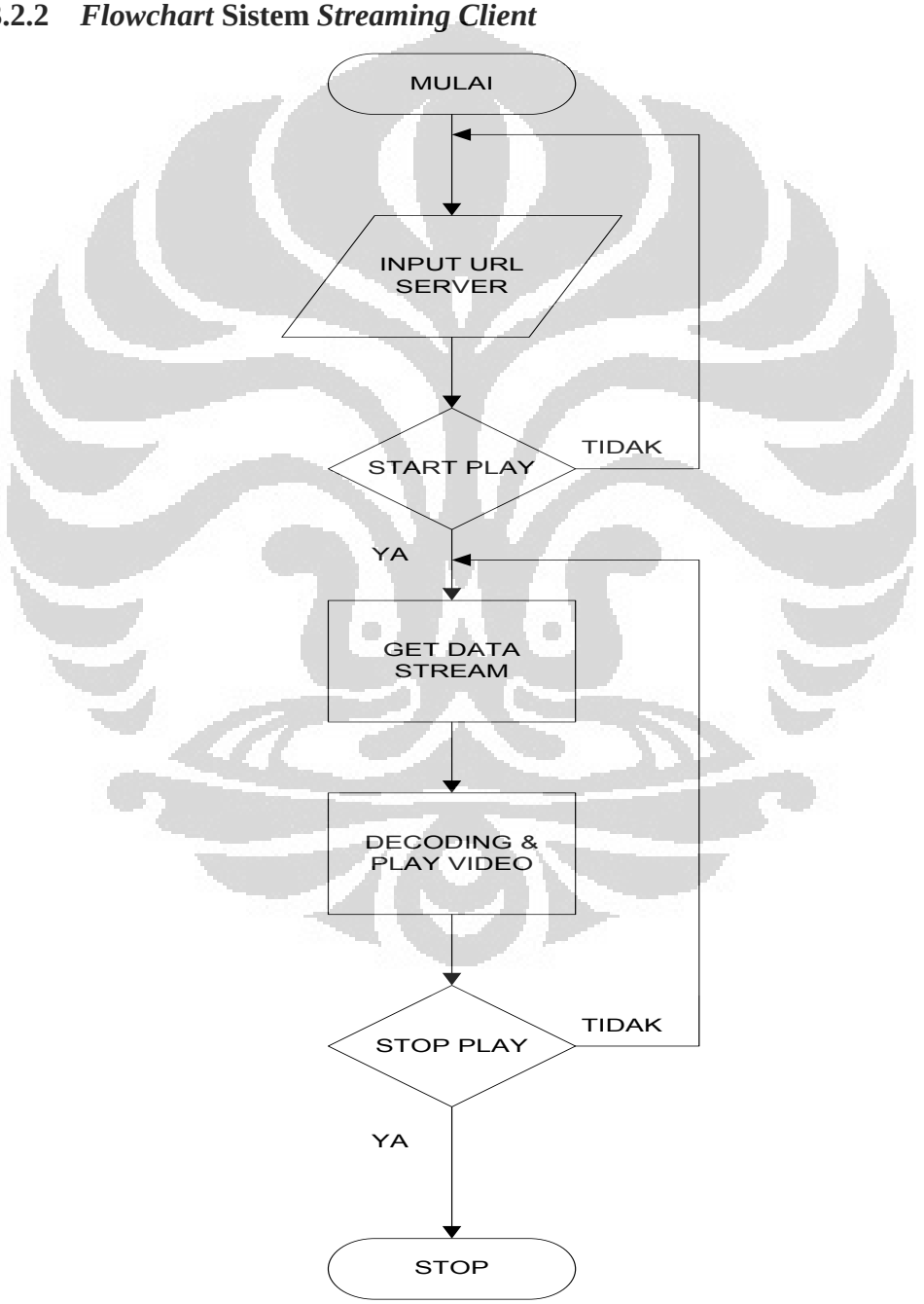


Gambar 3-3 *Flowchart Streaming Server*

Dari gambar flowchart sistem streaming server ini, setelah program dimulai, dilakukan penginputan device kamera dan konfigurasi parameter encoder seperti bitrate, frame rate, buffer size, key frame dan video smoothness. Setelah parameter sudah terkonfigurasi kemudian dimulai proses encoding data video

yang di tangkap oleh kamera. Data video itu dapat disimpan atau tidak tergantung kebutuhannya. Format Data video yang disimpan ialah .wmv. Setelah proses encoding, dilakukan proses broadcasting video atau proses pengiriman data video secara streaming ke jaringan. Proses ini akan berlangsung terus menerus dengan atau tanpa streaming client yang terhubung sampai proses encoding dihentikan secara manual.

3.2.2 *Flowchart* Sistem *Streaming Client*



Gambar 3-4 *Flowchart Streaming Client*

Pada streaming client, setelah program dimulai, dimasukkan alamat url streaming server agar dapat terhubung dengan streaming servernya. Setelah itu tekan tombol play untuk mengunduh data video secara streaming. Data video yang telah diunduh secara streaming tersebut kemudian di decoding lalu di mainkan. Proses ini akan berakhir apabila streaming server tidak mengirimkan data video lagi atau dihentikan secara manual di sisi streaming clientnya.

BAB IV
ANALISA DAN PENGUKURAN

4.1 Parameter QoS

Parameter QoS yang diukur adalah *Delay*, *Frame rate*, *Throughput* dan Kecepatan data *video*.

4.1.1 *Delay*

Adalah waktu yang dibutuhkan untuk sebuah data video untuk diproses atau untuk mencapai tujuan, karena adanya antrian yang panjang, atau mengambil rute yang lain untuk menghindari kemacetan. Delay pada sisi encoding adalah delay yang terjadi karena adanya proses kompresi data video. Delay ini tergantung pada kemampuan spesifikasi komputer dalam memproses kompresi. Sedangkan delay pada sisi decoding adalah total delay yang terjadi sejak awal data video di kompresi lalu dikirim secara streaming kemudian di dekompresi dan di play di sisi decoder.

$$\text{Delay Transmisi} = \frac{\text{Waktu pengiriman data (sec)}}{\text{Jumlah Data yang dikirim (bits)}}$$

Delay total (sec) = Delay Encoder + Delay Transmisi + Delay Decoder

4.1.2 *Frame rate*

Adalah banyaknya *frame* dalam satu detik pada data *video*. Pada dasarnya data *video* merupakan kumpulan dari banyaknya file gambar (*frame*) yang ditampilkan secara bergantian.

$$\text{Frame Rate} = \frac{\text{Banyaknya Frame yang dikirim (frames)}}{\text{Waktu Pengiriman data (sec)}}$$

4.1.3 *Throughput*

Adalah kemampuan sebenarnya suatu jaringan dalam melakukan pengiriman data. Biasanya *throughput* selalu dikaitkan dengan *bandwidth*. Karena *throughput* memang bisa disebut juga dengan *bandwidth* dalam kondisi yang sebenarnya. *Bandwidth* lebih bersifat *fix* sedangkan *throughput* sifatnya adalah dinamis tergantung *traffic* yang sedang terjadi atau loss data yang hilang selama transmisi berlangsung.

Rumus *Throughput* (*bits*/s) = Jumlah data yang terkirim (*bits*)

Waktu pengiriman data (sec)

4.1.4 Kecepatan Data *Video* atau *Bit rate* (*bits*/s)

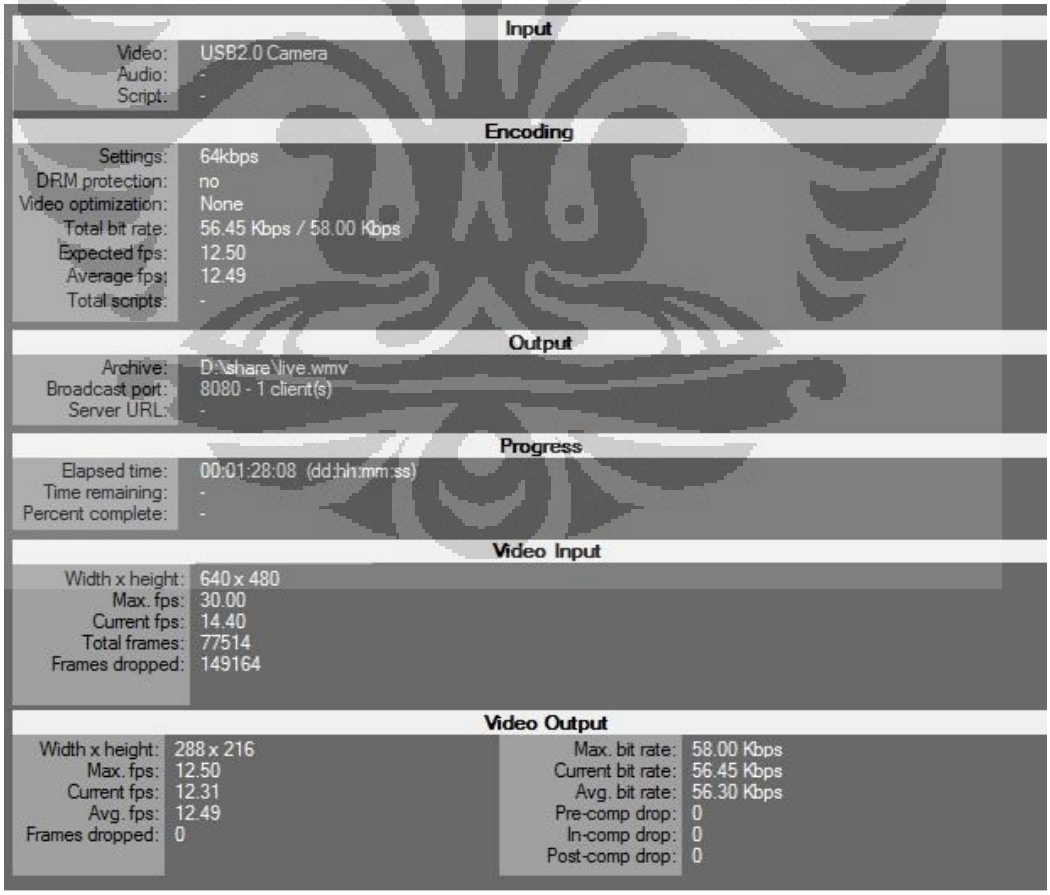
Adalah kecepatan data *video* yang dihasilkan oleh *encoder* dan *decoder*. Biasanya dihitung dari besar ukuran data *video* dibagi dengan waktu durasinya.

Bit rate = Besar ukuran data video (bits)

Waktu Durasinya (sec)

4.2 Pengukuran *Encoding*

Pengukuran QoS di sisi *streaming server* (*encoder*) dimana data *video* di *encoding* dengan konfigurasi yang sudah ditentukan sebelumnya. Pengukuran QoS diukur dengan melihat hasil statistik *encoder* selama beberapa waktu tertentu. *Encoder* yang digunakan dalam pengukuran ini yaitu aplikasi *Windows Media Encoder* 9.



Gambar 4-1 Statistik Proses *Encoder*

Dari Gambar 4-1 didapatkan parameter-parameter yang akan diukur seperti *bitrate*, *frame rate*, dan *throughput*nya. Untuk mengukur parameter *delay* ialah dengan membandingkan waktu pada *video* input dengan *video* outputnya.

input video output video

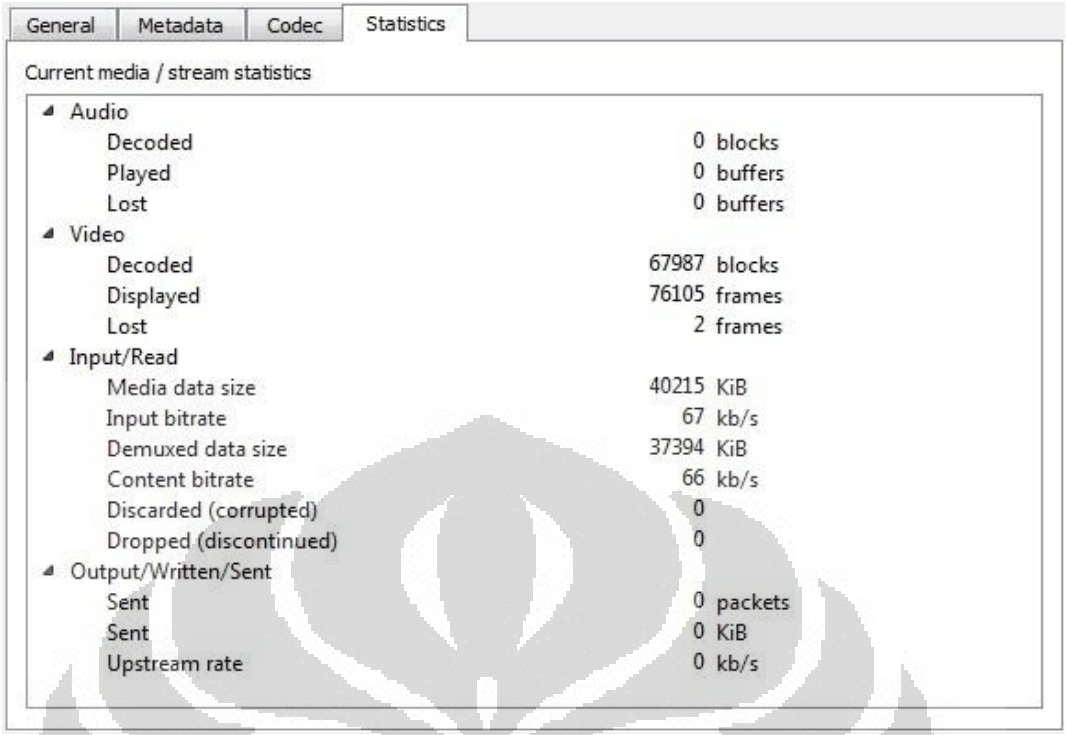
Gambar 4-2 Perbandingan waktu input *video* dengan output *video*

Tabel 4-1 Pengukuran QoS *Encoding*

Waktu Rekam (sec)	*Frame* rata-rata (avg. fps)	*Delay* (sec)	*Throughput* (kbps)	Kecepatan rata-rata (avg. kbps)
60	12.44	0.01	55.71	55,74
120	12.41	0.01	55.49	56,37
300	12.43	0.01	55.02	56.33
600	11,58	0.01	55.19	56.43

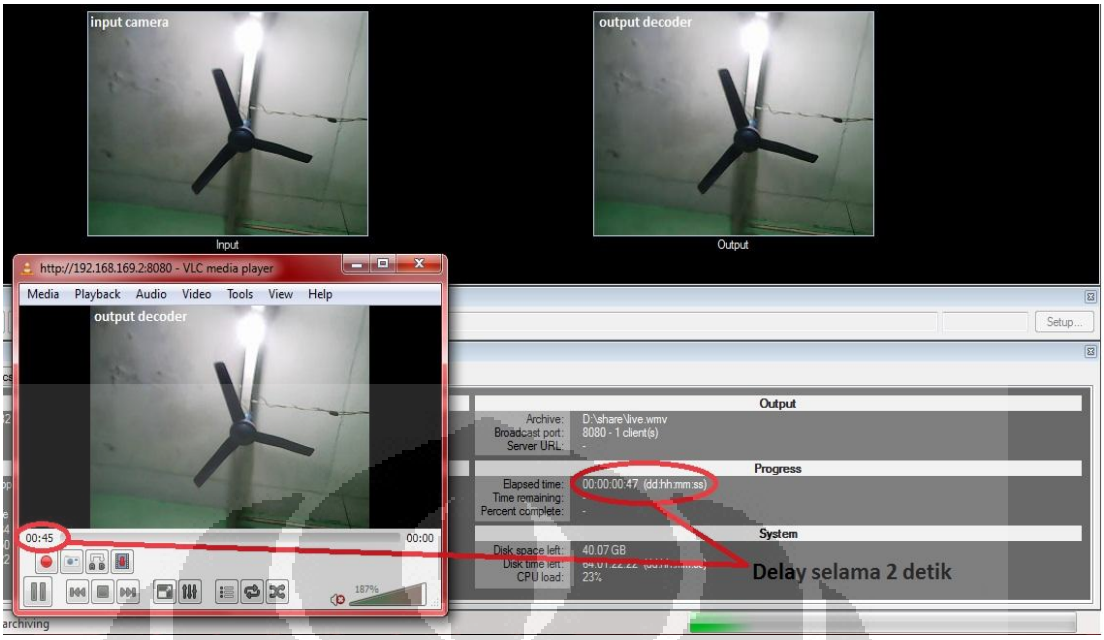
4.3 Pengukuran *Decoding*

Pengukuran QoS di sisi *streaming client* (*decoder*) dimana data *video* yang diterima lalu di decode dan di mainkan. Pengukuran QoS ini diukur dengan melihat hasil statistic *decoder* selama beberapa waktu tertentu. *Decoder* yang digunakan dalam pengukuran ini yaitu aplikasi *Video Lan Converter*.



Gambar 4-3 Statistik Proses *Decoder*

Dari Gambar 4-2 didapatkan parameter Statistik Proses *Decoder* - parameter yang akan diukur seperti *bitrate, frame rate*, dan *throughput*nya. Untuk mengukur parameter *delay* ialah dengan membandingkan waktu pada proses *encoder* dengan waktu proses di *decoder*.



Gambar 4-4 Perbandingan waktu proses *encoder* dengan *decoder*.

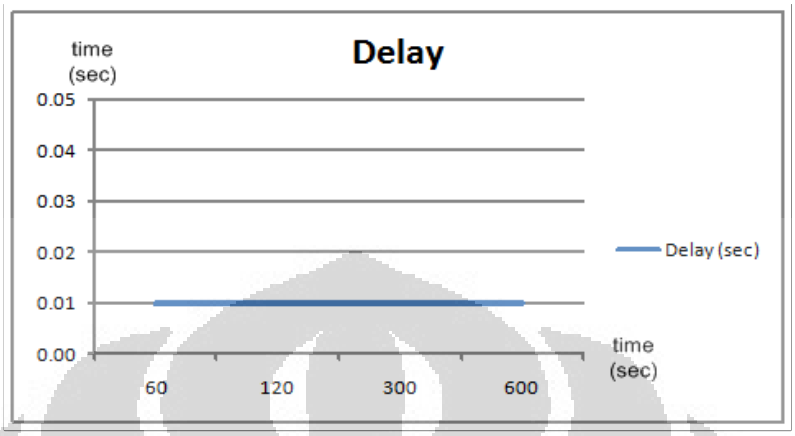
Tabel 4-2 Pengukuran QoS *Decoding*

Waktu Rekam (sec)	*Frame* rata-rata (avg. fps)	*Delay* (sec)	*Throughput* (kbps)	Kecepatan rata-rata (avg. kbps)
60	12.70	1.60	54.27	59.07
120	12.77	1.80	54.87	59.40
300	12.74	2.00	54.85	59. 15
600	12.76	2.10	55.00	59.21

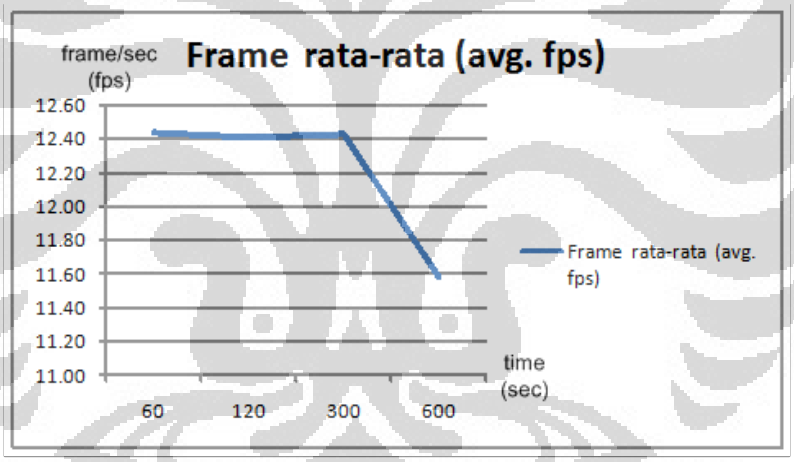
4.4 Analisa Hasil Pengukuran

Untuk pengukuran di sisi *streaming server* atau pengukuran pada proses *encoding*. *Delay* hampir tidak terlihat pada pengukuran setiap beberapa waktu tertentu. *Frame rate* mendekati target *frame rate* yang diatur pada *encoder* yaitu 12,5 fps. Untuk *throughput* dan kecepatan rata-rata hasil pengukurannya relative stabil yaitu sekitar 55 Kbps. Target *bitrate* memang diharapkan sebesar 58 Kbps, tetapi perbedaannya tidak terlalu besar, hanya sekitar 3 Kbps. Jadi untuk *bandwidth* transmisi sebesar 64 Kbps, data *video* dapat dikirim secara *live*

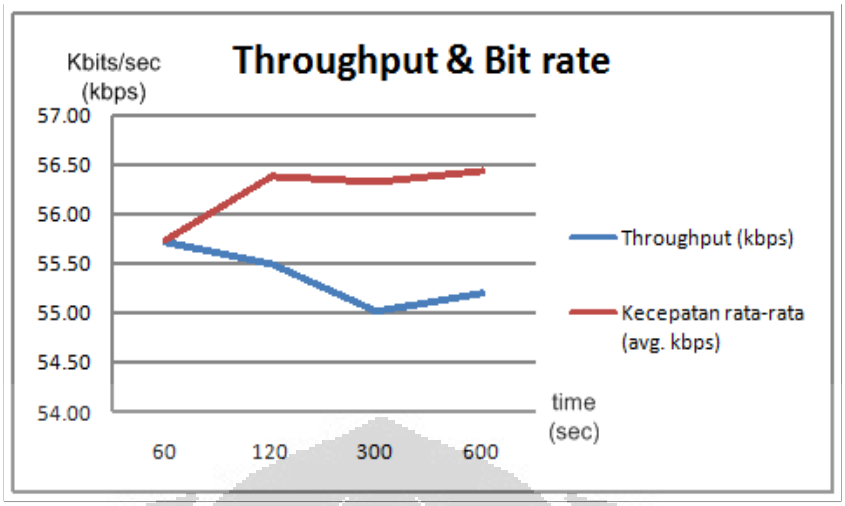
streaming tanpa harus mengalami *delay* karna kekurangan *bandwidth* jaringan. Berikut grafik hasil pengukurannya:



Gambar 4-5 Grafik *Delay* pada pengukuran *Encoding*

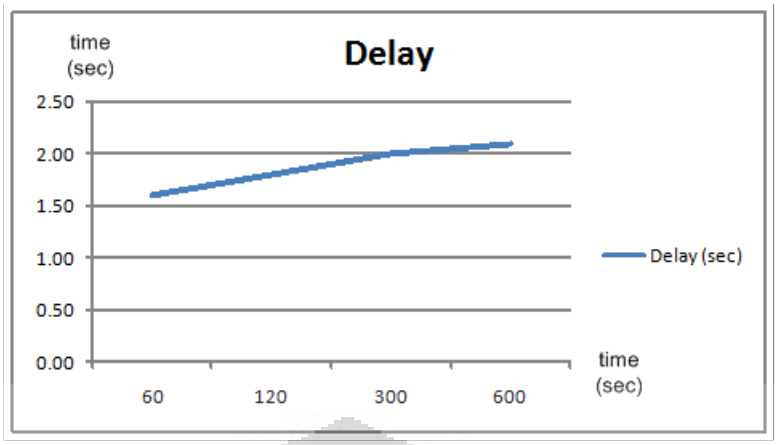


Gambar 4-6 Grafik *Frame rate* pada pengukuran *Encoding*

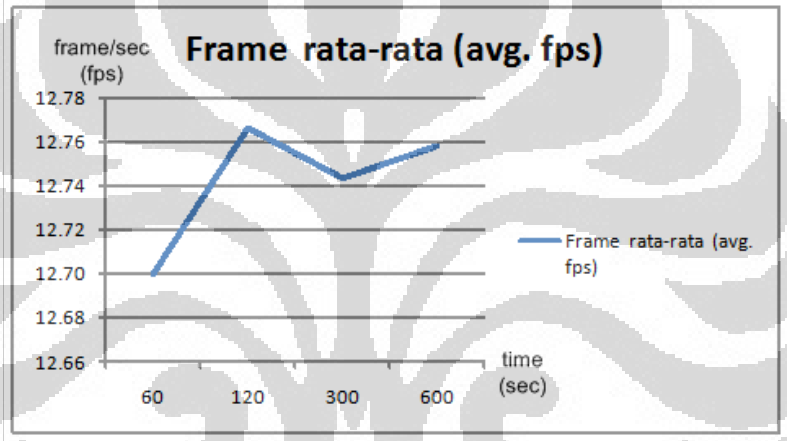


Gambar 4-7 Grafik *Throughput* dan *Bit rate* pada pengukuran *Encoding*

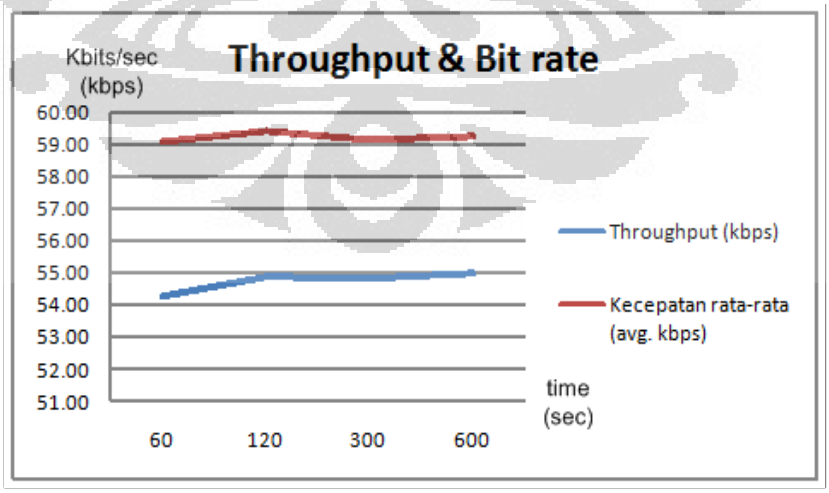
Untuk pengukuran di sisi *streaming client* atau pengukuran pada proses *decoding*. *Delay* sangat terlihat pada pengukuran setiap beberapa waktu tertentu. Semakin lama pengukurannya, *delay* akan bertambah namun tidak terlalu besar. *Delay* diukur dari waktu awal dikirim sampai waktu data *video* diterima dan di encode oleh *client*. Ada beberapa faktor yang mempengaruhi terjadinya *delay*, yaitu waktu transmisi data dari *server* ke *client*, kondisi *traffic* dan waktu proses *decoding*. Jadi semakin jauh jarak transmisi dan semakin padat *traffic* jaringan maka waktu *delay* akan bertambah. Untuk *Frame rate* nilainya melebihi target *frame rate* yang diatur pada *encoder* yaitu sekitar 12,7 fps. Hal ini karena *decoder* melakukan dekompres atau dengan kata lain meningkatkan qualitas data *video*nya. Untuk *throughput* dan kecepatan rata-rata hasil pengukurannya relative stabil yaitu sekitar 59 Kbps untuk *throughput*nya dan 54 Kbps untuk kecepatan rata-ratanya. Kecepatan rata-rata meningkat bila dibandingkan dengan hasil pengukuran di sisi *encoding*. Hal ini karena proses decoding. Berikut grafik hasil pengukurannya:



Gambar 4-8 Grafik *Delay* pada pengukuran *Decoding*



Gambar 4-9 Grafik *Frame rate* pada pengukuran *Decoding*



Gambar 4-10 Grafik *Throughput* dan *Bit rate* pada pengukuran *Decoding*

BAB V
KESIMPULAN

Dari analisa dan pengukuran yang telah dilakukan, dapat ditarik kesimpulan sebagai berikut:

1. Faktor dasar yang mempengaruhi besarnya *delay* yaitu *bandwidth* transmisi, jarak transmisi, kondisi *traffic* dan kemampuan spesifikasi komputer dalam memproses *video* (*encode-decode*)
2. Besar throughput di sisi decoder lebih kecil dibandingkan di sisi encoder. Hal ini dikarenakan adanya packet loss pada saat pengiriman data secara streaming dari server ke client.
3. Teknik encoding CBR menjaga bit rate dan frame rate tetap stabil dan konstan mendekati target bitrate dan target frame rate yang telah ditentukan sebelumnya sehingga kualitas video tetap maksimal sesuai dengan bandwidth jaringannya.

DAFTAR PUSTAKA

[1] Peter D. Symes, 2003. *Digital Video Compression*. USA

[2] Apostolopoulos, John G. Wai- tian Tan. Susie J. Wee. 2002. *Video Streaming: Concepts, Algorithms, and Systems*. USA

[3] Stephen A. Thomas, 2001. *HTTP essentials: protocols for secure, scaleable, Web sites.* USA

[4] Mauricio Arregoces, Maurizio Portolani, 2003. *Data Center Fundamentals*. USA

[5] Frederic P. Miller, Agnes F. Vandome, McBrewster John, 2010. Adaptive Bit Rate. USA

[6] Donny B.U., M.Si. 2002. *Streaming: Membuat File Besar Serasa Kecil*. Indonesia

[7] Lambert M. Surhone, Miriam T. Timpledon, Susan F. Marseken, 2010. *Windows Media Video*. USA

[8] Microsoft, 2009. *Windows Media Encoder Help.* USA

[9] Microsoft, 2012. *Enchoding Method*. USA

[10] Vijay K. Madisetti, 1995 *Signal Compression.* India

[11] Dr.E.KANNAN & G. Murugan, Vel Tech Dr.RR & Dr.SR Technical University. 2012. *Lossless Image Compression Algorithm For Transmitting Over Low Bandwidth Line*. India

[12] Iain E. G. Richardson, 2002. *Video Codec Design: Developing Image and Video Compression Systems.* USA

[13] Ashfaq A. Khan, 2005. *Digital signal processing fundamentals*. India

LAMPIRAN

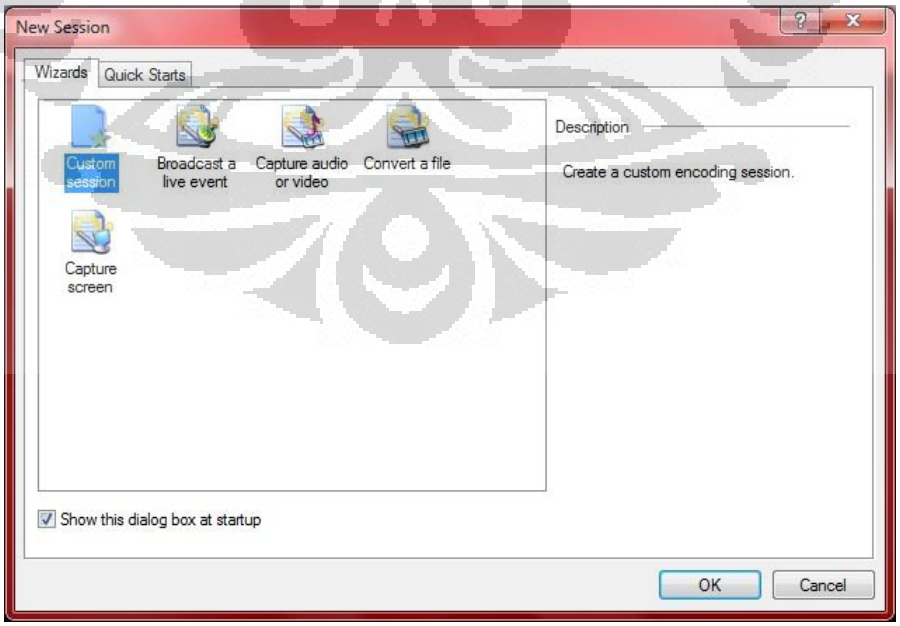
Prosedur Pengoperasian Simulasi Sistem Kompresi

1. Prosedur pengoperasian sistem Encoding

a. Tampilan awal aplikasi streaming *server*

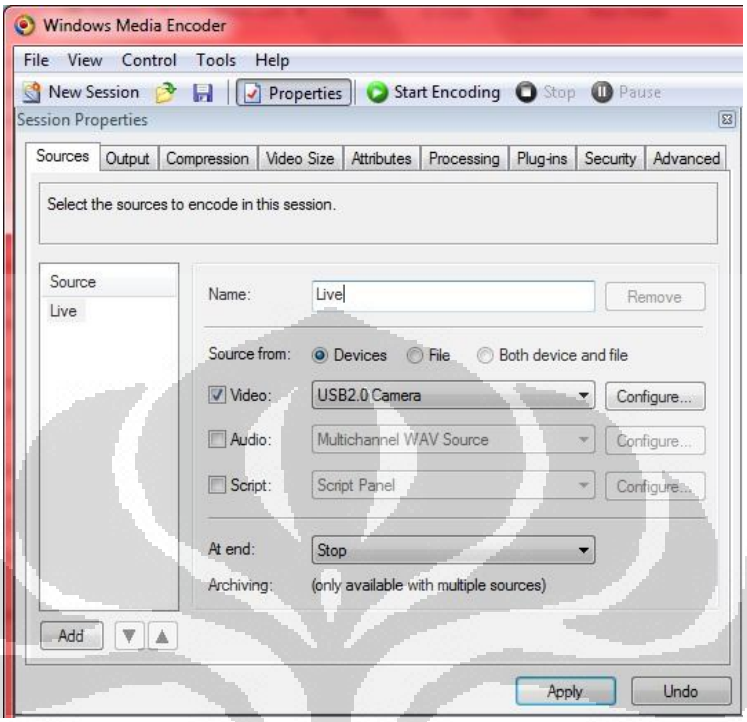
Gambar 1 Tampilan awal aplikasi streaming *Server*

b. Menu Session – pilih custom session lalu klik OK



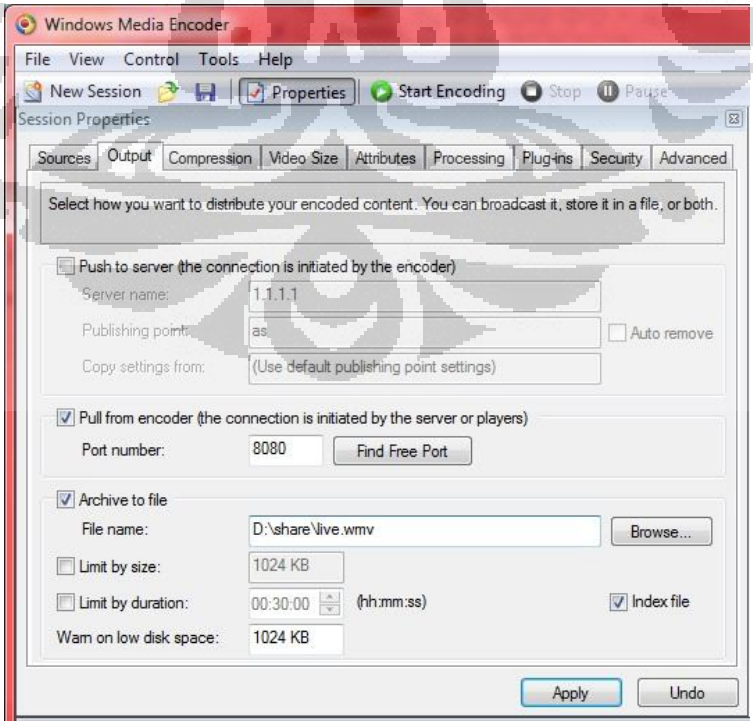
Gambar 2 Tampilan Menu Session

c. Tab Sources – pilih input source from Devices, hanya pilih input video saja dan pilih device kameranya (USB2.0 Camera)



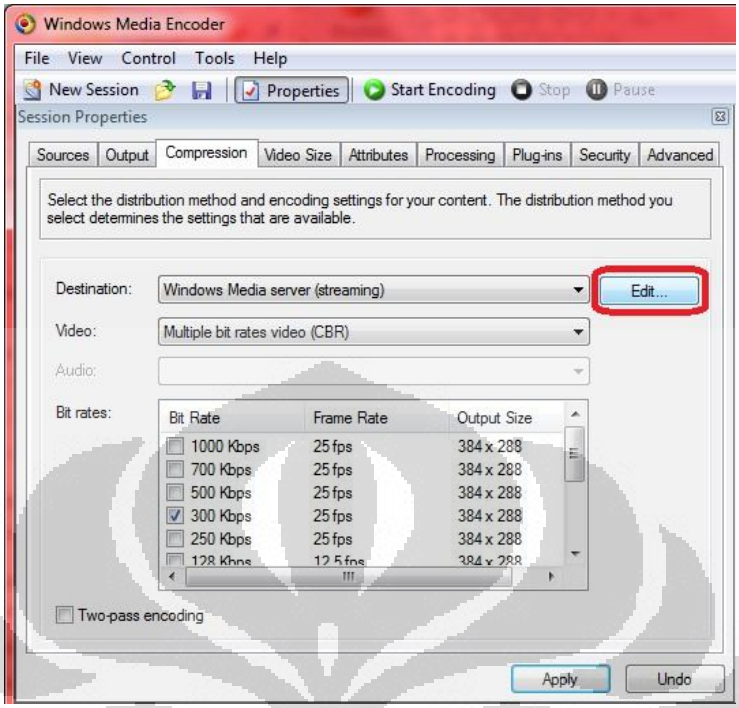
Gambar 3 Tampilan *Tab Sources*

d. Tab Output – masukkan port yang akan digunakan (8080) dan nama file video yang akan disimpan.



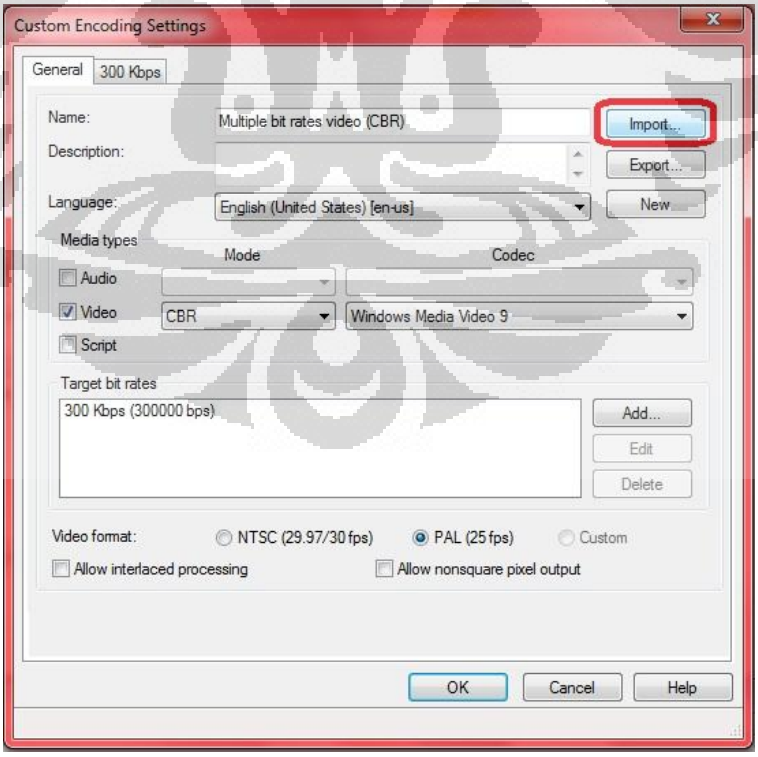
Gambar 4 Tampilan Tab Output

e. Tab Compression – klik tombol edit



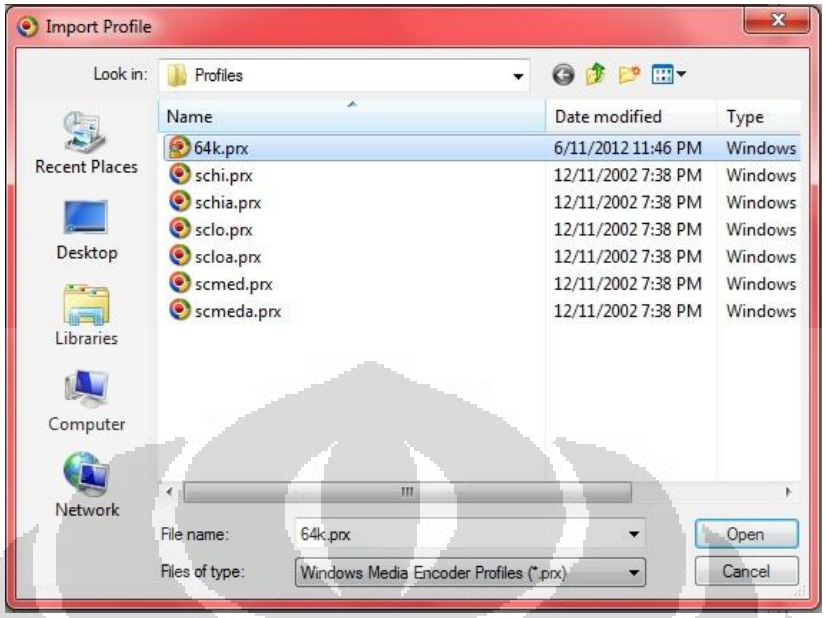
Gambar 5 Tampilan *Tab Compression*

f. Menu Encoding Option – klik menu Import untuk menggunakan profil atau konfigurasi kompresi yang sudah ditentukan sebelumnya



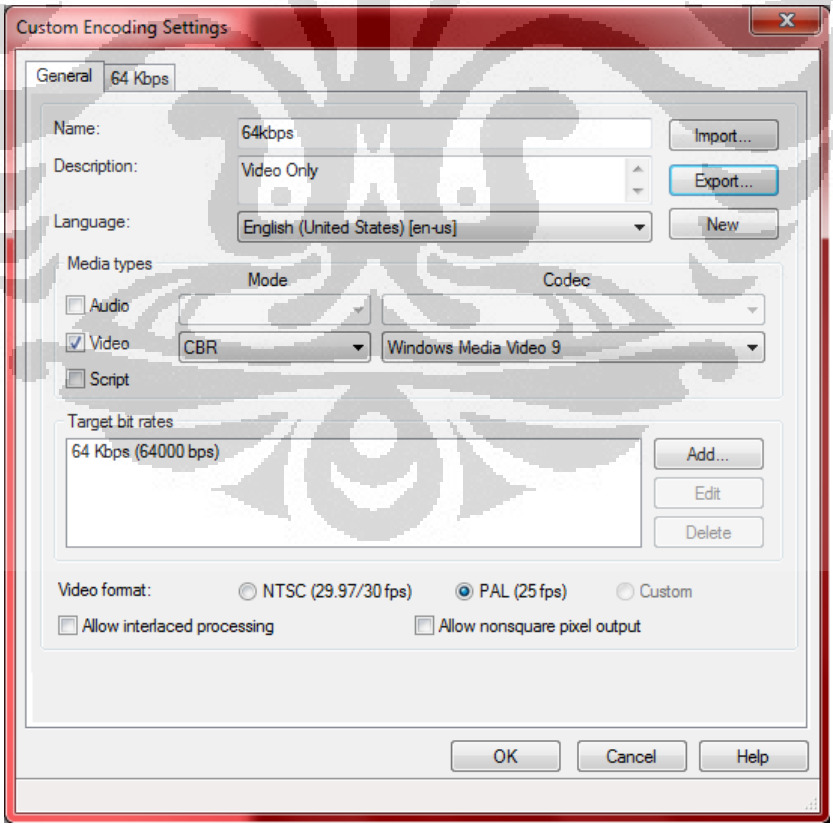
Gambar 6 Tampilan Encoding Option

g. Pilih profil yang sudah ditentukan sebelumnya kemudian klik Open



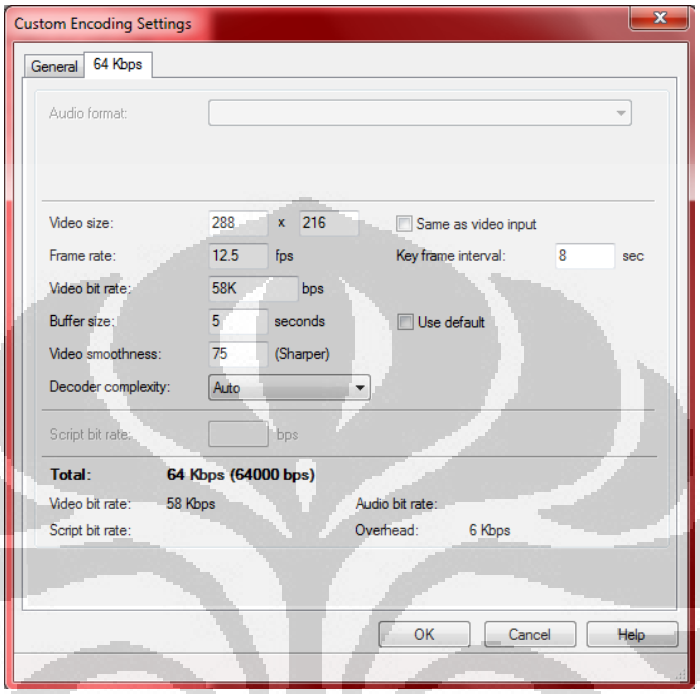
Gambar 7 Menu Profil Encoding

h. Profil encoding yang digunakan – Encoding Mode CBR, Codec Windows Media Video 9, Target bit rate 64000 bps dan Video Format PAL



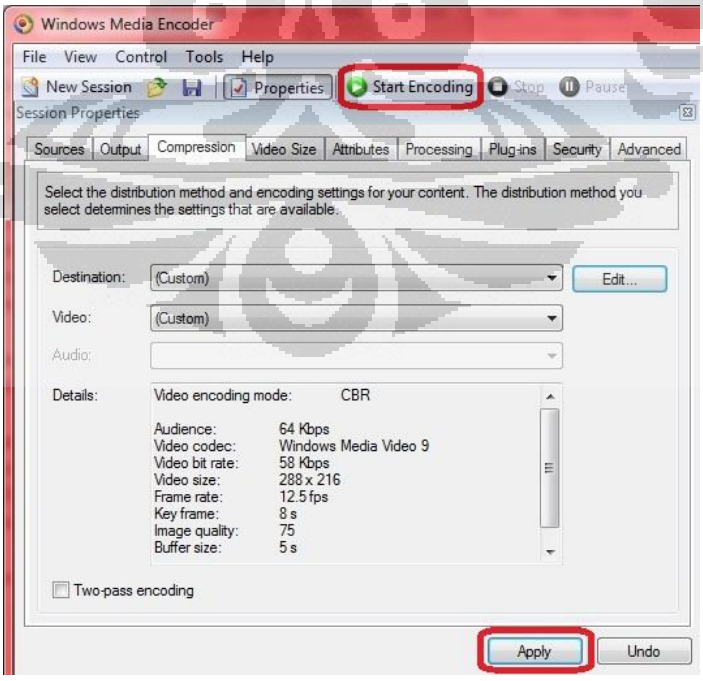
Gambar 8 Tampilan Profil Encoding

i. Konfigurasi output video – Video size 288px216p, frame rate 12,5 fps, video bitrate 58 Kbps+Overhead 6Kbps (64Kbps), Buffer size 5 secs, video smoothness 75 (Sharper), Key frame interval 8 secs, dan decoder complexity auto



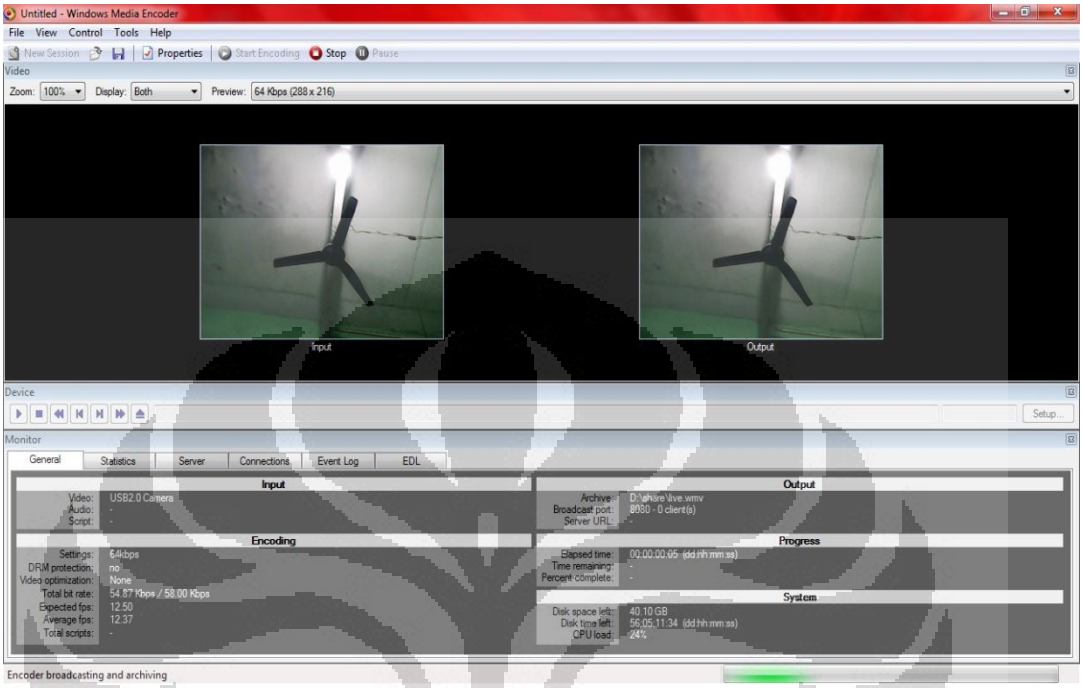
Gambar 9 Tampilan Konfigurasi Kompresi

j. Tab Compression – klik Apply kemudian Start Encoding.



Gambar 10 Tampilan Tab Compression

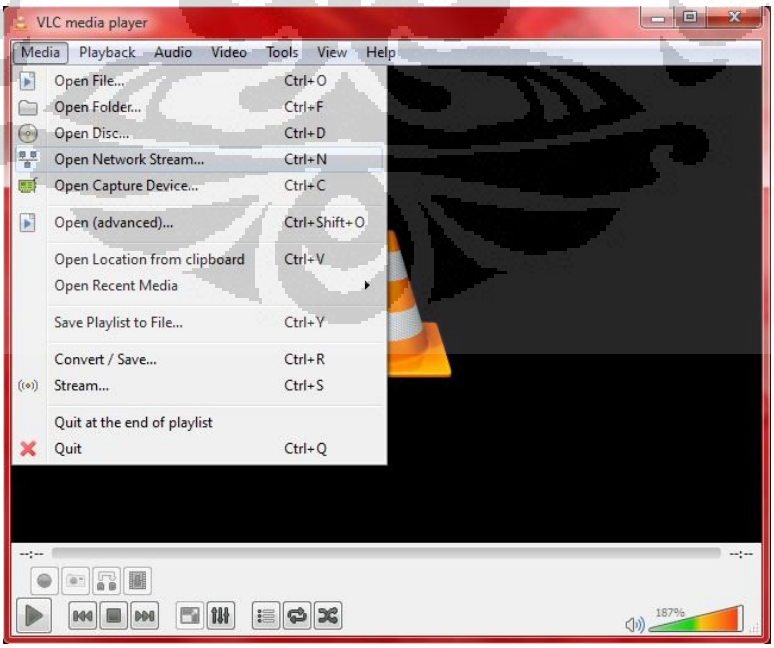
k. Proses Encoding dan Broadcast video streaming, video di sebelah kiri ada input video dan di sebelah kanan adalah output video yang telah diencoding



Gambar 11 Proses Encoding dan Broadcast Video Streaming

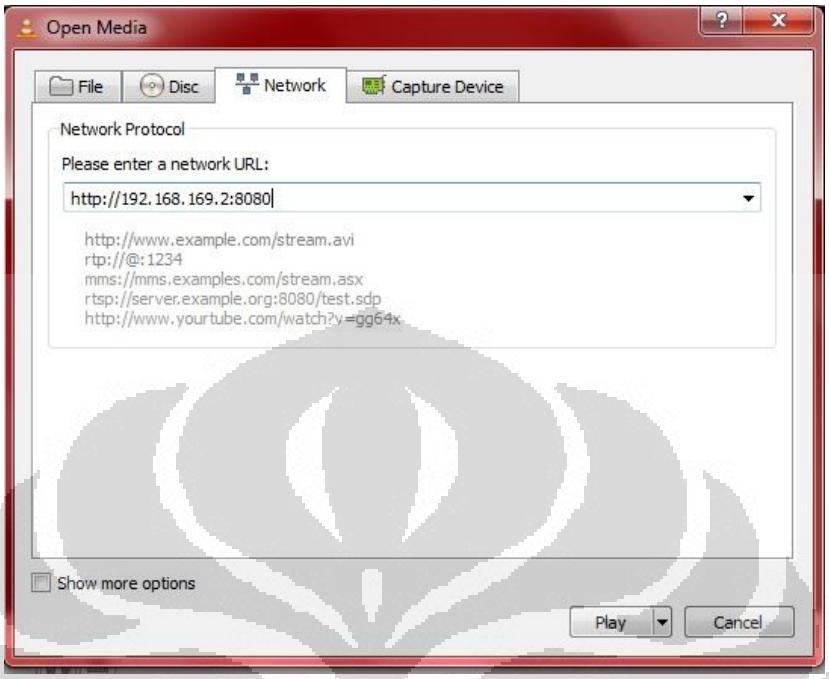
2. Prosedur pengoperasian sistem Encoding

a. Menu Media – pilih Open Network Stream



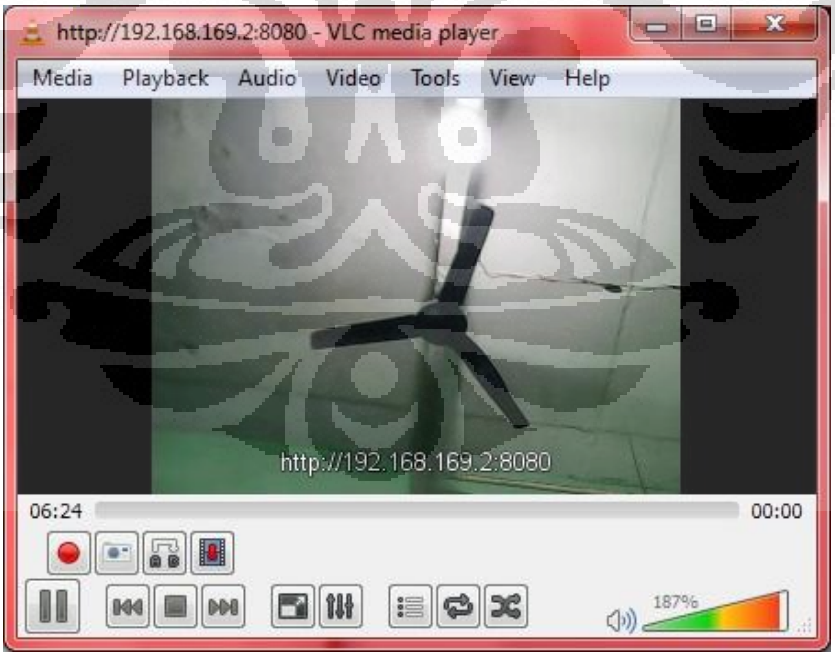
Gambar 12 Tampilan utama Sistem *Server*

b. Tab Network - masukkan alamat URL Windows Media Server dan port yang digunakannya (http://192.168.169.2:8080)



Gambar 13 Tampilan *Tab Network*

c. Tampilan Output Video Streamingnya



Gambar 14 Tampilan Output Video Streaming