Video Coding for Streaming Media Delivery on the Internet

Gregory J. Conklin, Member, IEEE,
Gary S. Greenbaum, Member, IEEE,
Karl O. Lillevold,
Alan F. Lippman, Member, IEEE,
and Yuriy A. Reznik, Member, IEEE

Presented by: Devon Krisman
History of Streaming Media

- Streaming media was first introduced when media standards already existed (MPEG-1)
- Users could download a full MPEG-1 file, store it and then watch, but this is a problem for users with low disk space, a slow internet connection, or a short attention span.
- The first attempts at streaming media used HTTP based servers, meaning data was streamed using TCP. Due to the limitations of TCP, especially its lack of control over data rates, these attempts at streaming media required a large (5-20s) preroll buffer. And when the next fragment of a stream was delayed beyond the preroll buffer (a frequent occurrence) the player had to refill the buffer before continuing.
In 1995, Real Audio was introduced. It used a combination of TCP and UDP (User Datagram Protocol), which allowed the audio data to be sent from the server out of order, and delivery of packets is not guaranteed, so there is no overhead of checking to make sure packets are received. The lost, delayed, and out of order packets did create other problems though. To deal with dropped packets, RealAudio implemented Automatic Repeat-Request (ARQ), which allowed the client to re-request missed packets through the bi-directional TCP connection. And if the server resent the missed packet within the preroll time, the client recovered from the loss. For the times when ARQ would fail, RealAudio used a frame interleaving technique. This randomized the distribution of audio frames within a packet, making a dropped packet less noticeable.
First Codec Specifically for Streaming Media

- One of the biggest problems was the need for a more Constant Bit Rate when sending media.
- Video encoded with MPEG have a very high bitrate during hard to compress areas, and a very low bitrate during slow moving scenes.
- Because of the preroll buffer used, the bitrate does not need to be strictly constant, it just needs to average out to the required bit rate, with this in mind, RealVideo designed the Variable-Bit-Rate rate control algorithm for their codec.
- Another requirement of the codec was support for random access, to be able to fast forward, rewind, or join live broadcasts.
- To do this, the codec periodically inserted Intraframes and modified the rate control to minimize fluctuation in quality of encoded frames.
- Recovering from lost packets was a harder task than in audio since video frames were too large to use interleaving.
- For packet loss RealVideo used forward error correction codes and error concealment mechanisms, this combination is known as unequal error protection.
Around the time RealVideo was introduced, streaming media was only delivered at a fixed bit rate (14/28 kbps).

Because of this, users with faster connections got no benefit from them, and those with slower connections could not view them at all.

As a temporary solution, content distributors created multiple streams of their content for varying connection speeds.

Not only did this add extra confusion, but it relied on actual bandwidth being the same as the rated bandwidth, and it did not factor loss statistics.

One simple fix which was applied to early streaming media servers was stream thinning, when the server was notified that the client was receiving packets slower than expected, the server began skipping transmission of some packets. This introduced some data loss and frame skips, but it helped to prevent rebuffering and dropped connections.
Introduced in 1998
Meant to serve multiple audiences adaptively
SureStream technology produced multiple streams of the content, optimized for different conditions
During the stream, the client monitors the actual bandwidth and losses of the connection and dynamically requests the optimal stream
This client side processing allowed the server to handle more connections and allowed for simple multicasting
This was the first streaming media system built on IETF and W3C standards
G2 used the standard RTSP for session control and the RTP standard for framing and transporting data.
Delivery of live content

Public IP-Based Delivery Network

Multiple-Access Splitter

Pass. Cache

Local Network

Local Network

Local Network

Clients

Public IP Network

Source Server

2\textsuperscript{nd}-tier Splitter

1\textsuperscript{st}-tier Splitter

Dedicated Delivery Network

SLTA

Encoder

Live source

Archived content
Delivery of on-demand content
Streaming Media Delivery
Mechanisms

- Video is streamed on the internet in two modes: live and on-demand

**Live Video**
- Live video is encoded on the fly, then is passed to the server, which distributes it to the clients and splitters
- Splitters are additional servers that distribute the workload of serving live data to all connected clients
  - Push splitting is initiated by server
  - Pull splitting is initiated by a client connecting to a local splitter it then sends its to the nearest active splitter
  - Multiple-access splitting allows lower-tier splitters to connect to multiple upper-tier splitters
- The server or splitter will either unicast the information to a client, or if its network has the capability, the server will send a multicast stream which is replicated to all clients on that network

**On Demand**
- Similar to live broadcast except
  - The server reads from a stored file instead of directly from the encoder
  - Proxy servers can cache the most frequently accessed streams
  - Clients can rewind and fast forward
Problems with Video Coding

- Variations in bandwidth
  - Solutions:
  - Adaptive client-driven serving
    - Client can monitor incoming packets and instruct the server on how to adjust encoding or transmission rate
  - Dynamic prediction of bandwidth
    - With a sufficient preroll time, statistical techniques can potentially be used to alter transmission rate based on the predicted bandwidth fluctuations
Problems with Video Coding

- Random Access
  - Solution: insert I-frames at intervals of 1-5 seconds, so random access will bring the video to the nearest I-frames
    - This hurts compression rate

- Processing Resources
  - Media clients have a wide range of processing power (PCs, PDAs, Cell Phones)
  - Solution: Complexity-scalable encoding/decoding
Motion-compensated hybrid video-coder

Diagram:
- Input frames
  - Spatio-temporal prediction filter
  - Temporal Pre-processor
  - Motion Estimation engine
- Motion Compensation engine
- De-blocking filter
- Rate Control Logic
- Motion vectors
  - Residual data
  - Transform
  - Inverse transform
  - Encoders
- Q
- Q⁻¹
- Compressed data
- QP
Scalability Models

- The first motion-compensated hybrid video-coding algorithms were designed to produce a single-rate encode for point-to-point transmission. This needed to be scaled to multicast and multiple-access transmissions.

- One solution: simulcast – send all versions of the stream through multicast and let the client choose a stream.
Adaptive channel encoding of streams allows the bit rate to correspond to the actual bandwidth and loss statistics of the channel.

- We don’t have channel statistics until the stream starts, so the server must manage adaptive strategies.
- The server cannot adapt the encoding on the fly, so the encoder must use scalable source coding techniques to allow the server to choose the correct bit rate before transmission.
SureStream components:
- Adaptive Stream Management (ASM) protocol
- SureStream file format access and rendering mechanisms
- Source and channel coding algorithms
SureStream - ASM

Server Core

Data Packets

File format plugin

Client Core

ASM Rule Subscription

File-system plugin

Rate / Distortion Analyzer

ASM Rule Book

ASM Rules Used

Renderer plugin

Compressed Media file
SureStream - ASM

- **Adaptive Stream Management (ASM)**
  - ASM allows the client to communicate the specifications of the encoding that should be “synthesized” by the server to minimize distortion
  - Compressed media files are accessed by the server, and the file format plug-in is capable of producing the proper combination of encoded streams to send to the client
  - The way the ASM protocol works is that the ASM rule book for the file is stored within the compressed media file. At the beginning of communication, the server sends a copy of the rule book to the client. During the stream, the client collects information about the channel, and sends a request to the server to subscribe to the combination of rules that would send the most optimal stream
RealVideo 8 Algorithm

Fig. 7. The structure of the RealVideo 8 encoding module.
Digitized video are sent along with their timestamps to a set of input filters.

A spatial resampler then downscales the output frames to spatial resolutions that are suitable for encoding in different bitrates (for selection of an optimal bitrate for the client).

The resampler chooses these resolutions based on the bitrates, and the type of content and type of acceptable distortion.

- Content creators can select from four modes: “smooth motion,” “normal motion,” “sharpest image,” and “slide show.”

After being resampled, video frames are send to video codecs, which encode the frames into streams based on the target bit rates.

Throughout this process, the CPU scalability control module monitors the system throughout the process, and notifies the algorithms if they should switch to lower complexity modes (very important for live feeds).
RealVideo 8 Algorithm

- Input filters: remove noise and potential artifacts introduced by edits and conversions of video signal
  - The inverse telecine filter searches for and removes the redundant 5.97 frames introduced by converting video from film to NTSC
  - De-interlace analog signals to produce progressive frames
  - Remove low-energy spatial noise

Fig. 8. Input filters in RealVideo 8.
RealVideo 8 Algorithm

- **Core Algorithm**
  - RealVideo 8 uses a motion-compensated hybrid scheme similar to the one shown earlier with a few improvements:
    - Motion prediction
    - Adaptive transform sizes
    - Advanced statistical models
RealVideo 8 Algorithm

- **Scene Detection and Rate Control**
  - Rate control determines which frames are encoded and the quality of the encoded frame
  - At higher bitrates it is necessary to maintain full framerates
  - Rate control has two modes: single-pass and two-pass
    - Single pass guesses the number of frames to skip until the next frame is encoded, and it guesses the correct quality and type (i.e. I-frame)
    - Two pass has knowledge of the previous frames and the next frame, so it can make more efficient choices
    - Live broadcasts must use single-pass encoding
RealVideo 8 Algorithm

- **Client-Side Video Postprocessing**
  - The frame rate upsampler is a temporal filter that attempts to interpolate intermediate frames
  - The deringing filter reduces ringing artifacts
  - User Requested filters include filters such as sharpening filter, color controls, etc.

Fig. 11. Client-side video processing in RealSystem 8.