Live Streaming Video Options
for Bugscope

Don Appleman
Visiting Research Programmer
Imaging Technology Group

Date Issued: February 4, 2003

The Beckman Institute
Imaging Technology Group
Technical Report 02-005

Copyright © 2002 Board of Trustees of the University of Illinois

The Beckman Institute for Advanced Science and Technology
Imaging Technology Group
405 N Mathews
Urbana, IL 61801
techreports@itg.uiuc.edu
http://www.itg.uiuc.edu
Live Streaming Video Options for Bugscope

Don Appleman

Imaging Technology Group
Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign

December 2002

Goals

Streaming video from the ESEM to the web-based client would provide the visual experience most like that encountered by an operator of the ESEM at the machine. However, due to the overhead associated with streaming video, it is overkill for the majority of tasks associated with operating Bugscope. Additionally, streaming video corresponds with TV mode on the ESEM, and the fact is that TV mode is used only intermittently by the ESEM operator.

Yet there are two specific tasks for which streaming video provides the most natural solution. First, with the ESEM in TV mode and the controller client streaming video, it should be possible to navigate the stage in a manner that is similar to navigating the stage in TV mode while directly at the ESEM. Second, with the ESEM in scan mode and the controller client streaming video, it should be possible to focus interactively, using the direct, mouse-based ESEM focus technique or one very similar.

Because of the demands video streaming places on the server, it is likely that its use will be restricted to the controller client and not the observer, except perhaps as an option. It will likely be toggled active automatically when the user chooses to perform either stage navigation or interactive focus control, rather than explicitly activated by the user.

These goals, due to their interactive nature, emphasize the need for low-latency in video transmission. For focus control, it may not be necessary to have a large video frame size, as a small size window is often used on the ESEM itself for this purpose. However, for stage navigation, a larger image facilitates quicker and more accurate navigation by the user.

Criteria

The many options for streaming video vary across a number of criteria, leading to trade-offs among the choices. The issues that are most important are

- Cost of acquisition and use
- Licensing

  The cost for licensing can run anywhere from $0 to several thousand dollars. Some packages are available for free as open-source projects. Also, some software may already be available in house, in which case the cost for utilizing the software for Bugscope is effectively $0. Licensing costs can also be affected
favorably by our status as an academic institution, and can go up or down depending on the number of concurrent video streams that we wish to support. This has implications as we consider whether to provide streaming video to both the controller and the observer, or restrict it to the controller only, as well as the possible use of multiple controllers.

- **Dedicated hardware**
  There is a wide range of performance among the various packages. Some are highly efficient, and place only small demands on their hardware. For these, it’s possible to run the streaming video server on the same hardware as the web server itself. Other packages trade higher CPU demands for higher quality video, and may require dedicated hardware. If additional hardware is unneeded or already available, then the effective cost of dedicated hardware is $0; if a fast machine with lots of memory must be acquired, then the cost of hardware will likely fall into the $1000+ range.

- **Bandwidth requirements and control**
  Streaming video is an inherently high-bandwidth practice. We must concern ourselves with the overall bandwidth available for output (i.e. bandwidth available to the web server in use), overall bandwidth available for input (multiple users at a single client location who may saturate that location’s network), and individual bandwidth and processing available for input (i.e. each individual user’s ability to receive, decode, and display the streaming video).

  Generally, higher quality video requires higher bandwidth. Different software packages take different approaches to accommodating bandwidth, among them
  - Stream at a number of bit rates, dynamically adjusting according to the apparent connection speed.
  - Produce multiple streams at different bit rates, and then provide the appropriate stream at connection time.
  - Produce multiple streams at different bit rates, and then allow the user to select among streams playing at different bit rates. Having the user select among streams at playback time is not an option for our purpose.
  - Support multiple bit rates, but require static configuration of the server based on expectations regarding the connection speed.
  - Do not allow adjusting the bit rate.

  In all cases, the trade-off is against video frame rate, frame size, image quality, or any combination of these.

  An additional consideration related to bandwidth is whether we wish to stream video to only a single user (e.g. as a feature of the controller client), or to stream it to any number of users (perhaps as an optional feature of the observer client).
• **Programmatic difficulty of implementation**
  The simplest implementations are simply plug & play. The vendor provides server software which takes a live video feed, compresses the input, and streams the result out on a URL under our control. In this case, the viewer is generally a separate application, also provided by the vendor, with its own user interface. These solutions are generally cash expensive, but labor cheap, and provide a high level of support.

  The most complex approach, from a programming standpoint, is adaptation of an open-source solution. There are several to choose from, but none does exactly what we need. Since they are open-source, though, we can choose to implement any functionality that we require. These solutions are “free” from a cash perspective, and come with little or no support; the package developers may be available to provide advice, but development support is unlikely. This approach is therefore costly in development effort.

  In the middle is a less-complex solution in which the vendor provides server software plus a software development kit (SDK) for constructing the client. These solutions are likely also expensive, provide a high level of support, and have a mid-level development cost.

• **Level of control over the appearance of video within the client**
  In the best of all worlds, the streaming video client would appear wrapped in a widget in the user interface of the Bugscope client. The following features are desirable

  • Frame the video within a Bugscope client widget.
  
  • Start and stop the video under Bugscope client program control (rather than under video player or other user interface control).
  
  • Use the video widget as a user input tool. When “driving” over the stage in TV mode, the ESEM operator uses the mouse to control the direction and rate of travel. To support this, the video widget would need to be able to accept mouse input and communicate with the ESEM to provide similar control during stage traversal. Alternatively, a separate widget, responsible for taking user input, could be displayed alongside of, over overlaid onto, the video widget.

  A client provided by a vendor is likely to be a free-standing media player. This format gives us the least control over the appearance and interactivity of the video widget. A client derived from an open-source project will provide total control, but may or may not require extended development time. A client derived from a commercial SDK should also be highly configurable, with potential for good interactivity, and with a manageable development time.
Latency (buffering, encoding/decoding, overall efficiency)

Streaming video is an inherently high-bandwidth practice. To deal with this problem, many video players buffer a portion of the video before beginning playback, in order to provide a smooth startup. For video on demand (VOD), or stored streaming video, the frame rate can be maximized by correctly determining the bandwidth available, then buffering enough frames to permit downloading the remaining frames while the available frames play. For live streaming video, the frame rate that is displayable is of course never higher than the frame rate that can be encoded by the server, so buffering does not result in an overall higher frame rate.

Several factors contribute to latency, or delay between the instant at which a video frame is generated by the ESEM and the time that same frame is displayed in the video widget.

- Acquisition by the frame grabber is essentially instantaneous.
- The new frame must be encoded; this is very fast for all applications meant for streaming live video.
- Most players spend time buffering frames for playback. For live video, this behavior should be minimized or suppressed entirely.
- The new frame must be decoded and displayed by the viewer; this also is very fast for applications that stream live video.

A solution should be chosen that does not entail significant latency. Note that although encoding and decoding are very fast, they do take time, and this time does affect the resulting frame rate.

Quality of the resulting video

Streaming video is an inherently high-bandwidth practice. Because of this, there is always a compromise in image quality between the video source and the displayed result. There are three areas where quality is reduced to facilitate live streaming.

- Frame rate
  By displaying video at a lower frame rate, it is possible to directly reduce all aspects of the demands engendered by streaming video. Processing demands on both the encoder and decoder are reduced, as well as bandwidth for both the server and viewer.

  During focusing, a low frame rate is not a great handicap. If the user can make a focus change, hesitate for a moment to see the change take effect, and then make another change based on that feedback, then this mechanism should work well. The greatest issue here is image latency. Note that if there is a low frame rate during focus, it is more important to get an image that shows the final result of the focus change than to get the intervening frames that show the change taking place.

- Frame size & resizability
  For streaming video the size of the viewer window is often small, and may be tightly restricted by the chosen implementation. As mentioned above, a large video frame is not required for fine focus control, but is desirable for stage navigation. The ability to resize the video display area is not a requirement, as
long as one or more acceptable frame sizes are available. Small frame sizes
directly reduce the demands of encoding, transmitting, and decoding the video
stream.

- **Image fidelity**
  Our source for video streaming is video only (no audio), and black and white (no
color). Any server that supports audio and/or color video streaming should have
these features disabled, if possible, unless they have no measurable impact on
latency or frame rate.

  By sacrificing image fidelity, it’s possible to indirectly reduce the demands of
streaming video. Though the frame rate remains unchanged, the per-frame
processing load on both the server and viewer software is reduced. Additionally,
bandwidth demands for both server and viewer are reduced, due to the smaller
compressed size of frames that carry less data. This latter effect is the most
important consideration when managing image fidelity.

  For focus control, it is important that image fidelity not be reduced to the extent
that it interferes with determining focus accuracy. Since focus control can be
restricted to a small video frame, it may be possible to have both a high frame rate
and high image fidelity while doing focus control.

  For stage navigation, image fidelity is somewhat less important. To aid
navigation, the frame area should be as large as possible. On the ESEM itself,
stage navigation in TV mode utilizes a significantly degraded image compared to
that seen in scan mode. Note that the noisy image in TV mode may in fact reduce
the ability of encoding software to compress the image. During development, we
need to ensure that image fidelity is sufficient to readily identify significant
artifacts on the sample. Latency must be minimized to allow the user to drive to a
location and stop without over-shooting the target.

## The Market

There are six main markets driving streaming video on the Internet. The vendors that
target these different markets take differing approaches, and make their money in
different ways. These market segments are identified below, listed in order of increasing
sophistication of the streaming video marketed to each.

1. **Personal web cam owners.** These users are often hobbyists, and have limited
   funding. The streaming video that is available in this market sacrifices one or more of
   the quality criteria above, frame rate, frame size, and image fidelity. For streaming,
   the best of these products are open-source. The simplest are packaged with off-the-
shelves web cams. Some software in this category may be worth our interest.

2. **Security.** Software for this market is generally targeted at personal and business web
   cam owners whose needs are not met by the software that comes with the cameras.
   This includes the ability to monitor a large number (4 to 16 or more) of cameras
   simultaneously, plus features such as motion detection and camera pan/tilt/zoom
   control. Software in this segment is not of interest for our needs.
3. **Internet pornography.** These customers have needs that are very much like the security market, except that they generally want to host a large number of streams, rather than just one or two. Due to their interest in multiple feeds, this market’s software is also not likely to meet our needs.

4. **Internet advertisers.** There are two types of Internet advertisers in this market, those who seek widespread dissemination of TV-like advertisements, and those who wish to stream marketing videos on corporate web sites. For my purposes, I place the latter group in the next category, Internet media sites. For the former group, whose interest is mass advertising, suppliers are often interested in providing the distribution channel (servers and bandwidth). These products are expensive and generally not suitable for our needs.

5. **Internet media sites.** Most of these sites, whether streaming music video clips, movie trailers, web-based monitoring systems, or corporate marketing materials, want to stream to numerous simultaneous viewers. The servers are usually hosted by the customer and are expensive, though vendors often charge on the basis of the number of simultaneous streams served. Our costs may therefore be quite low; in fact, some of these servers allow “proof-of-concept” use for a single stream, which may be all we need for Bugscope. This market segment seems to most closely share our needs.

6. **Video conferencing.** These vendors target corporate customers, and often make their money by hosting the streaming servers, charging for bandwidth used, or on the basis of the number of streams served. Many of their customers need these abilities only occasionally. For those customers who use video conferencing on a regular basis, server installations at the customer site are available. These tools are usually expensive, but some may be of interest for our purposes.

## Options

There are so many options for streaming video at this time that this document does not attempt to thoroughly review them all. Review and prioritization of the criteria mentioned above should make an examination of the options more profitable by allowing the choice to focus on those criteria of greatest importance to the success of the project. Where possible, I address these criteria in the descriptions below. Additionally, I have contacted some of the commercial vendors listed, and feedback from those exchanges is included as well.

Currently, there appear to be 3 general categories of vendors for the streaming of live video across the web.

1. **Major-player commercial streaming applications.** The major drawback with each of these is that the vendor’s player is required to render the video stream. Use of a standard player makes it impossible to render the video within a widget in the client interface, entirely under client program control. Fortunately, all of these vendors provide open-source or SDK solutions for viewer implementation.

   These products are marketed to Internet advertisers, Internet media sites, and corporations that need video conferencing ability.
• **Apple QuickTime**
  The Darwin server is available for free as an open-source solution. With QuickTime for Java (free), we can readily incorporate streaming QuickTime into a Java video widget.

• **Microsoft Windows Media**
  This format is compatible with the Windows platform only. However, the Windows Media Encoder and Windows Media Server applications are available for free from Microsoft, and can run on Windows NT 4.0 (NT Server not required). The Windows Media Player SDK includes an ActiveX control that can be embedded in web sites.

• **RealSystems RealMedia**
  The cost of the server can be quite high, although for our limited needs licensing costs may be low or non-existent. The availability of the RealSystem SDK means that it should be possible to integrate the viewer into the Bugscope client. The server and production tools, including those for live streaming, have just recently been released as open source.

2. **Open-source solutions.** None of these appears to be sufficiently mature to provide confidence in a straightforward implementation & deployment. These products are marketed to personal web cam owners, users of security cameras, and small Internet media sites.

  • **camsrv**
    This server streams to a Java-based client or saves JPEG images. It’s small, and was intended for monitoring a home web cam. This product may not support our video source, and is unlikely to meet our needs.

  • **FFmpeg**
    This server streams MPEG-4 over HTTP (RTSP reportedly under development). It supports formats compatible with Windows Media Player and RealPlayer. There are documented problems with playback in Windows Media format.

  • **mp4live (MPEG4IP project)**
    This server streams MPEG-4, and runs under Linux only. Although documentation indicates that it works with live broadcasts, it may not support our video source, and may not be compatible with our need to start and stop streaming quickly under program control.

3. **Alternative commercial streaming applications.** These products are marketed to Internet advertisers, Internet media sites, and corporations needing video conferencing ability.

  • **Cleaner Live**
    This server streams to Windows Media Player or to RealPlayer only, and is meant for streaming from a digital video camera. Its advertised cost is $2,995. Dedicated hardware is required to support the server. It is unclear whether this product would support our video source.
• **ClipStream**  
This $3,000 product streams over HTTP to a Java client. Dedicated server hardware is required. They are anxious to work with us, and will provide a 14-day license to try it out. Their client list includes a number of large corporations. This product appears to be a viable option.

• **Envivio Live Broadcaster**  
This product streams MPEG-4 using UDP (RTP & RTCP). This is a fairly high bandwidth product that might be able to meet our needs. The server requires dedicated, high-end hardware. It is unclear whether we would have sufficient control over the viewer, though an embedded player for MPEG-4 is likely available from one of the major players listed above.

• **Essential Viewing**  
This product streams a proprietary format over HTTP to a Java viewer that can be embedded. This appears to be a very capable product that could serve our needs. They advertise real-time capability for video conferencing, as well as total control over the viewer. Although they apparently do not have a licensing structure that addresses our needs, they have verbally offered to provide a single-stream license for free, in exchange for a favorable mention in any papers we write.

• **helloNetwork’s helloLive**  
This is another product that streams to a Java client. It appears to be full-featured, though it is unclear whether we would have sufficient control over the viewer, including controlling its size.

• **Kasenna MediaBase XMP**  
This company is anxious to do the actual distribution of the video stream, in exchange for a charge for bandwidth. Additionally, it is unclear whether this option requires use of their proprietary viewer, or if the display could instead be incorporated into a widget in the client interface. This product does not appear to be a viable option.

• **Palantir**  
This low-cost product runs on Linux only and streams to a Java viewer. They advertise the ability to communicate with hardware connected to the server. It is unclear what level of performance we could expect. They have a partners program that provides the product for free, if the server is going to be left on-line for viewing of “something interesting”.

• **StreamEZ**  
This product offers a great deal of functionality, but the Java-based live viewer is still under development.

**Conclusions**  
A review of these options leads to the following provisional conclusions and recommendations.
• We are not currently streaming any media from our web site. There is neither streaming server software nor dedicated hardware installed.

• We already have Cleaner in house; there may be a synergy related to selecting Cleaner Live.

• Some of the non-mainstream commercial vendors, particularly ClipStream, Cleaner Live, and Essential Viewing, are worth a closer look.

• Apple provides the Darwin open-source streaming media server for QuickTime. RealSystems distributes the Helix streaming media tools in open source. Microsoft gives away the Windows Media encoder, server, and SDK. These are combination products, the free or open-source off-spring of leading commercial streaming media companies. One of these solutions is likely to serve us well.

Depending on the amount of time that we want to dedicate to selecting among these solutions, I recommend getting trial software from ClipStream and Essential Viewing, a further investigation of Cleaner Live, and a trial of the software available from RealSystems.