Adaptive Data Delivery Over Disadvantaged, Dynamic Networks

Rickenbach, Brent; Griffin, Peter; Rush, Jason; Flanagan, John
General Dynamics Advanced Information Systems
Bloomington, MN, USA

Adamson, Brian; Macker, Joseph
Naval Research Laboratory
Washington, D.C., USA

Abstract—Most applications and network protocols are designed under the assumption of a stable and consistent physical network infrastructure. However, the communications infrastructures of most military tactical networks are dynamic and dependent on mission CONOPS and network topology. Further, they are highly variable in their capabilities from one data link to the next. Consequently, it is difficult for standard applications and services to maintain reliable quality of service when transitioning to a dynamic tactical network, or crossing the borders between different classes of military networks. Ultimately, this lack of network awareness impacts the quality and timeliness of actionable intelligence delivered to the warfighter.

General Dynamics and the Naval Research Laboratory (NRL) have collaborated on efforts to improve the reliable delivery of live video data across these disadvantaged tactical networks. We address the development of a network-aware video server, which senses network characteristics and dynamically transcodes video to meet the capacity of network between the server and clients.

End-to-end network characteristics are required to drive the video transcoder. These network attributes can either be derived from software methods by analyzing the network protocol, or using specialized network hardware (e.g. tactical radios, routers, or QoS devices). For this effort end-to-end network awareness was collected from the NACK-Oriented Reliable Multicast (NORM) [1] protocol operation. Hooks in the NORM protocol provide a network information service with access to path bandwidth, delay, and lost packets. The video server uses the network information service to adapt the video by adjusting frame size, frame rate, and compression according to current network conditions to deliver the video with minimal delay.

A comparison of how video is streamed over enterprise networks and the improved delivery methods made by General Dynamics and NRL for tactical networks is presented.

I. INTRODUCTION

Delivery of real-time video over heterogeneous, dynamic tactical networks is difficult to maintain at a reliable quality of service for the end user. Video is easily broken and unrecognizable, reducing the mission effectiveness in tactical environments. This paper addresses approaches to improve the delivery of data such as video over disadvantaged networks. Specifically we outline the use of the NACK-Oriented Reliable Multicast (NORM) protocol used in conjunction with a dynamic video transcoder that reacts to changing network characteristics to provide a high quality real-time video stream.

The improvement in video quality is quantified by the use of SSIM, described below. The results of the analysis will show that a dynamic transcoding approach based on network information provides an overall improved quality video stream than traditional streaming approaches over disadvantaged links.

II. BACKGROUND

This section discusses the building blocks required to develop a video delivery system capable of providing higher quality, real-time video than common approaches used today. Current transcoding approaches select a frame rate, frame size and bit rate at the start of the stream based on the type of traffic. Some types of traffic may send based on network and device limitations [2,3]. We introduce the concept of network-aware applications, common video streaming approaches, the NACK-Oriented Reliable Multicast (NORM) protocol, and finally the combination of each into a network-aware video server capable of changing video fidelity mid-stream.

A. Network Aware Applications

A network-aware application is a service that transmits data over the network and changes what it sends based on network characteristics. Common network characteristics may include: bandwidth, latency, and bit error rate, but can be dependent on the particular communications system in use. Network characteristics may come from external sources to the application such as radios and routers, or derived internal to the application with network protocols that expose such information or via application-specific methods. This data is used to calculate relative available network bandwidth allowing network-aware applications to adapt transmission rates.

There are a number of ways to change what an application sends based on the type of traffic. Some types of traffic may include: streaming data such as video and voice, periodic messaging such as status and heartbeat messages, asymmetric messaging such as chat, and bulk transfers such as large file sets, imagery, and more. Depending on the source and sink of each data type above, different approaches are required to adapt to different data types. For this paper video data will be sent from a network aware server to a player in a simulated real time environment and the server will attempt to adapt to various network conditions.

B. Video Servers

Many methods exist for the distribution and streaming of video over a variety of networks, and this paper focuses only on IP delivery of video. Current and emerging standards for
video delivery include web-based approaches such as HTML5 or Adobe Flash over an HTTP/TCP transport, as well as RTP and raw MPEG-TS streams over UDP.

Existing approaches for delivering uninterrupted video to clients today tend to involve buffering and/or pre-processing a video stream in to multiple streams of different quality thus consuming different bandwidth. One example of the latter is HTTP Streaming, a proposed IETF standard. Here existing videos are transcoded and stored on servers to fit multiple bandwidths and chunked into multiple smaller files. When the client requests a video, it receives a playlist file of the file chunks, which it then requests each file with a bandwidth parameter for the server to match against. This and other buffering techniques may be sufficient for archived, non-real time data; however it provides no benefit to the delivery of real-time sensor data.

C. NACK-Oriented Reliable Multicast

NORM is a reliable transport protocol for both unicast and multicast transport of streaming, bulk, and message data. NORM is a powerful protocol that allows for configuration options to suit various network environments. In general, NORM can provide end-to-end reliable transport of data, utilizes a negative acknowledgment mechanism reducing protocol overhead, supports congestion control to share network capacity with other protocols, and uses FEC-based repair packets to minimize NACKs in a multicast environment.

NORM features are also useful in enabling network-aware applications. The congestion control algorithm can be used to approximate the available bandwidth of the network, allowing the application's network provider to be driven internally as opposed to integrating with external components.

The NORM_CMD(CC) message contains information to allow measurement of RTTs, to inform the group of the congestion control “current limiting receiver” (CLR), and to provide feedback of individual RTT measurements to the receivers in the group. NORM uses the RTT and reported lost packet information to compute a receive rate. NORM compares the reported receive rate to the current send rate and adjusts the send rate as needed. While the NORM protocol and even the specified rate control algorithm have attributes well suited for the often wireless environments needed for tactical systems, the TCP-Friendly aspect of its design leads to some of the same limitations of TCP in wireless systems. Most notable is the negative effect of packet loss due to bit errors or contention that often occurs in wireless systems. Both TCP and the TCP-Friendly NORM-CC assume packet loss is due to buffer overflows of congestion and back-off their transmission rate. However, with NORM, its rate control is a separate, “plugable” component that can be replaced. [4]

The NORM protocol offers an interface to enable (or disable) the NORM sender congestion control operation for the session. For best operation, this function is called before the call to NormStartSender() is made, but congestion control operation can be dynamically enabled/disabled during the course of sender operation. If the value of the enable parameter is true, congestion control operation is enabled while it is disabled for enable equal to false. When congestion control operation is enabled, the NORM sender automatically adjusts its transmission rate based on feedback from receivers. If bounds on transmission rate have been set (see NormSetTxRateBounds()) the rate adjustment will remain within the set bounds.

The NORM Congestion Control can operate in a monitor and report mode with feedback collected from the receiver set and the CLR identified, except that no actual adjustment is made to the sender’s transmission rate. I.e., the transmission rate that was set by NormSetTxRate() is observed by the sender regardless of the feedback received. The NORM_TX_RATE_CHANGED notification will still occur as if the rate were being adjusted and the value returned by NormGetTxRate() reflects the rate that would have been used had the adjustRate parameter been enabled even though no actual rate change has occurred. The purpose of this variation of NORM Congestion Control operation is to allow applications to get a "suggested" rate from the NORMCC mechanism. [5]

This work leveraged the NORM Congestion Control’s monitor-and-report mode. NORM would report to a network information service using the NormGetTxRate() method. The Network information service smoothes the NORM transmit rate reports using a weighted moving average and notifies interested network aware applications periodically. The moving average filter improves sense and response operations over tactical wireless comms.

D. Network-aware Video Server

A network-aware video server is an implementation of a network-aware application with the purpose of delivering real-time video over a changing network through active transcoding. Specifically, our implementation is able to stream from multiple sources such as a file, live sensor, or existing network stream. Further it supports multiple output formats such as RTP/UDP, NORM, RTP via NORM, and more.

The network-aware video server encodes source video in real-time based on current network information using encoding profiles. Encoding profiles define compression characteristics (e.g. bit rate, frame size, frame rate, etc.) of the video during transcoding. As mission needs change, the profile is modified to meet the current mission requirements. Figure 2 shows a snippet of a configuration file highlighting the profiles for
video transcoding. A simple-step function is used to select video compression characteristics based on current network bandwidth. In the config file, profile 1 is defined to be used when the network path bandwidth is sensed to be between 0 and 1 Mbps. For example, profile 1 incorporates a stream bit rate of 256K, reduces the HD video frame to 640x360 pixels, and modifies the frame rate to five frames per second.

```
<profile name="profile1" bandwidth="1000000" maxTxRate="1000000"
bitRate="256000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="15" maxBFrames="5" />

<profile name="profile2" bandwidth="1500000" maxTxRate="1500000"
bitRate="512000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="30" maxBFrames="5" />

<profile name="profile3" bandwidth="2500000" maxTxRate="2500000"
bitRate="1000000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="60" maxBFrames="0" />

<profile name="profile4" bandwidth="4000000" maxTxRate="4000000"
bitRate="2000000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="150" maxBFrames="0" />

<profile name="profile5" bandwidth="6000000" maxTxRate="6000000"
bitRate="3000000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="300" maxBFrames="1" />

<profile name="profile6" bandwidth="8000000" maxTxRate="8000000"
bitRate="4000000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="300" maxBFrames="0" />

<profile name="profile7" bandwidth="10000000" maxTxRate="10000000"
bitRate="5000000" width="640" height="360" timeBaseNum="1"
timeBaseDen="1" gopSize="300" maxBFrames="0" />
```

Figure 2. XML Configuration file showing video transcoding profile block.

Specific to the tests shown in this paper, NORM is the network provider using its congestion control algorithm to approximate the available network bandwidth.

```
Figure 3. Example size reduction of video compression techniques
```

### III. APPROACH

This study was designed to investigate the ability of NORM to deliver a simulated live video stream across different network environments while maintaining video quality. The study tested the benefit of using NORM rate adaptation and forward error correction (FEC) on the overall quality of the received video stream. Transmission protocols tested included:

- Non-adaptive, non-NORM (control)
- Non-adaptive, NORM-enabled
- Non-adaptive, NORM-enabled with FEC
- Adaptive, NORM-enabled
- Adaptive, NORM-enabled with FEC

Comparing the received videos transmitted over these protocols to both the original source video as well as the video stream output by the server allowed for a quantitative analysis to be done of their ability to deliver quality video over various network conditions.

#### A. Testing

A single test consisted of a set of network configurations, the protocols to be tested, and whether or not to attempt to actively adapt to the network conditions.

The following four testing configurations were defined:

<table>
<thead>
<tr>
<th>Test</th>
<th>Available Bandwidth</th>
<th>Packet Loss</th>
<th>Protocols</th>
<th>Adaptive</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>10 Mb/s</td>
<td>None</td>
<td>UDP, NORM, NORM w/ FEC</td>
<td>NO</td>
</tr>
<tr>
<td>2</td>
<td>5 Mb/s</td>
<td>1%</td>
<td>UDP, NORM, NORM w/ FEC</td>
<td>NO</td>
</tr>
<tr>
<td>3</td>
<td>5 Mb/s</td>
<td>1%</td>
<td>UDP, NORM, NORM w/ FEC</td>
<td>YES</td>
</tr>
<tr>
<td>4</td>
<td>Changing: 10 Mb/s, 5Mb/s, 1 Mb/s, 512 Kbps</td>
<td>Changing: None, 1%, 2%, 3%</td>
<td>UDP, NORM, NORM w/ FEC</td>
<td>YES</td>
</tr>
</tbody>
</table>

These tests were designed to test a range of network conditions, as well as test NORM’s ability to actively adapt to a changing network. In order for NORM to be able to adapt to differing network speeds several profiles were defined for the server to chose from based on the amount of bandwidth being reported by NORM. These profiles include settings for video resolution, frame-rate, and encoding bit-rate to allow the server attempt to fit the video stream into the available bandwidth.

To ensure consistency and accuracy of testing a Perl script was written to remotely control the testing machines. This includes setting up the necessary network emulator, launching the test server and player, and gathering the log files for analysis. The network control took place over a physically separate network card to ensure that control traffic did not interfere with the results of the test.

#### B. Data Collection

The tests were conducted by transmitting a source video file between two computers over a dedicated network connection. A third machine capable of emulating impaired network conditions was connected between the two test machines in order to test communication over disadvantaged networks.

The inline network emulator is capable of simulating latency, packet corruption, packet loss, and available bandwidth; however, only the two latter parameters were adjusted in these tests.

In order to be able to perform analysis on the quality lost over the network both the server and the player software are designed to log out all network communication that occurred on their end of the link. This data was used to extract the video signal that was sent by the server and the video signal that the player was able to decode to compare the relative quality.

#### C. Quality Analysis

In order to objectively analyze the quality loss introduced by the network transmission a tool was developed to read in log
files generated by the player and server machine and produce a numerical score using an industry standard algorithm.

Both Peak Signal to Noise Ratio (PSNR) [6] and Structural Similarity (SSIM) [7] were tested as possible algorithms to compute an objective quality. PSNR and SSIM are both objective methods for measuring image quality. PSNR is based on the mean square error calculated between two images and SSIM is a combination of the loss of correlation, luminance distortion, and contrast distortion. SSIM was selected for our tests as it performs slightly better than PSNR in discriminating image quality degradation caused by frame compression [8].

The quality of video received by the player was compared to both the original source video to find total quality loss, and the transcoded video that was sent by the server to find quality lost resulting from network transmission.

IV. RESULTS

The test results illustrate a noticeable improvement in the quality of the video the player was able to decode as additional capabilities were added to the stream. Notice that RTP NORM w/ Parity was able to maintain a perfect duplicate at all but three short periods throughout the stream while RTP NORM and RTP have noticeably lower results.

Figure 4. SSIM results comparing received video frames to sent video frames

Figure 4 illustrates the result of comparing a video the player machine was able to decode to the stream that was sent over the network by the server. In the figure different network types are drawn in different colors as indicated by the key in the top right corner of the figure. A score of one on the graph indicates that at the given frame number the frame decoded by the player was identical to the frame sent by the server. Any score less than one represents network data.

Figure 4 depicts the results of a test run over a 5 Mb/s network with 1% packet loss introduced. The server machine is not running a network provider, so all three lines are attempting to encode the outgoing video at the same settings throughout the test.

RTP refers to a UDP stream wrapped with an RTP header to allow for video decoding timing. RTP NORM is also a UDP stream with RTP headers, but has the added functionality of NORM so that the player can NACK missed packets and attempt to repair the video in time for playback. Lastly RTP NORM w/ Parity is another RTP NORM stream, but with Parity/Forward Error Correction sent along with the stream to allow the player to try and repair the video without requiring the server to send new data.

RTP refers to a UDP stream wrapped with an RTP header to allow for video decoding timing. RTP NORM is also a UDP stream with RTP headers, but has the added functionality of NORM so that the player can NACK missed packets and attempt to repair the video in time for playback. Lastly RTP NORM w/ Parity is another RTP NORM stream, but with Parity/Forward Error Correction sent along with the stream to allow the player to try and repair the video without requiring the server to send new data.

Figure 5 shows the result of test 4 in which the network emulation machine cycles through four increasingly impaired network configurations as time increases. In both NORM tests the network provider is enabled to allow the server to attempt intelligently adapt to the changing network conditions. As the network conditions deteriorate the video server is able to lower the bit-rate of the video enabling it to successfully traverse the network. This does bring up the possibility that the lower bit-rate video, even though it is received in better shape, may be of lower overall quality than one with a higher bit-rate and more packets lost. This is addressed in Figure 6.

Figure 5. SSIM measurement comparing the received video to the transcoded video

Figure 5 also depicts the result from test 4. However, this graph is comparing the quality of the video received by the player to the original source video. This graph is to illustrate the combined effect of loss do to the server re-encoding the video before sending, in addition to the network loss. This graph illustrates that with the use of a network provider the NORM streams were able to maintain a higher overall quality, especially at the end of the test. This was accomplished
because the server was able to change encoding profiles to create a lower fidelity video to transmit, but one that would be able to successfully be received on the player side. The RTP line on the other hand shows what happens to overall quality when the server attempts to send a video at a higher bit-rate than the network can handle.

V. ADAPTIVE PARITY

Additional testing was done to evaluate an observed shortcoming in the NORM network provider. In situations where the network bandwidth cap grows rapidly higher than previously available the network provider may only reflect a relatively small amount of bandwidth over the current transmission rate. If the reported bandwidth cap is between two encoding profiles the server will continue using the lower of the two, even though the network may be able to support the higher profile, the network provider will not notice the extra room unless more of the reported bandwidth is consumed.

To address this issue an adaptive parity feature was added to the network aware video server. The adaptive parity feature gradually increases the number of parity packets added to the video stream, thereby increasing total bandwidth usage, when the available bandwidth is between two encoding profiles. Ideally the increased bandwidth use will help the network provider report a more accurate bandwidth cap. This process of adjusting the number of parity packets continues until the provider reports enough bandwidth to jump to the next encoding profile and the process starts over. Parity packets are used so that as additional bandwidth is being added so is signal redundancy. If adding more parity packets begins to cause packet loss the large number of parity packets will help the client repair the video.

A. Results

Figure 7 illustrates the visual quality of video received compared to the source video before encoding. This test consisted of five sections in which the network conditions alternated between an essentially unhindered network, and one with very low available bandwidth. Areas in which the Smart Parity line is above the standard NORM line represent an segment were the adaptive parity system was able to identify enough available bandwidth to increase the encoding profile while the standard NORM system was not.

The adaptive parity feature enabled the network provider to more accurately estimate the total available network bandwidth allowing the video server to use a higher quality encoding profile. This resulted in the visual quality of the video received by the client to be improved at times when the network conditions improve quickly versus ones transmitted by the standard video server.

VI. FUTURE WORK

Although the system for adaptive parity was developed and produced results, further adjustments and system tuning could allow for improvements in the server beyond those produced by the current implementation. Future enhancements could allow for the server to more quickly and accurately increase the number of parity packets added to the stream causing improved response time and decreased packet loss.

VII. CONCLUSION

Through the use of a network-aware video server the quality of the video received over compromised or changing networks can be greatly increased. The server is able to reduce the overall bandwidth consumption enabling the video transmitted to be received with better overall quality than one sent at a higher source quality that cannot be supported by the current network conditions.

REFERENCES


Figure 7. SSIM results showing the improvements of NORM with Smart Parity