

EE384b Project Report

A Simulation of MPEG-2 Video Delivery Quality under different Packet Loss Patterns

Serene Tan sereneta@stanford.edu Stanford University
Gary Duan garyduan@yahoo.com Cisco Systems

ABSTRACT

Real-time video stream transmitted over the Internet normally use UDP as its transport layer protocol. Packet loss is inevitable as the network traffic increases. Analysis shows that different packet loss patterns affect the perceived quality of delivered video stream differently, through several important factors, such as compression schemas, the nature of the content and concealment techniques.

The purpose of this project is to understand how different packet loss patterns will affect the quality of video delivery. We begin with an analysis of several factors that packet loss characteristics will affect video quality. The simulation results then are presented. Video streams with various bandwidth requirements and video content are used. Random loss model, bursty loss model and Gilbert loss model are investigated. Their effects on video quality are calculated. We conclude with the discussion of possible future works.

Keywords: MPEG-2 video, packet loss pattern, quality measurement, loss concealment

1. Introduction

Real-time multimedia delivery has become an important factor of today's Internet traffic. Because of its high overhead, TCP is not suitable for real-time applications. Video and Audio transmitted over the Internet typically use UDP, or RTP over UDP, as there transport layer protocol. UDP, as a "best effort" datagram service, gives no assurance for the streaming packets to be transferred and transferred in time to the destinations. Although the design MPEG-2 video format take the tolerance of occasional packet loss into consideration, the bursty nature of both packet loss and video stream itself make the real-time application especially sensitive to different loss pattern. Packet loss reduces video quality depending on the type of lost information, the location of lost data, and the temporal distribution of the lost information.

Losses of syntactic data, such as headers, greatly affect the perceptual quality. Rebuilding video stream with some system information, such as video dimensions or frame delimiter, missing is sometimes impossible. The location of packet loss also degrades the video quality because of the compression method used in MPEG-2. Packet losses are propagated spatially and temporally. One slice could be transferred in several consecutive packets. The loss of one packet often makes the rest of the slice unrecoverable. Because P and B frames are decoded depending on other frames, for example, packet losses in I frame will affect all frames in the same group of pictures.

Pure random packet losses do not capture what really happen in the real world. For instance, if packet n has lost or delayed, packet $n + 1$ is also likely to do so. The bursty loss of packets also plays important roles in quality degradation. Bursty loss makes stream rebuilding even harder. It also makes loss concealment work less effectively. Finally, bursty loss and bursty transmission delay are inter-dependent. For the receiving application, delayed packets are normally discarded. This fact further reduces the quality of the delivered video.

In this project, we investigated different packet loss patterns and their impacts on video quality degradation.

2. Packet Loss Pattern

IETF IPPM working group has proposed a mechanism ^[3] to describe the properties of packet loss. Several metrics ^[4] have been defined. The following metrics are of most significance in this project,

- Type-P-One-way-Packet-Loss-Average
- Type-P-One-Way-Loss-Distance-Stream
- Type-P-One-Way-Loss-Period-Stream
- Type-P-One-Way-Loss-Noticeable-Rate

Several experiments ^{[1][2]} that try to capture the characteristics of packet losses of the multimedia stream transmitted over the public network have been performed. These results help us not only find out some important parameters, such as average loss rate, average loss length and distance, but also build mathematic models of packet loss pattern for network simulation.

The follow packet loss patterns are investigated in this project,

2.1 Random Packet Loss

This pattern has only one parameter, average loss rate π . The packets are dropped randomly with no dependency of each other. Loss rate $\pi = 1\%$, 5% and 10% are tested.

Programming algorithm:

```
drop = false;
if ( rng.uniform() < pi )
    drop = true;

// uniform() - ns-2 function:
// generate a floating-point number uniformly distributed on [0, 1)
```

2.2 Bursty Packet Loss

This pattern is a simplified model to simulate the bursty nature of packet losses. Maintain the same average loss rate π , whenever a packet is dropped, the next consecutive $n-1$ packets are also dropped.

Programming algorithm:

```
drop = false;
if ( drop_len == 0 || drop_len == n )
{
    if ( rng.uniform() < pi )
    {
        drop = true;
        drop_len = 1;
    }
}
else
{
    drop = true;
    drop_len ++;
}
```

2.3 Gilbert Packet Loss

Sanneck, Yajnik and Bolot recommend use of a Markov model to capture temporal loss dependency. The 2-state Markov model is also known as the Gilbert model ^[5].

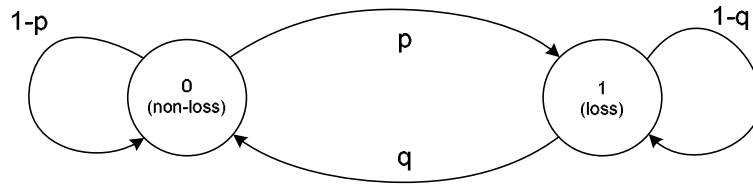


Figure 1: Gilbert Model: Two-State Markov Chain

p is the probability that the next packet is dropped, if the previous one has delivered. q is the opposite. $(1 - q)$ is the conditional loss probability (clp). We have the probability for state 0 and 1,

$$\pi_0 = \frac{q}{p+q}, \quad \text{and} \quad \pi_1 = \frac{p}{p+q}$$

Experiments show that Gilbert Model is a better description on packet loss distribution than Random Model.

Programming algorithm:

```

drop = false;
if ( drop_len == 0 )
{
  if ( rng.uniform() < p )
  {
    drop = true;
    drop_len ++;
  }
}
else
{
  if ( rng.uniform() < q )
  {
    drop = false;
    drop_len = 0;
  }
  else
  {
    drop = true;
    drop_len ++;
  }
}

```

2.4 Other Packet Loss Models

Extended Gilbert Model ^[5] replaces the 2-state Markov chain in Gilbert Model with $m + 1$ states Markov chain. It assumes the last m packet losses can affect the probability if the next packet will be dropped or not. Extended Gilbert Model is the most detailed model

for capturing loss run distribution. Because of its complexity and lack of experiments to provide enough parameters, we do not use this model in the simulation.

Inter-loss distance (ILD)^[4] is another metric to describe the distance between packet losses. If many of packet loss runs are close to each other, the sequence has small ILDs. Small ILD also worsen the video quality and degrade the performance of loss concealment. The model is not used in the simulation because there is no mathematic model available to calculate the packet loss sequence with given ILDs.

3. Simulation Design

The simulation is performed under ns-2 environment. ns-2^[6] is a discrete event network simulator that supports various protocols, such as TCP, UDP, multicast and wireless LAN. Implemented in C++, ns-2 provides many flexible and straightforward APIs for users to manipulate or extend the protocol stacks. The simulation scenarios are controlled by TCL scripts. Users can specify the parameters of the network nodes, links and protocol entities as well as simulation attributes in the TCL scripted.

3.1 Video Source

Two different video contents are tested. One is a clip of basketball game containing lots of fast object movements and environment switches; the other is a clip of political speech containing mostly hand and shoulder movements. Each content is then encoded with different bandwidth requirements of 2mb, 3mb, 4mb, 5mb and 6mb. The clips are encoded with Open-loop Variable Bit Rate (OL-VBR) scheme. The group of picture structures is IBBPBBPBBPBBI... i.e., it comprises 12 frames between I frames and 3 frames between anchor frames, ($M=3$ $N=12$). The frame size of the clip is 640x480 pixels. Each frame consists of 30 slices whose size is 40 macroblocks each. The frame rate of clips is 30fps. Each clip has 1800 frames, runs 1 minute.

3.2 Decoder and Quality Assessment

When a video stream is transmitted to the receiver endured packet losses and loss recovery, a new clip is generated. The new clip then is compared against the original clip to calculate the quality degradation. Several perceptual quality measurement metrics are proposed, such as MPQM^[7] and PDM^[8]. These metrics are based on the attributes of human visual system and provide relatively accurate measurement of video quality.

Peak Signal-to-Noise ratio (PSNR)^[9] is traditionally used as the quality assessment metric for video and voice data. Let $r(m,n)$ and $r'(m,n)$ stand for the luminance value of pixel (m,n) in a $M * N$ frame of the original clip and delivered clip, respectively. Assume pixels are one-byte coded,

$$MSE = E[|r(m, n) - r'(m, n)|^2]$$

$$PSNR = 10 \log_{10} \frac{255^2}{MSE}$$

PSNR is an objective metric which doesn't take into account at any level the human visual system comporment. Studies have shown the model is too primitive and poorly correlated with human perception. However, it does provide us general ideals about the quality of a video clip. For its simplicity, we still choose it as the video quality assessment metric.

We use *mpeg2decode* ^[10] software to calculate the luminance value of every frames in original clip and delivered clip. Then PSNR of each two corresponding frames are measured. We take the average PSNR of all frame pairs as the overall quality evaluation.

3.3 Network Design

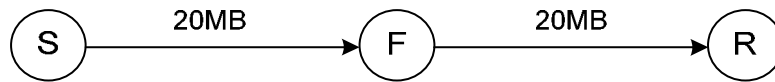


Figure 2: Network Simulation Topology

The simulation network is formed by three nodes, sender, forwarder and receiver. Sender is responsible for reading the MPEG-2 clip and initiating the video stream. Forwarder simulates different packet loss patterns and decides if a packet should be forwarded to the receiver. Receiver is responsible for concealing packet losses and re-building the video clip. The network parameters are designed so that no packet loss will take place because of congestion. Packet losses are totally control by the Forwarder application.

- Sender

Sender reads in video file as MPEG-2 elementary stream, parses it and packetizes it according to RTP payload format for MPEG video proposed in RFC 2250 ^[13]. The maximum payload size is 1500 bytes, plus a 12-byte RTP ^[12] header, a 4-byte MPEG video-specific header and a 4-byte MPEG video-specific header extension. Depends on the slice size, multiple small slices could be packetized into one packet, while bigger slice might take several packets to be transferred.

- Forwarder

Forwarder implements the logic of different packet loss patterns. When a video stream packet is received, the configured loss pattern computation is applied. If it is decided that the packet should be dropped, forwarder consumes the packet without forwarding it to the receiver.

- Receiver

Receiver accepts the video packet stream with some of packets missing. When losses can not be recovered by the transmission protocols, it is necessary to try to hide as much as

possible their visual impact. Loss concealment has to be done to achieve better playback quality. Based on the information in the RTP header and MPEG video-specific header, receiver can figure out if the stream is broken and what part of information is dropped, so that it is able to reconstruct the video clip.

Several error concealment techniques have been proposed, from the simple ones to the more sophisticated but also more computational expensive ones. Error concealment can be achieved by estimating the lost data from spatial-temporally adjacent data, or if layered coding presents, base layer data can be used to predicate other layer's packet losses. Another loss concealment method, Forward Error Correction (FEC) ^[14], transmit additional redundant packets which can be used for error recovery.

Error concealment schema used in this project combines temporal concealment and spatial concealment methods. Slice is the minimum unit for loss recovery, i.e. if a piece of slice data is missing, the whole slice is considered lost. Its impact on the quality of recovered video stream is discussed later. Receiver keeps a latest copy of successfully received slices for each frame type. When it detects a slice is broken, receiver replace the slice with the one in the same location in the previous frame of the same type. If the copy does not exist, the neighboring slice within the same frame will be used.

3.4 Software Design

Functionalities of Sender, Forwarder and Receiver are implemented in one ns-2 Agent class inherited from UDP Agent.

```
static class MpegAgentClass : public TclClass
{
public:
    MpegAgentClass() : TclClass("Agent/UDP/MPEG") {}
    TclObject* create(int, const char*const*) {
        return (new MpegAgent());
    }
} class_mpeg;

class MpegAgent : public UdpAgent
{
public:
    MpegAgent();
    int command( int argc, const char* const* argv );
    int timeout();
    void recv( Packet*, Handler* );
    void stop();

protected:
    enum MPEG_MODE { MPEG_SRC = 1, MPEG_FWD, MPEG_DEST };
    enum LOSS_PATTERN { LOSS_NOLOSS, LOSS_RANDOM, LOSS_GILBERT };
    .
    .
    .
}
```

- MPEG_MODE – Defines different roles (Sender, Forwarder and Receiver) of the agent.
- LOSS_PATTERN – Defines different packet loss patterns applied on forwarder.
- command() – Handles TCL commands triggered by TCL simulation script.
- Timeout() – A timer on sender is initialized as the simulation starts. The timeout value is set to 33ms. When timer expires, this function is called. Sender packetizes one video frame and sends out to forwarder.
- recv() – Function is called when a packet reaches the Agent. Different tasks are taken by forwarder and receiver.
- stop() – Function is called at the end of simulation. Housekeeping is performed at this time.

The following TCL verbs are implemented,

- mode** role – Defines agent roles. Possible values are “sender”, “forwarder” and “receiver”.
- debug** on/off – Turn on or off the agent’s debug output
- start** file – Specify the input video clip for sender agent; specify the output video file name for the receiver agent.
- stop** – Stop the agent, call `stop()` function.
- loss** pattern v1 v2 – Configure the loss pattern applied in forward agent. Possible patterns are
 - zero : No packet loss
 - random : Random packet loss. v1 is average loss rate; v2 is loss run length.
 - Gilbert : Gilbert mode. v1 is average loss rate; v2 is clp.

Appendix A lists a TCL script example used in the project.

4. Simulation Results and Analysis

We apply six different loss patterns to the “basketball” video sequence; four different loss patterns are used for the “speech” video sequence.

- r(1%, 1) Random packet loss. Average loss rate is 1%. No loss dependency.
- r(5%, 1) Random packet loss. Average loss rate is 1%. No loss dependency.
- r(10%, 1) Random packet loss. Average loss rate is 10%. No loss dependency.
- r(5%, 5) Bursty packet loss. Average loss rate is 5%. Loss run length is 5 packets.
- g(5%, 20%) Gilbert packet loss. Average loss rate is 5%. CLP is 20%.
- g(5%, 40%) Gilbert packet loss. Average loss rate is 5%. CLP is 40%.

Table 1 lists some typical packet loss distributions for Bursty and Gilbert model. Table 2 and Table 3 list the quality assessment of video clips for different bandwidth requirements under above loss patterns.

Table 1: Typical packet loss distribution

#	Model	0 (no loss)	1	2	3	4	5	6	7	8	9	10
1	r(5%, 5)	11059	0	0	0	0	104	0	0	0	0	2
2	r(5%, 5)	16404	0	0	0	0	176	0	0	0	0	6
3	r(5%, 5)	22464	0	0	0	0	228	0	0	0	0	0
4	r(5%, 5)	28662	0	0	0	0	293	0	0	0	0	2
5	g(5%,20%)	11011	383	71	14	4	1	0	0	0	0	0
6	g(5%,20%)	16464	548	112	25	5	2	0	0	0	0	0
7	g(5%,20%)	17273	542	116	28	4	1	0	0	0	0	0
8	g(5%,20%)	22476	746	141	8	1	0	0	0	0	0	0
9	g(5%,20%)	28582	1004	202	41	7	0	1	0	0	0	0
10	g(5%,40%)	10992	222	88	36	13	7	0	2	0	0	0
11	g(5%,40%)	16441	324	141	51	22	5	4	1	0	0	0
12	g(5%,40%)	17290	313	119	55	19	5	5	1	1	0	0
13	g(5%,40%)	22443	423	194	61	27	10	2	0	0	0	0
14	g(5%,40%)	28568	552	212	100	36	17	7	2	1	0	1

Table 2: Video quality assessment of the “basketball” clips

Basketball	2 MB	3 MB	4 MB	5 MB	6 MB
r(1%, 1)	4.28	4.20	4.20	3.91	3.84
r(5%, 1)	3.15	2.99	2.85	2.83	2.66
r(10%, 1)	2.60	2.50	2.50	2.42	2.25
r(5%, 5)	3.83	3.56	3.65	3.43	3.31
g(5%, 20%)	3.22	3.05	3.08	2.81	2.67
g(5%, 40%)	3.29	3.18	3.10	2.99	2.79

Table 3: Video quality assessment of the “speech” clips

Speech	2 MB	3 MB	4 MB	5 MB	6 MB
r(5%, 1)	3.48	3.42	3.39	3.35	3.30
r(5%, 5)	3.98	3.89	3.98	3.84	3.72
g(5%, 20%)	3.67	3.48	3.38	3.38	3.33
g(5%, 40%)	3.66	3.60	3.53	3.45	3.40

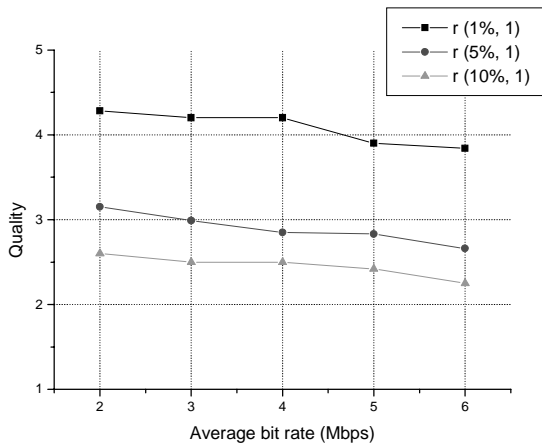


Figure 3: Video quality vs. average loss rate

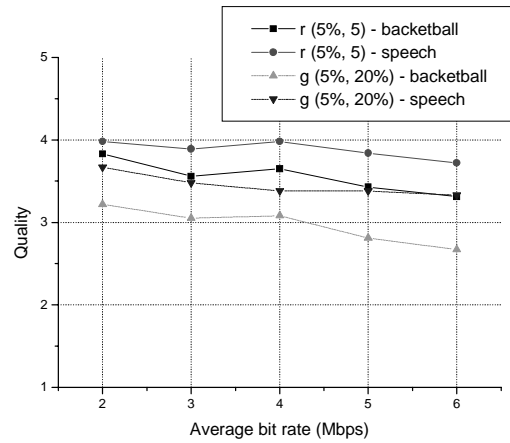


Figure 4: Video quality vs. content

1. Video quality vs. average loss rate

Figure 3 shows the impact of average packet loss rate on the video quality. It is clear that average loss rate plays the most important role in video quality degradation. 1% packet loss still gives us acceptable playback quality; when packet loss rate increases to 10%, video quality becomes poor.

2. Video quality vs. content

Figure 4 shows how video quality degradation is affected by video content. It is obvious that packet losses have greater effect for the clip with rich content. The reason that “speech” clip contains only slow object movement and most of area has the same color and uniform shape, where temporal error concealment works most effectively.

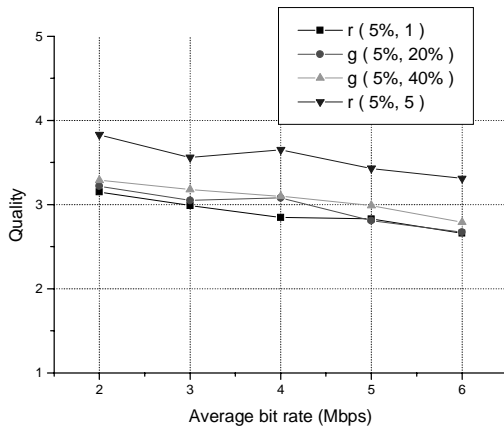


Figure 5: Video quality vs. loss run length (basketball clip)

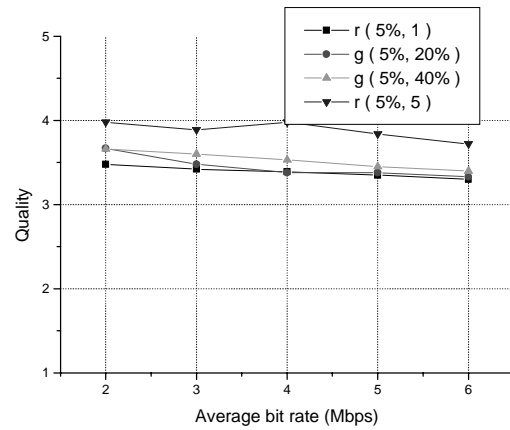


Figure 6: Video quality vs. loss run length (speech clip)

3. Video quality vs. loss run length

As we discussed, bursty losses makes broken video stream hard to recover, and in turn will further reduce the video quality. However surprisingly, from Figure 5 and Figure 6, we can see the result is reversed. Uniformly distributed packet losses $r(5\%, 1)$ has the worst quality, while the most bursty loss sequence $r(5\%, 5)$ shows the best result.

The unexpected result is rooted in the error concealment method used in the simulation. Lost packets which carry I frame have affect more on video quality. The error concealment works in this way, if one of the packets is lost, the whole slice is discarded. An I frame is normally transferred by several packets, so consecutive packet losses do not degrade the quality of that frame very much. As average packet loss rate keeps same, this means the burstier packet loss sequence has less loss runs, so the overall video quality is less affected.

4. Video quality vs. bandwidth requirement

From all of these curves, we can see that video quality does not change much with different bandwidth requirements, but consistently and linearly decreases as bandwidth requirements grows. This can be explained by PSNR metric we used to evaluate the playback quality. PSNR is only a subjective assessment metric, which does not consider how quality is “felt” by human visual system. The video clips with higher bit rate contain more detailed content in each frame. When these details are lost, the quality is measured low. We can expect when average bit rate reaches a certain limit so that higher bit rate does not provide more details any more, the quality curve will become flat.

5. Conclusions and Future Works

In this project, we built a network simulation environment for MPEQ-2 video delivery quality assessment. Different loss patterns and their effects on the playback quality are investigated. The simulation demonstrates the average loss rate and video content have the most significant impact on how much quality is degraded under packet losses. For implementation simplicity, we chose to use the very simple error concealment mechanism, and PSNR does not reflect perceptual video quality very well. Also, several other packet loss patterns can be studied, such as loss distance and noticeable packet loss metrics.

References

1. J. Boyce and R. Gaglianello, "Packet Loss Effects on MPEG video sent over the Public Internet", Proceedings of ACM MultiMedia, 1998
2. W. Jiang and H. Schulzrinne, "Modeling of Packet Loss and Delay and Their Effect on Real-Time Multimedia Service Quality"
3. G. Almes and S. Kalidindi, "A One-Way Packet Loss Metric for IPPM", RFC 2680, IETF, September 1999
4. R. Koodli and R. Ravikanth, "One-Way Loss Pattern Sample Metrics", Internet Draft, IETF, January 2001
5. H. Sanneck and G. Carle, "A Framework Model for Packet Loss Metrics Based on Loss Runlengths", Proceedings of the SPIE/ACM SIGMM MMCN, 2000
6. "The Network Simulator - ns-2", <http://www.isi.edu/nsnam/ns/>
7. C. J. Branden Lambrecht and O. Verscheure, "Perceptual Quality Measure using a Spatio-Temporal Model of the Human Visual System"
8. S. Winkler, "A Perceptual Distortion Metric for Digital Color Video", Proceeding of SPIE Human Vision and Electronic Imaging, vol. 3644, pp. 175-184, Jan. 1999
9. P. Frossard, "MPEG-2 over Lossy Packet Networks QoS Analysis and Improvement", SSC, July, 1998
10. MPEG Software Simulation Group, "MPEG-2 Encoder / Decoder", Version 1.2, July, 1996
11. J. Kimura, F. A. Tobagi, "Perceived Quality and Bandwidth Characterization of Layered MPEG-2 Video Encoding", Proceedings of the SPIE International Symposium on Voice, Video and Data Communications, September 1999
12. H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", IETF RFC-1889, Jan. 1996
13. D. Hoffman, G. Fernando, V. Goyal, and M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", IETF RFC-2250, Jan. 1998
14. V. Parthasarathy, J. W. Modestino and K. S. Vastola, "Design of a Transport Coding Scheme for High-Quality Video over ATM Networks", IEEE Trans. Circ. Syst. Video Techn. 7, pp. 358-376, April 1997

Appendix A: Simulation TCL script example

```
#Create a simulator object
set ns [new Simulator]

proc finish {} {
    global m0 m1 m2

    $m0 stop
    $m1 stop
    $m2 stop

    exit 0
}

set n0 [$ns node]
set n1 [$ns node]
set n2 [$ns node]

#Create a duplex link between the nodes
$ns duplex-link $n0 $n1 40Mb 10ms DropTail
$ns queue-limit $n0 $n1 200
$ns duplex-link $n1 $n2 40Mb 10ms DropTail
$ns queue-limit $n1 $n2 200

#Create a MPEG agent and attach it to node
set m0 [new Agent/UDP/MPEG]
$m0 set packetSize_ 2000
$m0 mode sender
$m0 debug off
$ns attach-agent $n0 $m0

set m1 [new Agent/UDP/MPEG]
$m1 set packetSize_ 2000
$m1 mode forwarder
$m1 loss random 0.05 1
$m1 debug on
$ns attach-agent $n1 $m1

set m2 [new Agent/UDP/MPEG]
$m2 set packetSize_ 2000
$m2 mode receiver
$m2 debug on
$ns attach-agent $n2 $m2

$ns connect $m0 $m1
$ns connect $m1 $m2

#Define output and input MPEG clip
$m2 start n-2-r-005-1.mpg
$ns at 0.01 "$m0 start news-2.mpg"

$ns at 70.0 "finish"

$ns run
```