TETCOS

Video Traffic Generator
White Paper

Video Traffic Architecture in NetSim

<table>
<thead>
<tr>
<th>Work</th>
<th>Author</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created</td>
<td>SP</td>
<td>20\textsuperscript{th} June 2012</td>
</tr>
<tr>
<td>Review</td>
<td>SS / PV</td>
<td>23\textsuperscript{rd} June 2012</td>
</tr>
<tr>
<td>Update</td>
<td>SP / PV</td>
<td>27\textsuperscript{th} June 2012</td>
</tr>
<tr>
<td>Review</td>
<td>SS / PV</td>
<td>3\textsuperscript{rd} June 2012</td>
</tr>
<tr>
<td>Update</td>
<td>SP</td>
<td>5\textsuperscript{th} June 2012</td>
</tr>
<tr>
<td>Review</td>
<td>SP/PV</td>
<td>11\textsuperscript{th} July 2012</td>
</tr>
<tr>
<td>Update</td>
<td>SP/PV</td>
<td>13\textsuperscript{th} July 2012</td>
</tr>
</tbody>
</table>

Authors:

Swati Pathak      IIT Kanpur
Shashikant Suman   TETCOS
Pranav Viswanathan TETCOS

©2012. TETCOS. This document is protected by copyright, all rights reserved.
Contact: sales@tetcos.com
## Table of contents

<table>
<thead>
<tr>
<th>Description</th>
<th>Page No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Abstract</td>
<td>3</td>
</tr>
<tr>
<td>2. The Need for a Sophisticated Video Traffic Generator</td>
<td>3</td>
</tr>
<tr>
<td>3. NetSim – Network Simulation software</td>
<td>3</td>
</tr>
<tr>
<td>4. NetSim’s Video Traffic Library</td>
<td>4</td>
</tr>
<tr>
<td>4.1. Model 1</td>
<td>4</td>
</tr>
<tr>
<td>4.2. Model 2</td>
<td>4</td>
</tr>
<tr>
<td>4.3. Model 3</td>
<td>5</td>
</tr>
<tr>
<td>4.4. Model 4</td>
<td>6</td>
</tr>
<tr>
<td>5. NetSim simulation results</td>
<td>8</td>
</tr>
<tr>
<td>5.1. Plot &amp; Performance Analysis</td>
<td>8</td>
</tr>
<tr>
<td>6. Inference</td>
<td>13</td>
</tr>
<tr>
<td>7. References</td>
<td>14</td>
</tr>
<tr>
<td>8. Appendix</td>
<td>14</td>
</tr>
</tbody>
</table>
Abstract

Video services have proliferated the internet thanks to a wide variety of video applications meeting the needs of both consumers and providers. Understanding the video traffic’s underlying architecture is the key to network design of the future. This paper provides an insight into the video traffic generators implemented in NetSim’s traffic generator library. The design, implementation and statistical analysis of the all video traffic generators have been carried out. The appendix contains the python routines used for the graphing. And, it contains the standard table of parameters (per the IEEE reference paper) used in fourth video traffic generator model.

The Need for a Sophisticated Video Traffic Generator

Digital video is expected to become a major traffic component on integrated services digital networks (ISDN’s). Applications such as video conferencing, video telephone, and switched TV, impose very large bandwidth requirements on public and private networks. Thus, video traffic constitutes a significant part of network traffic.

A sophisticated video traffic generator is essential for a variety of research needs such as network design, performance evaluation, bandwidth allocation, and bit rate control. A case in point would be network engineers intending to optimize traffic flowing over a network. Another interest could be to study the impact of bandwidth-reducing video compression algorithms on sharing network resources. For an instance, variable bit rate video coding can take advantage of dynamic bandwidth sharing via optimized statistical multiplexing and packet switching.

It is observed, however, that traditional models fall short in describing the video traffic because video traffic is strongly auto correlated and bursty. To accurately model video traffic, complex mathematical algorithms such as autocorrelations among data, IB&P frame generation, inverse transform of probability distributions, LRD and SRD characteristics of video ACF etc should be taken into consideration.

NetSim – Network Simulation software

NetSim- Network Simulation software is popular discrete event network simulator software, used widely for academic research. NetSim’s development environment infrastructure provides a unique opportunity to create custom models, simulate them and statistically analyze the results.

NetSim’s traffic library contains a wide variety of traffic generators for data, voice and video traffic. In this paper we have statistically validated NetSim’s video traffic library, covering four different traffic generation models, against standard reference papers.
NetSim’s Video Traffic Library

**MODEL 1: Continuous Normal VBR Video Traffic Model**

This model is the simplest of all models. It uses Normal Distribution for the generation of bits per pixel. In this model, consecutive packet sizes are independent of each other.

**Input parameters of the model**
- fps=frame/sec=10-50
- ppf=pixels/frame=10000-100000
- bpp=bits/pixel= Normal Distribution($\mu$, $\sigma^2$)

**Calculations**
- Inter Arrival Time (IAT)= 1/fps sec
- Generation rate(bits per sec)= bps =fps× ppf× bpp
- Generation rate(bytes per sec)= bps/8

**Output**
- Frame Size (bytes) = Generation rate (bytes per sec) × IAT

**MODEL 2: Continuous-State Autoregressive Markov Model**

This model incorporates the autocorrelation between the frames. Also, current packet size depends on the previous packet size via first order autoregressive Markov process. It models the coder rate as a continuous-state discrete time stochastic process. Let $\lambda(n)$ represents the bit rate of a single source during the nth frame. First order autoregressive Markov process $\lambda(n)$ can be generated by the recursive relation

$$\lambda(n) = a \lambda(n-1) + b w(n)$$

where $w(n)$ is a sequence of independent Gaussian random variables with mean $\eta$ and variance 1. And $a$ and $b$ are constants. It is assumed that $|a|<1$; thus the process achieves steady state with large $n$. The steady-state average $E(\lambda)$ and discrete auto covariance $C(n)$ are given by

$$E(\lambda) = \frac{b}{1-a} \eta$$

$$C(n) = \frac{b^2}{1-a^2} \eta^n \quad n \geq 0$$

The steady state distribution of $\lambda$ is Gaussian with mean $E(\lambda)$ and variance $C(0)$. One can use the experimental data to obtain $E(\lambda)$ and $C(n)$. And using these values it is easy to obtain the values of constant $a$ and $b$ and Gaussian parameter $\eta$. In the paper, values obtained from the measured data are $a=0.8781; b=0.1108; \eta=0.572$
**Note:** It is possible to get negative values for $\lambda(n)$ because of Gaussian distribution. To solve this issue, we regenerate $\lambda(n)$ until we get positive values.

<table>
<thead>
<tr>
<th>Input parameters</th>
<th>$\eta$ = mean of the Independent Gaussian random variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>$\lambda(n)$ = Frame size per second during the n\textsuperscript{th} frame</td>
</tr>
</tbody>
</table>

**MODEL3: Quantized-State, Continuous-Time Markov Model**

In this model the bit rate is quantized into finite discrete levels. This model takes uniform quantization step as $A$ bits/pixel. And there are $M + 1$ possible levels (0, $A$, ..., $MA$). Transitions between levels are assumed to occur with exponential rates that may depend on the current level. This model approximating the bit rate by a continuous time process $\lambda(t)$ with discrete jumps at random Poisson times. This model is analyzing a queue that is filled from $N$ variable rate sources each with rate $\lambda(t)$. This model considers the aggregate instantaneous input rate $\lambda_N(t)$ instead of the single source bit rate $\lambda(t)$. The total rate is the sum of $N$ independent random processes each with mean $E(\lambda)$ and variance $C(0)$ at steady state. Therefore, the steady state mean of $\lambda_N(t)$ will be $E(\lambda_N) = N \times E(\lambda)$ bits/pixel. The paper uses birth-death Markov model to describe the aggregate source bit rate. Then it follows that $\lambda_N(t)$ at steady state will have a binomial distribution with mean $E(\lambda_N)$ and variance $C_N(0)$ and exponential auto covariance $C_n(\tau)$.

\[
P\{\lambda_N(t) = kA\} = \left(\begin{array}{c} M \\ k \end{array}\right)p^k(1-p)^{M-k}, \quad p = \frac{\alpha}{\alpha+\beta}
\]

\[
E(\lambda_N) = MA p
\]

\[
C_N(0) = MA^2 p(1-p)
\]

\[
C_N(\tau) = C_N(0)e^{-(\alpha+\beta)\tau}
\]

Following values are obtained by matching above values with the measured values.

\[
\beta = \frac{a}{N \times a^2 \left(\frac{\lambda_N}{1+\frac{C_N(0)}{M}}\right)} = \frac{3.9}{1+\frac{2.6445BM}{M}}
\]

\[
\alpha = a - \beta = 3.9 - \beta
\]
with the number of quantization levels $M$ as a parameter, and for a given number of multiplexed sources $N$. The bit rate can span the interval $0 \leq \lambda_N(t) \leq MA$; $M$ should be selected large enough to span all likely bit rate values.

### Input parameters
- $N$ = Number of multiplexed sources
- $M$ = Number of quantization levels

### Output
$\lambda_N(t) = $ Bit rate

### Model 4: MPEG Video Traffic Model

The basic ideas behind this model is to decompose an MPEG compressed video sequence into several parts according to motion/scene complexity or data structure; each part described by a self similar process. Beta distribution is used to characterize the marginal cumulative distribution (CDF) of the self-similar processes. This model is a **simple IPB composite model**. In this model, the frames are organized as IBBPBBPBBPBB PBB... i.e., 12 frames in a Group of Pictures (GOP).

#### Decomposition of MPEG data

This model decomposes the MPEG compressed video sequence according to the MPEG Data Structure. The MPEG traffic is decomposed into three sub-sequences $X_I$, $X_P$ and $X_B$. $X_I$ consists of all I frames, $X_P$ consists of all P frames and $X_B$ consists of all B frames. We model each part and then combine them similar to the GOP pattern. To obtain a model that can capture the auto correlation functions of MPEG data, each part is modeled by a self-similar process. To generate the self-similar traffic asymptotically self similar fractional autoregressive integrated moving-average (F-ARIMA) process is being used. The algorithm of the **F-ARIMA process** is as follows:

1. Generate $X_0$ from a Gaussian distribution $N(0, \nu_0)$. Set initial values $N_0=0$, $D_0=1$.
2. For $k = 1, 2, ..., N-1$, calculate $\phi_{k,j}$, $j = 1, 2, ..., k$ iteratively using the following formulae

   $$N_k = r(k) - \sum_{j=1}^{k-1} \phi_{k-1,j} r(k-j)$$

   $$D_k = D_{k-1} - N_{k-1}^2 / D_{k-1}$$

   $$\phi_{kk} = N_k / D_k$$

©2012. TETCOS. This document is protected by copyright, all rights reserved.
Contact: sales@tetcos.com
\[ \phi_{kj} = \phi_{k-1,j} - \phi_{kk} \phi_{k-1,k-j} \quad j = 1, \ldots, k - 1 \]

\[ m_k = \sum_{j=1}^{k} \phi_{kj} X_{k-j} \]

\[ V_k = (1 - \phi_{kk}^2) V_{k-1} \]

Finally, each \( X_k \) is chosen from \( N(m_k, \nu_k) \). Thus we get a process \( X \) with ACF approximating to \( r(k) \).

The auto correlation function can be calculated in a recursive way as follows:

\[ r(0) = 1, \quad r(k + 1) = \frac{k+d}{k+1} r(k) \]

where \( d = H - 0.5 \).

\( H \) is called the Hurst parameter and \( k^{-\beta} \) is used as the ACF of a self-similar process. We get the value of \( H \) parameter for a self-similar process using the relationship,

\[ \beta = 2 - 2H \]

Distribution of these data is Gaussian. For data to be Beta distributed, the following mapping is being used

\[ Y_k = F^{-1}_\beta(F_N(X_k)), \quad k > 0 \]

\( X_k \): Self-similar Gaussian process,

\( F_N \): The cumulative probability of the normal distribution,

\( F^{-1}_\beta \): The inverse cumulative probability functions of the Beta model.

<table>
<thead>
<tr>
<th>Input parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \gamma, \beta ) and ( \eta ) for each of the I, B and P frames</td>
</tr>
<tr>
<td>Fps= frames per second</td>
</tr>
<tr>
<td>(sample values for ( \gamma, \beta ) and ( \eta ) are given in appendix)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>( Y_k = F^{-1}_\beta(F_N(X_k)) ) The packet size in bits per frame</td>
</tr>
</tbody>
</table>

©2012. TETCOS. This document is protected by copyright, all rights reserved.
Contact: sales@tetcos.com
Plot & Performance Analysis

Model 1

Plot of Bits per pixel Vs. Frame number

Plot of Generation rate Vs. Time

The python code for the following plots is given in the appendix.

©2012. TETCOS. This document is protected by copyright, all rights reserved.
Contact: sales@tetcos.com
Model 2

Plot of Packet Size Vs. Frame number

Plot of Frame size vs. Frame number

©2012. TETCOS. This document is protected by copyright, all rights reserved.
Contact: sales@tetcos.com
Model 4

Plot of I-Packet size vs. Frame number

Plot of B-Packet size vs. Frame number
Plot of P-Packet size vs. Frame number

Plot of GOP-Packet size vs. Frame number
Inference

NetSim simulations using these video models match the theoretical results and plots mentioned in the reference paper. For the convenience of the user, there is at least one parameter in all the models which can be varied by the user to get the desired bit rate. This is useful because as we increase the bit rate, the video quality improves. This section gives a brief description of the applications of these models:

- **Model 1**: This model can be used in the scenarios which require variable bit rate but no interdependence among frame sizes. For example, if we need to analyze the common buffer statistics, we need a model for the coded source rate. The rate depends on the compression algorithm and the nature of the video scene. For a scene without abrupt movement, such as the head of a person in a picture phone, we expect the rate to have a bell shaped stationary probability density. In such a scenario our first model can work very well as it uses Gaussian distribution and hence we get the bell-shaped probability distribution.

- **Model 2**: This model, which was used to generate source data in simulation experiments performed in the reference paper, can be best used to generate the bit rates of sources in queuing simulations experiments.
• **Model 3:** This model is a quantized state continuous time Markov process. The analytic model can be used in scenarios like fluid flow queuing analysis; analysis of the queuing behavior of statistical multiplexing of several independent, identically distributed video sources. This model also found a natural application in analyzing statistical multiplexing of voice sources. The analysis using this model is extremely fast and can be used to analyze multiple source situations with no extra effort and only a slight increase in running time.

• **Model 4:** MPEG coded video traffic model can very well capture the statistical properties of MPEG video data. The theoretical results show that this is the most practical one in terms of accuracy and complexity. For example, simulations based in the real MPEG compressed movie sequence StarWars have demonstrated that this model can capture the autocorrelation function and the marginal CDF very closely.

These models are not restricted to only above mentioned scenarios but can also be adapted to model other video coding scenarios.

The NetSim Video traffic generator can be used to generate statistically validated video traffic. Further, developers can make modifications to the traffic generator code, using NetSim’s primitives, to generate custom video traffic.

**References**


**Appendix**

• Simulations are run for the all four models and frame sizes for all models are saved. Using python’s matplotlib module we plot the frame size vs. frame number plot as shown in below source code.

```python
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
import numpy as np
import sys

frameSizeRepeated=[];
frameSize=[];
```
```python
frameNumber=[];
i=0;
j=1;
fp=open("Output\model1.txt", "r");
dataOriginal=fp.readlines();
fp.close();
for data in dataOriginal[1:len(dataOriginal)]:
    data1=data.rsplit('t');
    if(data1[16]!="
    frameSizeRepeated.append(float(data2[0]));
else:
    data2=data1[17].rsplit('n');
    frameSizeRepeated.append(float(data2[0]));

while(i<len(frameSizeRepeated)):
    frameSize.append(frameSizeRepeated[i]);
    frameNumber.append(j);
    j=j+1;
i=i+14;
plt.plot(frameNumber, frameSize);
plt.xlabel("Frame Number");
plt.ylabel("Frame Size (Bytes)"");
plt.title("Model 1: Plot of Frame Size(Bytes) vs. Frame Number");
plt.show();
```

- Table for the values of Hurst parameter, γ and η as given in the reference paper is as follows. And the corresponding values for the StarWars used for the above plots are given in first row.

### Estimated Parameters for the Simple IPB Composite Model Using Least Square Fit

<table>
<thead>
<tr>
<th></th>
<th>I Frame</th>
<th></th>
<th></th>
<th></th>
<th>B Frame</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th>P Frame</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>γ</td>
<td>η</td>
<td>Hurst</td>
<td>γ</td>
<td>η</td>
<td>Hurst</td>
<td>γ</td>
<td>η</td>
<td>Hurst</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>4.0605</td>
<td>10.4233</td>
<td>0.7669</td>
<td>1.6605</td>
<td>12.0277</td>
<td>0.8298</td>
<td>1.6431</td>
<td>14.0724</td>
<td>0.848</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4.9569</td>
<td>6.1068</td>
<td>0.7158</td>
<td>1.0084</td>
<td>6.1457</td>
<td>0.78075</td>
<td>1.9798</td>
<td>43.88</td>
<td>0.86845</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4.43</td>
<td>6.0825</td>
<td>0.6961</td>
<td>1.7008</td>
<td>5.7095</td>
<td>0.73645</td>
<td>1.6014</td>
<td>8.2953</td>
<td>0.8219</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4.7191</td>
<td>8.9661</td>
<td>0.57925</td>
<td>2.3581</td>
<td>12.7828</td>
<td>0.67295</td>
<td>3.4765</td>
<td>23.4751</td>
<td>0.7349</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>3.6932</td>
<td>9.7717</td>
<td>0.7711</td>
<td>2.4559</td>
<td>11.3658</td>
<td>0.81825</td>
<td>3.3685</td>
<td>17.5669</td>
<td>0.8219</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6.1765</td>
<td>10.6556</td>
<td>0.7112</td>
<td>1.3718</td>
<td>6.5978</td>
<td>0.81345</td>
<td>1.7872</td>
<td>17.2641</td>
<td>0.8544</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>2.9395</td>
<td>5.6374</td>
<td>0.5908</td>
<td>2.1856</td>
<td>6.7743</td>
<td>0.66345</td>
<td>2.3760</td>
<td>19.2525</td>
<td>0.69355</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>2.6122</td>
<td>5.7122</td>
<td>0.61505</td>
<td>1.3527</td>
<td>8.8534</td>
<td>0.67905</td>
<td>1.1375</td>
<td>18.8076</td>
<td>0.74795</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>2.9316</td>
<td>7.2989</td>
<td>0.66215</td>
<td>2.3029</td>
<td>11.6746</td>
<td>0.74335</td>
<td>1.666</td>
<td>12.7058</td>
<td>0.8145</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>2.6119</td>
<td>5.3078</td>
<td>0.64165</td>
<td>1.3900</td>
<td>7.7165</td>
<td>0.7335</td>
<td>2.7786</td>
<td>20.0555</td>
<td>0.81735</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

©2012. TETCOS. This document is protected by copyright, all rights reserved. Contact: sales@tetcos.com