Video and Audio Trace Files of Pre-encoded Video Content for Network Performance Measurements

Frank H. P. Fitzek Michele Zorzi Patrick Seeling Martin Reisslein*

DipInge, Università di Ferrara
via Saragat 1, 44100 Ferrara
Italy
[ffitzek|zorzi]@ing.unife.it

Department of Electrical Engineering
Arizona State University
USA
[patrick.seeling|reisslein]@asu.edu

Video services are expected to account for a large portion of the traffic in future wireless networks. Therefore realistic traffic sources are needed to investigate the network performance of future communication protocols. In our previous work we focused on video services for 3G networks. We provided a publicly available library of frame size traces of long MPEG-4 and H.263 encoded videos in the QCIF format resulting in low bandwidth video streams. These traces can be used for the simulation of 3G networks. Some future communication systems, such as the WLAN systems, offer high data rates and therefore high quality video can be transmitted over such higher speed networks. In this paper we present an addition to our existing trace library. For this addition we collected over 100 pre-encoded video sequences from the WEB, generated the trace files, and conducted a thorough statistical evaluation. Because the pre-encoded video sequences are encoded by different users they differ in the video settings in terms of codec, quality, format, and length. The advantage of user diversity for encoding is that this reflects very well the traffic situation in upcoming WLANs. Thus, the new traces are very suitable for the network performance evaluation of future WLANs.

Keywords: video and audio traces, multimedia, network performance

*Corresponding Author: M. Reisslein, Dept. of Electrical Engineering, Arizona State University, Goldwater Center, MC 5706, Tempe, AZ 85287-5706, phone: 480-965-8593, fax: 480-965-8325, email: reisslein@asu.edu, web: http://www.eas.asu.edu/˜mre

1 Introduction

Mobile communication networks of the second generation such as GSM were optimized for voice services. Future networks will also support enhanced services such as video communication or streaming video. Currently in Europe network providers start to provide video services on mobile phones. Due to the small end-system and the low bandwidth the video quality is typically low. For more sophisticated video services with TV quality higher bandwidth and more enhanced end-systems are needed. Both of these requirements are met in WLAN networks. The data rates go up to 54 Mbit/s for IEEE802.11a and a large variety of end-systems is available. Moreover a wide range of video applications based on IP is available for free on the Internet.

For the performance evaluation of future communication protocols realistic traffic sources are needed for simulations. In our previous works for H.261, H.263, H263+ (all presented in [5]), H.26L [6] and MPEG4 [5, 16] we have demonstrated that for video traffic the usage of traces is a good choice. After having investigated over 50 video sequences (covering sport events, movies, cartoons, surveillance, and news) at different quality levels, we concluded that the video traffic characteristic depends on the video content itself and the chosen encoder settings (frame types used, quality, variable or constant bit rate). Furthermore, we recognized that each video sequence differs from others, which makes modeling of these types of traffic sources very difficult. Several researchers took our traces and tried to build traffic models from the traces, e.g., [3, 13]. We encoded each video sequence with different settings such as the quality or the resulting bit rate to offer other researchers a large library satisfying their needs.

For all our previous measurements we played a video sequence on a VCR and grabbed each single frame with a video card. Interested readers regarding the grabbing process are referred to [5]. We stored the complete *original* video sequence on disk. Afterwards we encoded the *original* sequence with different encoder settings using different video codecs. E.g., for our H.263 measurements presented in [5], we encoded with 16k, 64k, 256k (all constant bit rate), and variable bit rate. The *encoded* bit stream was parsed bit-wise to retrieve the video frame with its play-out time, its frame size, and its frame type to obtain the video trace file. Each video codec had his own parser following the appropriate standard. The video trace file was used for the statistical analysis of the encoded video data. Afterwards we decoded the *encoded* bit stream and obtained the *decoded* bit stream. By comparing the *original* and the *decoded* bit stream, we were able to calculate the peak signal to noise ratio (PSNR). This procedure is based on a pixel–wise comparison and is given in more detail in [19]. With our VideoMeter tool [19] we were able to examine the *original* and the *decoded* video sequence simultaneously displaying the pixel differences and the actual PSNR values. In Figure 1 the output of the VideoMeter Tool is given: On top of the window the video sequences are played out. In the middle the pixel differences are given. Black pixels refer to no changes, while lighter pixels refer to changes. On the bottom the PSNR values are given versus the last twenty samples.

The disadvantage of the previous approach was that this type of investigation is very time consuming, which is due to two facts: Firstly, the entire grabbing and encoding

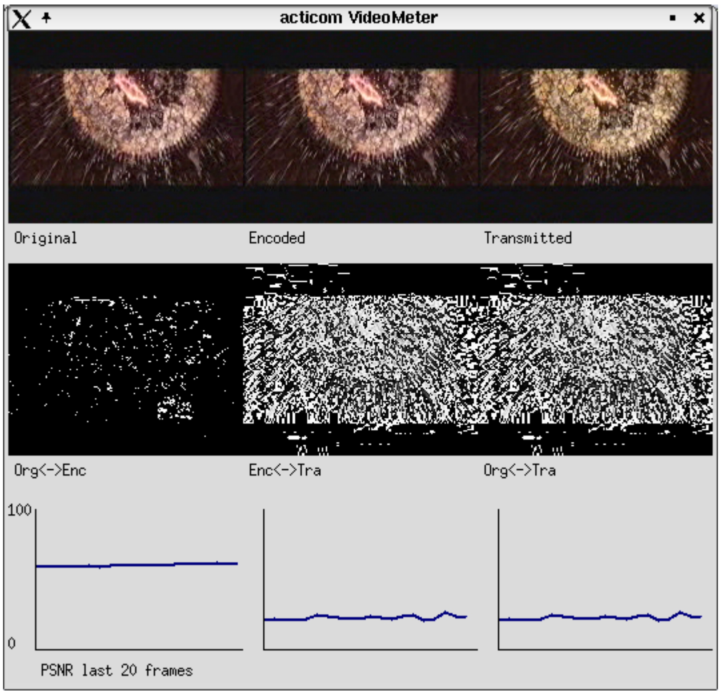

Figure 1: VideoMeter tool output.

process takes a lot of time and secondly due to the diversity in the encoder settings and the encoder itself (H.26x or MPEGx), the encoding process had to be repeated several times. Furthermore we face the problem that numerous video encoders are emerging. As an example current video players support about 100 different video codecs and their derivates. The most important encoders are `DivX;-)` (including DIV3, DIV4, DIV5, DIV6, MP43, etc), `Windows Media Video 7/8/9`, and the `RealPlayer` (including RV 20/30/40). Following our former approach this would require a new parser for each of them.

Even more time is needed if the video format is not limited to the QCIF (144x176 pixels) or CIF (288x352 pixels) format as it was used in our former work. The QCIF and CIF format fits well for the application in UMTS networks, where the wireless bandwidth is limited to an overall data rate of 2Mbit/s. WLAN networks can offer a lot of more bandwidth (up to 54 Mbit/s) and therefore may support a much higher video quality in terms of video format, frame rate, and quantization than cellular competitors. For the protocol design of WLAN networks video trace files of currently used codecs with higher quality are needed. To offer a large library even for networks with high bandwidth a new approach is needed.

2 Trace Generation For Pre-Encoded Video

We developed an approach to use pre-encoded video content, which is shared on the Internet between users, for the video trace generation. The advantage of this approach is that the entire grabbing and encoding process (including the choice of parameter settings) is already done by different users, who seemed to be satisfied by the quality of the video content after encoding. This type of video content is shared among users in the fixed wired Internet, but it appears that this content is an appropriate content even for streaming video in WLAN networks. The reason for this lies in the fact that the video content was encoded for transmission (full download) over MoDem like links (56k analog MoDem – 1M DSL) in a timely fashion.

For our measurements we collected over 100 pre-encoded sequences on the web [1]. We focused on different actual movies and TV series. A subset of all investigated sequences is given in Tables 1 and 2. The video sequences given in Table 1 are used also for the statistical evaluation, while sequences in Table 2 are listed because of specific characteristics found. The tables give the sequence name and video and audio information. The video information includes the codec type, the format, color-depth, frame rate, and data rate. We found a large variety of video codecs such as DX50, DIV4, DIV3, XVID, RV20, RV30, DIVX, and MPEG1. The video format ranges from from very small (160x120) to large (640x352). The frame rate ranges from 23.98 to 29.97 frames/sec.

Table 1: Investigated video streams: Movies.

sequence	video					audio
	codec	format	colordpth	frame rate	data rate	rate
		[pixel]	[bpp]	[1/s]	[kbit/s]	[kbit/s]
Bully1	DX50	576x432	24	25.00	1263.8	128.0
Bully3	DX50	512x384	24	25.00	988.6	128.0
Hackers	DIV4	720x576	24	23.98	794.8	96.0
LordOfTheRingsII-CD1	XVID	640x272	24	23.98	966.0	80.0
LordOfTheRingsII-CD2	XVID	640x272	24	23.98	965.2	80.0
Oceans11	DIV3	544x224	24	23.98	707.7	128.0
RobinHoodDisney	DIV3	512x384	24	23.98	1028.9	96.0
ServingSara	XVID	560x304	24	23.98	831.2	128.0
StealingHarvard	XVID	640x352	24	23.98	989.1	128.0
Final Fantasy	DIV3	576x320	24	23.98	823.9	128.0
TombRaider	DIV3	576x240	24	23.98	820.3	128.0
Roughnecks	DIV3	352x272	24	29.97	849.1	128.0
KissoftheDragon	DIV3	640x272	24	23.98	846.6	128.0

[1] To avoid any conflict with copyright we do not make the video sequences publicly available on our web page. Only the frame size traces and statistics are made available for networking researchers.

Table 2: Investigated video streams: TV series.

sequence	video					audio
	codec	format	colordpth	frame rate	data rate	rate
		[pixel]	[bpp]	[1/s]	[kbit/s]	[kbit/s]
Friends4x03	DIV3	512x384	24	25.00	1015.1	128.0
Friends4x04	DIV3	640x480	24	25.00	747.4	64.1
Friends9x13	DIV3	320x240	24	29.97	498.2	128.0
Friends9x14	DIVX	352x240	24	29.97	589.7	56.0
Dilbert1x06	MPEG1	160x120	-	29.97	192.0	64.0
Dilbert2x03	DIV3	220x150	24	29.99	129.4	32.0
Dilbert2x04	RV30	220x148	-	30.00	132.0	32.0
Dilbert2x05	RV20	320x240	-	19.00	179.0	44.1

These sequences were fed into the mplayer tool [10] version 0.90 by Árpád Gereöffy. The tool is based on the libmpg3 library and an advancement of the mpg12play and avip tools. Major modifications to the source codes were made such that the mplayer tool played the video sequence and simultaneously printed each frame with the frame number, the play-out time, the video frame size, the audio frame size, and a cumulative bit size into our trace files. An excerpt of a trace file is given in Table 3. By means of this approach we avoid having to write a parser for each video codec.

The trace files were used for the statistical analysis of the video data. Both, the trace files and the statistical analysis of the sequences given in the tables are publically available at [1]. We measured that the video file size is always slightly larger than the sum of the frame sizes produced by the video and audio encoders. To explain this fact, we have to state first that all video sequences are mostly distributed in the AVI format. Simply speaking the AVI format is a container. Due to the container information the file size is larger than the video and audio format. We do not include this overhead into our trace files. In case of multimedia streaming the video and audio information is packetized into RTP frames. The RTP header contains all important information for the playout process at the receiver. Therefore we assume that the additional container information is not needed and therefore not included into the trace file. In Section 5 we give a short introduction how to integrate RTP streams into simulations.

But our new approach has also some significant shortcomings in contrast to our former approach. The first drawback is that the PSNR quality values cannot be generated with the new approach as the original (unencoded) video content is not available. Consequently it is not possible to assess the video quality from these new traces. The second drawback is that the encoded video streams differ in terms of video format, resulting bit rate for audio and video, frame rate, and video quantization as they come from different users, see Tables 1 and 2. We assume that all sequences come from different users, because they were collected from different web sites. All sequences differ at least in one column of the table. On one hand we want realistic traces and different settings to reflect the real network traffic, but it might be more difficult to introduce them into

Table 3: Excerpt of the video trace file.

	2.043710	338	640	38870
51	2.085418	550	640	40060
52	2.127127	896	640	41596
53	2.168835	1342	640	43578
54	2.210544	709	640	44927
55	2.252252	817	640	46384
56	2.293961	807	640	47831
57	2.335669	786	640	49257
58	2.377378	728	640	50625
59	2.419086	807	640	52072
60	2.460794	652	640	53364
50	2.043710	338	640	38870
51	2.085418	550	640	40060
52	2.127127	896	640	41596
53	2.168835	1342	640	43578
54	2.210544	709	640	44927
55	2.252252	817	640	46384
56	2.293961	807	640	47831
57	2.335669	786	640	49257
58	2.377378	728	640	50625
59	2.419086	807	640	52072
60	2.460794	652	640	53364

simulations. Therefore we present also video sequences which have similar video settings such as the *Friends* episodes. Nevertheless, this issue has to be kept in mind by the researcher applying our traces.

A further problem is that the collected video sources are not encoded for real-time transmission. In our former work we used group of pictures (GoP) with length 12 (corresponding to about 480 ms for a full GoP), i.e., each 12th frame provided a full update of the video information. This is very important in presence of high bit error rates. The question arises how robust the investigated pre-encoded video streams are. We leave the answer to this question for further studies and note that the presented streams are well suited for video streaming over reliable links, but the application for real-time communication over error prone links is not clear yet.

3 Statistical Analysis of Video Traces

In this section we give a thorough statistical analysis of the frame size traces. We illustrate the salient results with two video sequences, namely *Serving Sara* and *Stealing-Harvard*. For the statistical evaluation of the traces we introduce the following notation. Let N denote the number of considered frames of a given video sequence. In case of the *Serving Sara* sequence this would be $N = 143837$. In Figures 2 and 3 the frame sizes versus time are given for the two video sequences.

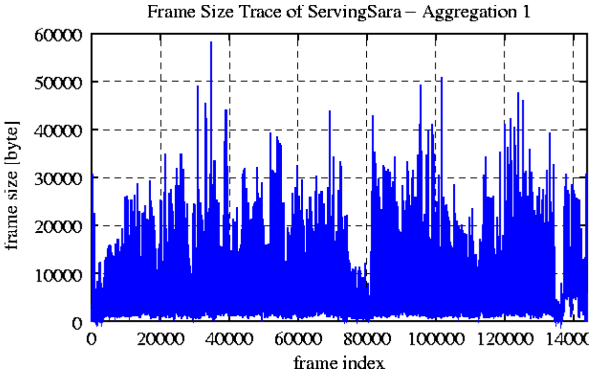

Figure 2: Frame sizes versus time for *Serving Sara.*

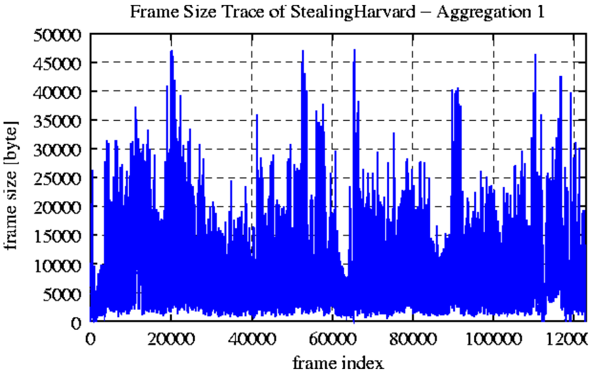

Figure 3: Frame sizes versus time for *Stealing Harvard.*

The individual frame sizes are denoted by $X_1, \ldots, X_N$. The mean frame size $\overline{X}$ is estimated as

$$\overline{X} = \frac{1}{N} \cdot \sum_{i=1}^{N} X_i. \tag{1}$$

The aggregated frame size is denoted by $\overline{X_a}(j)$ and is estimated as

$$\overline{X_a}(j) = \frac{1}{a} \cdot \sum_{i=(j-1)a+1}^{ja} X_i. \tag{2}$$

In Figure 4 and 5 the aggregated frame sizes versus time are given for the two video sequences for the aggregation level $a = 800$. The characteristics of the video sequences are much better illustrated than in Figure 2 and 3. From the aggragation plot we see that the video sequences are clearly variable bit rate (VBR) encoded.

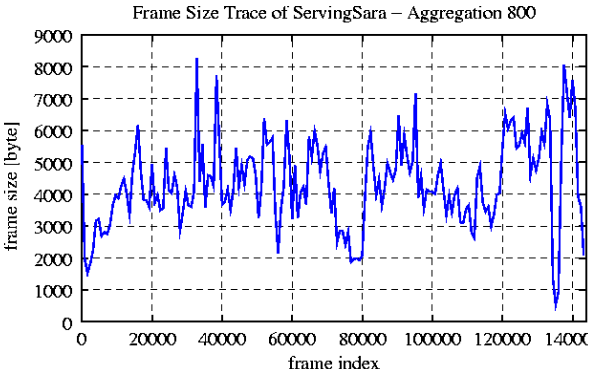

Figure 4: Aggregated frame sizes (aggregation level 800) versus time for *Serving Sara.*

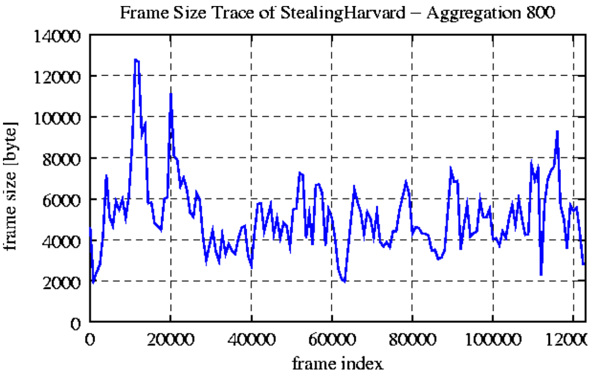

Figure 5: Aggregated frame sizes (aggregation level 800) versus time for *Stealing Harvard.*

The variance S_X^2 is estimated as

$$S_X^2 = \frac{1}{N-1} \cdot \sum_{i=1}^{N} \left(X_i - \overline{X}\right)^2 \tag{3}$$

$$= \frac{1}{N-1} \left[\sum_{i=1}^{N} X_i^2 - \frac{1}{N} \cdot \left(\sum_{i=1}^{N} X_i \right)^2 \right]. \tag{4}$$

The Coefficient of Variation CoV is estimated as

$$CoV = \frac{S_X}{\overline{X}}. \tag{5}$$

In Table 4 we give an overview of the frame statistic for several video sequences. The table presents the mean frame size, the Coefficient of Variation, and the peak to mean ratio of the frame size. Furthermore the mean and peak bit rates are given. Note, the data rates given in Table 1 are based on the output of the mplayer tool, while the data rates given below are an output of our evaluation tool. We observe that the streams are highly variable with peak to mean ratios of the frame sizes in the range from approximately 15 to about 25 for most of the video streams and three extremely variable streams with peak to mean ratios of up to 44. In our earlier trace studies the

Table 4: Overview of frame statistics of traces.

	frame sizes			bit rate	
	mean	CoV	peak/mean	mean	peak
sequence	$\overline{X}$	$S_X/\overline{X}$	$X_{\max}/\overline{X}$	$\overline{X}/t$[Mbit/s]	$X_{\max}/t$[Mbit/s]
Bully1	6319	1.27	35.68	1.26	45.09
Bully3	4942	1.24	31.01	0.98	30.66
Hackers	4150	0.62	43.78	0.79	34.85
LordOfTheRingsII-CD1	5036	0.59	15.69	0.96	15.16
LordOfTheRingsII-CD2	5032	0.60	16.77	0.96	16.19
Oceans11	3694	0.75	20.49	0.71	14.52
RobinHoodDisney	5364	0.74	26.06	1.02	26.82
ServingSara	4333	0.66	23.40	0.83	19.45
StealingHarvard	5156	0.59	15.91	0.98	15.74
FinalFantasy	4295	0.74	20.17	0.83	16.62
TombRaider	4289	0.76	22.47	0.82	18.49
Roughnecks	3541	0.57	14.10	0.84	11.97
KissoftheDragon	4413	0.61	16.63	0.84	14.08

peak to mean ratios of the frame sizes were typically in the range from 3 – 5 for videos encoded with rate control (in a closed loop) and in the range from 7 – 19 for the videos encoded without rate control (in an open loop). Clearly these new video streams are significantly more variable, posing particular challenges for network transport. We note also that the peak rates fit well within the bit rates provided by the emerging WLAN standards.

Besides the mean and variance of the frame sizes, the frame size distribution is very important for the network design. Furthermore, the distribution of the frame sizes is needed in order to make any statistical modeling of the traffic possible. Frame size histograms or probability distributions allow us to make observations concerning the variability of the encoded data and the necessary requirements for the purpose of real–time transport of the data over a combination of wired and wireless networks. In Figures 6 and 7 we illustrate the inverse cumulative frame size distributions G as a function of the frame size for two sequences. For the probability density function p as well as the probability distribution function F, we refer to [1].

Many researcher simply pick a frame size distribution and randomly generate frames as a traffic model. But such a model would not represent the characteristic of the video sequences as it does not include the dependencies between frames. Therefore one needs to take the autocorrelation into consideration. The autocorrelation [4] function can be used for the detection of non–randomness in data or identification of an appropriate time series model if the data are not random. One basic assumption is that the observations are equi-spaced. The autocorrelation is expressed as correlation coefficient, referred to as autocorrelation coefficient (acc). Instead of the correlation between two different

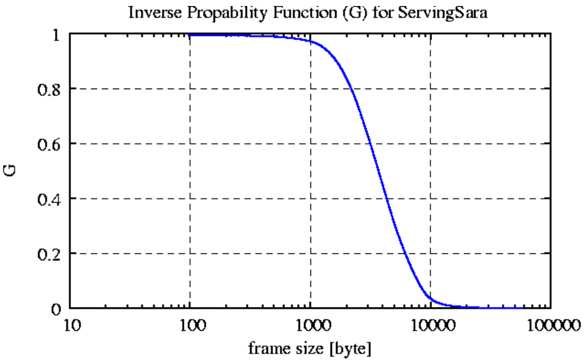

Figure 6: Inverse cumulative frame size distribution for *Serving Sara*

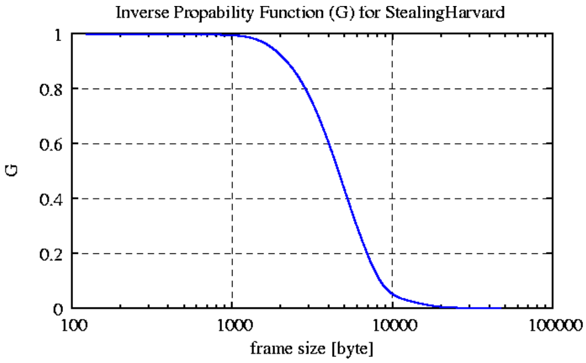

Figure 7: Inverse cumulative frame size distribution for *Stealing Harvard*

variables, the correlation is between two values of the same process (stream) at times X_t and X_{t+k}. When the autocorrelation is used to detect non-randomness, it is usually only the first (lag $k = 1$) autocorrelation that is of interest. When the autocorrelation is used to identify an appropriate time series model, the autocorrelations are usually plotted for a range of lags k. With our notation the acc can be estimated by

$$\rho_X(k) = \frac{1}{N-k} \cdot \sum_{i=1}^{N-k} \frac{\left(X_i - \overline{X}\right) \cdot \left(X_{i+k} - \overline{X}\right)}{S_X^2}, \qquad k = 0, 1, \ldots, N. \tag{6}$$

In Figures 8 and **??** we plot the frame size autocorrelation coefficients as a function of the lag k. We observe that the autocorrelation very rapidly drops from 1 to values between 0.4 and 0.6 and then drops off only very slowly. This indicates that there are significant correlations in the sizes between relatively distant frames. Which in turn results in traffic bursts that tend to persist for relatively long periods of time, making it very challenging to accommodate this traffic in networks.

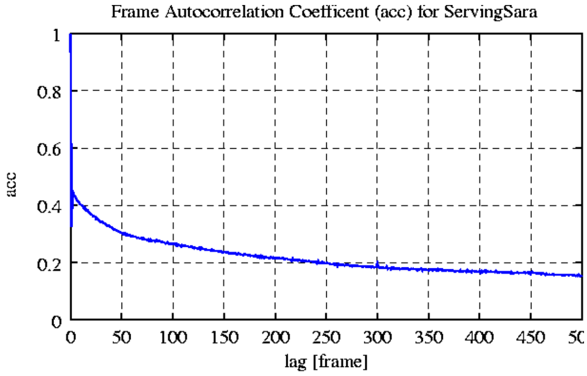

Figure 8: Autocorrelation coefficients for *Serving Sara*

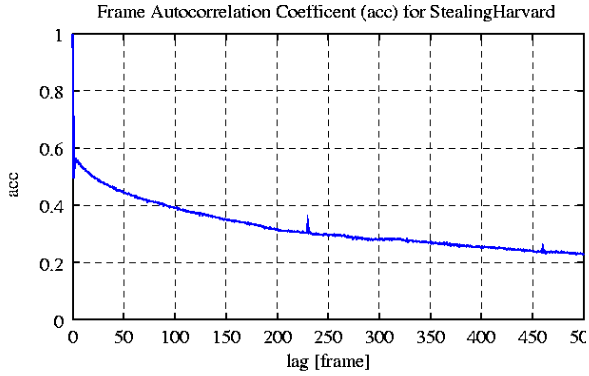

Figure 9: Autocorrelation coefficients for *Stealing Harvard*

The Hurst parameter, or self–similarity parameter, H, is a key measure of self-similarity [12, 14]. H is a measure of the persistence of a statistical phenomenon and is a measure of the length of the long range dependence of a stochastic process. A Hurst parameter of $H = 0.5$ indicates absence of self-similarity whereas $H = 1$ indicates the degree of persistence or a present long–range dependence. The H parameter can be estimated from a graphical interpolation of the so–called R/S plot. The R/S plot gives the graphical interpretation of the rescaled adjusted range statistic by utilizing the following method [2, 15]. The length of the complete series N is subdivided into blocks with a length of k, for which the partial sums $Y(k)$ are calculated as in Equation (7). Then the variance of all these aggregations is calculated. The resulting R/S value is evaluated as shown in Equation 9 for a single block.

$$Y(k) = \sum_{i=1}^{k} X_i \tag{7}$$

$$S_X^2(k) = \frac{1}{k} \cdot \sum_{i=1}^{k} \left[X_i^2 - \left(\frac{1}{k}\right)^2 \cdot Y(k)^2 \right] \tag{8}$$

$$\frac{R}{S}(N) = \frac{1}{S_X(k)} \left[\max_{0 \le t \le k} \left(Y(t) - \frac{t}{k} \cdot Y(k) \right) - \min_{0 \le t \le k} \left(Y(t) - \frac{t}{k} \cdot Y(k) \right) \right] \tag{9}$$

If plotted on an log/log scale for R/S versus differently sized blocks, the result will be several different points. This plot is also called the *pox plot* for the R/S statistic and is illustrated in Figures 10 and 11. The Hurst parameter H is then estimated by fitting a line to the points of the plot, usually by using a least square fit, neglecting the residual values at the lower and upper borders. The larger the resulting Hurst parameter, the higher the degree of long range dependency of the time series. We observe from the Figures 10 and 11 that for the single–frame aggregation ($a = 1$) the Hurst parameters stay well above 0.5, reflecting the presence of long–term dependence. We applied the 4σ–test [18] to eliminate all outlying residuals for a better estimation of the Hurst parameter.

In Table 5 the Hurst parameters of the frame size traces from the pox plots of the R/S statistics are given for each video sequence for different aggregation levels a. All investigated video sequences indicate a high degree of long-range-dependence. For the aggregation level $a = 1$, most sequences have Hurst parameters larger than 0.8. Only two sequences have remarkable smaller values. The *Hackers* sequences is the only sequence that has the letterboxes (black bars on top and at the bottom for the 16:9 adjustment). Furthermore this sequence has a large peak to mean ratio as given in Table 4. It appears that these two attributes have a large impact on the R/S calculation. The *Roughnecks* sequences is dominated by very dark scenes and very quick movements, which may also result in a small Hurst parameter. For all other videos we observe H values above 0.7 even for the large aggregation levels giving a very strong indication of long range dependence.

The variance time plot is applied to a time series to show the development of the variance as in Equation (4) over different aggregation levels. This provides another test

Table 5: Hurst parameters estimated from the pox diagram of R/S as a function of the aggregation level.

	aggregation level a										
sequence	1	12	50	100	200	300	400	500	600	700	800
Bully1	0.884	0.861	0.838	0.842	0.821	0.796	0.784	0.706	0.735	0.688	0.655
Bully3	0.870	0.861	0.856	0.889	0.908	0.904	0.940	0.936	0.930	0.907	1.030
Hackers	0.503	0.517	0.513	0.531	0.520	0.503	0.486	0.549	0.478	0.527	0.619
LotRingII-CD1	0.960	0.879	0.848	0.847	0.866	0.835	0.809	0.794	0.745	0.772	0.750
LotRingII-CD2	0.976	0.876	0.894	0.926	0.934	0.886	0.864	0.833	0.826	0.844	0.816
Oceans11	0.917	0.844	0.818	0.809	0.787	0.764	0.756	0.745	0.741	0.720	0.736
RobinHoodDisney	0.815	0.826	0.806	0.798	0.810	0.826	0.784	0.851	0.822	0.863	0.808
ServingSara	0.936	0.853	0.849	0.839	0.821	0.827	0.790	0.785	0.750	0.726	0.740
StealingHarvard	0.966	0.894	0.853	0.813	0.785	0.744	0.700	0.704	0.705	0.695	0.675
Final Fantasy	0.916	0.833	0.779	0.769	0.752	0.759	0.733	0.742	0.782	0.734	0.726
TombRaider	0.908	0.849	0.852	0.850	0.843	0.848	0.800	0.825	0.815	0.797	0.731
Roughnecks	0.647	0.650	0.650	0.631	0.633	0.625	0.690	0.683	0.703	0.718	0.771
KissoftheDragon	0.902	0.852	0.808	0.809	0.802	0.772	0.780	0.780	0.736	0.764	0.774

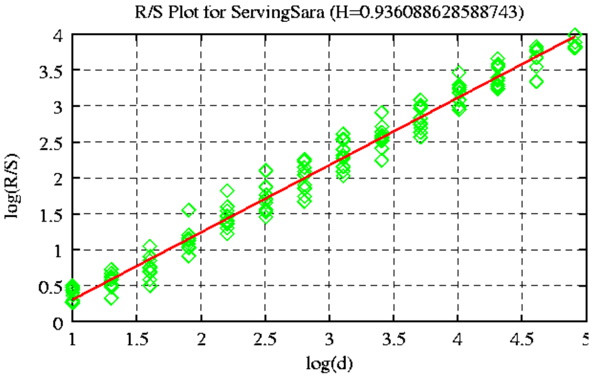

Figure 10: Single Frame R/S plot and H parameter for *Serving Sara*

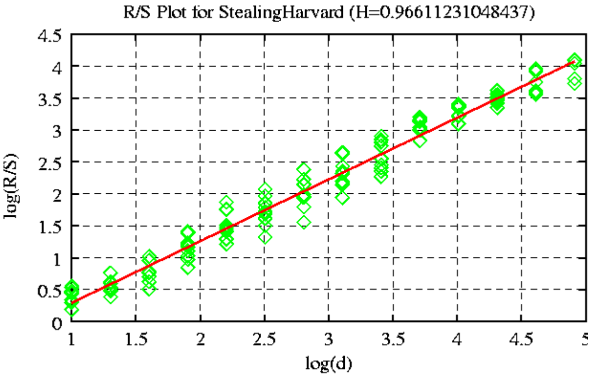

Figure 11: Single Frame R/S plot and H parameter for *Stealing Harvard*

for long–range dependency [17, 5, 14, 2]. It is furthermore used to derive an estimation of the Hurst parameter. In order to obtain the plot, the normalized variance as given in Equation (10) of the trace is plotted as a function of different aggregation levels k of the single frame sizes in a log–log plot. For each aggregation level k the N frames are grouped into blocks and the variance is calculated as shown before in Equations (7) and (8).

$$S_{\text{norm}} = \frac{S_X^2(k)}{S_X^2} \tag{10}$$

If no long range dependency is present, the slope of the function is -1. For slopes larger than -1, a dependency is present. For simple reference we plot a reference line with a slope of -1 in the figures. We did not apply any regression–fits up to now but plan to do so in the future. Our plots in Figures 12 and 13 indicate a certain degree of long term dependency since the estimated slope is less than -1.

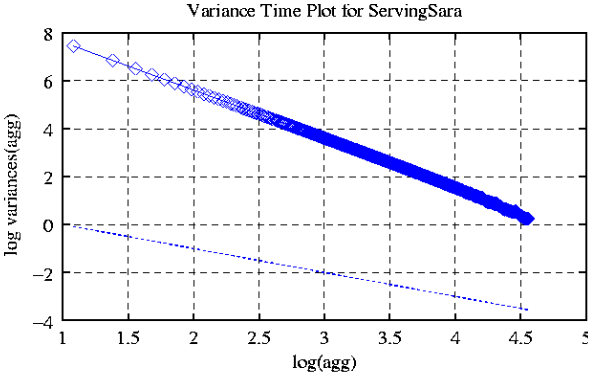

Figure 12: Variance time plot for *Serving Sara*

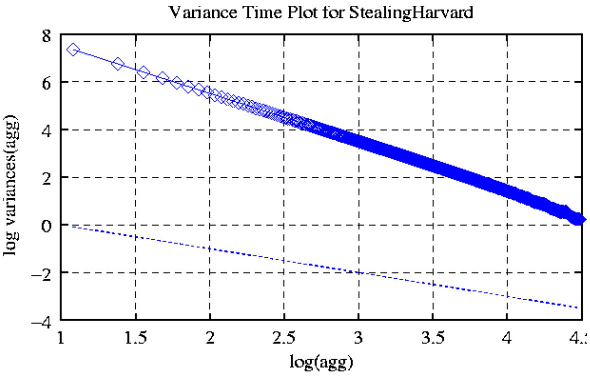

Figure 13: Variance time plot for *Stealing Harvard*

A periodogram is a graphical data analysis technique for examining frequency–domain models of an equi–spaced time series. The periodogram is the Fourier transform of the auto-covariance function. This calculation is currently employed in measuring the spectral density, following the idea that this spectrum is actually the variance at a given frequency. Therefore additional information about the magnitude of the variance of a given time series can be obtained by identifying the frequency component. This is done by correlating the series against the sine/cosine functions, leading to the Fourier frequencies [20]. For the calculation of the periodogram plot, the frame sizes x_i of N frames are aggregated into equidistant blocks k. For each block, the moving averages and their according logarithms are calculated as in Equations (11) and (12) with $n = 1, \ldots, N/k$.

$$Y_n^{(k)} = \frac{1}{k} \cdot \sum_{i=1}^{\frac{N}{k}} x_i \tag{11}$$

$$Z_n^{(k)} = \log_{10} Y_n^{(k)} \tag{12}$$

In order to determine the frequency part of the periodogram, we calculate λ_k as in Equation (13). The periodogram itself is then derived as given in Equation 14. For each different aggregation level, we plot the resulting $I(\lambda_k)$ and λ_k in a *log*/*log*–plot. The resulting plots are shown in Figures 14 and 15.

$$\begin{aligned} \lambda_k &= \frac{2\pi i}{\frac{N}{k}},\ i = 1, \ldots, \frac{M-1}{2} && (13) \\ I(\lambda_k) &= \frac{1}{2\pi \frac{N}{k}} \cdot \left| \sum_{l=0}^{\frac{N}{k}-1} Z_l^{(a)} \cdot e^{-jl\lambda_k} \right|^2 && (14) \end{aligned}$$

The Hurst parameter is estimated as $H = (1-\beta_1)/2$, using a least squares regression on the samples. The Equations (16) and (17) are applied to determine the slope of the fitted line $y = \beta_0 + \beta_1 x$ and H.

$$\begin{aligned} K &= 0.7 \cdot \frac{\frac{N}{k} - 2}{2} && (15) \\ \beta_1 &= \frac{K \cdot \sum_{i=1}^{K} x_i y_i - \left(\sum_{i=1}^{K} x_i\right) \cdot \left(\sum_{i=1}^{K} y_i\right)}{K \cdot \left(\sum_{i=1}^{K} x_i^2\right) - \left(\sum_{i=1}^{K} y_i\right)^2}, && (16) \\ \beta_0 &= \frac{\sum_{i=1}^{K} y_i - \beta_1 \cdot \sum_{i=1}^{K} x_i}{K} && (17) \end{aligned}$$

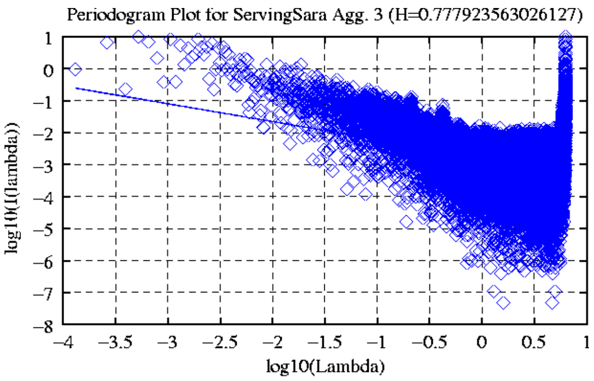

Figure 14: Periodogram plot for *Serving Sara*

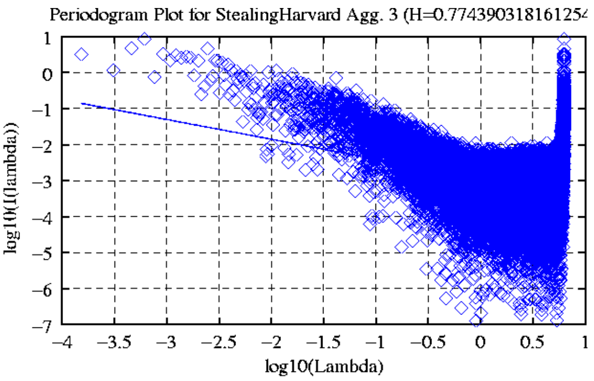

Figure 15: Periodogram plot for *Stealing Harvard*

4 Comparison with Other Traces

It is very hard to compare our former traces with the traces generated with the presented approach. This is due to the different encoder settings and video formats. Nevertheless, in Figure 16 a comparison of the frame sizes for the *RobinHoodDisney* video sequence is given using former results for MPEG-4 measurements and our new approach. For a better illustration we used an aggregation level of 800. The MPEG-4 measurements were done for three different quality levels (see also [5]) and the QCIF (144x176) video format using the MoMoSyS software [11]. Our new approach uses the DIV3 codec and the video format is 512x384. Clearly the data rate is smaller for the medium and low qualities, but the high quality QCIF video and the DIV3 video have nearly the same data rates. Interesting is the dynamic behavior of the frame sizes. The dynamics of the variable bit rate traffic are nearly identical for all four curves. Especially the comb during the period from $65000 - 70000$ frames and the peak at 21500 represent this similarity. We note that this dymanic behvior is the same for the H.263 encoded video, see [1]. We emphasize that these similarities are observed even though the videos were encoded completely independently (using different encoders applied to the sequences grabbed from a VCR with our previous approach and by someone posting a DIV3 encoding on the web with our new approach).

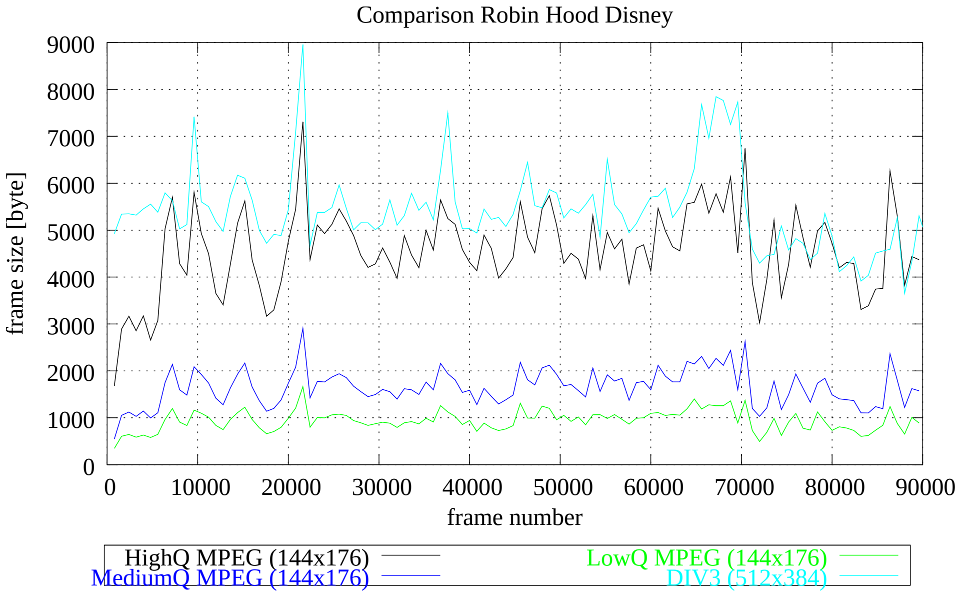

Figure 16: Comparison of former work with actual approach regarding the frame sizes.

5 Using Network Simulators with Video Traces

In [8] we give detailed instructions on how our video trace files may be used by other researchers in their own simulation environment. Examples of implementations for NS2, PTOLEMY, and OMNET++ are also illustrated in [8]. Regarding the presented work we note that besides the video trace information also audio information are available. Therefore two different streams will be used to transport the data in an IP environment.

Note, that each frame needs its own transport and network information, which can be an additional overhead (of 40 bytes for real-time transmission using RTP/UDP on top of IPv4) for each packet as given in Figure 17. This will increase the traffic significantly and has to be accounted for the implementation process.

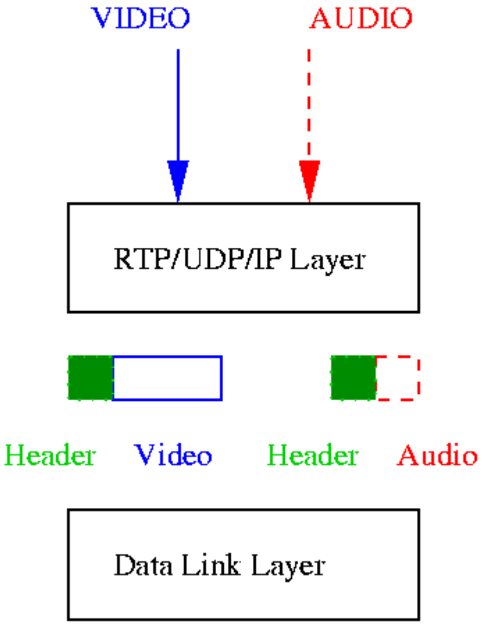

Figure 17: Using the trace file in simulation with RTP/UDP/IP environment.

6 Conclusion

We have generated and analyzed video and audio traces of video content currently being exchanged on the web. The traces reflect the wide variety of video encoders, video format and frame rates that are currently (and probably also in the near future) being transmitted over the Internet, including its wireless components, such as WLANs. Compared to our earlier traces which primarily relect the video suitable for transmission over 3G wireless networks to mobile devices with small displays, the new traces are for higher quality video (with large display formats) which is suitable for transmission over WLANs. Our statistical analysis of these new traces indicates that the WLAN suitable video is significantly more variable (bursty) than previously studied video streams; the peak to mean ratios of the frame sizes of the new traces are typically in the range from 15 to 35, whereas the range from 7 to 18 was typically observed before. We also observed that the new traces have very consistently high autocorrelations and Hurst parameters, further corroborating the burstiness of the traffic. We also observed that the audio bit rate is typically 8% to 15 % of the corresponding video bit rate. We make all our traces publicly available at [1] and provide instructions for using the traces in network evaluations.

7 Outlook

In our future work we want to investigate the robustness of the presented video sequences in the presence of transmission errors. As we stated before the investigated video sequences are not encoded for the real-time transmission over wireless links. Therefore we will investigate the robustness of the video sequences by applying elementary bit errors patterns on the video sequences and measuring the quality degradation using our VideoMeter tool [7]. We used this procedure already in our former work as presented in [9] for different video GoP structures. One interesting result would be to aquire an understanding of how to set the parameters for encoding in presence of the wireless errors and to evaluate the increase in bandwidth requirements.

References

[1] Arizona State University. Video traces for network performance evaluation. http://trace.eas.asu.edu.

[2] J. Beran. *Statistics for Long–Memory Processes.* Chapman and Hall, London, 1994.

[3] T. Borsos. A Pratical Model for VBR Video Traffic with Applications. *MMNS*, pages 85–95, 2001. Springer-Verlag Berlin Heidelberg.

[4] G.E.P Box and G. Jenkins. *Time Series Analysis: Forecasting and Control.* Holden-Day, 1976.

[5] F. Fitzek and M. Reisslein. MPEG–4 and H.263 video traces for network performance evaluation. *IEEE Network*, 15(6):40–54, November/December 2001.

[6] F.H.P. Fitzek, P. Seeling, and M. Reisslein. H.26L Pre-Standard Evaluation. Technical Report acticom-02-002, acticom – mobile networks, Germany, November 2002.

[7] F.H.P. Fitzek, P. Seeling, and M. Reisslein. VideoMeter tool for YUV bitstreams. Technical Report acticom-02-001, acticom – mobile networks, Germany, October 2002.

[8] F.H.P. Fitzek, P. Seeling, and M. Reisslein. Using Network Simulators with Video Traces. Technical report, Arizona State University, Dept. of Electrical Engineering, March 2003.

[9] F.H.P. Fitzek, P. Seeling, M. Reisslein, M. Rossi, and M. Zorzi. Investigation of the GoP Structure for H.26L Video Streams. Technical Report acticom-02-004, acticom – mobile networks, Germany, December 2002.

[10] Á. Gereöffy. mplayer tool. http://www.mplayerhq.hu, May 2003. Version 0.90.

[11] G. Heising and M. Wollborn. MPEG–4 version 2 video reference software package, ACTS AC098 mobile multimedia systems (MOMUSYS), December 1999.

[12] H.E. Hurst. Long–Term Storage Capacity of Reservoirs. *Proc. American Society of Civil Engineering*, 76(11), 1950.

[13] P. Kavallaris. Traffic Modelling for Mobile Multimedia Networks. In *Summit 2002*, 2002.

[14] W.E. Leland, M.S. Taqq, W. Willinger, and D.V. Wilson. On the self-similar nature of Ethernet traffic. In Deepinder P. Sidhu, editor, *ACM SIGCOMM*, pages 183–193, San Francisco, California, 1993.

[15] A.W. Lo. Long–term memory in stock market prices. *Economatria*, (59):1276–1313, 1991.

[16] M. Reisslein, F.H.P. Fitzek, et. al. Traffic and Quality Characterizaton of Scalable Encoded Video: A Large–Scale Trace–Based Study. Technical report, Arizona State University, Telecommunications Research Center, 2002.

[17] P. Morin. The impact of self-similarity on network performance analysis, 1995.

[18] L. Sachs. *Angewandte Statistik*. Springer, 2002.

[19] P. Seeling, F.H.P. Fitzek, and M. Reisslein. Videometer. *IEEE Network Magazine*, page 5, January 2003.

[20] R.H. Shumway and D.S. Stoffer. *Time Series Analysis and Its Applications*. Springer, New York, 2000.