A NO REFERENCE (NR) AND REDUCED REFERENCE (RR) METRIC FOR DETECTING DROPPED VIDEO FRAMES

Stephen Wolf

National Telecommunications and Information Administration (NTIA)

ABSTRACT

Digital video transmission systems consisting of a video encoder, a digital transmission method (e.g., Internet Protocol – IP), and a video decoder can produce pauses in the video presentation that result from dropped or repeated video frames. For example, a common response of a video decoder to dropped IP packets is to momentarily freeze the video by repeating the last good video frame. This paper presents a No Reference (NR) metric and a Reduced Reference (RR) metric for detecting these dropped video frames. These metrics may have application for in-service video quality monitoring.

1. INTRODUCTION

Digital video transmission systems consisting of a video encoder, a digital transmission method (e.g., Internet Protocol – IP), and a video decoder can produce pauses in the video presentation that result from dropped or repeated video frames. There are two primary reasons for this behavior. The first reason is that the video encoder may decide to reduce the video frame transmission rate in order to save bits. For example, an original video stream with a frame rate of 30 frames per second (fps) may be reduced to 15 fps by dropping every other video frame. The second reason is that the video decoder may decide to freeze the last good video frame when errors such as IP packet loss are detected. This is a simple error concealment algorithm that is used by many video decoders.

This paper presents a No Reference (NR) metric and a Reduced Reference (RR) metric for detecting dropped video frames. An NR metric only requires access to the destination video stream to make a measurement (i.e., no access to the source video stream). An RR metric requires access to both the source and destination video streams, and a method to communicate low bandwidth reference information between the source and destination ends. NR and RR metrics are both useful for in-service quality monitoring applications.

The NR and RR metrics in this paper are derived by examining the behavior of the motion energy in the video stream, which is computed from simple frame differences. Thus, these metrics should be easily implemented in real time by modern signal processing components.

2. ALGORITHM DESCRIPTION

The NR algorithm for detecting dropped video frames was developed using sequences of captured video frames (often called video clips) that ranged from 5 to 10 seconds in length. The behavior of the algorithm for shorter or longer video sequences should be analyzed before being applied. The NR algorithm will estimate the number of dropped video frames in the video clip and their temporal locations. The RR version of this algorithm is obtained by applying the NR algorithm to both the source and destination video streams and then comparing the number of dropped video frames in the source and destination video clips. The RR version of the algorithm can thus correct for mistakes that might be due to source content (e.g., still or very low motion video scenes, slow motion video where frames are repeated). However, the price for this increased robustness is the additional computational complexity of applying the algorithm to the source video, the need to temporally synchronize the source video clip with the destination video clip, and the need to communicate the reference information between the source and destination ends.

This section provides a step by step description of the NR algorithm as applied to one video clip. The RR version of the algorithm requires an optional extra step, which is also described. The NR algorithm only utilizes the luminance images of the video clip (e.g., the Y channel in an ITU-R Recommendation BT.601 sampled video stream [1]), which will be denoted in this paper as $Y(i, j, t)$, where $t = 1, 2, 3, ..., N$ (the total number of frames in the video clip) and i and j are the row and column indices of the images, respectively. The algorithm has two major components. The first component involves processing image pixels to produce a frame-by-frame motion energy time history for the video clip (Section 2.1). The second component involves examining this motion energy time history for dropped/repeated video frames (Section 2.2).

To assist the reader with proper implementation of the proposed algorithm, software source code is provided in [2].

2.1 Computing the Motion Energy Time History

The motion energy time history of a video clip requires three processing steps on the sampled video images.

Step 1) Compute the Temporal Information (*TI*) difference sequence given by

$$TI(i,j,t) = Y(i,j,t) - Y(i,j,t-1),\ (i,j) \in SROI, and\ t = 2,3,...,N.$$

Here *SROI* is the Spatial Region of Interest, which may be selected to be the central portion of the image to eliminate image border pixels that do not contain valid picture elements (e.g., some cameras may not fill the entire ITU-R Recommendation BT.601 frame, encoders may not transmit the entire frame).

Step 2) Zero image pixels in *TI* that have an amplitude less than or equal to an image motion threshold M_{image}.[1]

$$TI(i,j,t) = \begin{cases} TI(i,j,t) & if\ abs(TI(i,j,t) > M_{image} \\ 0 & otherwise \end{cases}$$

This step eliminates low level noise from being counted as image motion. M_{image} may also be adjusted higher to eliminate motion pixels that fall below the ability to perceive them.

Step 3) Square *TI* to convert from amplitude to energy and compute the mean of each video frame. Here the resulting time history of frame-by-frame values that contain the motion energy will be represented as *TI2* and computed as:

$$TI2(t) = \underset{over\ i,j}{mean}\left\{TI(i,j,t)^2\right\}.$$

The next section will address how to examine the *TI2* waveform to locate dropped/repeated video frames.

2.2 Finding Dropped/Repeated Frames

For several reasons, locating dropped/repeated video frames in the *TI2* time history is more complicated than it might first appear. Scenes can vary from being still or nearly still (e.g., a zoomed out video of someone talking where only their lips are moving) to having very large amounts of motion (e.g., pan or zoom). Dropped frames can have small amounts of residual motion due to very minor changes in the image pixel values. Video compression systems can perform partial frame updates where only a small fraction of the video frame is updated. Thus, there is an element of subjective perception involved in determining the level of motion where these minor frame updates are counted as dropped video frames and not as new video frames.

Examination of video scenes from many different compression systems has produced a method which utilizes a dynamic threshold to determine dropped video frames. This threshold is raised for scenes with more motion and lowered for scenes with less motion. When more motion is present in the video scene, the dropped frames can contain more residual motion yet still be perceived as dropped frames. In addition, brief frame drops of one frame duration (called *dips* in this section) can contain even greater amounts of residual motion than a series of dropped frames.

In light of the above discussion, the algorithm will use a fixed motion energy threshold M_{drop} to determine frame *drops* and a different fixed motion energy threshold M_{dip} to determine frame *dips*, where the dips must have an amplitude of at least A_{dip} (i.e., the *TI2* waveform must be higher on both sides of the dip by at least A_{dip}). Then, a dynamic factor *dfact* will multiply the three fixed constants (M_{drop}, M_{dip}, and A_{dip}) and these dynamically adjusted thresholds will be used to determine the actual drops and dips in the *TI2* waveform. The dynamic factor *dfact* will be derived from the average level of motion energy that is present in the video clip.

We continue with the step by step numbering used in Section 2.1 to complete the algorithm description.

Step 4) Compute the average *TI2* value (*TI2_ave*) as

$$TI2\_ave = \underset{over\ k}{mean}\{TI2\_sort(k)\},$$
$$ceil(F_{cut} * (N-1)) \le k \le floor((1-F_{cut}) * (N-1))$$

Here, *TI2_sort* is the *TI2* vector from step 3 sorted from low to high, with new index *k* rather than *t* (recall that the *TI2* vector has *N*-1 samples). F_{cut} is the fraction of scene cuts to eliminate before computing the average (e.g., $F_{cut} = 0.02$ will eliminate 2 scene cuts every 100 frames), *ceil* and *floor* are rounding functions that round up and down to the nearest integer, respectively. Scene cuts cause very high *TI2* values and unduly influence *TI2_ave*, particularly for low motion scenes. Both low and high *TI2* points are eliminated so that the average *TI2* value is not influenced when scene cuts are absent.

Step 5) Compute the dynamic factor *dfact* as

$$dfact = a + b * \log(TI2\_ave)$$
$$if\ (dfact < c)\ then\ set\ dfact = c$$

where *a*, *b*, and *c* are positive constants and log is the natural logarithm base *e*. The constant *c* is necessary to limit *dfact* to a small positive value. This equation says that the perception of frame drops and dips is linearly dependent upon the log of the average motion energy.

[1] There are a number of parameters (e.g., M_{image}) that control the behavior of the algorithm presented herein. Recommended values for these parameters are given in Table 1.

Frame drops for higher motion scenes can have a higher residual motion yet still be perceived as frame drops.

Step 6) Compute the Boolean variable *drops* (equal to 1 when a frame drop is detected, otherwise equal to 0) as

$$drops(t) = \begin{cases} 1 & if\ TI2(t) \le dfact * M_{drop} \\ 0 & otherwise \end{cases}$$

Step 7) Compute the Boolean variable *dips* (equal to 1 when a frame dip is detected, otherwise equal to 0) as

$$dips(t) = \begin{cases} 1\ if\ TI2(t) \le dfact * M_{dip}\ \&\ dips\_mag(t) \ge dfact * A_{dip} \\ 0 & otherwise \end{cases}$$

where *dips_mag* is a function that finds the magnitude of the dips and is given by

$$dips\_mag(t) = \min\{TI2(t-1) - TI2(t), TI2(t+1) - TI2(t)\}$$
$$if\ (dips\_mag(t) < 0)\ then\ set\ dips\_mag(t) = 0$$

The endpoints (i.e., t =2 and t=N) of the *dips* vector are set equal to 0 since the *dips_mag* function is undefined for these two data points.

Step 8) Some frames may be classified as both a drop and a dip. Thus, the logical OR of the *drops* and *dips* vectors from steps 6 and 7 gives 1's for those frames in the video clip that are detected as dropped/repeated frames and 0's otherwise. One can compute the Fraction of Dropped Frames (*FDF*) by summing the elements in the vector and dividing by the maximum number of samples that could be detected as drops or dips, namely

$$FDF = \text{sum}(drops\ \text{OR}\ dips)/(N-3).$$

Step 9) (Optional for RR measurement.) If one has access to the corresponding time-aligned original source video clip, one can adjust the *FDF* of the destination video clip downward to account for still frames and/or detected drops and dips in the source clip. Here

$$FDF_{RR} = \frac{FDF_{dest} - FDF_{source}}{1 - FDF_{source}}$$
$$if\ (FDF_{source} > 0.9)\ then\ FDF_{RR}\ is\ undefined$$
$$if\ (FDF_{RR} < 0)\ then\ set\ FDF_{RR} = 0$$

where FDF_{dest} and FDF_{source} are the *FDF*s of the destination and source video clips, respectively. The divisor (1 - FDF_{source}) is required since detected drops in the source video clip reduce the total number of samples that are available for estimating FDF_{RR}. If too many drops are detected in the source video clip (i.e., $FDF_{source} > 0.9$), then the divisor (1 - FDF_{source}) will approach zero. In this case, it would be prudent to treat FDF_{RR} as not having a value (i.e., undefined). Finally, FDF_{RR} needs to be limited at zero because the algorithm may detect drops and dips in the source video clip but not in the destination video clip.

Recommended values of the parameters that control the behavior of the algorithm are given in Table 1. These recommended values were obtained by examining plots of the *TI2* waveform and its associated drops and dips for different types of data sets that contained a wide variety of video encoders and transmission errors. When there was some doubt as to whether or not certain video frames should be classified as drops or dips, the video clips themselves were examined using frame-by-frame playback. While this approach is subjective, it produces a rapid convergence to parameter values that work well across many video clips, and indeed was the method used to determine the need for dynamic thresholds (step 5). A more optimal approach would be to have a set of human observers classify the drops and dips present in many video clips and then utilize a mathematical search to optimize the algorithm's parameters. However, this approach is very costly in terms of human labor.

Table 1. Recommended Values for Algorithm Parameters

Parameter	Value
M_{image}, step 2	30
F_{cut}, step 4	0.02
a, step 5	2.5
b, step 5	1.25
c, step 5	0.1
M_{drop}, step 6	0.015
M_{dip}, step 7	1.0
A_{dip}, step 7	3.0

3. EXAMPLE RESULTS

To obtain an estimate for the performance of the NR frame drop algorithm, a randomly selected set of clips from the Video Quality Experts Group (VQEG) Phase I Multi-Media (MM) VGA experiments was used [3]. Processed clips were randomly selected from thirteen data

sets of 128 clips each.[2] These data sets were chosen since they contain a wide range of modern video compression algorithms (e.g., H.264) and network impairments (e.g., dropped IP packets). Each clip processed by the automated algorithm was examined visually for correctly detected non-drops, correctly detected drops, falsely detected drops, and missed drops. Table 2 gives the results of the NR frame drop algorithm as increasing numbers of video clips are examined. The last line in the table for 80 clips represents about 5% of the total data. From this analysis, the algorithm has about a 2% false detection rate for dropped frames and about a 0.1% chance of missing a dropped frame.

Table 2. Sample Performance of NR Algorithm

Number of Clips	Correct Non-Drops	Correct Drops	False Drops	Missed Drops
20	67.0%	30.6%	2.2%	0.2%
40	64.1%	34.4%	1.4%	0.1%
60	64.8%	33.9%	1.2%	0.1%
80	67.3%	30.8%	1.8%	0.1%

Figure 1 gives an example plot of *TI2* for a 40-frame segment of a 30 fps video clip that was derived from 24 fps film. Here, every 5th video frame is a frame repeat from the previous frame (evident by the *TI2* dips present at frames 5, 10, 15, and 20). At frame 23 there is a scene change (causing a large *TI2* spike) to a nearly still scene that continues for the remainder of the segment. Detected drops are shown as red circles and detected dips are shown as green squares. The detected drops result from the scene being a still scene from frames 24 to 40. The detected dips at frames 5, 10, 15, and 20 result from the film conversion process. Visual examination of frame 22 (immediately prior to the scene cut at frame 23) reveals that this is indeed a frame repeat of prior video frame 21. Thus, frame 22 is correctly detected as a drop by the algorithm.

Figure 2 shows an example *TI2* waveform for a segment of a destination video clip with frame freezes (frame drops shown in red) resulting from dropped IP packets. This particular scene had very high motion (pan of a crowd at a football game). Error blocks were present in the video immediately before the frame freeze periods, and these error blocks show up as large spikes in the *TI2* waveform (at frames 6 and 10). The missing IP packets at frame 6 resulted in three dropped frames (i.e., frames 7, 8, and 9) while the missing IP packets at frame 10 resulted in only one dropped frame (i.e., frame 11). At frame 36, error blocks due to missing IP packets were also present in the video but a frame freeze period did not result from the dropped packets there.

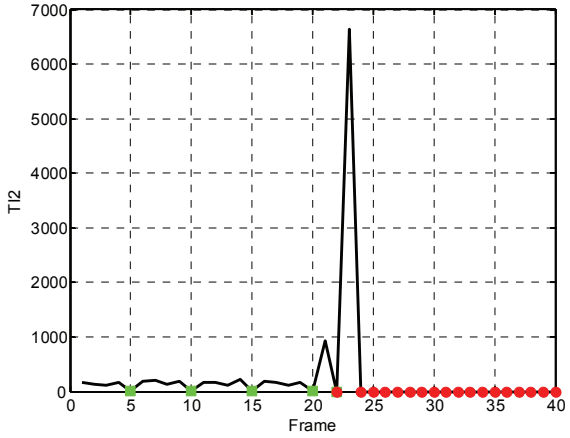

Figure 1. *TI2* with film conversion and still scene.

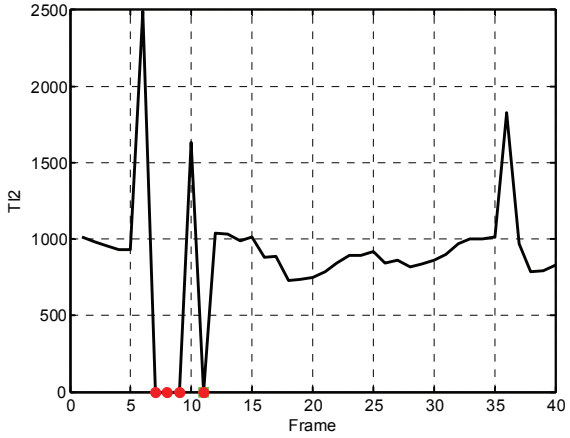

Figure 2. *TI2* with error blocks and frame freezes.

Figure 3 shows an example of a *TI2* waveform where the video coder performs dynamic frame dropping that depends upon the amount of motion in the video scene. Low motion portions of the scene are transmitted at full or near full frame rates (from frames 1 to 24, only one frame drop occurred at frame 10) while high motion portions are transmitted at reduced frame rates by dropping selected video frames (from frames 25 to 40, frame drops occurred at frames 25, 28, 30, 32, 34, 36, 38, and 40). This is a common method used to save on transmission bits.

[2] The original clips and common processed clips were excluded from each VGA data set to mitigate overrepresentation of these clips. This reduced the number of clips in each data set to 128.

Figure 4 shows an interesting example of a video coder that performs partial frame updates every other video frame. Here, the M_{dip} and A_{dip} thresholds that control the detection of the dips (shown in green) determine when the partial frame updates are too small to be true frame updates, and are hence declared to be dips. For this video clip, frame dips are detected before frame 36, which is detected as a frame drop (shown in red). Visual frame-by-frame examination of the video clip confirms this behavior where the detected dips (shown in green) are very minor frame updates (too minor to be viewed as new video frames) and the three detected drops (shown in red at frames 36, 38, and 43) are true frame repeats which are identical to the previous frames.

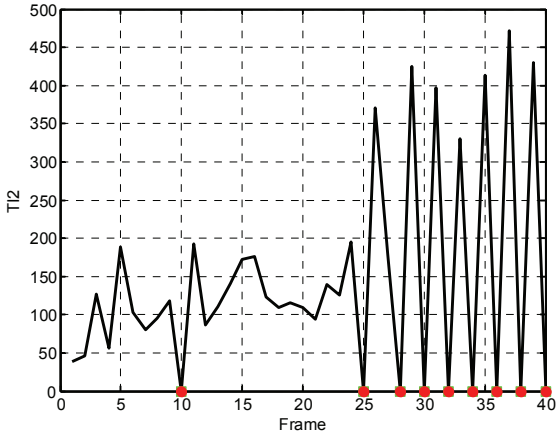

Figure 3. *TI2* with dynamic frame dropping.

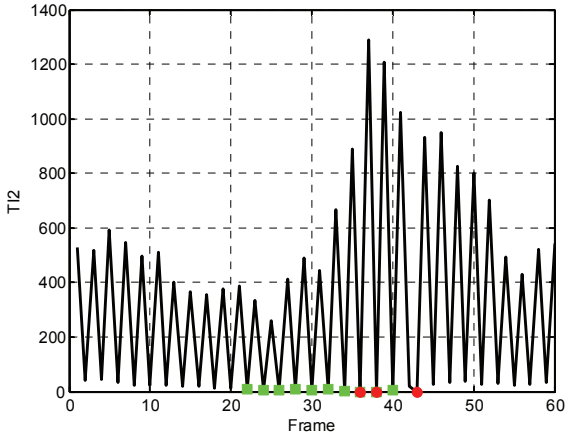

Figure 4. *TI2* with partial frame updates.

No thresholding scheme is perfect and the method presented in this document also has its problems. Figure 5 shows a *TI2* waveform for an original source scene of a head and shoulders shot where only the lips and eyelids are moving (note the extremely low magnitude of *TI2*). The scene has periods of very low motion which are mistakenly detected as frame drops (frames 29 to 31). Comparison of Figure 6 (from a video system that drops frames) with Figure 5 demonstrates one advantage of using the RR version of *FDF* given by step 9, namely, compensation for mistakes made by the algorithm for low motion scenes.

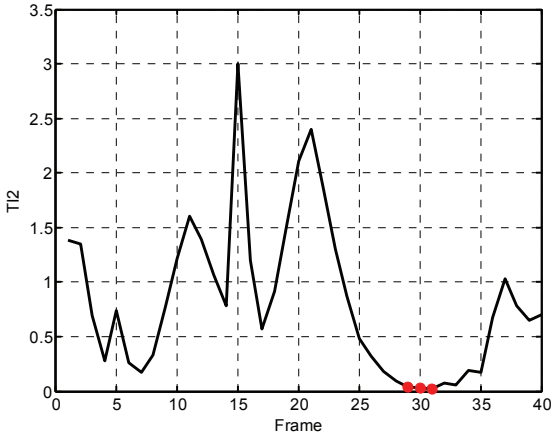

Figure 5. *TI2* of original scene with low motion.

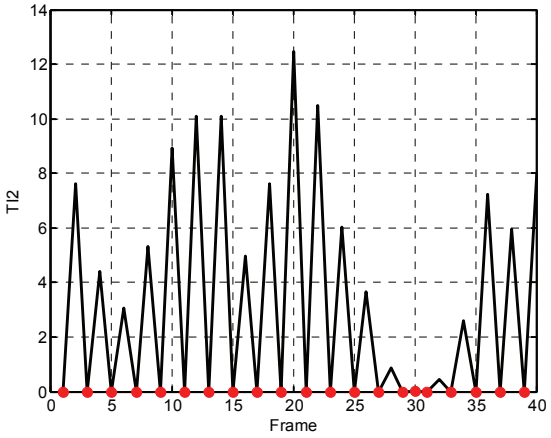

Figure 6. *TI2* of destination scene for Figure 5.

4. CONCLUSIONS

Reliable NR metrics for quantifying perceptual aspects of video quality are difficult to develop. This paper has presented an algorithm for quantifying one perceptual dimension of quality, namely, interruptions to the flow of motion in the video scene. This metric measures the fraction of dropped frames in a video clip by examining its frame-by-frame motion energy time history, a

computationally efficient process that should lend itself well to real time quality monitoring systems.

The problem of correctly identifying repeated/dropped frames is more difficult than it might first appear. This is because a repeated frame need not be an exact binary replicate of the prior frame (e.g., video compression systems can perform partial frame updates). This paper presents a dynamic thresholding scheme to identify these dropped frames. The algorithm works well for a wide range of video coders, transmission channels, and decoders.

One potential problem with the NR algorithm is that periods of low motion (e.g., moving lips) may be detected as dropped frames. An RR version of the algorithm can overcome this problem since the fraction of dropped frames of the destination video clip can be adjusted downward to account for still frames and/or detected drops in the source video (e.g., that might result from 24 fps film to 30 fps video conversion).

5. REFERENCES

[1] ITU-R Recommendation BT.601, “Studio encoding parameters of digital television for standard 4:3 and wide screen 16:9 aspect ratios," Recommendations of the ITU, Radiocommunication Sector.

[2] S. Wolf, "A No Reference (NR) and Reduced Reference (RR) metric for detecting dropped video frames," NTIA Technical Memorandum TM-09-456, Oct., 2008, available at http://www.its.bldrdoc.gov/pub/n3/video/index.php.

[3] "Final Report from the Video Quality Experts Group on the Validation of Objective Models of Multimedia Quality Assessment, Phase I," available at http://www.its.bldrdoc.gov/vqeg/projects/multimedia.