Network-Conscious Compressed Images over Wireless Networks*

Sami Iren, Paul D. Amer and Phillip T. Conrad

Computer and Information Sciences Department
University of Delaware, Newark, DE 19716 USA
Email: {iren,amer,pconrad}@cis.udel.edu
Phone: (302) 831-1944 Fax: (302) 831-8458

Abstract. We apply the concept of *network-consciousness* to *image compression*, an approach that does not simply optimize compression, but which optimizes overall performance when compressed images are transmitted over a lossy packet-switched network such as the Internet. Using an Application Level Framing philosophy, an image is compressed into path MTU-size Application Data Units (ADUs) at the application layer. Each ADU carries its semantics, that is, it contains enough information to be processed independently of all other ADUs. Therefore each ADU can be delivered to the receiving application out-of-order, thereby enabling faster progressive display of images. We explain why this approach is useful in general and specifically for wireless/heterogeneous environments.

1 Introduction

For many years, developments in image compression had one primary objective: obtaining the minimum image size. We argue that image compression algorithms should take into account that those images are likely to be transmitted over networks that will lose and reorder packets. Therefore, compression algorithms should not focus solely on achieving minimum image size; algorithms should be optimized to give the best performance when images are transmitted over such networks.

We apply the concept of *network-consciousness* [6] to *image compression*, an approach that takes network Quality of Service (QoS) into consideration when designing image compression. Network-conscious image compression is an application of the Application Level Framing (ALF) [3] principle. An image is divided into path MTU-size[1] pieces, called *Application Data Units* (ADUs), at the application layer, so that each piece carries its semantics, that is, it contains enough

* Prepared through collaborative participation in the Advanced Telecommunications/Info Distribution Research Program (ATIRP) Consortium sponsored by the U. S. Army Research Laboratory under the Fed Lab Program, Agreement DAAL01-96-2-0002. This work also supported, in part, by ARO (DAAH04-94-G-0093).

[1] MTU (Maximum Transmission Unit) is the maximum frame size that a link layer can carry. A path MTU-size ADU is one that can be transmitted end to end without the need for IP layer fragmentation and reassembly.

information to be processed independently of all other ADUs. As a result, each ADU can be delivered to the receiving application immediately upon arrival at the receiver, without regard to order, thereby potentially enabling faster progressive display of images.

Our research demonstrates that with a combination of innovative transport protocols and only a small penalty in compression ratio, today's standard compression algorithms can be modified to provide significantly better overall display of images, and hence performance, in the Internet and wireless environments.

Section 2 further motivates network-conscious image compression and defines what a network-conscious compressed image is. Section 3 explains why network-conscious image compression is well suited for wireless/heterogeneous environments. Section 4 summarizes a prototype implementation of our approach: network-conscious GIF. Section 5 concludes the paper with a summary.

2 Motivation — Network-Conscious Compressed Images

Historically, images are compressed and either stored locally (e.g., on a hard disk or CD-ROM) and accessed via high-speed error-free channels, or they are stored remotely and accessed via a relatively error-free, low-bandwidth circuit-switched network, (i.e., a phone line). In the latter case, the low bandwidth causes an increase in the time between requesting an image and seeing the image displayed (that is, the *response time*). As the response time increases, so does the appeal of compression methods that allow for *progressive display*. Applications using progressive display present an initial approximate image and refine it as more image data arrives, thereby allowing the user to begin comprehending the image sooner.

In both of these past environments, the channel between the stored compressed image and the client is essentially *reliable*. All pieces of image data are delivered without loss and in the precise order they were transmitted. Today's Internet introduces a new and different channel. As depicted in Figure 1, images are often transmitted over the intrinsically lossy, variable speed, order-changing Internet. The Internet's unreliable IP network layer protocol regularly loses and reorders pieces of image data during transmission. Furthermore, sometimes the last hop of the channel is a wireless link which is inherently unreliable because of bit errors and hand-off problems.

On top of IP, the Internet offers two transport protocols: UDP and TCP. UDP is simple but does little to enhance IP's service. Together UDP/IP provide an unordered/unreliable[2] transport service. Pieces of image data or the entire image can be lost, reordered, or duplicated. This environment is generally unacceptable for image retrieval.

TCP enhances IP's service to provide an ordered/reliable transport service. Image delivery is error-free and ordered, with no loss and no duplicates. However,

[2] "Reliable" refers to a "no-loss" service. Likewise, "unreliable" refers to a "lossy" service.

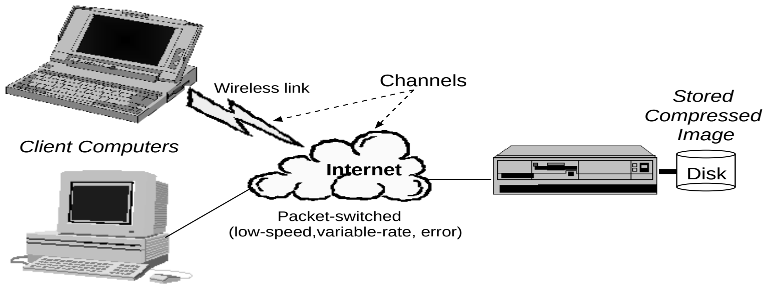

Fig. 1. Image Retrieval over Packet-switched/Wireless Internet

the overall end-to-end channel speed slows down because of TCP's retransmissions. While an image does arrive fully intact at the client, we argue that TCP service is *too* good, and that Web users are paying an unnecessary performance penalty by using it. Depending on the Internet's loss and delay characteristics, the cost of using more service than is needed can be significant [13].

Consider the following TCP-based scenario: a client Web browser retrieves a progressive image from a remote HTTP server over the Internet. After connection establishment, the client sends a request message and the server starts submitting chunks of image data to the transport sender. At the server, the transport sender segments these chunks of data and transmits individual Transport Protocol Data Units (TPDUs) over the Internet. As long as TPDUs arrive at the transport receiver in order, each TPDU is immediately delivered to the client application. On arrival, the client decompresses the data and progressively refines the display.

Figure 2a depicts the timing of such a scenario but assumes the network loses the original transmission of $TPDU_2$. Refinement of the progressive image at the client application is delayed until this lost TPDU is retransmitted and received at the transport receiver. At time *T2*, the user sees the first partial image on the display. Not until *T7* when $TPDU_2$'s retransmission arrives are TPDUs *2, 3, 4* delivered to the client so that the decompressed complete image can be seen at *T8*.

Between times *T3* and *T7*, the client application was unable to use $TPDU_3$ and $TPDU_4$ for two reasons: (1) TCP cannot deliver these TPDUs to the application because they are out-of-order, and (2) even if TCP could, the application (decompressor) would not be able to process these TPDUs because in general, today's decompression algorithms cannot process out-of-order data. For the time period *T4* - *T8*, the user sees no progression in the displayed image.

In an ideal case (Figure 2b), a client application would be able to process and display all TPDUs as soon as they arrive at the transport receiver regardless of their order. When $TPDU_3$ arrives at the transport receiver, it is immediately delivered to the application. The application decompresses the TPDU and refines the image *(T4)*. At *T4* the user sees a better image than in the previous scenario at the same point in time. Similarly, $TPDU_4$ arrives at the transport receiver

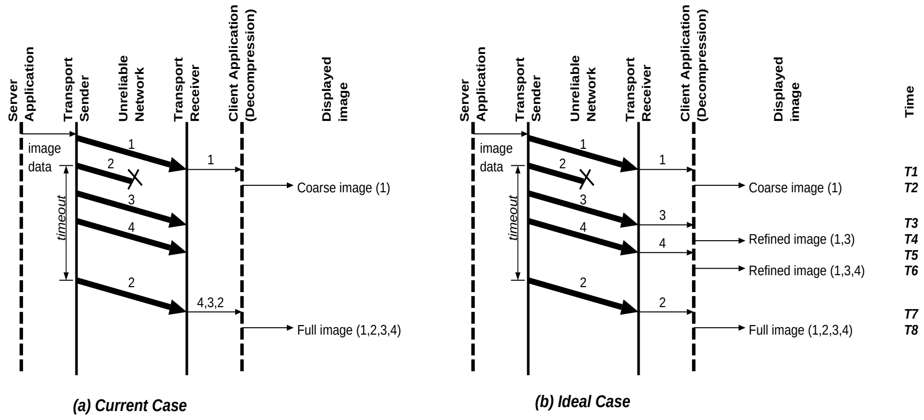

Fig. 2. Progressive Image Transmission over a Lossy Network (2 Cases)

(T5) and is decompressed and displayed by the application *(T6)*. At *T6*, the user sees yet again a better image than at the analogous point in time with TCP.

In Figure 2b, there is better progressive display for two reasons:

- the transport protocol delivers TPDUs *3* and *4* to the application sooner, and,
- the application can perform their decompression even though $TPDU_2$ is missing.

There are two basic requirements for the ideal case to work: (1) An alternative transport protocol must be available to provide an *unordered*, no-loss service. TRUMP [9] and POCv2 [4, 5] are two such transport protocols under development. (2) The receiving application must be able to decompress and display TPDUs independent of each other. That is, the image compression algorithm should allow out-of-order processing of incoming image data.

Definition 1. A network-conscious compressed image is one that is encoded *not* simply to give the *smallest size* for a specified image quality, but to give the *best (i.e., smallest) response time - image quality* combination to an end user retrieving the image over a packet-switched network.

As described in Section 1, the key feature of network-conscious image compression is that it produces path MTU-size, self-contained *blocks* (ADUs) that can be decompressed independent of each other. The basic characteristics of a network-conscious compressed image are: (1) application level framing, (2) progressive display (preferably multi-layered), and (3) robustness and adaptiveness to different user needs, and various networking conditions.

3 Wireless Networks and Network-Conscious Compressed Images

A challenging task for both transport protocol and application developers is to develop a protocol or an application that performs well over a heterogeneous network. Wireless networks have at least two inherent differences from wired networks that require special attention.

The first difference is that wireless networks suffer from high bit error rates and hand-off problems resulting in more TPDU losses than a wired network [2]. This implies that more out-of-sequence arrivals at the transport receiver can be expected. If an application cannot decompress these out-of-sequence data, they have to be buffered delaying the processing and painting of these data on the screen. Network-conscious image compression solves this problem by eliminating the in-sequence processing requirement.

The second inherent difference is that wireless devices are constrained in both their computing and communication power due to limited power supply [10]. Therefore, the transport protocols used on these devices should be less complex to allow efficient battery usage. With network-conscious image compression, a more efficient and simple transport protocol that needs not preserve order can be used. The expected buffer requirements at the transport receiver for an unordered protocol are always less than the buffer requirements for ordered protocols assuming a lossy network [13]. Ordered transport protocols require more computation to reorder ADUs before delivering them to the application thereby delaying the delivery of ADUs and resulting in a longer occupancy of the transport receiver's buffers. Furthermore the nature of the delivery is bursty and results in higher jitter. This burstiness may result in bottlenecks at the receiving application [7].

In a heterogeneous network that consists of variable speed wired and wireless links, we hypothesize that network-conscious compressed images will perform well because they adapt to different network conditions. In the heterogeneous and unstable Internet, one can expect applications which gracefully adapt to changing congestion levels to outlive those which cannot. Adaptability guarantees that applications will use all available resources and make efficient usage of these resources. This is especially important for multicast communication where each participant has different requirements. One research team argues that "network-consciousness" should be adopted as a design principle for many future applications [8].

Another advantage of using network-conscious image compression is that transmission control can be tailored to the characteristics of each ADU [7]. Not all parts of image data are uniform and require the same QoS. If the image compression algorithm provides layering of information, different layers of image data require different reliabilities and priorities. In a network-conscious compressed image, each block (i.e., ADU) can be associated with a set of QoS parameters and can be transmitted according to those parameters.

4 Experiments and Results

Our approach to testing the network-conscious image compression hypothesis consists of two phases. In phase one, we developed the Network-Conscious Image Compression and Transmission System (NETCICATS) [12] to observe the relation between compression algorithms and transport protocols over different network characteristics. We want to see how different compression techniques, when combined with different transport QoS, behave at different network loss rates. In phase two, we modified two popular image compression techniques, namely GIF89a[3] and SPIHT [14] (wavelet zerotree encoding), to make them network-conscious.

Because of lack of space, we will summarize only the network-conscious GIF algorithm. Interested readers may refer to the references [1, 11, 12] for more information on NETCICATS and algorithms.

4.1 Network-Conscious GIF

We modified the GIF89a standard to make it network-conscious. The result, called GIFNCa [1], removes the ordered delivery requirement (and for some applications the reliability requirement) of GIF89a by framing image data at the compression phase (i.e., application level).

The tradeoff between GIFNCa and GIF89a is one of compression vs. progressive display performance. GIF89a's advantage is its expected better compression. GIFNCa's advantage is its expected *faster progressive display* at the receiver when transmitted over an unreliable packet-switched network.

We ran a set of experiments comparing (1) GIF89a over a reliable, ordered transport protocol called *Sequenced Protocol (SP)* vs. (2) GIFNCa over a reliable unordered protocol called *Xport Protocol (XP)*. Both SP and XP are implemented at the user-level over UDP, and use the same code for all functions (including connection establishment/tear-down, round-trip-time estimation, retransmission timeout, acknowledgments, etc.); the only difference is that SP provides packet resequencing (i.e., ordered service) at the receiver, while XP does not.

Each experiment downloads a compressed image from server to client using an interface similar to familiar web browsers (see Figure 3). Packets are routed through a *Lossy Router*, a modified IP router that can simulate three loss models (Bernoulli, burst (2-Step Markov), or deterministic), and a *Reflector* that delays forwarding of IP packets to simulate a lower bandwidth link (28.8 Kbps for these experiments).

Due to space limitations, we provide only a portion of the results in this paper. Interested readers may refer to [1] for more information and results.

The average percentages of image data being displayed at various points in time for 0%, 5%, 10%, and 15% IP packet loss rates are graphed in Figure 4. At 0% loss rate, GIF89a performs better due to its higher compression ratio. As

[3] GIF89a is a Service Mark of CompuServe, Inc., Columbus, OH.

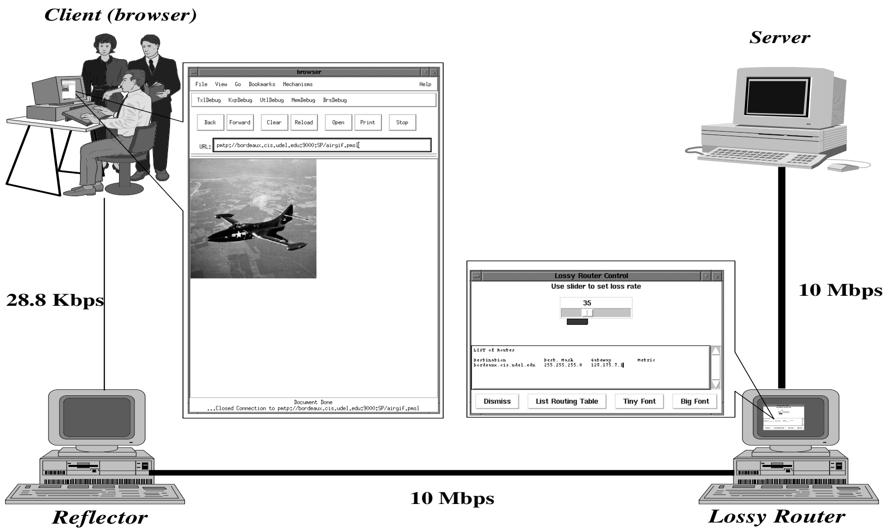

Fig. 3. Network-Conscious GIF Testing Environment

the loss rate increases, both GIFNCa and GIF89a take longer to display but the GIFNCa performance does not degrade as quickly, i.e., it improves *relative to* GIF89a.

To appreciate the significance of these numbers, Figure 5 illustrates the progressive display of an aircraft image using both GIF89a and GIFNCa, in the left and right columns, respectively. The figure shows the image that the application displayed at 5, 10, 15 and 20 seconds for the most "typical" run at a 10% loss rate; This run is closest in mean-squared distance to the average over all experiments for 10% loss rate. In all intermediate times, GIFNCa progressive image is better.

While more serious and exhaustive emprical study is currently underway, these initial results highlight the potential benefit of using GIFNCa over GIF89a under lossy network conditions.

5 Conclusion and Future Work

Traditional image compression algorithms are not designed for lossy packet-switched networks and heterogeneous environments with wired and wireless links. They are optimized to minimize image size only. However, minimum image size does not necessarily provide the best performance when those images are transmitted over lossy networks. The ordered-delivery requirement of traditional compression algorithms cause unnecessary delays at the receiving end.

We apply network-consciousness to image compression so that the compression algorithms will not be optimized only to give the minimum image size;

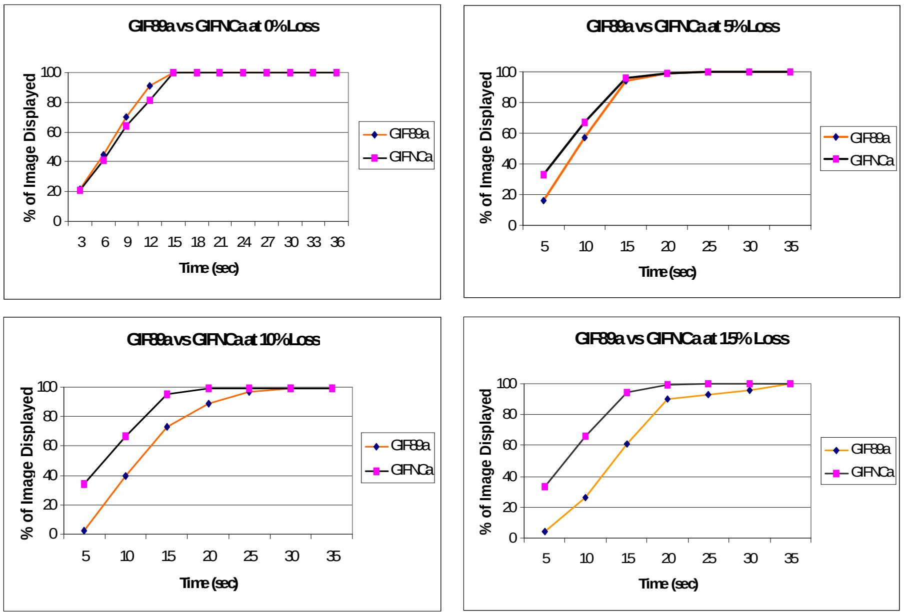

Fig. 4. Comparison of GIF89a and GIFNCa at Various Loss Rates

they will be optimized to give the best performance when transmitted over lossy networks. Network-conscious image compression is especially useful for wireless environments because (1) wireless networks inherently suffer from high bit errors and hand-off problems resulting in more TPDU losses, and (2) limited power problem of wireless devices can be addressed by using simpler transport protocols which do not have to buffer and reorder missing TPDUs.

We have developed two compression algorithms that utilize this approach: network-conscious GIF and network-conscious wavelet zerotree encoding. Initial experiments for network-conscious image transmission are promising.

References

1. P. Amer, S. Iren, G. Sezen, P. Conrad, M. Taube, and A. Caro. Network-conscious GIF image transmission over the Internet. In *4th International Workshop on High Performance Protocol Architectures (HIPPARCH'98)*, June 1998. (To appear).
2. H. Balakrishnan, V. Padmanabhan, S. Seshan, and R. Katz. A comparison of mechanisms for improving TCP performance over wireless links. In *ACM SIGCOMM '96*, Stanford, CA, August 1996.
3. D. Clark and D. Tennenhouse. Architectural considerations for a new generation of protocols. In *ACM SIGCOMM '90*, pages 200–208, Philadelphia, PA, September 1990.

4. P. Conrad. Order, reliability, and synchronization in transport layer protocols for multimedia document retrieval. PhD Dissertation, CIS Dept. University of Delaware, (in progress).
5. P. Conrad, E. Golden, P. Amer, and R. Marasli. A multimedia document retrieval system using partially-ordered/partially-reliable transport service. In *Multimedia Computing and Networking 1996*, San Jose, CA, January 1996. `www.eecis.udel.edu/~amer/PEL/poc/postscript/mmcn96full.ps`.
6. W. Dabbous and C. Diot. High performance protocol architecture. In *IFIP Performance of Computer Networks Conference (PCN '95)*, Istanbul, Turkey, October 1995. IFIP.
7. C. Diot and F. Gagnon. Impact of out-of-sequence processing on data transmission performance. Technical Report Project RODEO RR-3216, INRIA - Sophia Antipolis, France, July 1997. `ftp://www.inria.fr/rodeo/diot/rr-oos.ps.gz`.
8. C. Diot, C. Huitema, and T. Turletti. Multimedia applications should be adaptive. In *HPCS Workshop*, Mystic (CN), August 1995. IFIP.
9. E. Golden. TRUMP: Timed-reliability unordered message protocol, December 1997. MS Thesis, CIS Dept., University of Delaware.
10. Z. Haas and P. Agrawal. Mobile-TCP: an asymmetric transport protocol design for mobile systems. In *International Conference on Communications*, Montreal, Quebec, Canada, June 1997. IEEE.
11. S. Iren. Network-conscious image compression. PhD Dissertation, CIS Dept., University of Delaware, (in progress).
12. S. Iren, P. Amer, and P. Conrad. NETCICATS: network-conscious image compression and transmission system. (Submitted for publication).
13. R. Marasli, P. Amer, and P. Conrad. An analytic model of partially ordered transport service. *Computer Networks and ISDN Systems*, 29(6):675–699, May 1997. `www.eecis.udel.edu/~amer/PEL/poc/postscript/cn-isdn96.ps`.
14. A. Said and W.A. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. In *IEEE Transactions on Circuits and Systems for Video Technology*, volume 6. IEEE, June 1996.

This article was processed using the LaTeX macro package with LLNCS style

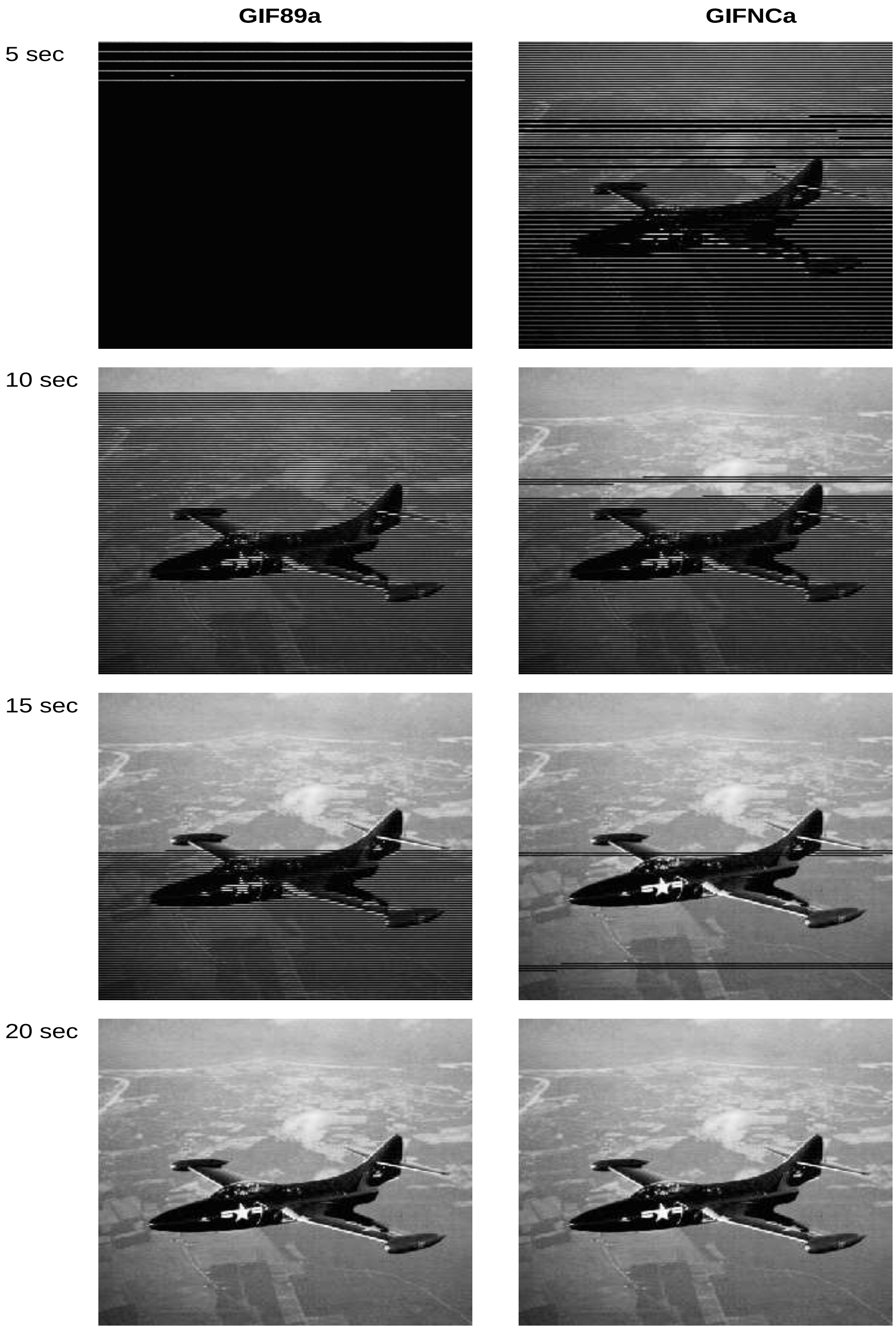

Fig. 5. Sequence of GIF89a and GIFNCa Images at 10% Loss