ERROR-RESILIENCE IN MULTIMEDIA APPLICATIONS OVER AD-HOC NETWORKS

L.E. Doyle, A.C. Kokaram

Department of Electronic & Electrical Engineering
Trinity College, Dublin 2
Ireland

D. O'Mahony

Department of Computer Science
Trinity College, Dublin 2
Ireland

ABSTRACT

Ad-hoc networking has been of increasing interest in recent years. It encapsulates the ultimate notion of ubiquitous communications with the absence of reliance on any existing network infrastructure. This paper presents a concept for robust operation of multimedia applications over such networks. Error resilient communication is achieved by using a new error detection and concealment technique that exploits information *from the decoded image data itself* as well as using information from the underlying network. This approach unifies information from both traditional Computer Science and Signal Processing domains. A layered architecture framework for the implementation of the proposed system is also described.

1. INTRODUCTION

1.1. Ad-hoc Networks

An ad-hoc network is a collection of wireless mobile nodes that will dynamically form a temporary network without the use of any existing network infrastructure. Ad-hoc networks replace the centralised hierarchical administration structure of contemporary networks with a distributed approach to routing management that allows each network to grow, reduce in size or fragment in real-time without referencing any central authority. The lack of a dependence on pre-existing infrastructures makes the ad-hoc network very appealing for many reasons. Infrastructures may not exist due to the lack of appropriate resources or alternatively due to their destruction or degradation, whether due to neglect, obsolescence or war. For example, a group of business people may want to conduct an impromptu meeting at an airport, where a short-lived ad-hoc network easily facilitates the exchange of documents and information between their hand-held terminals. Another relevant example is the occurrence of a natural disaster, such as an earthquake, which results in the destruction of existing network facilities. Ad-hoc networks fill the communications void created in such a situation allowing emergency services to operate in a coherent and productive manner.

Enabling multimedia communications over such networks is an interesting proposition. To date research in the area of ad-hoc networks has focussed on such issues as routing and media access with very little focus on the application side. There are clear applications in emergency services, consumer games and military scenarios.

1.2. Ad-hoc Networks and Multimedia Applications

To enable multimedia applications over ad-hoc networks the issue of error resilient data transmission must be addressed. There are many well established schemes for combating errors in communications networks. In general these schemes fall into three categories: (1) schemes that make the transmitted bitstream resilient to errors through the use of source and channel coding; (2) schemes that involve interaction between the transmitting and receiving nodes to either effect a change in the properties of the transmitted data (based on communication channel conditions) or involve retransmissions; and (3) schemes that focus on error detection and correction.

The use of type (1) schemes is essential in wireless communications. However ad-hoc networks are not suitable for type (2) schemes. For example (for non real-time applications) retransmission has widespread implications for ad-hoc networks. The topology of an ad-hoc network can be under constant and fast moving change. A network may consist of as few as two nodes or may encompass thousands of nodes and these nodes can be moving slowly or very quickly. Between the time of the original transmission and reception of the data and the request for retransmission it is quite possible that the route between the source and destination node has changed. Therefore before the retransmission takes place, a new route must again be established. A suitable ad-hoc routing protocol such as DSR, AODV or ZRP [1, 2] must be used to determine the route. This adds substantial signalling overhead to the network. In real time applications that make use of protocols such as RTP where RTPC is used to create a feedback mechanism to facilitate sender-based adaptation to channel conditions, the same problems occur in creating the feedback route.

What is most appropriate for ad-hoc networks is for applications to operate without heavy reliance on interaction with the node that is transmitting the data. In effect, the best applications for ad-hoc networks are those which operate effectively in *downstream* mode with a minimal or no *upstream* relationship to the transmitting node. This means that error detection and concealmeant are the only valid methods for enhancing the performance of the ad-hoc network with a view to enabling multimedia communications.

1.3. Error Resilience Concepts

Consider MPEG4 as an example of a standard multimedia stream that is required to be transmitted over an ad-hoc network. Error resilience means that the decoder must be able to *detect* errors, continue to decode *despite* errors and finally *correct* errors in the received bitstream.

Error resilience schemes in MPEG4 have been the subject of much recent work [3, 4]. The standard itself has incorporated the notion of resynchronisation which allows the decoder to continue operation *despite* errors in the bitstream. The work done on *detection* of errors in MPEG4, however, has concentrated on spotting

inconsistencies in the bitstream itself. This means, for instance, detecting errors in the bitstream syntax. Thus an error is flagged when more DCT Blocks are being decoded than encoded, or another data partition being encountered before decoding of the last partition was complete.

Furthermore, even when errors are detected the localisation of the error in the bitstream is still an issue [5, 6]. This is because the symptom indicating the error in the bitstream may have been observed much later than the processing of the actual error. The use of Reversible Variable Length Codes in MPEG4 can improve this situation by allowing the decoder to decode the bitstream in the reverse direction and so assist in the localisation of errors [5].

The left hand side of Figure 1 shows half of the 8th frame (B-VOP) of the Foreman sequence decoded from a corrupted MPEG4 bitstream[1] at a rate of .8 bits/pixel, using IBBPBB . . . . It shows the effect of an error simulated in the DC coefficients of the 10th macroblock of the intra coded frame. As can be seen the DC information up to the 10th macroblock is correctly decoded and displayed, but all the DC information up to the next resync_marker is lost. All of the AC information for this Video Packet is also lost as the decoder must jump to the next Video Packet Header to reestablish synchronization, thus skipping all of the AC information.

1.4. A coherent approach to resilience

The key points here are that errors in the bitstream do not *necessarily* correspond to visible errors, and furthermore, that errors in the decoded image can be detected *in that domain* and so help to localise errors in the bitstream even further.

This paper introduces a new mechanism for error detection and concealment for efficient video transmission over ad-hoc networks. It exploits information *from the decoded image data itself* and also uses information from the underlying network. When applied to MPEG4, the method can localise errors to an even greater extent than with RVLCs alone. An efficient, and dynamic implementation framework is also described.

2. ERROR DETECTION AND CONCEALMENT

In this paper the two problems of detection and concealment are treated as separate issues.

2.1. Error Detection

It is required to design a process for detecting errors by incorporating information from the image data, the bitstream syntax and the networking system. To begin this design, first it is necessary to produce a model for the observed, corrupted data.

Denote the decoded, image data at site $\mathbf{x}$ in frame n, which is received with errors, as $Y_n(\mathbf{x})$, and the uncorrupted (original) data as $I_n(\mathbf{x})$. Then

$$Y_n(\mathbf{x}) = (1 - b_n(\mathbf{x}))I_n(\mathbf{x}) + b_n(\mathbf{x})\mu_n(\mathbf{x}) \qquad (1)$$

where $b_n(\mathbf{x})$ is a binary variable, set to 1 to indicate a corrupt site, and set to 0 otherwise and $\mu_k(\mathbf{x})$ is the corrupted data at the site $\mathbf{x}$.

Now, the problem of error detection is to estimate $b_n(\cdot)$ given the observed data $Y_n(\cdot)$ and bitstream $\mathbf{s}$. This can be done by manipulating

[1] Thanks to L. Mcmanus for the decoded MPEG data.

$p(b_n|Y_n, I_n, s)$, the probability that a particular pixel is corrupted given the observed image data and bitstream. Proceeding using Bayes rule results in

$$p(b_n|Y_n, I_b, I_f, s) \propto p(Y_n|b_n, I_b, I_f, s)p(b_n|s) \qquad (2)$$

where I_b, I_f are previous and next decoded frames (either I or P frames in the case of MPEG). Assuming that the correctly decoded data follows the translational model $I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + d_{n,n-1}(\mathbf{x}))$, the likelihood $p(Y_n|b_n, I_b, I_f)$ can be written as

$$\propto \exp -\left((1 - b_n)[(Y_n - I'_b)^2 + (Y_n - I'_f)^2]/(2\sigma_e^2)\right) \qquad (3)$$

where I'_k is compensated for motion. This expression implies that, given $b_n = 0$ (no corruption), the motion compensated frame difference is a Gaussian distributed random variable $\sim \mathcal{N}(0, \sigma_e^2)$.

2.2. The prior

It is in the definition of the prior probability of corruption $p(b_n)$ that information from the networking layer and bitstream syntax can be incorporated. Thus a useful formulation of this prior is $p(b_n|s) \propto p_{\text{net}}(b_n)p_{\text{MPEG}}(b_n) \exp -(\alpha b_n)$ where $p_{\text{net}}(b_n)$ and $p_{\text{MPEG}}(b_n)$ refer to the probability of an error as indicated by the error detection and sequencing information from the networking layer and MPEG syntax respectively. The last term ensures that there is a penalty α for setting $b_n = 1$. α can be considered to be a threshold on the DFD, $(Y_n - I_k)$. This type of formulation is similar to that used for treatment of archived picture data in [7].

The choice of expressions for $p_{\text{net}}(b_n)$ and $p_{\text{MPEG}}(b_n)$ can depend on the complexity of the node hardware platform. The simplest are uniform distributions which are non-zero only over image regions corresponding to a lost packet (as indicated by the network layer) or lost synchronisation (as indicated by the MPEG syntax). More complex formulations can take into account the Block coding nature of MPEG and allow for b_n to be a spatially coherent field corresponding to those blocks.

2.3. Solution

A MAP solution for b_n is required that is of variable complexity depending on the hardware platform. The simplest (and arguably most effective) solution is to test each pixel in the decoded image, and evaluate $p(b_n = 0)$ and $p(b_n = 1)$ in equation 2 (cf ICM). The solution with the higher probability is chosen as the desired result. More complex but complete solutions would allow for simultaneous decoding and error detection which would involve flipping bits in the bitstream according to some Markov Chain Monte Carlo scheme.

The essence of this unifying strategy is to flag a pixel as corrupt when *both* the forward and backward motion compensated intensity differences at a site $\mathbf{x}$ are high AND when the networking and/or MPEG syntax indicates that something is amiss.

What is extremely important about this formulation of the error *detection* problem is that knowledge is incorporated about the the *decoded* image. This is done through the use of the likelihood expression derived from equation 1. This highlights the point that the decoder should not care whether bits are received in error if there is no *visible* error in the resulting image.

2.4. Illustration

The centre pane of Figure 1 shows the detection of errors using only the temporal image based error detector. Here $p(b_n) \propto \exp -(100 b_n)$, and $p_{\text{MPEG}}(b_n),\ p_{\text{NET}}(b_n)$ set to constants. In effect the detector is citing errors when the DFD is greater than 10 in both the forward and backward directions. Observe that already, only *visible* errors are being detected. But observe also, that errors have been detected in areas which were *not* corrupted at all. This is because of the inaccuracy of motion information, and of course the errors in the motion model itself.

The right hand side figure 1 shows the result when precisely the same algorithm is applied except with the use of $p_{\text{MPEG}}(b_n)$ which is set to 1 between the resynchronisation markers and 0 elsewhere. In effect this causes a logical AND operation between the image based detector and the detected erroneous areas flagged by the MPEG syntax. Right away, the detection of errors in the decoded image is improved from either the purely image centric case or the purely syntax based case. Only the regions which have been **visibly** distorted by loss of AC coefficients have been flagged.

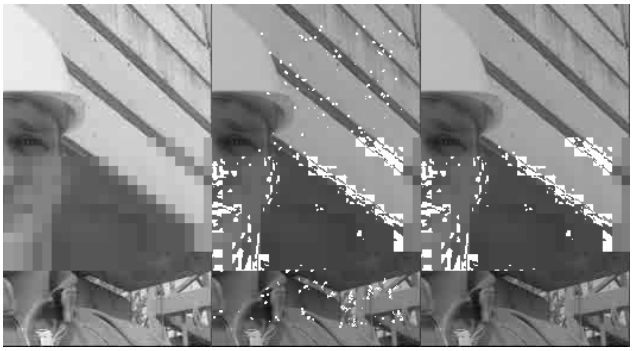

Fig. 1. From left to right: decoded 8th frame with DC coefficient error, detected errors using image only detection and detected error locations using information from the bitstream and the image data.

2.5. Error correction

In order to correct errors, first note that it can be the case that huge areas are lost in the decoded image. This could certainly imply lost frames. Reconstruction of image data therefore implies reconstruction of the underlying motion field in this latter case, since no spatial data can be relied on to infer such large amounts of missing data. In the worst case (considered here) all motion and image information for the current frame is lost.

Consider that a missing (or corrupted) B-VOP has been detected in the video stream. It is assumed that the basic image sequence model is as follows:

$$I_n(\mathbf{x}) = I_{n-1}(\mathbf{x} + \mathbf{d}_{n,n-1}(\mathbf{x})) + e_n(\mathbf{x}) \qquad (4)$$

where $\mathbf{d}_{n,n-1}$ refers to a motion vector mapping frame n into $n-1$ and $e_n(\mathbf{x}) \sim N(0, \sigma_e^2)$.

The idea is to choose the best estimate for the missing motion by manipulating $p(\mathbf{d}_{n,n-1}|I_{n-1}, I_{n+1})$ in a Bayesian fashion, given the model above. Assuming corrupted B frames, I_{n-1} would be the last correctly received Intra coded frame, and I_{n+1} would be the next P frame. Then the idea is to reconstruct the missing data in effect by cutting and pasting into the current frame given the reconstructed motion field. There is not enough space here for a detailed consideration of the algorithm and the reader can see [7] for details of a related process applied to archived video. The use of a Bayesian framework allows the coherent manipulation of spatial *and* temporal information in the decoded image data and this is a definite improvement over the relatively strong spatial-only focus of previous work.

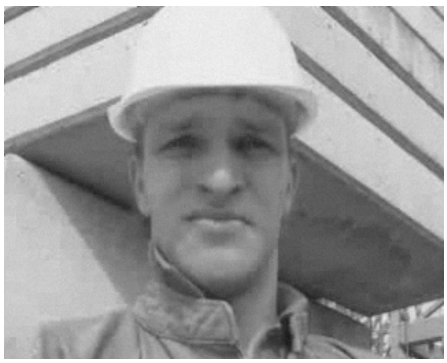

Fig. 2. Frame 8, reconstructed using the motion vectors from frame 9-10 and frames 7-6.

The main points are as follows.

1. Interpolation of the motion field is fully spatio-temporal and relies on the specification of priors for motion which encourage temporally smooth motion fields. In the simplest case of no acceleration, this prior represents a penalty for motion vectors which show a substantial differentce between motion compensated locations.
2. Given that the motion field only varies slowly over space and time, a fast algorithm is possible by using previous and next frame motion information to propose a set of motion candidates at each missing site. These are chosen by motion compensating vectors used for decoding previous and next frames.
3. The best motion candidate is chosen based on penalties for spatial and temporal smoothness (encouraged by the motion priors), as well as contiguity of image data as indicated in equation 4.

This algorithm was applied to reconstructing frame 8 of the Foreman sequence. Figure 2 shows the frame reconstructed using only the motion vectors from frame 9-10 and frame 7-6. This corruption is clearly removed and visible distortion is suppressed.

3. A FRAMEWORK FOR AD-HOC NETWORK COMMUNICATION

It is reasonable to assume that implementations of multimedia communications over ad-hoc networks will in general employ some kind of packet based protocol. There are two schools of thought on the protocols to be used - either a world-wide standard is created or nodes that wish to communicate in an ad-hoc fashion become capable of configuring themselves appropriately. In an era

of reconfiguarble and programmable communication systems we believe that a truely ad-hoc network will adapt to the network in which it wants to communicate.

To cater for this we have created a 'generic layer' interface that allows the dynamic assembly of a stack from application to physical layer consisting of hardware and software elements. The layers are independently configurable. Multi-thread and synchronisation facilities allow each layer to run as an independent thread and interact with its neighbouring layers in very simple, clearly defined manner. The layers of the stack can be altered to suit the underlying network. A pool of options for each layer exists; the layers can implement different standards or proprierty protcols. Figure 3 shows some examples and more details of this system can be found in [8]. The layers are assembled at runtime to form a complete protocol stack.

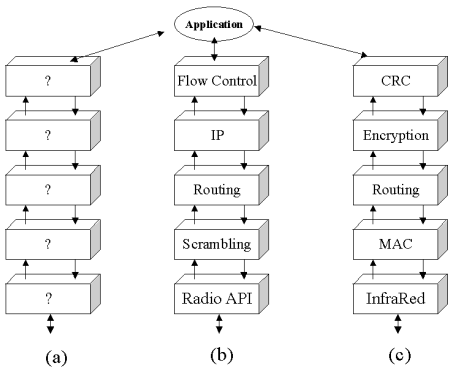

Fig. 3. The Generic layer Structure.

From the point of view of enabling mutimedia applications on an ad-hoc network the layers of the stack can also be configured to suit the application. In other words using this framework we can insert layers that will facillitate more efficient error detection and concealment and thereby reduce the need for interaction with the sending node. It is reasonable to assume that the size of the data packets transmitted from source to destination would be smaller than that employed in the packet structure of the encoder producing the multimedia bitstream. If this is the case, then packet error information extracted from a layer could be passed to the decoder or multimedia application and used to identify more clearly the location of errors in the bitstream. A simple example of this is to insert a CRC layer into the stack as shown in Figure 3 (c). This layer could perform a CRC on the packet (containing a section of an MPEG-4 stream) on the transmit side and perform the reverse operation on the receive side. However rather than dumping faulty packets, it would pass them upwards and flag the application that errors have occurred. This very modular approach avoids creating a new protocol to deal with multimedia communications over ad-hoc networks and instead allows the use of existing techniques to build a system that best suits the needs of the application.

4. FINAL COMMENTS

Because of their scope for extending wireless communications beyond the bounds of infrastructural network, ad-hoc networks are an important framework for investigation with respect to multimedia transmission. This paper has considered the requirements for error-resilient communications in such networks, illustrating the need for the maximum advantage to be taken of information available to the destination node and thereby reducing the need for retransmissions by the source or the establishment of feedback paths between the source and destination. The paper introduced a new *quantitiative* approach to error detection for compressed bitstreams which unifies information from image data, MPEG bitstream and the networking infrastructure. A flexible layered architecture to build an optimal communication stack to support the application was described. This approach has the advantage of limiting processing power to only *visible* distortion in the decoded images caused by bitstream errors, and also allows better localisation of errors.

5. REFERENCES

[1] E.M. Royer and C.K. Toh, "A review of current routing protocols for ad hoc mobile networks," *IEEE Personal Communications*, pp. 46–55, April 1999.

[2] J.Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking*, October 1998, pp. 85–97.

[3] C. W. Chen, P. Cosman, N. Kingsbury, J. Liang, and J. W. Modestino, "Special issue on error-resilient image and video transmission.," *IEEE Journal on Selected Areas in Communications*, June 2000.

[4] Yao Wang, Stefan Wegner, Jiangtao Wen, and Aggelos K. Katsaggelos, "Error resilient video coding techniques," *IEEE Signal Processing Magazine*, pp. 61–82, July 2000.

[5] Iole Moccagatta, Salma Soudagar, Jie Liang, and Homer Chen, "Error-resilient coding in JPEG-2000 and MPEG-4," *IEEE Journal on Selected Areas in Communications*, , no. 6, pp. 899–914, June 2000.

[6] K. W. Kang, S. H. Lee, and T. Kim, "Recovery of coded video sequences from channel errors.," in *Proc. SPIE Visual Communications and Image Processing*, May 1995, pp. 19–27.

[7] A. C. Kokaram, *Motion Picture Restoration: Digital Algorithms for Artefact Suppression in Degraded Motion Picture Film and Video*, Springer Verlag, ISBN 3-540-76040-7, 1998.

[8] T. Forde, L.E. Doyle, and D. O'Mahony, "An evaluation system for wireless ad-hoc network protocols," in *Irish Signals and Systems Conference*, June 2000, pp. 219–224.