A Spatially Enhanced Error Concealment Technique and its Potential Alternative Application to Reduce H.264 Stream Sizes

Steven Beesley and Christos Grecos

***Abstract* - With more and more video content being transmitted digitally and with user expectations continually rising, error concealment is becoming an increasingly important part of streaming media. Often overlooked in the past, even now manufacturers are often only doing the bare minimum necessary in order to avoid complexity. This paper first presents a combination of simple techniques that when combined produce an extremely effective concealment method that maintains spatially correlated edges throughout any lost data; this in turn gives an increase in both mathematical and visual performance when compared against the commonly used bilinear concealment technique. Secondly this paper looks at an alternative use of the bilinear passive error concealment algorithm that is often used by H.264 decoders. Occasionally a concealed macroblock is mathematically closer to the original than an encoded and decoded one, by removing these from the stream at the encoder and thus forcing the decoder to conceal the missing data, a significant reduction in the bit stream size (up to 5%) can be achieved with almost no loss in quality.**

***Index terms* - Error Concealment, Spatial, Sobel, Rate Reduction, Edge Detection**

I. INTRODUCTION

H.264 / MPEG-4 Part 10 Advanced Video Coding (AVC) is the latest video coding standard from the Joint Video Team (JVT), a collaboration between the ITU-T Video Coding Experts Group (VCEG) and the ISO/IEC Moving Picture Experts Group (MPEG) [1]. The standard has been designed to be suitable for a wide range of applications including 2-way communications such as video telephony plus 1-way streams such as for storage and broadcast purposes. It has also been designed with enhanced compression in mind such that it achieves considerably improved rate-distortion efficiency over existing standards such as MPEG-2 and H.263 [2].

A number of new coding features were incorporated in order to achieve this high level of compression, allowing up to 32 reference pictures for prediction, quarter pixel precision for motion compensation, variable block sizes including exact match integer transforms at 4x4 and 16x16, logarithmic quantisation step sizes and an in-loop deblocking filter to name just a few. However it is the new entropy encoding design that is of most interest to this paper, H.264 is structured such that all of the data below slice headers utilizes either Context-Adaptive Binary Arithmetic Coding (CABAC) or the lower complexity Context-Adaptive Variable Length Coding (CAVLC) compression techniques [3].

Stream errors may occur for a number of reasons, for example noise on a transmission line, hard drive corruption or scratches on an optical disc, alternatively excessive line congestion may cause unacceptable delays to a real time system. The very nature of entropy encoding makes a stream extremely vulnerable to such errors; a single bit error can render the entropy data useless until the stream can be resynchronised. In the case of H.264 this resynchronisation point will be the next fixed length coded pattern, for example the next data partition or slice header, therefore severe degradation can occur within an erroneous slice [4]. Several options exist to counteract such damage such as Forward Error Correction (FEC) which adds redundant information to the stream and Automatic Retransmission reQuest (ARQ) which allows erroneous data to be requested again, however both of these require additional bandwidth and introduce additional latency [5].

There are times when adding latency to a stream is unacceptable (for example video telephony applications where a fluent transmission is more important than exact reconstruction of the data), bandwidth unavailable or communication only possible in one direction. In these cases the aforementioned resiliency tools are unavailable and so a decoder has to rely on alternative flexibilities plus error concealment methods. The H.264 standard [3] also adds the option to use Flexible macroblock ordering (FMO); this allows a picture to be partitioned into multiple slices allowing for the creation of macroblock patterns that are better suited for error concealment. For example by splitting a picture into two slices in a checkerboard pattern, even if an entire slice is lost the damaged macroblocks will still have their horizontal and vertical neighbours from which to conceal from, using a half checkerboard pattern also provides diagonal neighbours.

Detection of errors is achieved by checking that control codes are valid and that video semantics are correct (for example that the number of macroblock coefficients received matches the expected number based on the macroblock type). In order to conceal erroneous data, damaged macroblocks are discarded and once the entire picture is decoded are then initially estimated from correctly received data in order to try and hide the visual repercussions of errors from the end user. Should less than 2 neighbouring blocks contain correctly decoded data then previously concealed macroblocks will also be used, as concealing is only an approximation it is important to provide as many neighbouring macroblocks as possible [6].

Manuscript received June 01, 2007, revised October 30, 2007 and December 21, 2008.Steve Beesley is with the Loughborough University, Loughborough, Leicestershire, LE11 3TU, UK. Christos Grecos (corresponding author) is with the University of Central Lancashire, Preston, Lancs, PR1 2HE, UK

1845-6421/08/7090