An Unequal Error Protection Method for Progressively Transmitted 3-D Models

Ghassan AlRegib, Member, IEEE, Yucel Altunbasak, Senior Member, IEEE, and Jarek Rossignac

Abstract—In this paper, we present a packet-loss resilient system for the transmission of progressively compressed three-dimensional (3-D) models. It is based on a joint source and channel coding approach that trades off geometry precision for increased error resiliency to optimize the decoded model quality on the client side. We derive a theoretical framework for the overall system by which the channel packet loss behavior and the channel bandwidth can be directly related to the decoded model quality at the receiver. First, the 3-D model is progressively compressed into a base mesh and a number of refinement layers. Then, we assign optimal forward error correction code rates to protect these layers according to their importance to the decoded model quality. Experimental results show that with the proposed unequal error protection approach, the decoded model quality degrades more gracefully (compared to either no error protection or equal error protection methods) as the packet-loss rate increases.

Index Terms—Joint source and channel coding, multimedia networking, packet loss resilient transmission, unequal error protection (UEP), virtual reality over IP networks, 3-D graphics streaming.

I. INTRODUCTION

An increasing number of Internet applications utilize highly detailed three-dimensional (3-D) models, giving rise to a large amount of data to be stored, transmitted, and rendered within a limited time frame. To alleviate these limitations, several single-layer compression techniques have been developed [1]–[7]. These algorithms produce bitstreams that still require a significant amount of time to be fully downloaded before the model can be displayed on the client’s screen. To reduce this time, progressive compression techniques have been designed so that a coarse model is downloaded and displayed first and then refinement information is transmitted later [8]–[12].

Even though compression techniques reduce the number of bits transmitted, they do not account for network losses that affect the decoded model quality. In a typical packet-switched network (e.g., the Internet), packets can be lost because of congestion and buffer overflow. Error-resilient streaming techniques can be classified as either closed-loop, where lost packets are re-transmitted or open-loop, where pre- and post-processing of data are employed [13]. Generally, the first approach is applicable when there is no time limitation since it ensures that all packets are to be delivered (to the client) error-free. The transport control protocol (TCP) is an example of this approach because all lost TCP packets are re-transmitted until they are correctly received at the client. On the contrary, open-loop solutions are applicable for time-sensitive applications where re-transmission cannot be tolerated. When a 3-D graphics application is time-sensitive, a closed-loop approach may not be possible. Furthermore, closed-loop approaches such as TCP are not suitable when the round-trip-time (RTT) is large. Similarly, TCP is not suitable in multicast environments because by requiring feedback information from all receivers, TCP-based methods suffer from the feedback implosion problem, which occurs when the source is overwhelmed by feedback messages from the receivers. Therefore, in this paper, we only consider the open-loop error-resilient approaches for delay-constrained applications where no feedback is required from the receivers.

Open-loop techniques can be further classified as pre- or post-processing techniques that are applied on the data at the transmitter or the receiver, respectively [13]. In post-processing techniques, the receiver applies algorithms to recover lost parts from the correctly received parts. Such techniques require processing time and computational power at the receiver. Therefore, these techniques are not suitable for delay-constrained 3-D applications when different users with different machine capabilities communicate within a virtual world that has time constraints on user’s actions. Furthermore, post-processing techniques require certain parts of the bitstream to be correctly received in order to recover lost parts. As a result, post-processing techniques perform best when combined with pre-processing techniques. In this paper, we only consider pre-processing techniques.

One pre-processing approach that has been found to be effective in video streaming is to employ forward error correction (FEC) codes to protect the bitstream against packet losses [13]. These methods fall into two classes: equal error protection (EEP) and unequal error protection (UEP). EEP methods apply the same FEC code rate to all parts of the bitstream regardless of the contribution of each part to the decoded model. EEP is effective when the channel has a low packet-loss rate. However, at higher packet-loss rates, important parts of the bitstream might be lost, which results in a considerable degradation in the decoded model quality. In this case, UEP is more suitable
since important parts of the bitstream get higher level of error protection than other parts. The proposed UEP approach optimizes FEC code rates for each layer by considering the channel state as well as the encoded model characteristics. In this case, the server calculates the optimum FEC code rates for a spectrum of packet-loss rates and stores them together with the corresponding model. At runtime, the server selects FEC code rates that correspond to the actual channel. This off-line processing is applicable for most 3-D applications.

In this paper, the encoded bitstream consists of a base mesh and a number of enhancement layers \( (M) \) that refine the base mesh. Each layer, including the base mesh, is assigned an FEC code rate depending on its contribution to the decoded model quality. The distribution of these FEC code rates is based on a statistical distortion measure that quantifies the distortion at the displayed model on the client’s screen. This measure’s arguments are: 1) the 3-D model; 2) the total number of error-protection bits \( (C) \); and 3) the end-to-end channel. Based on this distortion measure, we define a mapping that determines the number of error-protection bits to be assigned for the \( M + 1 \) layers (i.e., \( C^0, \ldots, C^{(M)} \)) such that the distortion at the displayed model is minimum. In order to accommodate for error-protection bits, geometry coordinates are encoded using coarser quantizers.

The FEC codes used in this paper are the Reed-Solomon (RS) codes. These error-control block codes are perfectly suited for error protection against bursty packet losses because they are maximum distance separable codes, i.e., there are no other codes that can reconstruct erased symbols from a smaller number of code symbols [14]. An \( (n,k) \) RS-code defined over the Galois Field \( GF(2^q) \) encodes \( k \) information symbols where each symbol is represented by \( q \) bits. These \( k \) symbols are encoded into a codeword of \( n \) symbols. The codeword length, \( n \), is restricted by \( n \leq 2^q - 1 \). The code rate of this RS-code is \( k/n \), which represents the fraction to which the source data rate has to be reduced in order to maintain a constant total (source and channel) data rate. As soon as \( k \) symbols are received, all lost symbols can be reconstructed.

This paper is organized according to the following plan. Section II summarizes 3-D graphics streaming techniques that have been proposed in the literature. Section III describes the progressive compression methods that have been used to generate a hierarchical bitstream while Section IV illustrates the packetization method. We derive the statistical distortion measure in Section V. The channel model we employ is described in Section VI. Section VII explains the distribution of the error-protection bits among transmitted layers by solving a constrained optimization problem. Experimental results comparing the proposed UEP method with the methods of EEP and no error protection (NEP) are presented in Section VIII. Finally, conclusions are summarized in Section IX.

II. ERROR-RESILIENT 3-D GRAPHICS STREAMING

Error-resilient streaming of 3-D graphics is a new research area and only few researchers have recently tackled the problem. The existing techniques in the literature range from robust source coding to re-transmission based methods. In robust source-coding methods, the model is partitioned into independent trimmed surfaces and each portion is encoded independently from others. Such error-resilient transmission method has been proposed by Bajaj et al. in [15] for compressed Virtual Reality Modeling Language (VRML) files. It is a source-based error-control method, where the encoded bitstream is classified into layers according to the depth-first order of the vertices. As a result, each layer is independent of all other parts of the model and losing any layer will not affect the decoding of other layers. Even though this method adds a level of protection to the transmitted bitstream, it is not scalable with respect to the channel packet-loss rate. For example, at high packet-loss rates, many parts will be lost and the decoded model will suffer from high distortion because the method does not scale up at such high packet-loss rates.

Another robust encoding method has been proposed by Yan et al. in [16] and is based on the constructive traversal compression scheme proposed by Li and Kuo [17]. In this method, a 3-D model is partitioned into several segments and each segment is independently transmitted. These segments are then stitched at the decoder using joint-boundary information. This error-resilient method classifies the joint-boundary information as the most important part of the bitstream. However, the number of error-protection bits assigned to the different parts of the bitstream were experimentally selected (i.e., no analytical approach was provided for error-protection bit-allocation). In contrast, our proposed method assigns these error-protection bits in a statistically optimal manner.

The above two methods in [15] and [16] partition the model into trimmed surfaces at full resolution. As a result, only portion of the model is displayed on the client’s screen first and then more portions are rendered and displayed before the user can identify the model. In contrast, coarse-to-fine representation of the model is more appealing since the user can identify the model from the first received portion, which is in this case the base mesh. The proposed method, in this paper, follows the latter approach, which is more appropriate for delay-constrained applications.

MPEG-4 proposed an error-resilient coding of 3-D models [18] that is similar to the one proposed in [15]. In MPEG-4, the compressed data is partitioned into segments and each segment is encoded and transmitted independently from other segments. As a result, when a segment is lost, it will not affect other segments and hence this method prevents error propagation. Even though this method adds error-resilience to the bitstream, it is not scalable with respect to the channel characteristics.

A re-transmission-based error-resilient method has been proposed by Bischoff and Kobelt in [19] where the receiver exploits the geometry coherence of the received data. The model on the sender side is re-sampled to ensure such coherence between samples. On the client side, the received samples are used to construct an approximation of the original 3-D model. Furthermore, the base mesh is re-transmitted with every enhancement layer to guarantee that it is correctly received at the client. Even though this method provides a reasonable protection level at high packet-loss rates, it is inefficient when the packet-loss rate is low. In other words, it provides the same level of robustness regardless of the channel behavior. In
contrast, our approach is scalable with respect to the channel packet-loss rate.

In applying FEC codes to transmitted 3-D bitstreams, the number of source coding bits needs to be reduced in order to accommodate for the error-protection bits. Such reduction can be achieved by using coarser quantizers or by sending fewer triangles. In this paper, we employ the former approach. The proposed error-resilient algorithm improves upon the above methods since it: 1) does not require any special processing on the client side; 2) is scalable with respect to both channel bandwidth and channel packet-loss rate; 3) applies FEC codes optimally on the transmitted bits; 4) is applicable to any progressive compression method that produces a hierarchical bitstream; and 5) does not require any feedback from the receivers. In this paper, we address the problem of streaming static 3-D models only. The next section explains in detail the progressive compression algorithm that produces a hierarchical bitstream.

III. PROGRESSIVE COMPRESSION OF 3-D MODELS

The recently proposed compressed progressive mesh (CPM) [9] simplifies the model into a base mesh and a number of enhancement batches that transform the base mesh into a number of finer meshes that represent different levels of detail. Even though a specific algorithm is used in this paper to produce the progressive bitstream, the proposed error-resilient streaming method can be applied to any other progressive compression with minor modifications. This section summarizes the CPM algorithm to provide insight on the encoded bitstream content.

CPM applies a well-known simplification method that depends on two operations: edge-collapse and vertex-split [20], [21] that are applied at the encoder and the decoder, respectively. These two operations are illustrated in Fig. 1 while Fig. 2 illustrates the CPM codec structure. The encoding process is iterative. At the beginning of every iteration, a subset of edges is chosen to be collapsed. These edges need to satisfy two conditions to be collapsed within the current iteration. First, in order to uniquely identify a vertex-split operation, at most two vertices are collapsed into one vertex. Second, let edge $e_1$ connect vertices $v_1$ and $v_2$, and similarly let edge $e_2$ connect vertices $u_1$ and $u_2$. If these four vertices form a quadrilateral, then these two edges cannot be collapsed within the same iteration. Such restrictions make the edges being collapsed independent of each other, and hence, the decoding process (vertex-split operation) for a given vertex is independent from others. On the other hand, these restrictions limit the reduction rate between consecutive levels-of-detail. Simulations show a reduction of about 30% in the number of vertices between two consecutive levels-of-detail for a typical model.

Each edge-collapse operation is represented by three fields. These fields contain two categories of information, namely geometry and connectivity information. These fields are: the split-status bit, which is one bit per vertex that specifies whether the vertex should be split or not, cut-edges bits per edge-collapse operation to enable the decoder to identify the two edges (among all edges incident on the vertex) to be split, and a position-correction vector, per edge-collapse operation, which is the difference in geometric coordinates between the split vertex and the corresponding vertex predicted by the prediction algorithm. This difference between coordinates is quantized and entropy coded for efficient representation. All edge-collapse operations are encoded using the above three fields while the base mesh can be compressed using any single-level mesh compression technique that exists in the literature [1]–[7]. If part of the connectivity data is lost, then the decoding process terminates.

At this stage in the transmission system, the encoded bitstream is ready for transmission after the RS codes are appended. The packetization method is illustrated in the next section.

IV. PACKETIZATION

In our work, we adapt a packetization method known as block of packets (BOP) [22], [23]. In this method, the data is placed in horizontal packets and then RS code is vertically applied across BOP packets as illustrated in Fig. 3. Such a method is most appropriate for packet networks where burst errors are common [22], [23]. The packets are transmitted in horizontal order using the user datagram protocol (UDP).

1Requiring a feedback from receivers is not appropriate in several scenarios. For example in multicast applications where many users are connected to the virtual world and requiring feedback from every user causes an implosion problem. Another scenario is when the RTT is large and waiting for feedback from the receiver causes a considerable end-to-end delay.
be the RS code applied to level-
the decoded model is the product
. In equation form, the expected distortion at the
, and
BOPs and their cor-
is the error be-
packets are received, then the decoder will be able
is the probability of
, and
RS code rates. The choice of these RS code
is the probability of having irrecoverable packet
. Then, if
kj
packets are received, then the decoder will be able
to recover all lost packets in this BOP. Otherwise, the decoder
considers these packets as lost and irrecoverable [13]. Since the
progressive compression algorithm produces
M
batches in addi-
tion to the base mesh, there are
M + 1
BOPs and their cor-
responding
M + 1
RS code rates. The choice of these RS code
rates is determined using a statistical distortion measure that is
derived in Section V.2

V. STATISTICAL DISTORTION MEASURE

The proposed UEP method is scalable with respect to varia-
tions in both channel bandwidth and channel error charac-
teristics. Furthermore, this method distributes the error-protection
bits among transmitted layers to maximize the decoded model
quality. To this effect, in this section, we present a theoretical
approach to determine the relative importance of transmitted
layers for a given model and an end-to-end channel. First, we
develop a statistical distortion measure that estimates the dis-
tortion introduced on the displayed model. Then, we minimize
this distortion function with respect to
k0, k1, . . . , kM
(\text{where} \ (n, k_j) \ \text{is the RS code applied to the} \ j^{th} \ \text{layer of the bitstream}).

Generally speaking, model distortion estimation depends
on the decoding strategy. As discussed in Section III, each
batch contains both connectivity and geometry information.
The former specifies the vertices to be split and the edges to
be cut while the latter provides the coordinates of the newly
added vertices. Therefore, although losing part of the geometry
information (but not the connectivity information) affects the
decoded model quality, the decoding process can still continue
because the decoder can keep track of vertices to be split. In con-
trast, if part of the connectivity information is lost, the
decoder will be working on the incorrect set of vertices and
the decoded model features will be different from the original.
Hence, the decoding process terminates whenever lost packets
cannot be recovered.

Following the above standard decoder operation, the expected
distortion at level-
-j
of the decoded model is the product
P_j E_j
given that all coarser layers have been received correctly. \(E_j\)
is the distortion that would result when the decoding stops at
level-
-j
, and \(P_j\) is the probability of having irrecoverable packet
loss at level-
-j
. In equation form, the expected distortion at the
displayed model is written as follows:

\[ D_r = P_0 E_0 + \sum_{j=1}^{M} P_j E_j \prod_{i=0}^{j-1} (1 - P_i) \]  \hspace{1cm} (1)

where \(M\) is the number of batches, \(P_j\) is the probability of having irrecoverable packets at level-
-j
, and \(E_j\) is the error be-
tween the original model and the mesh produced after decoding
the bitstream of level-
-j
.

The error between the original model and any of the meshes
produced by decoding a batch can be found by measuring the
distance between the surfaces of the two meshes. A well-known
distance measure between two sets of points in the 3-D space
is the Hausdorff distance. Fig. 4 depicts the rate-distortion
(RD) curve obtained by measuring the Hausdorff distance
between the transmitted SMALL BUNNY model and each of the
ten levels-of-detail generated by the progressive compression
algorithm.\(^3\) Notably, the Hausdorff distance is an expensive
operation and it requires considerable processing power as well
as computational time. Therefore, we estimate the distortion,
\(E_j\), by using a conservative upper bound on the maximum
distortion that results from edge collapses in a batch, which
might be obtained using the sum of the squares of the distances
between each vertex \(V\) of the simplified mesh and the planes
that support the original triangles that were incident upon all
the vertices that collapsed to \(V\) [24]. The RD curve in this
case is shown in Fig. 5. Even though this RD curve is not identical

\(^2\)In this paper, each packet is 128 Bytes in size.

\(^3\)Every vertex split operation at the decoder reduces the expected error by
certain amount. In these RD curves, we instead show the reduction in the error
resulted by decoding the complete batch. This results in the stair-like shape for
the rate-distortion curves.
to the one obtained using the Hausdorff distance and shown in Fig. 4, they both reflect similar relative importance among transmitted layers. Moreover, the former is obtained at no cost since this quadric error measure is used in choosing the set of edges to be collapsed at each iteration in the encoding process.

$P_j$, in (1), is the probability of having irrecoverable lost packets at level-$j$. $P_j$ computation depends on the $(n,k_j)$ RS-code being used to protect this level as well as the channel model. Since each level is packetized into one BOP as illustrated in Fig. 3, lost packets can be recovered whenever $k_j$ packets are received. Therefore, $P_j$ is the probability of losing more than $n - k_j$ packets and it can be calculated in terms of the block error density function as

$$P_j = \sum_{m=n-k_j+1}^{n} p(m,n)$$

(2)

where $p(m,n)$ is the block error density function, i.e., the probability of losing $m$ symbols within a block of $n$ symbols. Replacing the block error rate (BER), $P_j$, in (1) with the expression given in (2) results in the following distortion function:

$$D_r = \sum_{m=n-k_0+1}^{n} p(m,n) \times E_0 + \sum_{j=1}^{M} E_j \times \sum_{m=n-k_j+1}^{n} p(m,n) \times \prod_{l=0}^{j-1} \left(1 - \sum_{m=n-k_{l+1}}^{n} p(m,n)\right).$$

(3)

So far, the only quantity in (3) that has not yet been calculated is the block error density function, $p(m,n)$, which depends on the channel model. In this paper, we employed the Gilbert-Elliot model, which is described in Section VI.

After formulating the expected distortion introduced at the decoded model, we need to optimize the function given in (3) to maximize the decoded model quality. The setup of the optimization problem and its solution are presented in Section VII.

VI. CHANNEL MODEL

The underlying processes that lead to packet losses in networks are quite complex. However, even a simple and analytically tractable Markov model with only two states approximates network’s packet loss behavior fairly well for our purpose. The two-state Markovian Gilbert–Elliot model has been thoroughly investigated in the literature [22], [23]. This model is illustrated in Fig. 6. Here, we will briefly re-iterate the main results. Of particular interest within the scope of this paper is the probability of having $m$ errors in $n$ symbols when an RS code defined over $GF(2^t)$ with rate $k/n$ is used. The G-E model possesses a characteristic distribution of error-free intervals, which are also called gaps. To be precise, the gap is defined as the interval of length $v - 1$ packets between two consecutive erroneously received packets. Let $g(v)$ be the probability of having a gap of $v$ received packets between two lost packets. Then, $g(v)$ is given by

$$g(v) = \begin{cases} 1 - p_{BG}, & v = 1 \\ p_{BG}(1 - p_{GB})^{v-2} p_{GB}, & v > 1 \end{cases}$$

(4)

where $p_{BG}$ represents the transition probability from the bad state to the good state while $p_{GB}$ represents the transition probability from the good state to the bad state.

Similarly, let $G(v)$ be the probability of having a gap greater than $v - 1$ packets. Then, $G(v)$ is given by

$$G(v) = \begin{cases} 1, & v = 1 \\ p_{BG}(1 - p_{GB})^{v-2}, & v > 1. \end{cases}$$

(5)

Furthermore, let $R(m,n)$ denote the probability of having $m - 1$ packet losses within the $n - 1$ packets following a lost packet. Then $R(m,n)$ is given by

$$R(m,n) = \begin{cases} G(n), & m = 1 \\ \sum_{i=1}^{n-m+1} g(v)R(m - 1, n - v), & 2 \leq m \leq n. \end{cases}$$

(6)

Finally, the probability of losing $m$ symbols, each of which is of $q$ bits in length, within a block of $n$ symbols can be written as

$$p(m,n) = \begin{cases} \sum_{i=1}^{n-m+1} P_{LR}G(v)R(m, n-v+1), & 1 \leq m \leq n \\ 1 - \sum_{i=1}^{n} p(v,n), & m = 0 \end{cases}$$

(7)

where $P_{LR}$ is the packet-loss rate.

VII. UNEQUAL ERROR PROTECTION

Equation (3) estimates the expected distortion introduced on the decoded model in a statistical sense. Now, the objective is to minimize this distortion with respect to the set of $k_j$’s in (3). Intuitively, the base mesh is usually regarded as the most important layer in the encoded bitstream, followed by the coarsest batch, and so on, until the finest batch because losing a certain layer will result in terminating the decoding process and making all consecutive batches undecodable. Therefore, we expect the optimization process to allocate more error-protection bits to the base mesh and the first few coarse layers when the packet-loss rate is high.

The $M + 1$ quantities, $k_j$, in (3) must satisfy two main conditions. First, the error-protection bit-budget is upper-bounded by $C$, the number of available error-protection bits. The second constraint is that $k_j$ cannot be greater than $n$, which is a natural constraint for any error-control block code. Combining (3)
and (7) together with the above two conditions results in a constrained optimization problem given as follows:

\[
\min_{k_0,k_1,\ldots,k_M} \arg : \quad \sum_{m=n-k_0+1}^{n} p(m,n) \times E_0 + \sum_{j=1}^{M} E_j \times \sum_{m=n-k_j+1}^{n} p(m,n) \\
\times \prod_{i=0}^{j-1} \left( 1 - \sum_{m=n-k_{i}+1}^{n} p(m,n) \right) \tag{8}
\]

subject to

\[
\sum_{j=0}^{M} \frac{S_j(n-k_j)}{k_j} = C \quad 0 \leq k_j \leq n \quad j = 0, \ldots, M \tag{9}
\]

where \(S_j\) is the size of level-\(j\) in bits, and \(C\) is the total number of error-protection bits, i.e., \(C = \sum_{j=0}^{M} S_j(j)\).

The first constraint in (9) forces the solution to be scalable with respect to the error-protection bit-budget, i.e., with respect to the channel bandwidth. Similarly, incorporating the block error density function, \(p(m,n)\), in (8) forces the solution to be scalable with respect to the channel packet-loss rate.

The above nonlinear constrained optimization problem is difficult to solve analytically because the arguments, \(k_j\)'s, exist in the summation limits. Furthermore, this constrained-optimization problem involves recursion in calculating \(p(m,n)\). Therefore, we concentrate on solving this problem numerically. Such a constrained-optimization problem can be solved using a transformation into an unconstrained problem and zero-order search methods [25]–[27]. However, since \(k_j\)’s take discrete number of integer values, solving the constrained-optimization problem by full-search is feasible. Nevertheless, the computational time can be reduced by using the prior-knowledge that assigning more error-protection bits to coarser layers reduce the expected distortion on the decoded model, \(D_r\). Therefore, given an error-protection bit-budget and a channel with a certain packet-loss rate, the steps of solving the constrained-optimization problem in (8) and (9) are as follows.

1) Fix \(n\), the total number of encoded symbols. Choosing \(n\) is a design issue. In this paper, we choose \(n\) to be 100 and each symbol is represented with 8 bits, i.e., the used RS codes are defined over \(GF(2^8)\).

2) Distribute the error-protection bits, \(C\), among the transmitted \(M + 1\) layers such that \(k_0 = k_1 = \ldots = k_M\); compute the corresponding expected distortion, \(D_r\).

3) Re-distribute the error-protection bits among the transmitted layers such that \(k_0 \leq k_1 \leq \ldots \leq k_M\). Iterate all possibilities and at each iteration, compute the expected distortion, \(D_r\), from (8).

4) Choose the minimum \(D_r\) and the corresponding set of \(k_j\)'s as the optimum solution.

It is anticipated that for an error-free channel, the solution of this optimization problem produces equal \(k_j\)'s because all layers have equal importance. This does not necessarily require a unique solution of the optimization problem. On the other hand, at a high packet-loss rate channel, the base mesh and the coarse layers consume most of the available error-protection bits while other layers get a very small portion of it, if any. The above optimization problem has been solved for a number of models. Tables I and II list the optimal RS codes for the two models SMALL BUNNY and TRICERATOPS, respectively, for two

<table>
<thead>
<tr>
<th>(P_{LR})</th>
<th>base mesh</th>
<th>batch 1</th>
<th>batch 2</th>
<th>batch 3</th>
<th>batch 4</th>
<th>batch 5</th>
<th>batch 6</th>
<th>batch 7</th>
<th>batch 8</th>
<th>batch 9</th>
<th>batch 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
<td>0.85</td>
</tr>
<tr>
<td>04</td>
<td>0.75</td>
<td>0.76</td>
<td>0.77</td>
<td>0.79</td>
<td>0.81</td>
<td>0.83</td>
<td>0.85</td>
<td>0.86</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
</tr>
<tr>
<td>08</td>
<td>0.65</td>
<td>0.70</td>
<td>0.81</td>
<td>0.83</td>
<td>0.85</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td>0.93</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>12</td>
<td>0.55</td>
<td>0.67</td>
<td>0.70</td>
<td>0.83</td>
<td>0.85</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td>0.93</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>18</td>
<td>0.55</td>
<td>0.65</td>
<td>0.81</td>
<td>0.83</td>
<td>0.87</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td>0.93</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>25</td>
<td>0.55</td>
<td>0.61</td>
<td>0.83</td>
<td>0.85</td>
<td>0.87</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td>0.93</td>
<td>0.95</td>
<td>0.97</td>
</tr>
<tr>
<td>40</td>
<td>0.50</td>
<td>0.61</td>
<td>0.73</td>
<td>0.80</td>
<td>0.87</td>
<td>0.89</td>
<td>0.91</td>
<td>0.93</td>
<td>0.95</td>
<td>0.97</td>
<td>1.00</td>
</tr>
</tbody>
</table>

(a)
error-protection bit-budgets. As can be seen from this data, when the channel is error free, all layers are assigned the same error-protection level. On the other hand, at high packet-loss rate, coarse layers consume most error-protection bit-budget while fine layers are left unprotected. In summary, the solution of the constrained optimization problem is scalable with respect to channel error characteristics. Similar conclusions are derived when the error-protection bit-budget is increased as in Tables I(b) and II(b), where the base mesh and the coarse layers consume all error-protection bits. The following section discusses the experimental results in detail.

VIII. SIMULATION RESULTS

To demonstrate the efficacy of the proposed error-resilient transmission method, we used both subjective and objective methods of comparison. In particular, we used the Hausdorff distance between densely sampled points on the original (transmitted model) and the displayed model on the client’s screen as an objective comparison measure.

In the following experiments, the error-protection bit-budget (i.e., \( C \)) is assumed to be given. In order to keep the total bit-budget fixed, we reduce the number of source coding bits using coarser quantizers for geometry precision. As an example, the original model will be quantized using a very fine quantization step size (in this case, 14-bit quantizer). For a given \( C \), we re-quantize the model such as the reduction in the number of source coding bits is equal to \( C \). In these experiments for the SMALL BUNNY model, we used 8-bit and 10-bit quantizers for \( C = 25920 \) bits and 12 400 bits, respectively. We compare the performance of the proposed UEP algorithm with the EEP algorithm. In EEP, all RS codes that are assigned to transmitted layers are the same, i.e., \( k_0 = k_1 = \ldots = k_M \). We further compare the proposed UEP algorithm with the case of not applying any error protection (NEP). In all these three approaches, total number of bits spent on source and channel coding is the same. Also, all models are compressed into ten levels (i.e., \( M = 10 \)). In all experiments, \( (n_t,k) \) RS codes with \( n_t = 100 \) have been employed and the average burst length, \( L_B \), is set to 5. Moreover, for each point, we run the experiment 150 times and we compute the average. We applied the proposed method on several 3-D models and in this section we summarize the results for the SMALL BUNNY and the TRICERATOPS models that have 9580 and 5660 triangles, respectively.

A. Objective Comparison

Fig. 7(a) depicts the error between the transmitted and the decoded models as a function of the packet-loss rate \( P_{LR} \) for the SMALL BUNNY model.\(^4\) Three curves in this figure represent the cases of NEP, EEP; and UEP.\(^5\) The error-protection bit-budget is \( C = 12 400 \) bits, corresponding to a channel rate of 85/100 in the EEP approach. As shown in Fig. 7(a), when the packet-loss rate is less than 2%, NEP performs better than both EEP and UEP because it uses more bits for source coding, and the error-protection bits (used in EEP and UEP) are largely redundant at such low packet-loss rates. In this figure, EEP uses the RS-code (100,85) to protect all layers. For the case of UEP, the RS codes that are assigned for each layer are tabulated in Table I(a). These RS code rates are calculated by solving the

\(^4\)We are using a quadric error measure at the encoder to estimate the error introduced by collapsing an edge. Nevertheless, we use the Hausdorff distance to compare the proposed method with other streaming techniques because the Hausdorff distance measures the maximum error between two meshes.

\(^5\)Even though \( P_{LR} \) of 40% are not common, we are interested in demonstrating the results at such extreme packet-loss rates.
constrained optimization problem given in (8) and (9). Note that as the packet-loss rate increases, more powerful RS codes are assigned to the coarse layers while less powerful RS codes are assigned to finer levels.

Comparing the three curves in Fig. 7(a), we can conclude that the UEP method outperforms both NEP and EEP when the packet-loss rate is higher than 2%. Moreover, using the proposed method, the quality of the decoded model degrades more gracefully as the packet-loss rate increases. As expected, at low packet-loss rates (less than 4%), the performances of both EEP and UEP methods are close to each other since the RS code is capable of recovering lost packets in both cases. On the other hand, at higher packet-loss rates, the base mesh in the EEP method is subject to packet losses that cannot be recovered by the RS code. Nevertheless, in the UEP case, these lost packets can still be recovered since the error-protection level that is associated with the base mesh is significantly higher than the corresponding one in the EEP case. This results in a better performance of the UEP method compared to the EEP method as the packet-loss rate increases.

Similarly, Fig. 7(b) depicts the case when the error-protection bit-budget is 25 920 bits, which is higher than the case in Fig. 7(a). In both cases, the same number of source coding bits is used. In this case, the corresponding EEP RS code rate is 70/100 while the optimal RS code rates assigned for each layer in the UEP approach are tabulated in Table I(b). In this case, UEP method outperforms NEP when the packet-loss rate is higher than 1% while it outperforms EEP when the packet-loss rate is higher than 4% since in both cases there is enough error-protection bits to recover lost packets. Fig. 7(c) depicts the error between the original and the decoded models for the UEP case when two error-protection bit-budgets are used. From this plot, it is evident that the proposed method is scalable with respect to the bit-budget. Note that when the channel is error-free, both UEP methods have the same distortion level. This is because the same number of source coding bits is used in both cases while in one case the error-protection bit-budget is higher than the other and therefore when the packet-loss rate increases the former outperforms the latter.

The same experiments were repeated for the TRICERATOPS model and the error between the transmitted and the decoded SMALL BUNNY models when the error-protection bit-budget is (a) 12 400 bits and (b) 25 920 bits, respectively. Curves in (c) compare the performance of the proposed UEP method for two error-protection bit-budgets. Number of iterations is 150.

Fig. 8. Maximum error (Hausdorff distance) between the transmitted and the decoded TRICERATOPS models when the error-protection bit-budget is (a) 5867 bits and (b) 8864 bits, respectively. Curves in (c) compares the performance of the proposed UEP method for two error-protection bit-budgets. Number of iterations is 150.
increased to 8864 bits, UEP outperforms NEP when the packet-loss rate is higher than 1% while it outperforms EEP when the packet-loss rate is higher than 4% as shown in Fig. 8(b). The performance of UEP for two different error-protection bit-budgets is depicted in Fig. 8(c). The optimal RS code rates assigned for each layer in the UEP method are tabulated in Table II(a) and (b).

B. Subjective Comparison

We also compare the three methods, NEP, EEP, and UEP, subjectively. Fig. 9 shows the experimental results for the SMALL BUNNY model when the error-protection bit-budget is 12400 bits. The caption under every image gives the error-protection method and the channel packet-loss rate. (a) Original SMALL BUNNY model. (b) NEP - $P_{LR}=00\%$. (c) EEP - $P_{LR}=00\%$. (d) UEP - $P_{LR}=00\%$. (e) NEP - $P_{LR}=04\%$. (f) EEP - $P_{LR}=04\%$. (g) UEP - $P_{LR}=04\%$. (h) NEP - $P_{LR}=12\%$. (i) EEP - $P_{LR}=12\%$. (j) UEP - $P_{LR}=12\%$. (k) NEP - $P_{LR}=25\%$. (l) EEP - $P_{LR}=25\%$. (m) UEP - $P_{LR}=25\%$.

Fig. 9. Subjective results of applying NEP, EEP, and UEP methods on the SMALL BUNNY model when the error-protection bit-budget is 12400 bits. The caption under every image gives the error-protection method and the channel packet-loss rate. (a) Original SMALL BUNNY model. (b) NEP - $P_{LR}=00\%$. (c) EEP - $P_{LR}=00\%$. (d) UEP - $P_{LR}=00\%$. (e) NEP - $P_{LR}=04\%$. (f) EEP - $P_{LR}=04\%$. (g) UEP - $P_{LR}=04\%$. (h) NEP - $P_{LR}=12\%$. (i) EEP - $P_{LR}=12\%$. (j) UEP - $P_{LR}=12\%$. (k) NEP - $P_{LR}=25\%$. (l) EEP - $P_{LR}=25\%$. (m) UEP - $P_{LR}=25\%$.

bits. The model in Fig. 9(a) is the original model. The first column on the left shows the decoded models in the NEP case for different packet-loss rates. Similarly, the second and the third columns show the decoded models in the EEP and the UEP cases, respectively. As shown, UEP keeps a reasonable decoded model quality level as the packet-loss rate increases. Similarly, running the same experiments for the TRICERATOPS model produced similar results as shown in Fig. 10. Notice that at high packet-loss rate, the displayed TRICERATOPS model lost parts of its head in the NEP and the EEP cases while the whole base mesh and one batch are preserved in the UEP case.

These results reflect the fact that using the proposed UEP method, the quality of the decoded model degrades more gracefully as the packet-loss rate increases. Furthermore, the proposed UEP method maximizes the decoded model quality for: 1) a given model that has been progressively compressed into $M$ batches; 2) a certain bit budget ($C$ bits) that is reserved for error protection; and 3) the channel error characteristics. However, this improvement in the system performance has the cost of increasing complexity because of the need to solve the derived constraint optimization problem.
IX. CONCLUSIONS

In this paper, we presented an error-resilient method for 3-D models transmission. The proposed method is scalable with respect to both the channel bandwidth and the channel error characteristics. The proposed system consists of a 3-D progressive model codec, a single-level 3-D model codec, a RS-code encoder/decoder and a bit-rate allocation algorithm. The bit-budget allocation method assigns optimal RS-code rates for different layers in the encoded bitstream to maximize the decoded model quality. These optimal RS codes depend on: the error-protection bit-budget, the channel packet-loss rate, and batch-by-batch rate-distortion characteristics of the source model. Experimental results show that with our UEP approach, the quality of the decoded model degrades more gracefully as the packet-loss rate increases.

In the following, we will comment on few features of the proposed system for 3-D model transmission.

- The applicability of the proposed UEP method does not depend on a particular 3-D model compression algorithm, although we used a progressive compression that employs edge-collapse and vertex-split operations in this paper. It is applicable whenever the proposed statistical distortion measure given in (1) is calculable. Thus, the proposed framework can be used with other progressive 3-D model compression methods as well.

- The applicability of the proposed UEP method does not depend on a particular channel model, although we used a two-state markovian model in this paper. The proposed method is applicable to any channel model where $P_j$’s in (1) can be calculated.

- Generally, the proposed UEP method outperforms the other two methods (i.e., NEP and EEP) for all packet-loss rates higher than $r\%$, where $r$ is a function of the error-protection bit budget, $C$, and the source model characteristics. This is due to the fact that our distortion measure produces optimal distribution of the error-protection bit-budget by incorporating both source and channel characteristics into the distortion function.

- The proposed 3-D model transmission system is more complicated than a standard system with NEP since for every 3-D model to be transmitted, the constrained optimization problem given in (8) and (9) needs to be solved for a spectrum of packet-loss rates. This complexity varies according to the numerical algorithm used to solve this nonlinear optimization problem. Nevertheless, this increase in complexity is compensated by the significant improvements in the decoded model quality as depicted in Figs. 7 to 10. Furthermore, in many cases, this processing can be performed off-line. Then, the plots in Figs. 7 and 8 are stored at the server together with the progressive bitstream of the corresponding model. Such extra data is only 0.6% and 1.1% of the total stored data for the SMALL BUNNY and the TRICERATOPS models, respectively.

- In this work, the error-protection bit-budget, $C$, at a particular packet-loss rate is assumed to be given. That is, the bit-allocation between the source and channel coding is already determined. However, this joint source and channel bit allocation can best be determined using a rate-distortion analysis. That is, rather than $C$, the total number of bits is specified, and the optimization algorithm determines the percentage of these bits to be spent on error-protection in addition to the distribution of these bits among transmitted layers. This extension is part of our future study.

REFERENCES


Ghassan AlRegib (M’97) received the Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology (Georgia Tech), Savannah, in 2003. His Ph.D. dissertation focused on developing error-resilient techniques to stream three-dimensional (3-D) graphics over lossy channels. He is an Assistant Professor at the School of Electrical and Computer Engineering, Georgia Tech. He joined the faculty at Georgia Tech in August 2003, where he is currently working on projects related to multimedia communications, video and 3-D graphics streaming, shared reality, multimodal communications, and distributed sensor systems.

Dr. Al-Regib is the recipient of the Outstanding Graduate Teaching Award in 2000–2001 at the School of Electrical and Computer Engineering at Georgia Tech, the Center for Signal and Image Processing Research Award in spring 2003 at Georgia Tech, and the Center for Signal and Image Processing Service Award in spring 2003 at Georgia Tech. He will serve as the Special Sessions Chair at ICIP’06.

Yucel Altunbasak (M’97–SM’02) received the M.S. and Ph.D. degrees from the University of Rochester, Rochester, NY, in 1993 and 1996, respectively.

He joined the School of Electrical and Computer Engineering, Georgia Institute of Technology (Georgia Tech), Atlanta, in 1999, where he is currently an Assistant Professor. He is currently working on industrial- and government-sponsored projects related to multimedia networking, wireless video, video coding, genomics signal processing, and such inverse imaging problems as super-resolution and demosaicking. He was previously with Hewlett-Packard Research Laboratories, Stanford University, Stanford, CA, and San Jose State University, San Jose, CA, as a Consulting Assistant Professor. His research efforts have resulted in over 100 peer-reviewed publications and 15 patents/patent applications. Some of his inventions have been licensed by the Office of Technology Licensing at Georgia Tech.

Dr. Altunbasak is an Associate Editor for the IEEE TRANSACTIONS ON IMAGE PROCESSING, IEEE TRANSACTIONS ON SIGNAL PROCESSING, Signal Processing: Image Communications, and for the Journal of Circuits, Systems and Signal Processing. He served as the lead Guest Editor for the Special Issue on Wireless Video of Image Communications. He serves as the Vice President for the IEEE Communications Society Multimedia Communications Technical Committee. He was elected to the IEEE Signal Processing Society IMDSP Technical Committee. He served as a Co-Chair for “Advanced Signal Processing for Communications” Symposia at ICC’03, as a Track Chair at ICME’03 and ICME’04, as a Panel Sessions Chair at ITRE’03, as a Session Chair at various international conferences, and as a Panel Reviewer for government funding agencies. He will serve as the Technical Program Chair for ICIP-2006. He is a co-author for a conference paper that received the Best Student Paper Award at ICIP’03. He received the National Science Foundation (NSF) CAREER Award in 2002 and the 2003 Outstanding Junior Faculty Award from the School of Electrical and Computer Engineering.

Jarek Rossignac is a Professor of Computing and Chair of IRIS at the Georgia Institute of Technology, Atlanta, GA. His Topological Surgery is the core of the three-dimensional (3-D) compression standard in MPEG-4. He has published 30 papers, given 14 keynotes, and delivered six courses at SIGGRAPH and at many other conferences on the compression and progressive transmission of surfaces, volumes, and animations. He has authored over 100 papers and 17 patents, received 12 awards, chaired 18 conferences, served on boards of seven journals, edited eight special issues, and served on 51 program committees.

Dr. Rossignac is a member of ACM/SIGGRAPH and a Fellow of the Eurographics Association.