Abstract—In this paper, we propose a visual important-driven interactive rendering method for 3D model over 802.11 WLAN for overcoming the shortcomings of wireless network’s narrow bandwidth, high transmission error rates and mobile devices’ low power supply. This paper first proposes an efficient simplification method based on an improved visual important region detection technique. Then, we develop an efficient hybrid FEC and MAC-Lite model transmission protocol which will transmit the model data by their importance respectively. Finally, we propose a real-time interactive rendering method by an efficient model coding. Experimental results demonstrate that we can obtain better rendering result among lossy environment and gain real-time interactive rendering result.

Index Terms—Visual detection, Model Simplification, FEC, MAC_Lite, interactive rendering

I. INTRODUCTION

With both the mature of the mobile network infrastructure and wide use of mobile handheld devices, 3D applications based on mobile devices among wireless network have got rapid development. However, the shortcomings of narrow bandwidth, high transmission error rates among wireless network and the limitations of limited power supply and low computing performance for mobile devices cannot meet the requirements for the real-time interactive rendering of the 3D model on mobile devices.

Recently, the technologies for model transmission and interactive rendering have been received more attentions from researchers. The typical technique is the progressive coding and transmission method [1] for 3D model that can transmit the model data on demand of the user’s quality requirement for 3D model. Usually, the model should be simplified into progressive model. Thus, how to simplify this model efficient is more important. Now, researchers are paid more attention to simplify the model from the aspect of visual optimization.

As we knew, lossy wireless network is considerably different from wired networks. The transmission over lossy wireless links stays challenging due to narrow bandwidth, fading and obstacles which result in high transmission error rates.

To address these problems, retransmission is scheduled. Evidently, this retransmission mechanism affects the network’s throughput and end-to-end delay badly. Factually, most dropped packets are caused by bit-errors in the frame during transmission in the wireless network and a packet with this kind of errors can still be utilized for some error-tolerant transmission such as audio, video and graphics. In this regard, a new transport layer protocol called UDP Lite[2] which is tailored for real-time applications over error-prone networks has been proposed. Unlike UDP, UDP Lite allows for partial checksums that only covers part of a datagram, and will therefore deliver packets that have been partially corrupted. However, in WLAN, lots of corrupted packets are discarded in the MAC layer without reaching UDP layer. The CRC in the MAC layer also should be altered to allow corrupted frames being passed to higher layers, which is known as Mac-Lite [3]. Similar to UDP-Lite, the coverage of Mac-Lite's checksum can be set freely.

What’s more, the key problem of interactive rendering for 3D model on mobile device is how to decrease the transmitted data during the transmission and lower the computing and rendering task in mobile device.

In this paper, we propose a simplification method based on an improved salient detection method. In the implementation, we present some optimization techniques to accelerate the progressive mode reconstruction. Then, we propose an efficient hybrid FEC and MAC-Lite model transmission protocol which will transmit the important graphics data and less important data by modified protocol based on FEC and sensitive data dependent Mac-Lite respectively. Finally, a real time transmission and interactive rendering method by an effective model coding method is proposed. In this method, the mobile client only executes the low-lever rendering operation such that shortens the waiting time before the rendering.

The rest of the paper is organized as follows. Section II is the related work. Section III describes perceptually-based progressive model construction method. Section IV
designs a hybrid transmission protocol. Section V explains the progressive transmission and interactive real time rendering method. The experimental result is shown in section VI. Finally, we summarize our work in section VII.

II. RELATED WORK

A. Simplification and Visual important Computation

Researchers have presented many simplified methods which aim to decrease the visual quality differences of the simplified model and source model. The typical method are QEM[4] and improved QEM[5]. However, those methods focus only on the geometry features and the visually important area cannot be preserved longer while simplifying.

In order to solve this problem, the simplified methods from the aspect of visual optimization are proposed. Lindstrom [6] proposed the simplified method based on the CSF model. Luebke and Hallen[7] proposed a method that employed the visual psychology model to control the 3D model simplification procedure. Qu [8] etc. al proposed a visual mask computing method, which would direct the simplifying for the textured 3D model. Unfortunately, above methods did not consider the topology information of the model itself.

As we knew, the idea of salient region has been developed to help identify distinct spatial regions from their surrounding neighborhoods. Also, the saliency technique has been applied to 3D models. In general, the detection of 3D salient regions can be treated as an extension of identifying salient regions on a 2D image. Based on the model developed for 2D image [9], Lee et al. [10] proposed the idea of mesh saliency as a measure of regional importance for 3D models based on the center-surround mechanism and feature integration theory [11]. This method focuses only on the curvature of each vertex. Gal et al. [12], however, computed a salient region based not only on the curvature, but also the variance of curvature, the number of curvature changes and the size relative to the whole object. By these saliency methods, they achieved better simplification result.

B. UDP_Lite and MAC_Lite Transmission Protocol

Recently, there are lots of literatures about UDP Lite or Mac-Lite applications. In Ref.[13-17], UDP Lite is deployed to transmit multimedia data. Errors in the sensitive part of a multimedia packet should result in dropped packets, while errors in the insensitive part are forwarded to application layer. To allow packets containing errors to be forwarded to the UDP layer, the 802.11 MAC level errors checking feature is completely disabled. Regarding to WLAN, however, the MAC level checksum cannot be completely disabled due to the high bit error rates during transmission. Moreover, the MAC layer plays much important role than UDP layer because the data can be forwarded to the destination by the MAC protocol even without UDP protocol in WLAN.

Mac-Lite is used to transmit voice in WLAN [18]. The checksum only covers headers data such as MAC header, IP header and UDP header, but for voice data, no checksum is applied on it. The experiments results show that compared with the original CRC checking scheme, better performance of networks is achieved. In [19], the authors use different coverage of MAC layer's checksum to transmit speech and compare their experimental results. For video transmission, video coding technology is adopted to divide the video data into different parts according to their different importance and then use Mac-Lite to transmit it [20]. In order to transmit data correctly and fast by Mac-Lite, the forward error-correcting (FEC) technology is used [21]. If the partial checksum detects errors in important data such MAC header data, no retransmission but correcting it instead.

However, there is no discussion in literatures about adopting Mac-Lite or its modified version to transmit mesh of 3D model in WLAN.

C. Streaming and Interactive Rendering

Different from the desktop pc device over wired network, the main shortcoming for the mobile device is the limited power supply and computing ability. Therefore, the key problem of 3D model rendering on mobile device is how to decrease the transmitted data from the server to client and lower the computing and rendering task in mobile device. Luo[22] proposed the progressive transmission and model simplification methods for mobile device. However, these methods needs local reconstruction operations which will take up lots of computing costs and cause the rendering delay at the client. Actually, this method is not good for mobile device. Thus, we can translate the 3D model into image or video and adopt the successful image or video coding technique to transmit the model data. For example, reference [23] and [24] respectively proposed the MPEG-4 coding and JPEG 2000 coding methods to transmit the 3D model. Unfortunately, these methods are not suitable for the application of 3D model representation in mobile e-commerce because these method can not obtain the whole 3D model data but the static images.

III. PERCEUTALLY-BASED PROGRESSIVE MODEL CONSTRUCTION

Loosely speaking, a salient region of a model is the area that is distinct from its surroundings. In this paper, we propose a saliency computation method to effectively obtain salient regions of a model. Similar to [10], the saliency map is created by center-surround mechanism. Usually, center-surround differences are calculated as an across-scale difference between coarse and fine scales. For each scale, a filter window to include neighbouring vertices samples should be designed.

The implementation of our saliency computation method is depicted as follows:

**Step 1:** Compute the mean curvature $MC$ at each vertex $v_i$ ($i = 1...n$, $n$ is the number of vertices of the mesh).
Step 2: Define the local filter window for vertex \( v_i \) and choose its neighboring vertex set \( NS(v_i) \).

Step 3: According to \( NS(v_i) \), calculate the Gaussian weighted average \( GW(v_i) \) at different scales.

Step 4: Get the difference \( GW_m(n) \) between the two scales \( m \) and \( n \) for \( v_i \) and then compute the geometry feature map \( G_i(m, n) \).

Step 5: Make use of the non-linear suppression operator to combine the feature maps \( G_i(m, n) \) into the final geometry saliency map \( GF \).

In step 1, we use the method proposed in [15] to get the \( MC \). Then, we utilize the local filter design method to acquire \( NS(v_i) \).

Given the \( GW(V_i) \) of each vertex and radius \( r \), its Gaussian-weighted average is

\[
GW(V_i, r) = \frac{\sum_{ij} MC(x) \exp \left[ -\frac{\|v_i - v_j\|^2}{2r^2} \right]}{\sum_{ij} \exp \left[ -\frac{\|v_i - v_j\|^2}{2r^2} \right]}
\]

(1)

Then, each feature map \( G_i(m, n) \) is calculated as:

\[
G_i(m, n) = GW(V_i, r_m) - GW(V_i, r_n)
\]

(2)

Finally, those feature maps will be combined into one geometry map by the nonlinear suppression operator. We improve the method proposed in [10] by not only acquiring the block salient region but also the details, such as the exact boundary of the salient region. In our case, we take the mean curvature of each vertex into consideration while combining the above four scales into the final salient region. We also adjust the weight \( \alpha \) and \( \beta_i \) to get the final geometry feature map using the following formula.

\[
GF(V_i) = \alpha N(MC) + \sum_{i=1}^{4} \beta N(G_i)
\]

(3)

To preserve the visually important vertices longer, we will adopt above salient detection method. By this method, we can get the salient importance values \( S(V_i) \) of each vertex. We have modified the QSlim algorithm [2] by weighting the quadrics with mesh saliency.

After the creation of simplification metric, the collapsed queue(CQ) is initialized and the collapsed operations are executed for importing the new vertices and edge pairs(EP). Thus, we can build the full collapsed queue namely the vertex split list. Meanwhile, two data structures of the collapsed record stack and split record stack will be introduced to meet the needs of interactive rendering. In the following, we explain the optimization tactics in the implementation.

1) Initialization of the CQ. According to \( w(v_i) \), the suitable EPs are chosen and the CQ is built. Usually, the CQ adopts the heap data structure. This structure is simple from logical. However, experimental result demonstrates this method is slowly while lots of EPs are appeared. In this paper, we will adopt the dynamical array structure. Different from the heap data structure, the sorting operation is executed after all the insertion operations. By this method, the whole sorting time is saved.

2) Executing the collapsed operation and building the final CQ. From the initial CQ, we can find the collapsed edge(CE) with the smallest value of \( w(v_i) \) and generate the new vertices and EP. Clearly, the new vertices and EP will effect the sorting operation of this dynamical array. In our implementation, the CQ will not be sorted immediately. Factually, these new vertices and EP will effect the sorting of dynamic array. Thus, we do not carry out the sorting for the CQ immediately but pushes these collapsed vertices into CQ and sorts the dynamic array again. Experimental result shows that this method will improve the collapse speed 30%-40% and does not affect the model’s simplification result.

3) Introducing of collapsed record stack (CRS) and split record stack (SRS). In order to achieve the interactive rendering, the server should provide the function of switch between different resolutions of model quickly. However, the existed simplification method [1] will consume a great deal of collapse and split operations while switching from one resolution to another resolution. Therefore, our method presents the CRS and SRS data structure, which will record each collapsed and split operation and push them into the CRS and SRS while executing the simplification operation. While the model needs the switch between different resolutions, we just fetch these records from the CRS or SRS and execute the corresponding rendering operation.

IV. TRANSMISSION PROTOCOL DESIGN

The basic idea of our modified protocol is to formulate the transmission protocol according to the different importance of the 3D model. Progressive Mesh, as a good solution to the transmission of 3D models over network, is represented by a base mesh \( M0 \) followed by an ordered list of vertex split (VSplit), which is in the form of \{M0, {VSplit 1, VSplit 2, . . . , VSplitn}\}. There exists dependency relationship among these VSplit operations. In practice, these VSplitS will be packed into packets for transmission over networks. Hence, these packets also have dependency relationship. Consequently, VSplitS could not be rendered unless their dependent VSplitS arrived at the client. If some of the received packets are dependent on the lost packet, the client will endure a rendering delay since the lost packet retransmission will be invoked. On the contrary, if no or just a small number of the received packets are lost, the client will endure a rendering delay since the lost packet retransmission will be invoked.
packets are dependent on the lost packets, the client could render more vertices at a given period of time and the delay will be reduced.

Thus, in our past work [29], we have presented a novel packetization scheme that is to decrease the dependencies among packets. In this packetization method, two steps will be performed. First, a Non-Redundant Directed Acyclic Graph (NR-DAG) will be constructed to encode all the necessary dependencies among the VSplit operations. Second, a Global Graph Equipartition Packing Algorithm (GGEPA) is applied to minimizing the dependencies among different partitions while separating the whole dependency DAG into k equal size partitions.

Though this method can decrease the dependencies among these packets, the dependencies are still existed. If the dependencies between one packet with other packets are higher, more VSplits, which are included in the dependent packets, should wait this packet be arrived at the client side. Thus, we here regard this packet are rendering-important packet. As we knew, if the VSplits belongs to the base mesh or upper levels, the packets that contain these VSplits are also rendering important packets. Unfortunately, these packets maybe not have many packets that dependent on them. To assign this kind of packets those have many dependent packets and the packets those have in the upper level of our model, we will deal with them in a unified way.

In our GGEPA, we will record each packet’s dependencies noted as PD. As we knew, the NR-DAG we built is a graph. We will translate them into a tree structure thus all nodes will have been arranged as level by level. Manifestly, the nodes in the upper level are the parents of the lower level’s nodes. To assign each packet with a rendering important (RI) value, we browse this tree level by level with depth-first visiting method and calculate each packet’s RI value. While finishing this depth-first visiting, we can obtain each packet’s RI. However, this method only records the RI between neighboring levels. To obtain the RI among all the levels, we should add all the children’s RI into their parents. Now, we can give each node with an accurate RI value.

While the packet is packed into frames, we will assign them with the perceptually importance value RI. It means if the frame’s RI is high, it is important data. According to the different importance of the frame, the MAC layer uses two different ways, MAC-FEC and MAC-Lite protocol, to transmit them respectively. The details of both methods are as follows.

a) **MAC-FEC**. For visual important data, to ensure the data transmitted correctly, the forward error-correcting (FEC) technology is employed in the MAC layer, as shown in Fig.1. When a frame has arrived, checksum mechanism is used to check it. If the checksum failure, retransmission is not used but FEC for error correction. While using FEC, the actual data transmitted is larger than the original base mesh data because the additional redundancy data is added. However, the ratio of amount of base mesh data in the entire model is much low, so using this method to transmit the base model does not affect the speed of the entire model transmission obviously. By this method, it can guarantee that a base model is transmitted correctly.

b) **Mac-Lite**. For less important model data, we adopt the Mac-Lite rather than traditional MAC.

However, the key of the Mac-Lite is to set the coverage of checksum for a frame. Usually, all headers information should be covered because of the following reasons:

1) If there are bit-errors in MAC header, the frame may be sent to other destinations because of source and destination address information in it.

2) If there are bit-errors in IP header, the packet will be discarded when it is forwarded in the IP layer, because the IP layer also has checksum mechanism which covers the IP header.

3) If there are bit-errors in the UDP header, the packet may be transmitted to other applications because the UDP header contains the source and destination port information.

Nevertheless, the checksum just covering the headers data is not enough. Factualy, the data can be divided into topology data and geometry data. While the topology information is lost during transmission, visual errors to the rendered model, such as the surface self-intersection, will be incurred. Thus the topology data should be transmitted as safely as the frame headers information. Therefore, the coverage of Mac-Lite checksum is the summary of MAC header (28bytes), IP header (20bytes) and UDP header (8bytes) and topology data as shown in Fig. 2.

![Fig. 1. MAC frame with FEC](image1)

<table>
<thead>
<tr>
<th>MAC Header</th>
<th>IP Header</th>
<th>UDP Header</th>
<th>Base Mesh data+FEC</th>
<th>MAC CRC</th>
</tr>
</thead>
</table>

![Fig. 2. MAC frame with VSpits](image2)

<table>
<thead>
<tr>
<th>MAC Header</th>
<th>IP Header</th>
<th>UDP Header</th>
<th>Topology data</th>
<th>Geometry data</th>
<th>MAC CRC</th>
</tr>
</thead>
</table>

V. PROGRESSIVE TRANSMISSION AND INTERACTIVE RENDERING

In this section, we presented a progressive transmission and interactive rendering method on mobile devices based on above progressive model. Different from the typical interactive rendering method, we will design a tactics for computing task allocated on the server side and client side respectively.

A. Computing Task Assign and Rendering

To reduce the mobile client's computing and storage burden, we will save and run the multi-resolution model in the server side as client does. For the client, it would only execute the rendering operation. First of all, we construct the multi-resolution model with the method in section 2, and the server will run and save this model. At the same time, using the CRS and SRS appeared in the
multi-resolution modeling to achieve the rapid switch of different resolution for the model. Secondly, when the client needs certain resolution model, the server makes use of the CRS and SRS to obtain the collapsed edge and split vertex, which will be formed into vertex index array and sent to the client. Finally, the client will execute the rendering procedure after obtaining this vertex index array and the data stored at the client already.

In the following, we will describe the real-time interactive rendering procedure from the aspect of the server and mobile client.

1) Server store the multi-resolution model data and response the client’s request.
   a) Responses the client’s request and sends the model’s geometry information to mobile client.
   b) Server run and store the multi-resolution model the client needed.
   c) Response the request for the certain resolution model from the client. Making use of the CRS and SRS to compute the multi-resolution model and get the vertex index array at this certain resolution.
   d) Sends the vertex index array to mobile client.
   e) Return to Step c and waits the request for another resolution model of the client.

2) Mobile client make requests for the server to obtain the certain resolution model and rendering locally according to the returned model.
   a) Client makes a rendering request to server.
   b) Receives the model geometry information and stores them locally.
   c) According to the user’s request, sends the rendering request for certain resolution.
   d) Receiving the vertex index array from the server side and rendering them without any reconstruction operation locally.
   e) Return to Step c:

During the interactive rendering procedure, it can be seen that our method just transmits a few vertex index array such that the total data over the network is decreased manifestly. More importantly, our method need not perform the local reconstruction operation thus that the waiting time for rendering of the model is decreased aggressively (see the experimental result as shown in Table 1). By this method, we can achieve interactive rendering for 3D model at the mobile client side.

VI. EXPERIMENTAL RESULT AND DISCUSSION

A. TestBed Design

We adopt the C/S model to validate our method. The PC server will transmit the data to mobile client through the D-Link wireless router among the Wireless 802.11.b network. The average network bandwidth is 0.5MB/s. The network layout is shown in Figure 2. At the server side, the multi-resolution model is created by our simplified method. In the mobile, we will adopt the rendering library M3D [25] we developed before which conforms the OpenGL ES specification. As we knew, the wireless network is varied so that the experimental result for the transmission time are measured as the average of 10 times.

To verify our transmission performance, what’s more, we use the ns-2 to build the simulation testbed. In our simulation model, the nodes have no mobility. This is primarily because our interest in his paper is to focus on the effectiveness of the modification of the MAC layer. Three nodes A, B and C are used as an ad hoc network and the topology is shown in Fig 2. In order to set the different loss rate in the PHY layer, the Gilbert Error model [27,28] should be added. The bit errors generated by this model are introduced to MAC frame.

The test models in our experiment that are stored by PLY format is Lauran, Bunny, and Horse as shown in Fig 4. The number of total vertices and corresponding storage space of each model, the ratio for the geometry information and the ratio for the topology information is shown in Table I.
We will report the experimental result for Horse model, which will be divided into 15233 $V_{Split}$. What’s more, the size of Horse model’s base mesh and details $V_{Split}$ is about 0.31M Bytes and 2.76M Bytes respectively.

In this paper, we only compare model quality received at the client side while the packet loss rate is 5%. Fig. 5 demonstrates the close-up wireframe view of horse model. In order to show comparative results obviously, we have to control the location of the packet loss rate. Then packets losses happen from the same area on the model. In these simulations, when part of a packet is lost and unrecovered, the packet is discarded. However, in order to show how many packets are lost at different packet-loss rates, we used a visualization trick by discarding part of the received packets in case the packets were lost. It can be seen that our transmission method can achieve the rendering result than the traditional MAC with retransmission mechanism.

C. Comparisons for Interactive Rendering

This subsection compares the proposed interactive rendering method with the typical method in reference [1]. We will compare the sum of transmission time from server to client and the rendering time on mobile device while transmitting the base model, which occupied 20% of the total model data, 60% and 100% of the model data. In the typical method, the whole time includes the transmission time, local reconstruction time and rendering time. Fortunately, our method’s whole time just includes the transmission time and rendering time. For the certain resolution model, the rendering time is assured so that the rendering time is not listed in Table II.

Since our method adopts the smart coding technique and computing task assignment, the transmitted data over network can be decreased and the local reconstruction time would be cut off. Thus, our method can achieve interactive rendering result. For example, while the full model are transmitted from the server to client and display, the time our method consumed are 41%, 21% and 37% in contrast to the typical method.

<table>
<thead>
<tr>
<th>3D Models</th>
<th>Vertices number/total data (KB)</th>
<th>Ratio for Geometry Information</th>
<th>Ratio for topology information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laurana</td>
<td>14499/1334</td>
<td>32.5%</td>
<td>67.5%</td>
</tr>
<tr>
<td>Bunny</td>
<td>20376/2963</td>
<td>32%</td>
<td>68%</td>
</tr>
<tr>
<td>Horse</td>
<td>16029/1382</td>
<td>33%</td>
<td>67%</td>
</tr>
</tbody>
</table>

VII. CONCLUSION AND FUTURE WORK

This paper proposes a visual important interactive rendering method for 3D model over 802.11 WLAN. This paper first proposes an efficient simplification method based on an improved saliency detection technique. By the introducing of the data structure including collapsed record stack and split record stack, we can finish the construction of multi-resolution model. Then, we develop an efficient hybrid FEC and MAC-Lite model transmission protocol which will transmit the model data by their importance respectively. Finally, we propose a real-time interactive rendering method by an efficient model coding. For decreasing the transmitted model data over the wireless network, we proposed an efficient model coding method and computing task assign method. By this method, we can transmit the model from the server to client quickly. What’s more, the mobile client can save the local reconstruction operation which would consume lots of CPU resource.

In the future work, we will adopt the geometry compression technique which will decrease the model data aggressively. Also, the dynamical transmission mechanism that just transmits the part of model user can see will also reduce the model data transmitted over the wireless network.
TABLE II. The Comparisons for Our Method and Typical Method of Transmission Time and Rendering Time

<table>
<thead>
<tr>
<th>Model</th>
<th>Typical Method (Full model transmission time and model reconstruction time)</th>
<th>Our method (Full model transmission time and the ratio comparing for the typical method)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Laurana</td>
<td>13889</td>
<td>5768 (41%)</td>
</tr>
<tr>
<td>Bunny</td>
<td>18892</td>
<td>6891 (37%)</td>
</tr>
</tbody>
</table>

ACKNOWLEDGMENT

This work is supported by in part Zhejiang natural science key foundation (Grant No. Z1080232, Z1101340), national natural science foundation of China (Grant No. 60873218), and the Scientific Research Fund of Zhejiang Provincial Education Department (Z201018041).

REFERENCES


**Bailin Yang** received the Doctor’s degree in department of computer science from Zhejiang University in 2007. He is an associate professor in the department of computer and electronic engineering of Zhejiang Gongshang University. His research interests are in mobile graphics, real-time rendering and mobile game.

**Zhiyong Zhang** received the master’s degree in department of Mechanical and Energy Engineering from Zhejiang University, Hangzhou, P.R. China in 2001. In 2005, He received the Doctor’s degree in department of computer science from Zhejiang University. Now, He is an associate professor in the department of computer and electronic engineering of Zhejiang Gongshang University. His research interests are in information retrieval, pattern recognition, and statistical machine learning.

**Xun Wang** received the Doctor’s degree in department of computer science from Zhejiang University. He is a professor in the department of computer and electronic engineering of Zhejiang Gongshang University. His research interests are in multimedia information retrieval, pattern recognition, and mobile networks and statistical machine learning.