3D Point Cloud Transmission and Visualization

by

Feng Chen

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science
University of Alberta

Abstract

The need for increasingly high resolution 3D models and the cost in processing high resolution polygonal meshes have made 3D point clouds an alternative for many researchers in recent years. Considerable research has been devoted to various areas of point clouds. This thesis presents three studies on point cloud transmission and visualization respectively. The first study designs a metric to evaluate each layer's quality contribution to the decoding result of the progressively compressed 3D point-based model. With this metric, the rendering quality of a reconstructed 3D model degrades more gracefully as the packet-loss rate increases. The second study proposes a new method for incorporating 3D point cloud models into multi-viewpoint video. With this method, 3D point clouds can be seamlessly inserted into a multi-viewpoint video and a realistic effect can be obtained. The third study presents a hierarchical RBF based approach to interactively visualize 3D point cloud. This aids users to achieve better resolution in a Region of Interest(ROI) without having to transmit and render the entire object in high detail.

Acknowledgements

I would like to thank my wife Min Yang and two daughters, Francesca and Gabrielle, for their love during my study. I would thank my father and mother for their nurturance and long time support. I also thank my father-in-law and Mother-in-law for their support.

I would like to thank my supervisor Dr. Anup Basu and Dr. Irene Cheng for their attentive guidance and kind help during my study.

I would also thank my committee member Dr. Nilanjan Ray for his careful examination and valuable suggestions on my thesis.

Table of Contents

List of Figures

List of Tables

Chapter 1

INTRODUCTION

1.1 MOTIVATIONS AND CONTRIBUTIONS

This thesis presents three studies on 3D point cloud transmission and visualization. The motivation and contributions of the thesis are introduced in this section. This thesis is based on three refereed conference papers [**?**] [**?**] [**?**].

In recent years 3D geometry model has become popularized as a new form of digital media. It plays an increasingly important role in many applications, such as e-commerce, entertainment, industrial design and education. As a primitive to represent 3D geometry model, point cloud evolved into a valuable alternative to polygonal meshes, for its simplicity and flexibility. Considerable research has been devoted to point cloud representation, modeling, processing, rendering and transmission [**?**] [**?**] [**?**] [**?**] [**?**] [**?**]. These three studies act as three different components in a 3D point cloud system, as shown in **??**.

1.1.1 ERROR RESILIENT TRANSMISSION

The goal of error resilient transmission is to achieve the best possible performance, given the errors introduced by the channel. To enable error detection and error concealment, redundant bits are needed in source coding or channel coding. Forward error correction (FEC) adds redundant bits to the original model, providing it the capability to recover lost model information at the receiver without the need for retransmission.

In this study, we utilize QSplat as the progressive compression strategy. This strategy represents 3D point cloud with a hierarchical tree. In this tree, leaf nodes

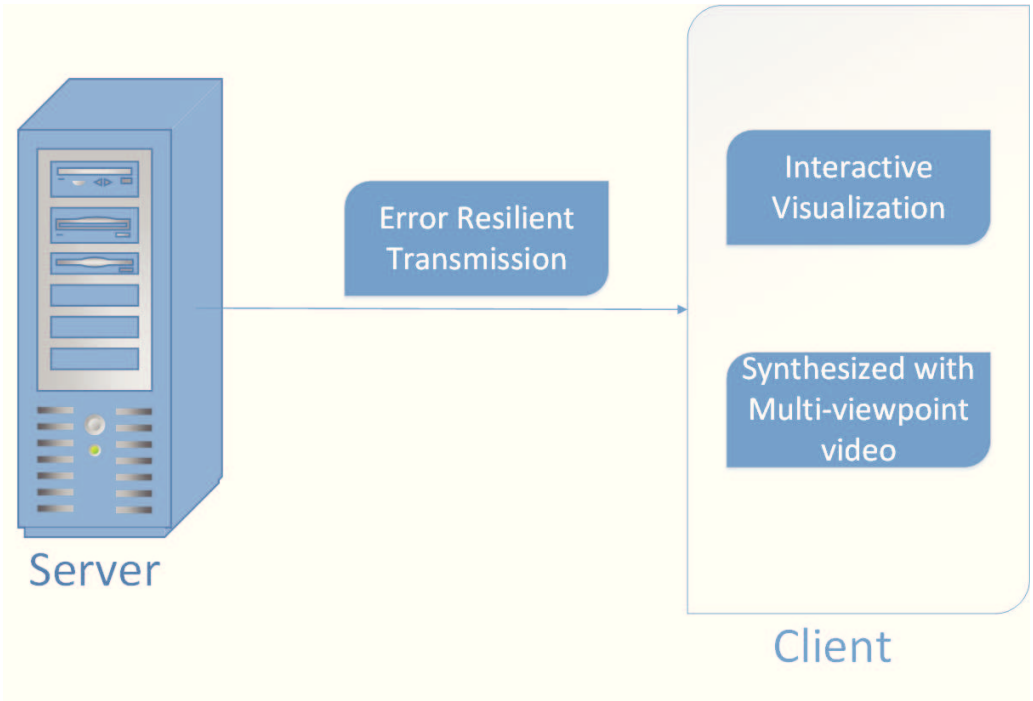

Figure 1.1: The three studies covered in this thesis.

correspond to vertices of the model, and the non-leaf nodes' properties equal to the average value of their children, which means each parent is an approximation of all its children. Thus, layers in this hierarchy represent the model at different levels of detail.

We address the problem of how to allocate redundant bits into different layers in this chapter. By analyzing the structure of a progressively compressed point cloud model, we design a metric to evaluate each layer's quality contribution to the decoding. With this metric, we then minimize the expected distortion when applying an Unequal Error Protection (UEP) strategy to allocate channel bits to different layers of the model. The performance of employing UEP and Equal Error Protection (EEP) are compared with respect to the expected distortion. Experimental results show that by incorporating our distortion estimation metric with UEP, the render-

ing quality of a reconstructed 3D model degrades more gracefully as the packet-loss rate increases.

1.1.2 SYNTHESIZED WITH MULTI-VIEWPOINT VIDEO

Multi-viewpoint video has recently gained significant attention in academic and commercial fields. It enables users to interactively access a scene from different viewpoints, which enhances the viewing experience, especially with the graphics models inserted into the video.

There are many techniques that have been proposed for synthesizing multi-viewpoint video. Zitnick *et al.* blended the individual projection results from the two closest cameras with a custom pixel shader that is given a weight for each camera based on the camera's distance from the new virtual view [**?**]. In [**?**], however, only the best surface areas from all available reference images are selected for rendering. Each pixel (area) in the virtual image is approximated by the reference image with the lowest value in the corresponding pixel (area). Kim *et al.* first applied the projective texture mapping method to project the texture image onto 3D objects that were modeled with a shape-from-silhouette (SFS) based method in the previous steps. They then merge the resulting objects with the background [**?**].

Point based graphics has potential in real-time interactive applications, including free viewpoint video [**?**]. The composition of multi-viewpoint video and 3D point cloud model is examined in this study. When inserting a rendered model into a multi-viewpoint video, one important problem is how to handle the possible

occlusions between the 3D model and an existing scene. Ignatov *et al.* exploited an identical coordinate system for camera, virtual and graphics views. Thus, the occlusion problem is automatically tackled by the graphics engine [**?**].

In our study, we first synthesize virtual multi-viewpoint video utilizing depth and texture maps of the input video. Then, we integrate 3D point cloud models with the resulting multi-viewpoint video generated in the first step by analyzing the depth information. As shown in our experiments, 3D point clouds can be seamlessly inserted into a multi-viewpoint video and a realistic effect can be obtained.

1.1.3 INTERACTIVE VISUALIZATION

With the fast growing volume of 3D point-cloud data, innovative point cloud visualization techniques are in demand for efficient and accurate information presentation and navigation. In general, 3D models are digitally archived, transferred on telecommunication networks, and visualized on computer screens. To download and render the whole model is time consuming, and sometime unnecessary for the user. For example, the reading physician focuses on a region of interest(ROI) when interpreting medical images. The region of interest, a tumor or an organ for example, often occupies less than 10% of all the data[**?**]. Thus, approaches that deliver higher resolution ROI and a low resolution context become a balanced solution.

Implicit modeling with Radial Basis Functions (RBFs) remains an active research area. J. C. Carr *et al.* showed that fitting scattered data by local, compactly supported RBFs leads to a simpler and faster computation procedure, while global

RBFs are useful in repairing incomplete data [**?**]. Y.Ohtake *et al.* used compactly supported RBFs to interpolate a given 3D point set surface in a hierarchical manner [**?**]. They employed locally supported functions that produced an efficient computational procedure, and the coarse-to-fine hierarchy to make their method insensitive to the density of scattered data.

In this study the 3D point cloud is stored by an octree-based hierarchy and interpolated progressively using RBF. Our experiment shows that users are able to achieve better resolution in a ROI without having to transmit and render the entire object.

1.2 THESIS ORGANIZATION

The remainder of this thesis is organized as follows. In the next chapter, a distortion metric for robust 3D point cloud transmission is introduced. Then we present the study of integrating 3D point cloud with multi-view point video in Chapter **??**. In Chapter **??** RBF based interactive visualization is introduced. The discussions and conclusions appear in Chapter **??**.

Chapter 2

DISTORTION METRIC FOR ROBUST 3D POINT CLOUD TRANSMISSION

2.1 INTRODUCTION

The need for increasingly high resolution 3D models and the cost in processing high resolution polygonal meshes have made 3D point clouds an alternative for many researchers in recent years. This chapter addresses the problem of error resilient transmission of progressively compressed 3D point clouds. Progressive compression approaches divide the 3D model data into a sequence of layers for transmission. Compared to single-layer compression approaches, they are more suitable for bandwidth-limited networks [**?**] [**?**] [**?**] [**?**] [**?**] [**?**]. At the server, different layers are packetized and transmitted separately. At the client, the rendering of the model begins after the first layer is received. Subsequent layers refine the rendering result.

In this study, we utilize QSplat as the progressive compression strategy, which is a multiresolution rendering system that employs a bounding sphere hierarchy structure and splat rendering [**?**]. The preprocessing procedure of QSplat removes all the connectivities of the input polygonal mesh and represents the vertices with a hierarchical tree. In this tree, leaf nodes correspond to vertices of the model, and the non-leaf nodes' properties (position, normal, color...) are equal to the average value of their children, which means each parent is an approximation of all its children. Thus, layers in this hierarchy represent the model at different levels of detail: all the leaf nodes intuitively form the finest layer, and nodes closer to the root represent coarser versions. Hence, this hierarchy tree can be transmitted progressively from coarse to fine, layer by layer, based on available bandwidth.

In addition to data compression, recovery from packet loss should also be con-

sidered for unreliable networks to ensure a certain level of quality of the rendered data. These error protection methods have been studied for audio, image and video communications. Their goal is to achieve the best possible performance given the errors introduced by the channel. These methods can be classified into two categories: sender based and receiver based.

Receiver based techniques are often referred to as ”error-concealment”. They exploit the property of smoothness in the model and produce a replacement for a lost packet that is similar to the original one. These techniques can be split into three categories [**?**]:

- Insertion-based techniques repair losses by inserting a fill-in packet. This packet is usually a repetition of the previous packet. This approach is easy to implement, but generally provides poor performance.
- Interpolation-based techniques use certain pattern matching and interpolation to create a replacement packet that is supposed to be similar to the lost one. This approach requires more computational power at the client and provides better performance than insertion-based technique.
- Regeneration-based techniques extrapolate the decoder state from packets surrounding the lost packet and generate a replacement for the lost packet from that. This approach is more expensive in term of implementation difficulty and computational power, but can provide good result.

Sender based methods can be divided into two major classes: active retransmission and passive FEC. Retransmission approaches retransmit all the lost packets

until they are received correctly on the client side, which is not applicable for time-sensitive application we focus on in this study. FEC adds redundant bits to the original model, giving it the capability to recover lost model information at the receiver without the need for retransmission. For delay-constrained applications, the combination of application layer FEC and user datagram protocol (UDP) is widely used.

Rusinkiewicz *et al.* demonstrated how to implement the streaming visualization of QSplat [**?**] . They focused on the selection of the transmitted data and the coordination of the transmission queue between the client and server. They also presented other interaction techniques for effective progressive visualization, including prefetching and color-coding by resolution. However, the potential for transmission error was not taken into account.

AlRegib *et al.* proposed a joint source and channel coding scheme for the progressive transmission of 3D meshes [**?**]. In their system, a 3D mesh is composed of a base mesh and several refinement layers. Their study defined a statistical measure to determine the channel bit allocation to each layer. We extend their approach to the transmission of 3D point clouds by introducing a quality distortion estimation metric.

The distortion metric proposed in this study estimates each layer's importance for decoding the progressive compression of 3D point cloud. Based on this metric, appropriate channel bits are allocated to different layers in the UEP strategy. The differing qualities of the rendered models generated by the UEP and EEP strategies are compared and examined in our experiments.

In our 3D transmission scheme shown in Figure. **??**, the preprocessing procedure of QSplat is applied to compress a 3D model into a progressive multi-layer point cloud. Before these layers are transmitted, data in different layers are packetized and encoded with FEC. At the receiver, a channel decoder is used to reconstruct packets in each layer according to the correction capability of the applied FEC code. QSplat is then utilized as the source decoder to render the model received on a display screen.

The remainder of this chapter is organized as follows. Section **??** presents background information on our 3D transmission method. Section **??** proposes a distortion estimation metric for QSplat model. In Section **??**, how to allocate channel bits among different layers is explained. Section **??** shows simulation results, which is followed by the conclusion in Section **??**.

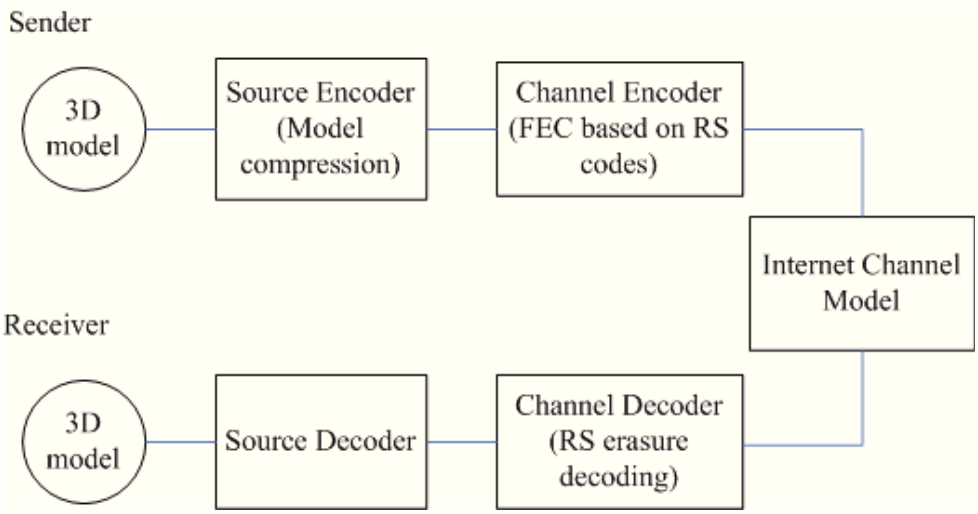

Figure 2.1: Outline of our 3D transmission scheme.

2.2 BACKGROUND

The object of this work is to achieve an error resilient transmission method that is scalable with respect to both the bandwidth and loss rate. Given a time constraint and an approximation of network bandwidth, the total amount of data bits that can be transmitted over the network can be budgetted. When the FEC scheme is applied to protect against packet loss, we need to determine how to allocate the channel coding bits to different layers of the progressive compressed model. In this section, we briefly present the background work for our proposed strategy.

2.2.1 PACKETIZATION METHOD

In a packet-switched network, both model data and error protection data are stored into packets before transmission. To reduce the adverse effects of packet loss, a packetization method known as block of packets (BOP) is presented in [**?**]. Figure **??** shows the construction of a BOP with FEC scheme based on Reed-Solomon (RS) codes. In this method, a channel encoding process is executed across different transmission packets instead of within a single packet. Using $RS(n, k)$, one BOP with n packets, including k packets of model data and $n - k$ packets of error protection data, can be recovered when any less than $n - k$ packets are lost during the transmission. Here k varies according to the different protection requirement.

In our strategy, different layers of a 3D point cloud model are encoded into different BOPs. In an EEP scheme, the same amount of error protection bits are allocated to each BOP, whereas in a UEP scheme, BOPs corresponding to different

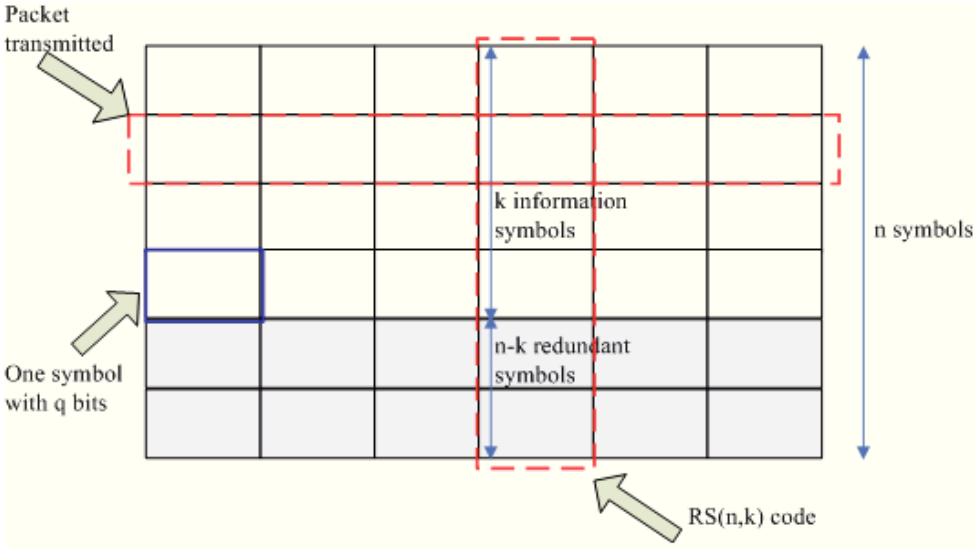

Figure 2.2: The layout of one BOP.

layers have different amounts of error protection bits. We will show that, for a 3D model represented by a progressive structure, UEP works better than EEP by applying stronger protection to the model data in more important layers. The importance of each layer is defined by a distortion metric introduced in Section **??**. Given the importance of each layer, we can minimize the expected distortion discussed in Section **??** and optimize the bit-allocation among different layers.

2.2.2 CHANNEL MODEL

With the BOP packetization method introduced in Section **??**, for the ith layer which is protected by RS(n_i,k_i), the lost packets can be recovered if the number of lost packets does not exceed $n_i - k_i$. Thus, the probability P_i of an irrecoverable ith BOP is defined in [**?**] as:

$$P_i = \sum_{m=n_i-k_i+1}^{n_i} p(m, n_i) \tag{2.1}$$

Here $p(m, n_i)$ denotes the probability of losing m packets within a block of n_i packets during transmission. In [**?**], B. Girod. *et al.* calculated $p(m, n_i)$ by utilizing a two-state Markovian model shown in Figure **??** to approximate package loss behavior of the underlying network. Here we shall give a brief introduction of their method.

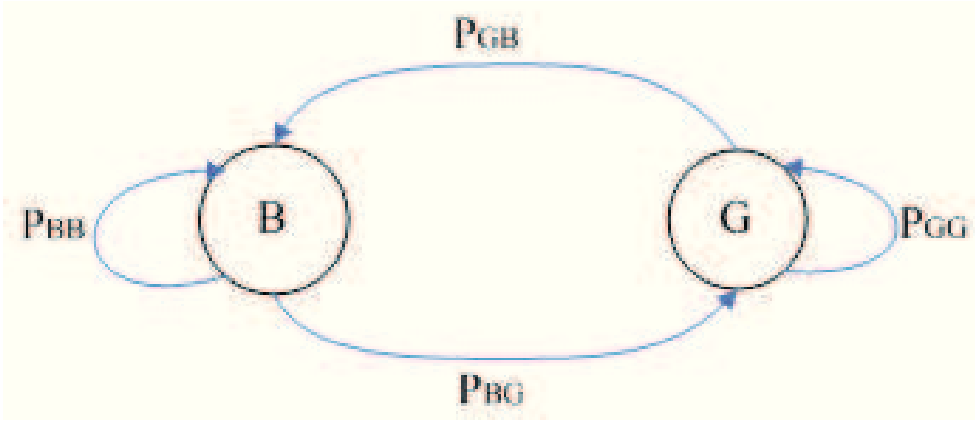

Figure 2.3: Gilbert-Elliot two-state Markovian channel model, modified from [**?**].

In Figure **??** state G denotes the case that a packet is received correctly and timely, and state B denotes the case that a packet is lost, either due to network congestion or due to exceeding the maximum allowed transmission delay. p_{GB} from state G to state B represents the probability that the packet following a timely received packet is lost. The average loss probability $P_{LR} = Pr(B) = p_{GB}/(p_{GB} + p_{BG})$ and the average number of consecutively lost packets $L_B = 1/p_{BG}$. Let $g(v)$ denote the probability of correctly receiving $v-1$ packets between two lost packets, and $G(v)$ denote the probability of correctly receiving more than $v-1$ packets after a lost packet, then

$$g(v) = \begin{cases} 1 - p_{BG}, & v = 1; \\ p_{BG}(1 - p_{GB})^{v-2} p_{GB}, & v > 1; \end{cases} \tag{2.2}$$

$$G(v) = \begin{cases} 1, & v = 1; \\ p_{BG}(1 - p_{GB})^{v-2}, & v > 1; \end{cases} \tag{2.3}$$

Let $R(m, n)$ denote the probability of losing $m - 1$ packet within the next $n - 1$ packets following a lost packet. It can be calculated by using $g(v)$ and $G(v)$ as:

$$R(m, n) = \begin{cases} G(n), & \text{m=1}; \\ \sum_{v=1}^{n-m+1} g(v) R(m - 1, n - v), & 2 \leq m \leq n; \end{cases} \tag{2.4}$$

Then the probability of losing m packets within n packets is

$$p(m, n) = \sum_{v=1}^{n-m+1} P_{LR} G(v) R(m, n - v + 1), 1 \leq m \leq n \tag{2.5}$$

With the $p(m, n)$ calculated by Equation **??**, P_i defined in Equation **??** can be obtained.

2.2.3 STATISTICAL DISTORTION MEASURE

For the QSplat model, a node and its subtree cannot be rendered if the packet containing the node is lost and cannot be recovered. We adopted the definition of expected distortion presented by Alregib *et al.* in [**?**] to estimate the distortion caused by an unrecoverable layer. They define the expected model distortion as

$$\mathcal{D}_r = P_0 E_0 + \sum_{j=1}^{M} P_j E_j \prod_{i=0}^{j-1} (1 - P_i) \tag{2.6}$$

In this definition, M is the number of layers. P_j is the probability of an unrecoverable jth layer, which can be calculated using Equation **??**, and E_j is the importance of the jth layer which is, defined by Equation **??**. The probabilistic measure reflects how much the rendered model deviates from the original model. Based on this measure, we minimize the expected distortion.

2.3 DISTORTION METRIC FOR $QSplat$ MODEL

In the compression of a $QSplat$ model, the properties (position, radius, etc.) of each node are encoded relative to its parent [**?**]. When the data of one node is irrecoverable, all its descendants cannot be rendered. The impact of losing one node is always larger than losing any of its descendants. Keeping this in mind, we define the importance e_N of one node N as the average Euclidean distance between N itself and all the leaves of the tree rooted at N, and the importance of the ith layer E_i as the average importance of all the nodes in this layer, which is:

$$E_i = \sum_{k:depth(k)=i} e_k/N_i, \tag{2.7}$$

Here N_i is the number of nodes in the ith layer. We use this importance to measure the estimated distortion between the original model and the model decoded with partial layers.

For the simple example shown in Figure **??**, the importance e_{N_0} of N_0 can be computed as:

$$e_{N_0} = \frac{\sum_{i=0}^{3} Dis(N_0, N_{0i})}{4} \tag{2.8}$$

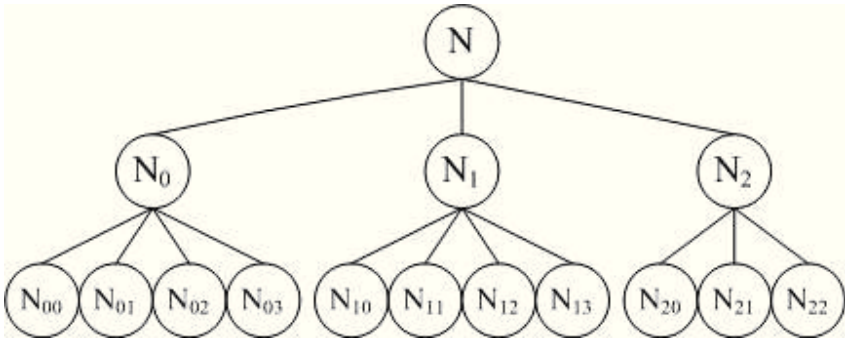

Figure 2.4: A simple example of QSplat hierarchical tree.

and the importance of the second layer can be calculated using:

$$E_2 = \frac{\sum_{i=0}^{2} e_{N_i}}{3} \tag{2.9}$$

Here $Dis(A, B)$ gives the Euclidean distance between A and B.

After preprocessing with QSplat, Bunny model is represented by a 10 level hierarchy tree. The importance of each level calculated by Equation **??** is given in Figure **??**.

2.4 CHANNEL BIT ALLOCATION

Given the channel bit budget C ,

$$C = \sum_{i=0}^{L-1} C_i \tag{2.10}$$

Where C_i denotes the number of channel bits allocated to protect the ith layer, and L is the total number of layers transmitted.

How to allocate the total budget C into C_i, for varying i, depends on the protection strategy applied. For FEC based on EEP, we divide C into L layers equally,

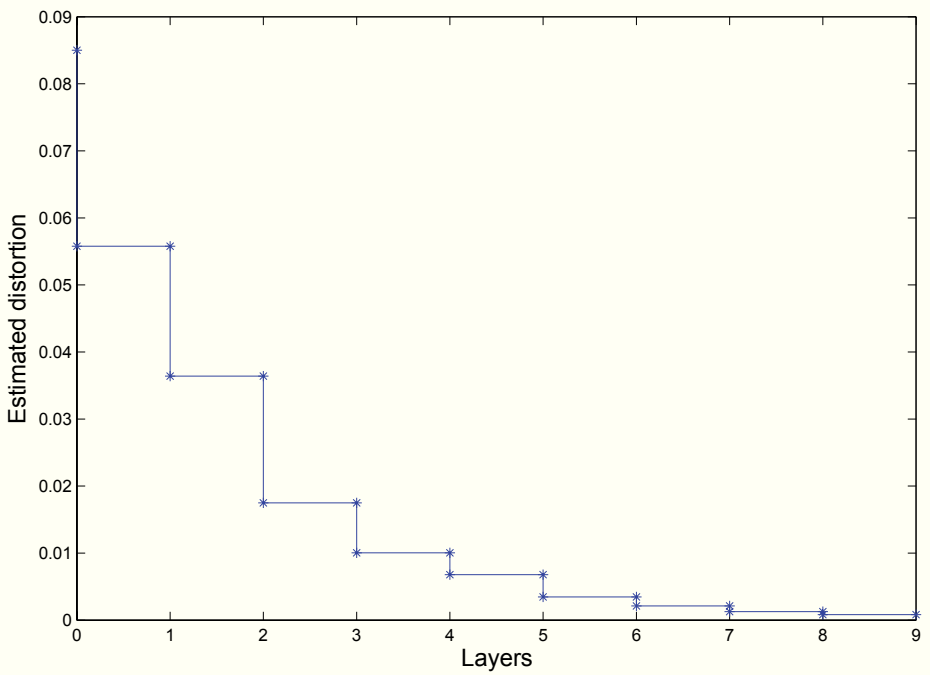

Figure 2.5: The estimated distortion between the original Bunny model and the model terminating decoding at different levels.

which means $C_0 = C_1 = ... = C_{L-1} = \frac{C}{L}$. For FEC based on UEP, we can determine the C_is by the following steps:

(1) For each possible combination of C_0 , C_1 ,.., C_{L-1}, we calculate corresponding k_0 , k_1,.., k_{L-1} for the RS codes as:

$$k_i = \lfloor \frac{S_i \cdot n}{C_i + S_i} \rfloor \tag{2.11}$$

Here S_i is the amount of source bits in the ith layer and n is a specified number of packets. After computing k_i and with the loss rate known, the probability P_i of the occurrence of an irrecoverable BOP in the ith layer can be obtained using the method presented in Section **??**.

(2) The importance E_i of the ith layer in the QSplat model can be calculated by

Equation **??**.

(3) Given the irrecoverable probability P_i and the importance of each layer E_i, the expected distortion defined in Section **??** can be calculated.

(4) From the result of the previous step, choose the minimum expected distortion, and the corresponding assignment of C_0 , C_1 ,..., C_{L-1} as the optimal solution.

In order to validate our quality distortion estimation metric, we applied the strategy on a number of 3D models and the results are presented in the next section.

2.5 EXPERIMENTAL RESULTS

Experiments were conducted to compare the performance of EEP and UEP. In our experiments a value of 255 was assigned to n as suggested in [**?**], which means that all BOPs employed $RS(255, k)$ as the channel encoding method. After preprocessing using the QSplat approach, the Bunny model was structured into a $L = 5$ layer hierarchy. A total bit budget of 10000 bytes was used in the experiment. Since the source coding bits had a size of 6692 bytes, there were 3308 left for channel coding. For UEP, the channel coding bits allocated to each layer were determined with respect to the optimal expected distortion. Figure **??** shows the expected distortion of EEP and UEP for different loss rates in our experiment. Observe that the expected distortions of both EEP and UEP increase as the loss rate increases. However, the distortion of UEP is smaller than EEP for all loss rates.

We used the QSplat rendering system to display the distorted models. When

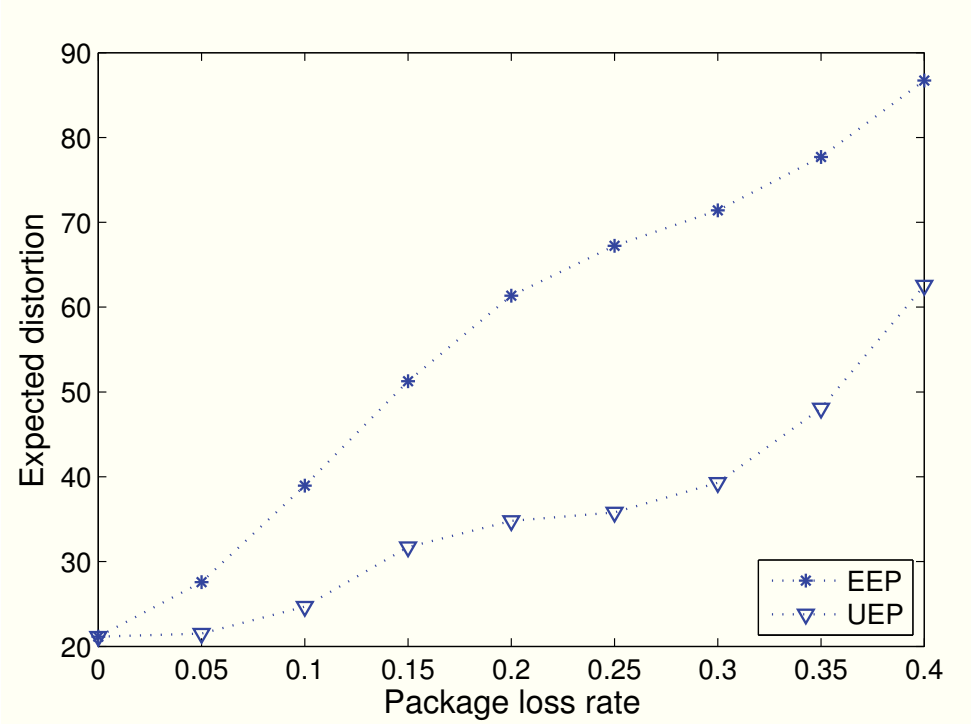

Figure 2.6: The expected distortion for the decoded Bunny model as a function of different packet-loss rates.

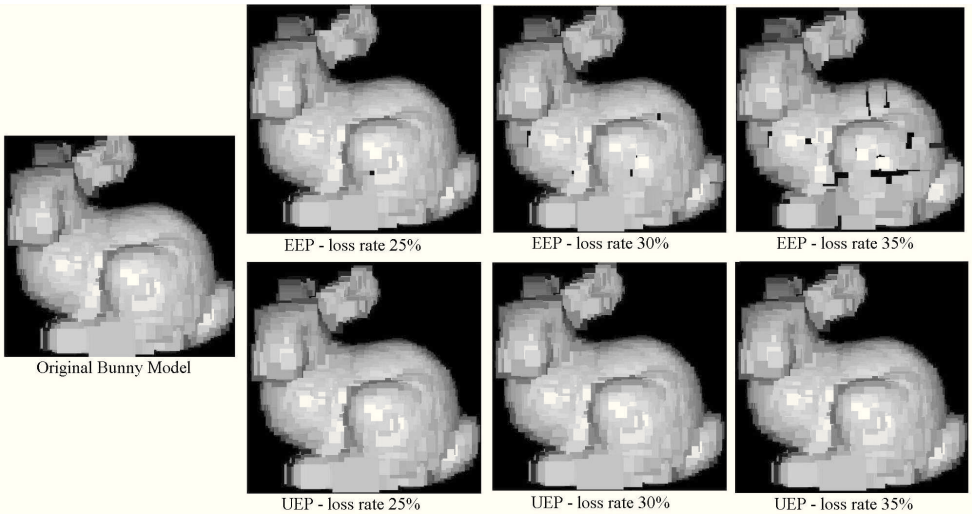

Figure 2.7: The decoded Bunny model (1334 points) based on EEP and UEP at different packet-loss rates.

one node was lost, both the node and its descendants were not rendered. Figure **??** shows that when the packet-loss rate is more than 25%, holes begin to appear in the model transmitted with the EEP method. On the contrary, the UEP method protects the model quite well even when the packet-loss rate is as high as 35%. We also performed experiments on other 3D models. Figure **??** shows an example using the 5451 point dragon model, Figure **??** shows the 5369 point lion model, and Figure **??** show the 5448 point Lucy model. All these experimental results illustrate that UEP incorporating our distortion metric outperforms EEP at various packet loss rates.

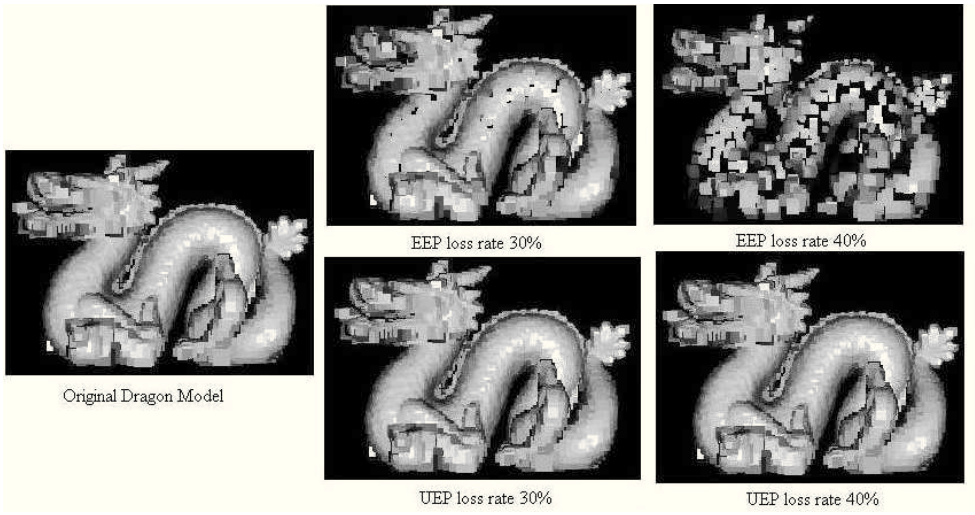

Figure 2.8: The decoded Dragon model (5451 points) based on EEP and UEP at different packet-loss rate.

2.6 CONCLUSION AND DISCUSSION

In this chapter we discussed the transmission of progressively compressed 3D point clouds and the use of FEC error resilient transmission to protect against packet loss. We proposed a quality distortion estimation metric to evaluate the adverse effects caused by an irrecoverable node in the QSplat layered hierarchy. Based on the

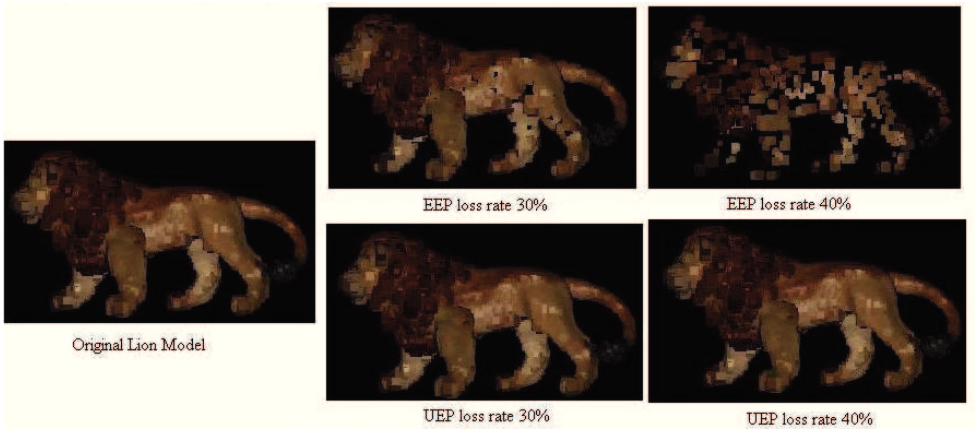

Figure 2.9: The decoded Lion model (5369 points) based on EEP and UEP at different packet-loss rate.

bandwidth constraint and packet loss rate, our metric helps to allocate channel bits in different layers based on their relative importance to the rendered quality. We compared the UEP approach incorporating our metric with the EEP approach using different 3D models. Experimental results show that the UEP strategy adapting our metric outperforms EEP.

In the current implementation, we treat each node in the same layer of the QSplat hierarchy equally. This may lead to underestimation of the distortion effect for nodes associated with higher surface curvatures. It is possible to improve the performance of our algorithm by considering curvature weighting when computing the importance factor E_i. This issue will be addressed in future work.

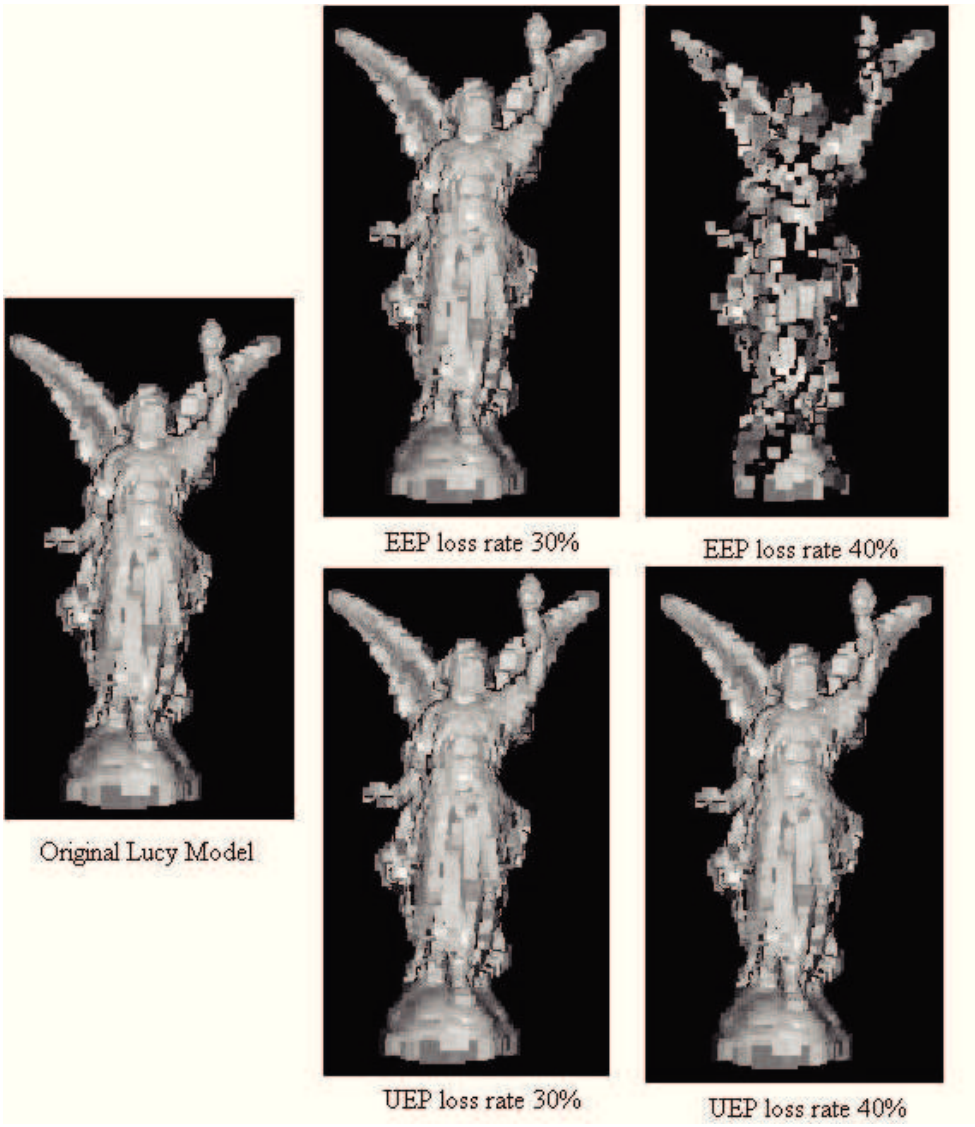

Figure 2.10: The decoded Lucy model(5448 points) based on EEP and UEP at different packet-loss rate.

Chapter 3

INTEGRATING 3D POINT CLOUDS WITH MULTI-VIEWPOINT VIDEO

3.1 INTRODUCTION

Multi-viewpoint video enables users to interactively access a scene from different viewpoints, which enhances the viewing experience, especially with graphics models inserted into the video. A multi-viewpoint video system contains four main components - camera calibration, correspondence matching, virtual view synthesis and graphics model integration.

One of the key problems in such a system is to reconstruct the depth information of a captured scene. In most methods proposed to solve this problem, recovering the depth information Z is converted to estimating the disparity d, which is inversely correlated to the depth.

$$Z = \frac{fB}{d} \tag{3.1}$$

where f is the focal length of the camera, and B is the baseline distance. Disparity estimation is one of the most active research areas in computer vision and numerous algorithms have been published in this area. Brown *et al.* reviewed advances in correspondence methods [**?**]. Scharstein and Szeliski presented a taxonomy of existing stereo algorithms [**?**].

There are many techniques that have been proposed for synthesizing multi-viewpoint video. Zitnick *et al.* blended the individual projection results from the two closest cameras with a custom pixel shader which is given a weight for each camera based on the camera's distance from the new virtual view [**?**]. Whereas, in [**?**], only the best surface areas from all available reference images are selected for

rendering. Each pixel (area) in the virtual image is approximated by the reference image with the lowest value in the corresponding pixel (area). Kim *et al.* first applied the projective texture mapping method to project the texture image onto 3D objects which were modeled with a shape-from-silhouette (SFS) based method in the previous steps, then they merge the resulting objects with the background [**?**].

Point based graphics has potential in real-time interactive applications, including free viewpoint video [**?**]. The composition of multi-viewpoint video and 3D point cloud model is addressed in this study. When inserting a rendered model into a multi-viewpoint video, one important problem is how to handle the possible occlusions between the 3D model and an existing scene. Ignatov *et al.* exploited an identical coordinate system for camera, virtual and graphics views. Thus, the occlusion problem is automatically tackled by the graphics engine [**?**].

The basic structure for our view synthesis and 3D model composition is illustrated in Figure **??**. Given camera calibration parameters and correspondence data generated by [**?**], we estimated the depth map and then reconstructed virtual intermediate images. Following this, in order to perform the image composition, the locations of graphics cameras were fixed at the same position as the corresponding base/virtual cameras so that the real scene and graphics models are correctly registered. Then, the texture image and depth data were obtained for both virtual and graphics views, and a final composite image was created by comparing the pixel-wise depth values.

The remainder of this chapter is organized as follows: Section **??** explains the virtual view synthesis method in detail. 3D point cloud integration is described in

Section **??**. Section **??** presents the experimental results of our system, before the conclusion and discussion of future work.

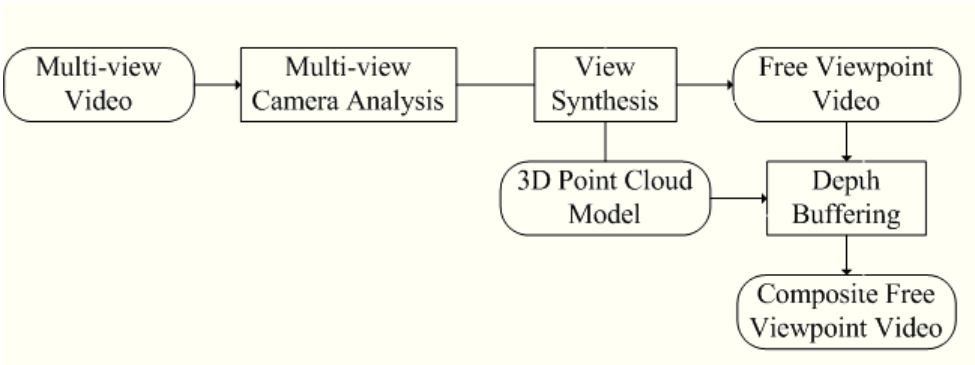

Figure 3.1: Framework of the free viewpoint synthesis and integration system.

3.2 VIRTUAL VIEW SYNTHESIS

We adopt the parallel setup for all cameras. In this case, we can generate the virtual images of these cameras through the following steps (similar to [**?**]):

1) Specify the two nearest base cameras: say, Base Cameras 1 and 2 (Figure **??**);

2) Project pixels from Base Camera 1 back to the 3-D space.

3) Project all 3D points reconstructed in 2) to the virtual camera and obtain a new image;

4) Repeat Steps 2) and 3) for Base Camera 2; and,

5) Calculate the weighed average of resulting images from the two base cameras, based on the distance between the virtual camera and each base camera.

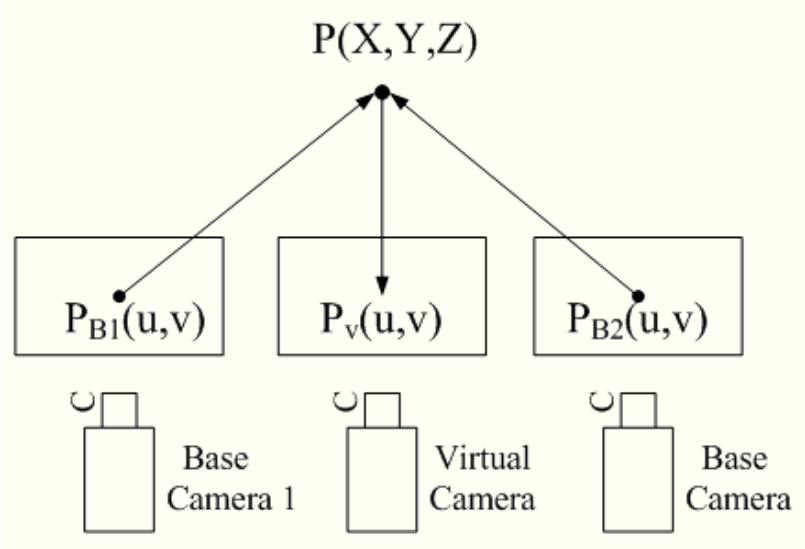

Figure 3.2: The principle of virtual camera synthesis.

Both intrinsic matrix K and extrinsic matrix $[R|t]$ are required for the above algorithm based on the methods illustrated in [**?**].

In Step 3, for one 3D world point X and its corresponding image point x represented by homogeneous vectors, their mapping can be expressed as $x = MX$ where $M = K[R|t]$.

For Step 2, the mapping between one image pixel x and its corresponding 3D point X is given by $X = C + zM^{+}x$. Here z is the depth value that is calculated by Equation (**??**), M^{+} is the pseudo inverse of camera projection matrix M, and C represents coordinates of the camera center.

3.3 3D POINT CLOUD INTEGRATION

After generating all the virtual views, we integrate 3D point cloud models into the multi-viewpoint video. The key problem in this integration is how to handle the occlusions between a graphics model and a real world scene, which requires a pixel-wise comparison of depth information. Since the images of the base and virtual cameras are in the image field, and the rendering results of 3D point cloud models are in the graphics field, there are two approaches to solve the occlusion problem: graphics-based and image-based. Graphics-based methods copy the texture and depth maps of cameras into the graphics pipeline, then the Z-buffer resolves the problem automatically. Image-based methods, on the contrary, save the rendering result of 3D models as texture maps and depth maps. The image composition is then performed between the camera view and the graphics view. Identical compositions are applied to base cameras, virtual cameras and graphics cameras.

To maintain consistency between the real world space and the graphics space, the corresponding base/virtual camera and graphics camera should have identical parameters. Given the intrinsic camera matrix K and the extrinsic camera matrix $[R|t]$, we need to determine the graphics viewing matrix V and the projection matrix P accordingly.

First, we convert the extrinsic camera matrix $\begin{bmatrix} R|t \end{bmatrix}$ to the viewing matrix V by adding $\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}$ to the bottom row.

$$V = \begin{pmatrix} & R & & t \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.2}$$

Then, we consider the conversion from the intrinsic camera matrix:

$$K = \begin{pmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.3}$$

to the projection matrix P.

In the implementation, P is fully determined by the near plane N, far plane F, aspect ratio and vertical view angle. We set the near plane N equal to a focal length f, and the far plane F farther than the depth of the background in the camera views. The aspect ratio is obtained by $\frac{\alpha_y}{\alpha_x}$ from Equation **??**, and vertical view angle is determined based on user preference.

For the 3D point cloud rendering, we adopt the rendering method presented in [**?**]. The QSplat system supports the real-time progressive rendering of large models, providing the graphics image used as the input for image composition in our system.

With the methods illustrated above, we implemented the image composition between the base/virtual camera and the graphics camera with the following steps:

1) Based on the K and $[R|t]$ of Base Camera 1, calculate the P and V of Graphics Camera 1;

2) Render the 3D point cloud model from Graphics Camera 1 using a QSplat type viewer, then save the rendered result as a texture map and a depth map;

3) Perform the image composition between the texture image of Base Camera 1 and the texture image of Graphics Camera 1, based on a pixel-wise comparison of their respective depth maps;

4) Repeat Steps 1) to 3) for all other base/virtual cameras, producing the composite image for each viewpoint; and,

5) Compress all the images from Steps 3) and 4) to get the multi-viewpoint video integrated with the point cloud model.

3.4 EXPERIMENTAL RESULTS

In this section, we present experimental results based on the methods discussed in Sections **??** and **??**. First, the result of virtual viewpoint synthesis is presented. Then, we show the integration of a 3D point cloud model into the multi-viewpoint video. At the end of this section, we compare two images of the same virtual camera generated with different synthesis strategies. For comparisons we show the *breakdancing* multi-view sequence provided by Microsoft Research [**?**].

Figure **??** shows three synthetic cameras between two Base Cameras 1 and 2. In this experiment, the distance between Base Camera 1 and Base Camera 2 is divided into 3 equidistant intervals. The result is acceptable with blurring occurring only close to the boundaries of objects.

Figure **??** shows the composition result of six different frames of Camera 1 and the 3D point cloud model "*lion*." The graphics model *lion* is designed to be part of the scene. As shown among the frames, the synthesized graphics model becomes an integral part of the scene, and the occlusion problem is handled properly by our algorithm. In addition, our method can show part of a moving object (the breakdancer) being in front of a point cloud model, while another part of the object

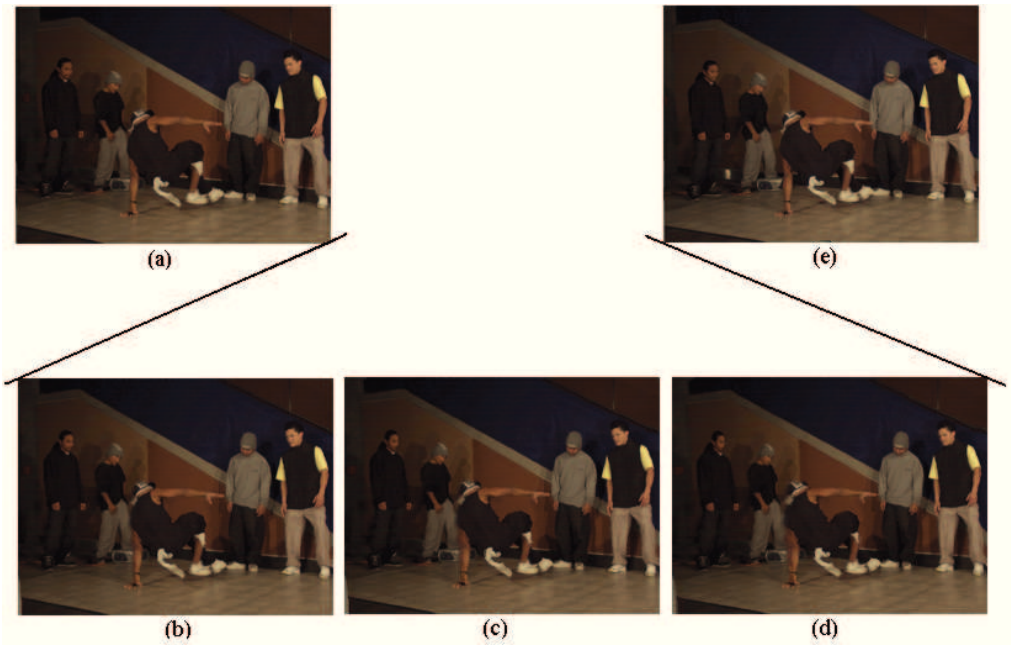

Figure 3.3: Three virtual viewpoints((b),(c),(d)) between Base Camera 1(a) and Base Camera 2(e).

(the breakdancer's towel) is behind the model (Figure **??**).

Thus far the virtual views were all obtained by weighed interpolation between the re-projection images of the two nearest base cameras. As an alternative, we generated the virtual view by using only one of the nearest base cameras. Figure **??** compares the result of these two approaches applied on the same frame. From this figure, we can observe that the virtual view generated by camera interpolation is less noisy in the motion part compared to the virtual view generated by only the nearest one (see the left arm of the dancer), but it is less sharp in the static part (see the face of the person standing on the right). This result supports what is intuitively expected.

3.5 CONCLUSION AND FUTURE WORK

In this chapter we implemented a system for synthesizing virtual views and inserting 3D point cloud models into multi-viewpoint sequences. 3D depth and texture information of base cameras were utilized to generate the virtual viewpoint. The conversion between camera matrix and graphics matrix was considered for consistency between the graphics camera and base/virtual camera. We used pixel-wise depth comparison to handle the occlusion problem. Experimental results show that our method generates high-quality synthesized images using information obtained from only two neighboring base cameras, and the occlusion problem is solved reasonably well during the integration of a 3D point cloud model with a multi-viewpoint video. In the future, we plan to improve three aspects of the proposed system:

- For view synthesis, different strategies will be explored to generate finer virtual views;
- Motion 3D point cloud models will be integrated into the multi-viewpoint video; and,
- Further optimization will be performed to support online rendering.

Figure 3.4: Free viewpoint video integrated with the 3D point cloud model “lion” near the center.

Figure 3.5: Breakdancer in front of “lion” with his towel behind.

Figure 3.6: Virtual viewpoint (a) interpolated between Base Camera 1 and Base Camera 2 (b) reprojected from Base Camera 1.

Chapter 4

INTERACTIVE VISUALIZATION OF 3D POINT CLOUD USING RBF BASED REPRESENTATION

4.1 INTRODUCTION

With the fast growing volume of 3D point-cloud data, innovative point cloud visualization techniques are in demand for efficient and accurate information presentation and navigation. This project addresses the problem of interactively visualization of 3D point cloud. In general, 3D models are digitally archived, transferred on telecommunication networks, and visualized on computer screens. To download and render the whole model is time consuming, and sometime unnecessary for the user. For example, the reading physician focuses on a region of interest(ROI) when interpreting medical images. The region of interest, a tumor or an organ for example, often occupies less than 10% of all the data[**?**]. Thus, approaches that deliver higher resolution ROI and a low resolution context become a balanced solution.

In this chapter, we propose a hierarchical RBF based approach to interactively visualize the 3D point cloud. With this approach, users are able to achieve better resolution in an ROI without having to transmit and render the entire object in high detail.

Implicit modeling with Radial Basis Functions (RBFs) remains an active research area. Whether to choose local or global RBFs is the problem that needs to be addressed. J. C. Carr *et al.* showed that fitting scattered data by local, compactly supported RBFs leads to a simpler and faster computation procedure, while global RBFs are useful in repairing incomplete data [**?**].

Y.Ohtake *et al.* used compactly supported RBFs to interpolate a given 3D point set surface in a hierarchical manner [**?**]. They employed locally supported functions

that lead an efficient computational procedure, and the coarse-to-fine hierarchy to make their method insensitive to the density of scattered data. In this chapter we use similar approach as theirs.

The remainder of this chapter is organized as follows. Section **??** presents background and related work. The implementation of our approach is discussed in Section **??**. We then present the experiment results and discussion in Section **??**.

4.2 BACKGROUND AND RELATED WORK

4.2.1 IMPLICIT SURFACE

An implicit surface is defined by an implicit function, a continuous scalar-valued function over the domain R^3. If a surface M consists of all the points (x, y, z) that satisfy the equation

$$f(x, y, z) = 0, \tag{4.1}$$

then we say that f implicitly defines M.

This may be generalized to a scattered data interpolation problem, as follows. Give a set of position $\boldsymbol{x}$ and corresponding values $\boldsymbol{h}$, solve for a function such that

$$f(\boldsymbol{x}) = \boldsymbol{h} \tag{4.2}$$

Radial basis functions are circularly-symmetric functions centered at a particular point. Scattered data interpolation can be achieved using RBFs centered at the constraints. By using n RBFs centered at n points, we can interpolate a function as

follows,

$$f(\boldsymbol{x}) = \sum_{j=1}^{n} d_j \phi(\boldsymbol{x} - \boldsymbol{c}_j) + P(\boldsymbol{x}) \tag{4.3}$$

In the above equations, $\boldsymbol{c}_j$ is the position of known values, d_j is the weight of the RBF positioned at that point, and $P(\boldsymbol{x})$ is a first-degree polynomial to account for the linear and constant portions of f.

4.2.2 INTERPOLATING IMPLICIT SURFACE USING RADIAL BASIS FUNCTIONS

Scattered data interpolation using RBFs has been studied from at least the 1980s [**?**]. Essentially, a 3D object surface is represented implicitly (where the RBFs have the value zero), which provides a compact representation with the RBF being defined everywhere in $\mathcal{R}^3$. The approach has various applications like automatic mesh repair in range-scanned graphical models [**?**], surface reconstruction in ultrasound data [**?**], and animated face modeling [**?**].

M.S. Floater and A. Iske presented a hierarchical scheme for smoothly interpolating scattered data with RBFs of compact support [**?**]. A nested sequence of subsets of data is computed efficiently using successive Delaunay triangulations. The scale of the basis function at each level is determined from the current density of points using information from the triangulation. The methods is rotationally invariant and has good reproduction properties. Moreover, the solution can be calculated and evaluated in acceptable computing time.

J.C. Carr *et al.* used RBFs to reconstruct smooth, manifold surfaces from point-cloud data and to repair incomplete meshes [**?**]. They introduced a greedy algorithm

in the fitting process which reduces the number of RBF centers required to represent a surface. This algorithm provides results in significant compression and further computational advantages. With fast methods for fitting and evaluating RBFs, they modeled large data sets, consisting of millions of surface points, by a single RBF.

H.Q. Dinh *et al.* presented a new method to construct a 3D implicit surface, which is formulated as a sum of weighted RBFs [**?**]. The implicit functions they constructed estimated the surface well in regions where there is little data. The reconstructed surface is locally detailed, yet globally smooth, because they use RBFs that achieve multiple orders of smoothness.

B.S. Morse *et al.* described algebraic methods for creating implicit surfaces using linear combinations of radial basis interpolants to form complex models from scattered surface point [**?**]. They explored and extended previous implicit interpolating methods by using compactly supported radial basis interpolants. The use of compactly supported radial basis elements generates a sparse solution space, reducing the computational expense and making the technique practical for large models. This reduction in computational complexity enables the application of these methods to the study of shape properties of large complex shapes.

Conventional methods for RBF implicit surface fitting to N points requires $\mathcal{O}(N^3)$ operations and $\mathcal{O}(N^2)$ storage. Pears *et al.* employs the fast multi-pole method(FMM) developed by L. Greengard and V. Rokhlin [**?**] and used by J.C. Carr *et al.* [**?**] for interpolating 3D object surfaces [**?**]. In [**?**], approximations are allowed in both the fitting and evaluation of the RBF. For example, for RBF evaluation at a given point, the centres are clustered into 'near field' and 'far field'.

The contribution of only those centres ‘near’ to the evaluation point are directly evaluated and those ‘far’ from the evaluation point are approximated, allowing a globally supported RBF to evaluated quickly to some prescribed accuracy.

Y. Ohtake *et al.* proposed a hierarchical approach to 3D scatted data interpolation and approximation with compactly supported RBFs [**?**]. Their numerical experiments suggest that the approach integrates the best aspects of scattered data fitting with locally and globally supported basis functions. Employing locally supported functions leads to an efficient computational procedure, while a coarse-to-fine hierarchy makes their method insensitive to the density of scattered data and allows restoring large parts of missed data.

4.2.3 INTERACTIVE VISUALIZATION WITH REGION OF INTEREST

Eric LaMar *et al.* presented a multi-resolution technique for interactive texture-based volume visualization of very large data sets [**?**]. The method uses an adaptive scheme that renders the volume in a ROI at a high resolution and the volume away at a progressively lower resolution. The algorithm is based on the segmentation of texture space into octree, where the leaves of the tree define the original data and the internal nodes define lower-resolution versions. Rendering is undertaken adaptively by selecting high-resolution cells close to a center of attention and low-resolution cells away from this area.

S. Guthe *et al.* proposed a novel algorithm for rendering very large volume data sets at interactive frame rates on standard PC hardware [**?**]. The algorithm accepts

scalar data samples on a regular grid as input. The input data is converted into a compressed hierarchical wavelet representation in a preprocessing step. During rendering, the wavelet representation is decompressed on-the-fly and rendered using hardware texture mapping. The level of detail used for rendering is adapted to the local frequency spectrum of the data and its position relative to the viewer.

S. Piccand *et al.* addressed the problem of interactive visualization [**?**].They focused on visualizing a complete resolution ROI and a low resolution context, by processing all wavelet coefficients that compose the ROI and dropping some of the wavelet coefficients that compose the context. The interactive visualization process uses the wavelet representation and user-defined region to render the result directly from the wavelet space through wavelet splatting, thus avoid volume reconstruction.

S. Piccand *et al.* proposed a novel wavelet splatting approach for multi-resolution 3-D visualization [**?**]. Their method renders the context with a low resolution at first, and then subsequently, refines it progressively to attain full resolution, while ensuring that a specific region of interest is rendered at full resolution at all times. It is based on the splatting approach for its computational efficiency and uses the localization property of the wavelet transform to simultaneously render a full-resolution region of interest with a coarser context.

4.3 IMPLEMENTATION

Given a point cloud distributed on a surface, during the preprocessing stage, we first down sample the point cloud and construct a coarse-to-fine hierarchy based on

octree.

Then during the fitting stage, we interpolate the point sets starting from the coarsest level. Other levels of the hierarchy are interpolated by an offset of the interpolating function computed at the previous level.

In the rendering stage, models are polygonized by Bloomenthal's method [**?**] [**?**].

Figure **??** shows the interface of our program. When users click on a specific position (say P) and choose to add points (say n points) around it, the n closest points to the selected P are calculated and added to the current point set. Then, the current point set is re-rendered to generate new result.

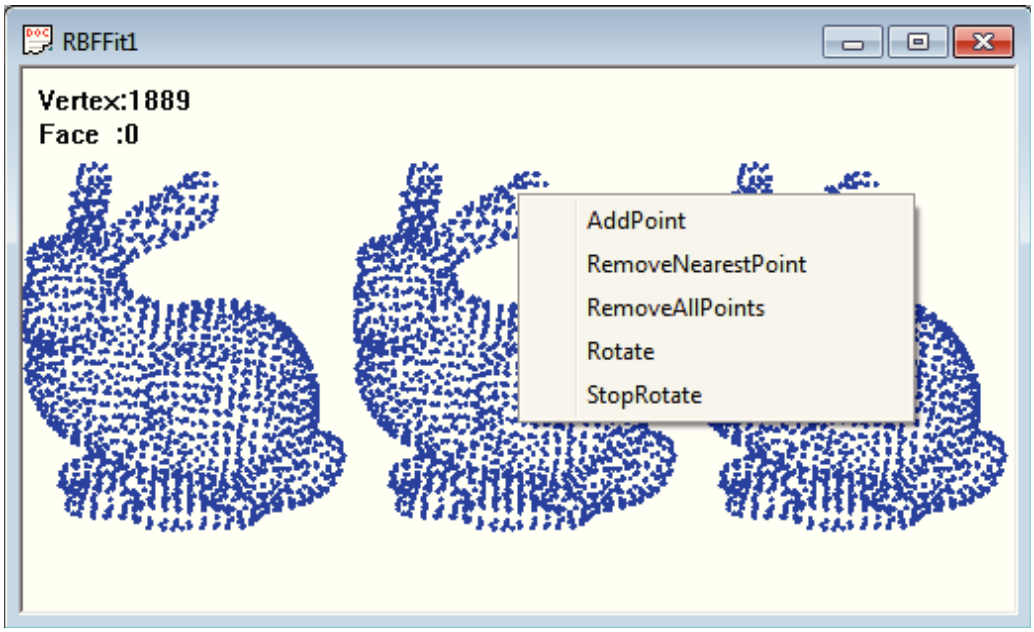

Figure 4.1: Interface of the program.

4.4 EXPERIMENT RESULT

With our approach, users can add extra points into any selected ROI when they are viewing the 3D point-cloud model. With these points added, a new, higher resolution of the ROI is presented to the user, with the context staying at the previous resolution.

Figure **??**, **??** and **??** show the experiment results of our program:

In the experiment results, the left column shows the original point-cloud (blue) with new points (red) added to the designated area. The middle column shows the rendering results of the new model (with points added), compared to the right column which shows the rendering result of the original one. From the experiment results, we can observe that the rendering results show more detail on the region with points added (face, mouth, eye, claw respectively).

Let N_c be the number of points of the coarse level of the model, and N_a be the number of points added. Then $N_c + N_a$ represents the points needed to render the ROI in the finest resolution with other area staying coarse. The number of the full model, N_t, is the number of points needed to achieve the finest resolution of the ROI without using this interactive rendering approach. Table **??** shows these numbers for the Bunny, Dragon and Buddha in the previous experiments. From Table **??** we can observe that in order to achieve a fine resolution of only the ROI, our approach needs a small amount of points comparing to transmitting and rendering the full model.

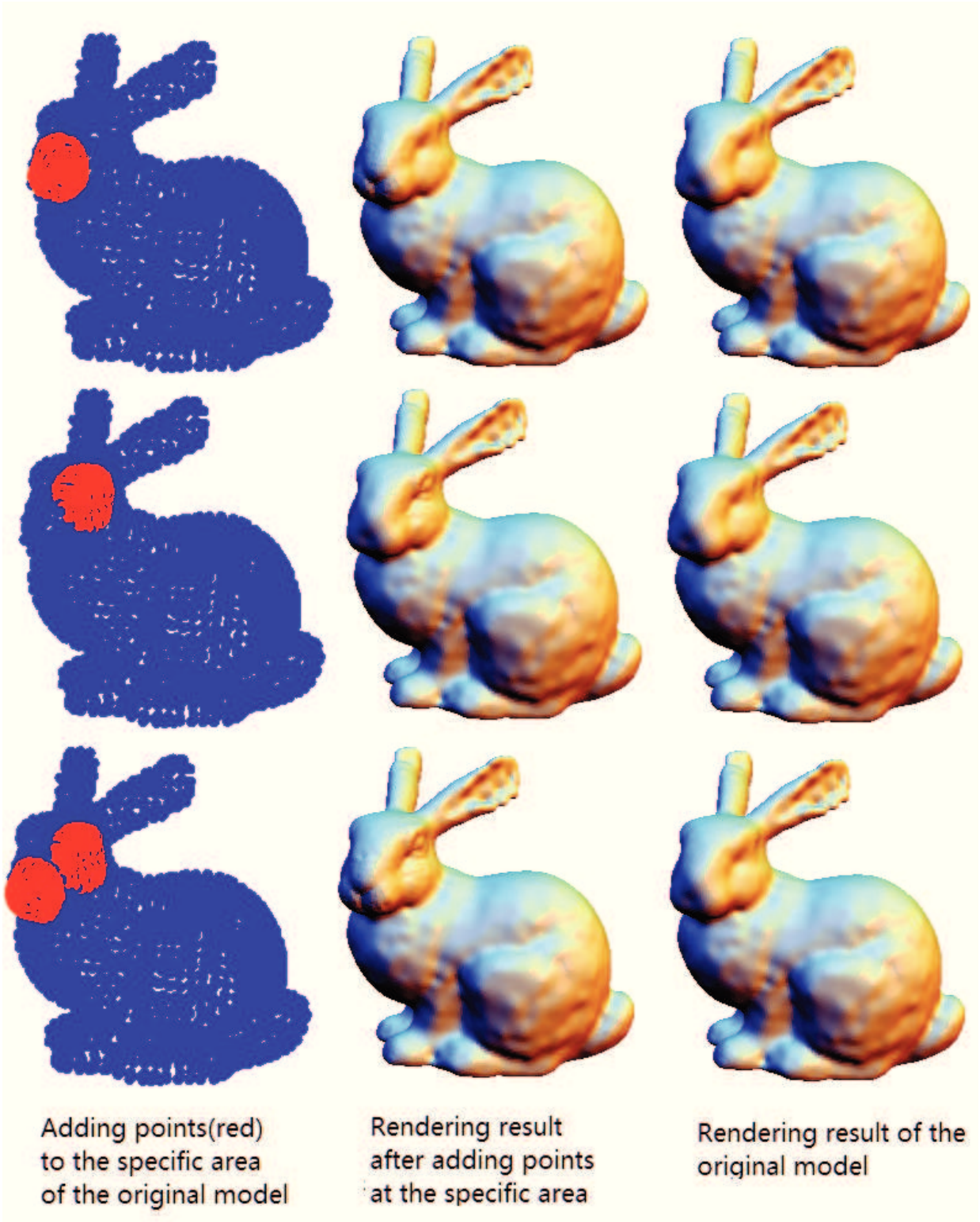

Figure 4.2: Result of *Bunny* model (3778 points) with points added on the mouth (500 points), eye (500 points) and mouth (500 points) plus eyes (500 points), respectively.

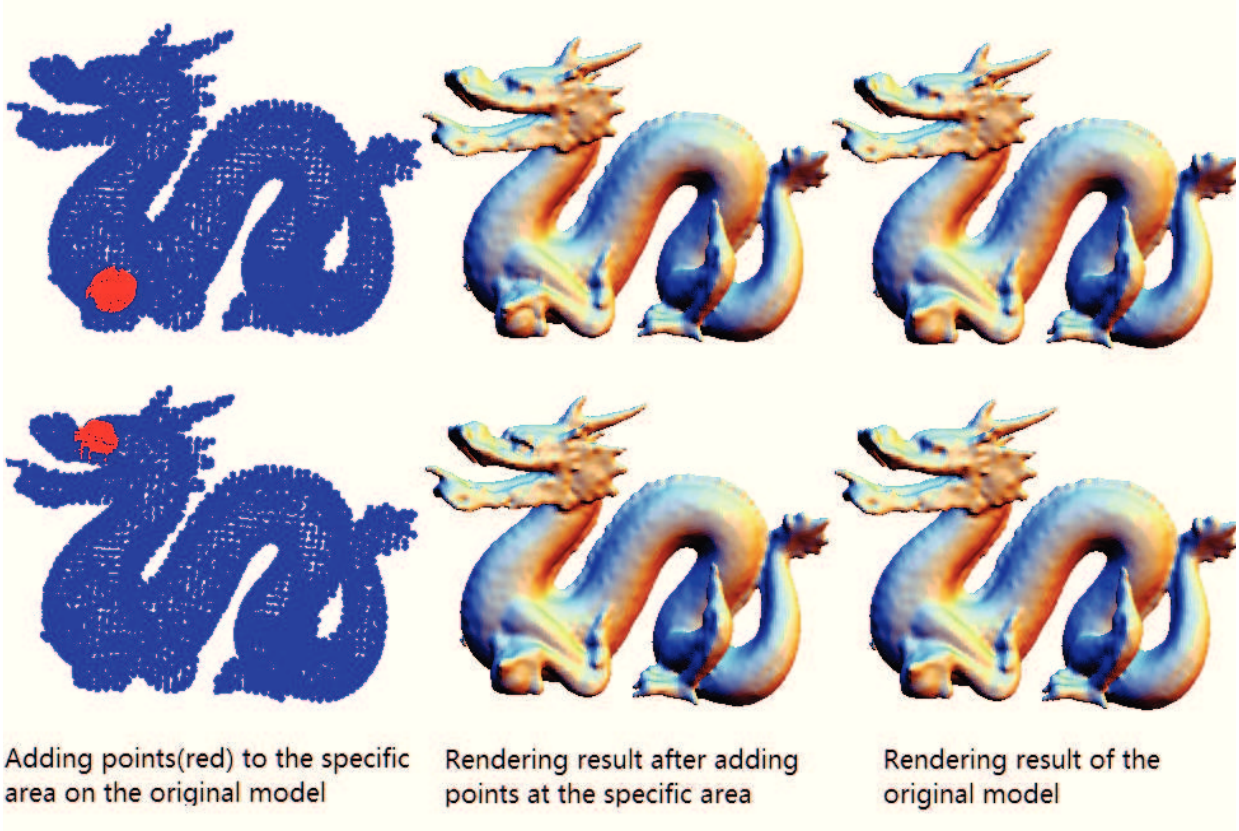

Figure 4.3: Result of *Dragon* model (10410 points) with points added on the claw (2000 points) and eye (2000 points), respectively.

Table 4.1: Point Number of the Rendered Model

Model Name	N_c	N_a	N_t	$R = (N_c + N_a)/N_t$
Bunny	$3,778$	500	$718,94$	5.9%
Dragon	$10,410$	$2,000$	$875,290$	1.4%
Buddha	$14,216$	$3,000$	$1,087,304$	1.5%

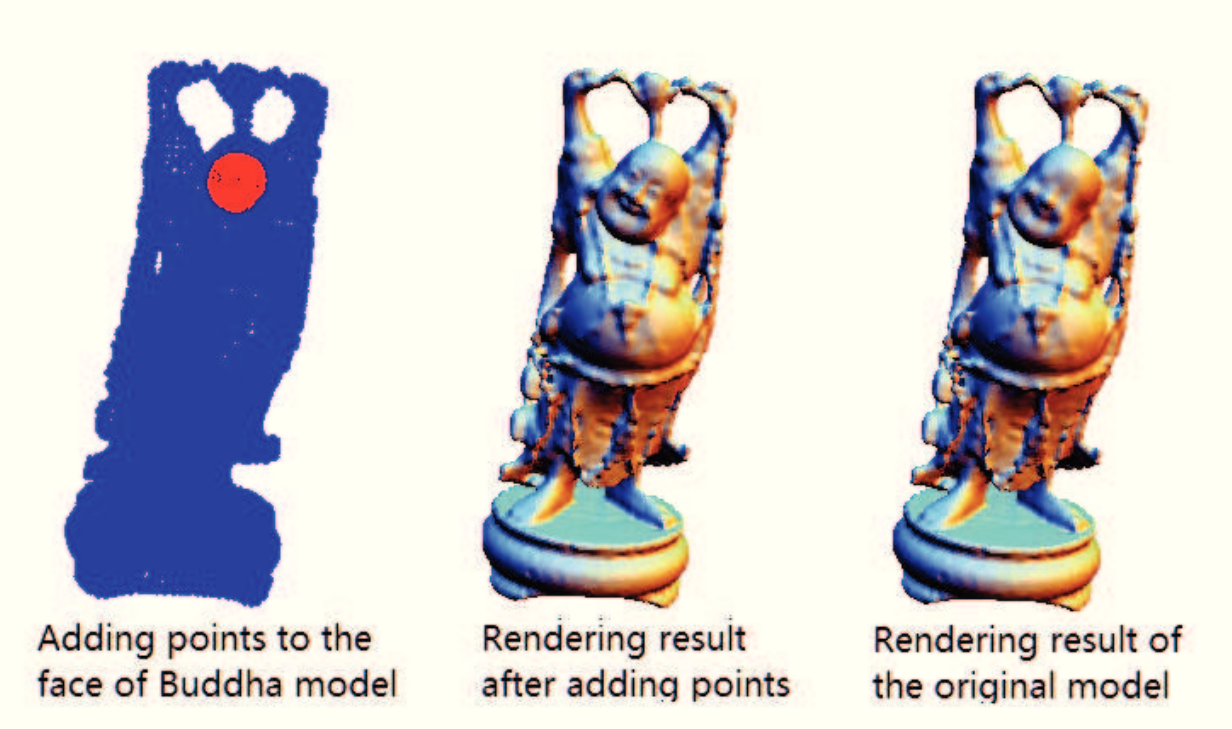

Figure 4.4: Result of *Budhha* model (14216 points) with 3000 points added on the face.

4.5 CONCLUSION AND FUTURE WORK

As the experiment result indicates, users are able to achieve a better resolution in a ROI without having to transmit and render the entire object in high detail.

We are currently working on techniques to automatically detect parts of the object that need more details based on a combination of visual and quantitative metrics. This new technique will make our approach automatic.

Chapter 5

CONCLUSION

5.1 CONCLUSION

The first study in this thesis discusses the use of a FEC algorithm to protect the transmission of progressively compressed 3D point clouds against packets loss. We design a metric to evaluate each layer's quality contribution to the decoding result of the progressively compressed model. With this metric, we minimize the expected distortion when applying an UEP strategy to allocate channel bits to different layers of the model. The performance of employing UEP and Equal Error Protection EEP are compared with respect to the expected distortion. Experimental results show that by incorporating our distortion estimation metric with UEP, the rendering quality of a reconstructed 3D model degrades more gracefully as the packet-loss rate increases.

In the second project, we propose a new method for incorporating 3D point cloud models into multi-viewpoint video. First, we synthesize virtual multi-viewpoint video utilizing depth and texture maps of the input video. Then, we integrate 3D point cloud models with the resulting multi-viewpoint video generated in the first step by analyzing the depth information. Our experiments shows that 3D point clouds can be seamlessly inserted into a multi-viewpoint video and realistic effect can be obtained.

In the third study, we propose a hierarchical RBF based approach to interactively visualize the 3D point cloud. The 3D point cloud is stored by an octree-based hierarchy and interpolated progressively. With this approach users are able to achieve better resolution in an ROI without having to transmit and render the entire object

in high detail.

5.2 PUBLICATIONS

This thesis is mainly based on the following publications.

Feng Chen, Irene Cheng and Anup Basu, ”Distortion Metric for Robust 3D Point Cloud Transmission” IEEE International Conference on Multimedia Expo (ICME), 2009;

Feng Chen, Irene Cheng and Anup Basu, ”Integrating 3D Point Clouds with Multi-Viewpoint Video”, 3DTV Conference, 2009;

Feng Chen, Irene Cheng and Anup Basu, ”Demo Paper: Interactive Visualization of 3D Point Cloud using RBF Based Representation”, IEEE International Conference on Multimedia Expo (ICME), 2013.

Bibliography

[1] M. Alexa, S. Rusinkiewicz, M. Waschbsch, M. Gross, F. Eberhard, E. Lamboray, and S. Wrmlin. Progressive compression of point-sampled models abstract. In *Proceedings of the Symposium on Point-Based Graphics*, 2004.

[2] G. AlRegib, Y. Altunbasak, and J. Rossignac. An unequal error protection method for progressively transmitted 3d models. *Multimedia, IEEE Transactions on*, 7(4):766–776, Aug. 2005.

[3] J. Bloomenthal. Polygonization of implicit surfaces. *Comput. Aided Geom. Des.*, 5(4):341–355, 1988.

[4] Jules Bloomenthal. Graphics gems iv. chapter An Implicit Surface Polygonizer, pages 324–349. Academic Press Professional, Inc., San Diego, CA, USA, 1994.

[5] Mario Botsch, Andreas Wiratanaya, and Leif Kobbelt. Efficient high quality rendering of point sampled geometry. In *Proceedings of the 13th Eurographics workshop on Rendering*, pages 53–64, 2002.

[6] M.Z. Brown, D. Burschka, and G.D. Hager. Advances in computational stereo. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(8):993 – 1008, aug. 2003.

[7] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and Representation of 3D Objects with Radial Basis Functions. In *Computer Graphics (SIGGRAPH '01 Conference Proceedings).*, pages 67–76. Springer, 2001. 1.3.001.

[8] JC Carr, RK Beatson, JB Cherrie, TJ Mitchell, WR Fright, BC McCallum, and TR Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, page 76. ACM, 2001.

[9] J.C. Carr, W.R. Fright, and R.K. Beatson. Surface interpolation with radial basis functions for medical imaging. *IEEE Transactions on Medical Imaging*, 16(1), 1997.

[10] C. Chen and E.C. Prakash. Face Personalization: Animated Face Modeling Approach using Radial Basis Function. *TENCON 2005 2005 IEEE Region 10*, pages 1–6, 2005.

[11] Feng Chen, I Cheng, and A Basu. Distortion metric for robust 3d point cloud transmission. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 770–773, June 2009.

[12] Feng Chen, I Cheng, and A Basu. Integrating 3d point clouds with multi-viewpoint video. In *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2009*, pages 1–4, May 2009.

[13] Feng Chen, I Cheng, and A Basu. Demo paper: Interactive visualization of 3d point cloud using rbf based representation. In *Multimedia and Expo Workshops (ICMEW), 2013 IEEE International Conference on*, pages 1–2, July 2013.

[14] E. Cooke and N. O'Connor. Multiple image view synthesis for free viewpoint video applications. *ICIP*, 1:I–1029–32, Sept. 2005.

[15] Huong Quynh Dinh, Greg Turk, and Greg Slabaugh. Reconstructing surfaces by volumetric regularization using radial basis functions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(10):1358–1371, 2002.

[16] Michael S Floater and Armin Iske. Multistep scattered data interpolation using compactly supported radial basis functions. *Journal of Computational and Applied Mathematics*, 73(1):65–78, 1996.

[17] R. Franke. Scattered Data Interpolation: Tests of Some Methods. *Mathematics of Computation*, 38(157):181–200, 1982.

[18] B. Girod, K. Stuhlmuller, M. Link, and U. Horn. Packet loss resilient internet video streaming. In *Proceedings of SPIE Visual Communications and Image Processing '99*, pages 833–844, 1999.

[19] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations* 1. *Journal of Computational Physics*, 73(2):325–348, 1987.

[20] Markus Gross and Hanspeter Pfister. *Point-based graphics*. Morgan Kaufmann, 2011.

[21] Stefan Guthe, Michael Wand, Julius Gonser, and Wolfgang Straßer. Interactive rendering of large volume data sets. In *Visualization, 2002. VIS 2002. IEEE*, pages 53–60. IEEE, 2002.

[22] R.I. Hartley and A. Zisserman. *Multiple View Geometry in Computer VIsion*. Cambridge University Press, 2nd edition, 2004.

[23] H. Hoppe. Progressive meshes. In *Proceedings of ACM SIGGRAPH'96*, pages 99–108, 1996.

[24] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992.

[25] Artem Ignatov and Manbae Kim. Multi-view video composition with computer graphics. *LNCS*, 4282:292–301, 2006.

[26] Martin Isenburg and Peter Lindstrom. Streaming meshes. In *Proceedings of Visualization'05*, pages 231–238, 2005.

[27] Aravind Kalaiah and Amitabh Varshney. Statistical geometry representation for efficient transmission and rendering. *ACM Trans. Graph.*, 24(2):348–373, 2005.

[28] Hansung Kim, Donghyun Kim, Dongbo Min, and Kwanghoon Sohn. A 3d modeling and free-view generation system using environmental stereo cameras. *Int. J. Imaging Syst. Technol.*, 17(6):367–378, 2008.

[29] Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Computers & Graphics*, 28(6):801–814, 2004.

[30] Eric LaMar, Bernd Hamann, and Kenneth I Joy. Multiresolution techniques for interactive texture-based volume visualization. In *Electronic Imaging*, pages 365–374. International Society for Optics and Photonics, 2000.

[31] Bryan S. Morse, Terry S. Yoo, Penny Rheingans, David T. Chen, and K. R. Subramanian. Interpolating implicit surfaces from scattered surface data using compactly supported radial basis functions. In *ACM SIGGRAPH 2005 Courses*, SIGGRAPH '05, New York, NY, USA, 2005. ACM.

[32] Yutaka Ohtake, Alexander Belyaev, and Hans-Peter Seidel. A multi-scale approach to 3D scattered data interpolation with compactly supported basis functions. In *Shape Modeling Int., 2003*, pages 153–161. IEEE, 2003.

[33] R. Pajarola and J. Rossignac. Compressed progressive meshes. *IEEE Transactions on Visualization and Computer Graphics*, 6:79–93, 2000.

[34] Mark Pauly, Richard Keiser, and Markus Gross. Multi-scale feature extraction on point-sampled surfaces. In *Computer graphics forum*, volume 22, pages 281–289. Wiley Online Library, 2003.

[35] Nick Pears, Tom Heseltine, and Marcelo Romero. From 3d point clouds to pose-normalised depth maps. *Int. J. Comput. Vision*, 89(2-3):152–176, September 2010.

[36] C. Perkins, O. Hodson, and V. Hardman. A survey of packet loss recovery techniques for streaming audio. *Network, IEEE*, 12(5):40–48, Sept 1998.

[37] H. Pfister and M. Gross. Point-based computer graphics. *Computer Graphics and Applications, IEEE*, 24(4):22–23, July-Aug. 2004.

[38] Sebastien Piccand, Rita Noumeir, and Eric Paquette. Efficient visualization of volume data sets with region of interest and wavelets. volume 5744, pages 462–470. SPIE, 2005.

[39] Sébastien Piccand, Rita Noumeir, and Eric Paquette. Region of interest and multiresolution for volume rendering. *Information Technology in Biomedicine, IEEE Transactions on*, 12(5):561–568, 2008.

[40] R. Rohling, A. Gee, L. Berman, and G. Treece. Radial basis function interpolation for freehand 3D ultrasound. In *Information Processing in Medical Imaging*, pages 478–483. Springer, 1999.

[41] Szymon Rusinkiewicz and Marc Levoy. Qsplat: A multiresolution point rendering system for large meshes. In *Proceedings of ACM SIGGRAPH 2000*, pages 343–352, July 2000.

[42] Szymon Rusinkiewicz and Marc Levoy. Streaming QSplat: A viewer for networked visualization of large, dense models. In *2001 ACM Symposium on Interactive 3D Graphics*, pages 63–68, March 2001.

[43] Miguel Sainz and Renato Pajarola. Point-based rendering techniques. *Computers & Graphics*, 28(6):869–879, 2004.

[44] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1-3):7–42, 2002.

[45] Jean-Philippe Thirion. New feature points based on geometric invariants for 3d image registration. *International journal of computer vision*, 18(2):121–137, 1996.

[46] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23(3):600–608, 2004.