Progressive Decomposition of Point Clouds Without Local Planes*

Jag Mohan Singh and P.J. Narayanan

Center for Visual Information Technology
International Institute of Information Technology
Hyderabad, India
{jagmohan@research., pjn@}iiit.ac.in

Abstract. We present a reordering-based procedure for the multiresolution decomposition of a point cloud in this paper. The points are first reordered recursively based on an optimal pairing. Each level of reordering induces a division of the points into approximation and detail values. A balanced quantization at each level results in further compression. The original point cloud can be reconstructed without loss from the decomposition. Our scheme does not require local reference planes for encoding or decoding and is progressive. The points also lie on the original manifold at all levels of decomposition. The scheme can be used to generate different discrete LODs of the point set with fewer points in each at low BPP numbers. We also present a scheme for the progressive representation of the point set by adding the detail values selectively. This results in the progressive approximation of the original shape with dense points even at low BPP numbers. The shape gets refined as more details are added and can reproduce the original point set. This scheme uses a wavelet decomposition of the detail coefficients of the multiresolution decomposition. Progressiveness is achieved by including different levels of the DWT decomposition at all multiresolution representation levels. We show that this scheme can generate much better approximations at equivalent BPP numbers for the point set.

1 Introduction

Polygon-based graphics is useful when the properties – such as color, normals, depth – can be interpolated linearly along a plane, from their values at the vertices. This results in the approximation of the shape for many natural objects. As the graphics capability improves, they get represented using finer and finer polygons. Recently, points have attracted renewed attention as the basic graphics representation primitives [1]. The interest in point-based representations is due to the increase in the resolution of polygon models. The polygons in the graphics models have been shrinking in size for greater accuracy and visual fidelity. Per pixel calculations are made in the graphics hardware in most cases to improve the shading

* This research was carried out with partial funding from the Naval Research Board, India.

P. Kalra and S. Peleg (Eds.): ICVGIP 2006, LNCS 4338, pp. 364–375, 2006.

effects. The display resolution has, on the other hand, been saturating. Thus, it is common for the polygons of a model to be of the same order as a screen pixel in many situations. This makes polygons cumbersome and inefficient to handle. Point-based representations could be more natural and efficient in such situations.

Point based models contain a very large number of points often running into the millions. Representing them in a compressed manner is therefore essential. Multiresolution representation and progressive decoding are important to point based models even more than geometric models. Methods proposed for that typically compute a local plane of support to induce a regular grid to the points [2,3]. This facilitates the application of many standard signal and image compression algorithms to points. These local planes are computationally intensive to find and can cause approximation errors. Since points have no connectivity, they are conceptually independent of one another and can be reordered with no loss in information.

In this paper, we present a simple multiresolution decomposition of a point set based on their proximity without the need for a local plane. Our scheme is based on the reordering of the points that naturally provides multiresolution representation and progressive decoding. Our method reorders and decomposes points successively into approximation and detail sets, loosely similar to the wavelet decomposition. The approximation results in representation of the point set in a different levels of decreasing detail. The detail sets are vector differences between point-pairs. We further reduce the representational complexity of the approximations by changing the quantization based on the sampling rate, keeping a fine balance between sampling and quantization. We provide a progressive representation for the point set. This is done by decomposing the details using DWT up to a certain number of levels and is used to obtain an approximation of the detail. While reconstructing the point set we use the approximated details instead of the original details. This scheme is used for generating an approximation of the point set with given number of bits.

We survey the literature related to the compression of point-based representations in Section 2. Section 3 presents the details of our multiresolution decomposition scheme and decompression. Section 4 shows results and is followed by a few concluding remarks in Section 5.

2 Related Work

Compression techniques have mainly focused on triangle meshes. Triangle based mesh compression mainly focuses on encoding connectivity [4,5,6]. Vertex positions are obtained by quantization followed by predictive coding. Progressive coders allow for better prediction of positions which allow for reconstruction of intermediate shapes by using a prefix of encoded bit stream. These include progressive meshes [7] which uses greedy edge collapses to arrive at lower resolution mesh. Progressive geometry compression [8] eliminates the need for connectivity compression by using semi-regular meshes, wavelet transforms, and zero-tree encoding.

Point based representations allow us to work directly on point data without worrying about connectivity. QSplat [9] uses a multiresolution data structure based on a bounding volume hierarchy. This is optimized for rendering of large point based models such as those obtained through the Digital Michelangelo Project [10]. The preprocessing allows dynamic level of detail selection on the fly. Layered Point Clouds [11] handles large point based models and adjust their sampling according to their projected size on the screen. The sampling technique [12] used is able to handle complex and procedural geometry. Point set surfaces [3] use Moving Least Square (MLS) which is a projection operator which can be used both for upsampling and downsampling of the surface. MLS operator can be used for generating multiresolution point sets [13]. They use a polynomial and a local plane and generate multiple resolutions by varying the degree of the polynomial. The number of choices available for encoding between two consecutive levels are however limited. MLS is a smoothing operator and it smooths the sharp features of the point set surface. Efficient Simplification of Point Sampled Surfaces [14] estimates the curvature of points using local planes. They use curvature and quadric error metric for simplification which is computationally expensive. Progressive compression of point sampled models [15] finds an optimal pairing of points and replace them by their average at lower level of approximation. This is followed by differential coding where the residues are decreased further by the use of local planes and a prediction operator. The residues are coded using a zerotree coder which gives a progressive stream and finally using arithmetic coding. This scheme can be used for generating progressive levels from a given point set at a fixed rate and progressive rate. This scheme is able to handle point attributes such as geometry, color and normals. However, averaging sends the points to outside the manifold and local plane computation which is needed is expensive. Predictive point-cloud compression [16] uses a prediction tree, which is a spanning tree over the vertices. Rooting of the tree defines a partial order. The tree is built greedily by adding those nodes that predicts the new point with smallest residue such as constant and linear. The residues are then encoded by arithmetic coding. This generates multiresolution hierarchy of the original point set. This scheme handles only geometry.

3 Multiresolution Decomposition Using Reordering

We propose a lossless, multiresolution decomposition of the geometry of the point set. Our scheme is based on a reordering of the points. Points are first paired up optimally such that the sum of distances between the pairs is minimum. The pairing induces a partitioning of the points into Odd and Even halves. The Odd half provides a lower resolution representation of the model. This process is repeated recursively with successive Odd sets to get a lossless multiresolution decomposition of the point set. We also adjust the quantization at the lower levels to match the sampling. Detail and quantized approximation are encoded by arithmetic encoding. The multiple resolutions of the representation provide discrete LODs of the original point cloud with decreasing number of points.

3.1 Reordering of Points

Our algorithm uses the minimum weighted perfect matching [17] for pairing up the points. A perfect matching in a graph G is a subset of edges such that each node in G is met by exactly one edge in the subset. Given a real weight c_e for each edge e of G, the minimum-weight perfect-matching problem is to find a perfect matching M of minimum weight $\Sigma(c_e : e \in M)$. An implementation of Edmond's original algorithm will run in time $O(n^2m)$ where n is the number of nodes in the graph and m is the number of edges. Minimum weighted perfect matching uses an improved version of Edmond's algorithm and is bounded by $O(nm\log(n))$. The pairing at each level minimizes the total Euclidean distance between all pairs. The edge weights can additionally include distances between colors and normals if those attributes are also being compressed. We do not construct the complete graph but only an adjacency graph connecting each vertex with its $k = 16$ nearest neighbours.

After applying perfect matching we get odd and even point sets.While choosing the odd point set and even point set after perfect matching we enhance coherence in the even point set. The odd point is chosen such that the vector from the odd point to the even point has positive X, Y, and Z components in that order of priority. This will increase the correlation between the pairing vectors since our intention is to replace the even points with them. The perfect matching would reorder the point sets such that the total distance between the odd and even point sets. After, getting the lowermost odd point set by repeatedly applying perfect matching. We would reorder the point sets such that odd points are in positions $(1, \cdots, N/2)$ and the corresponding even points are in the positions $(N/2+1, \cdots, N)$. This reordering is applied recursively to the odd half of the points so that after reordering the points remain matched. When a point in the odd section is reordered its matching even point should also be reordered to maintain the pairing. If only one level of decomposition is done then the even point set has to be reordered only once. However, if k levels of decomposition is done or perfect matching is applied k times, then the movement of a point at the lowest level can result in reordering of 2^{k-1} points.

3.2 Approximation and Detail

The even point can be replaced by a vector from the matching odd point. If the pairing is done well, these vectors will have similar values and can compress well.

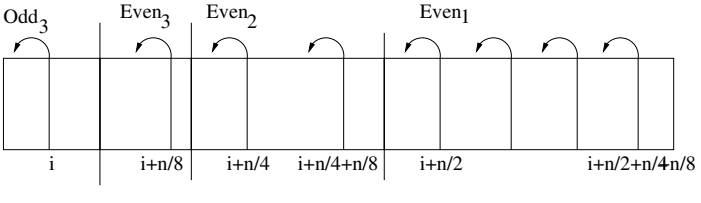

Fig. 1. Point set reordering with three levels of decomposition arrows show the movement of matching points. The positions show where the matched points appear after reordering.

This is performed in all hierarchical decomposition levels. The approximation point set is same as the odd point set. The detail point set is obtained by taking vector difference between the matched even point and the odd point for every point in the even point set. Thus, after k levels of decomposition the approximation is A_k and the detail is D_k. We denote j^{th} point of detail D_i as $D_i[j]$ and j^{th} point of approximation A_i as $A_i[j]$. Note that the lower resolution approximations have the larger index among A_is and D_is. The approximation and detail are shown after three levels of decomposition Figure 1.

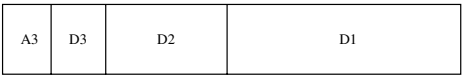

Fig. 2. Approximation and detail after decomposition. $A3$ is the lowest resolution level. $A3$ and $D3$ make up $A2$ in a lossless manner and so on recursively.

If each even point $i + N/2^j$ at level j is replaced by the vector from its odd counterpart i at each step, the decomposition divides the original point set successively into $A_k, D_k, D_{k-1}, D_{k-1}, \cdots D_1$.

The approach used by [15] is similar to the full edge collapse used in triangle meshes. Our approach is similar to the half-edge collapse used in triangle meshes. In triangle meshes the full edge collapse might result in converting a manifold mesh to a non-manifold mesh [18]. The same can happen in point sampled models as the average position is not expected to lie on the manifold. Figure 3 shows the back part of the of the bunny model with positions as average on the left one and retained on the right. The edge weight of the graph constructed in both the cases is the Euclidean distance and attributes are averaged in one case and retained in another. The averaged representation appears more regular, but has the points that are clearly away from the original manifold. This problem is more serious at lower levels of approximation.

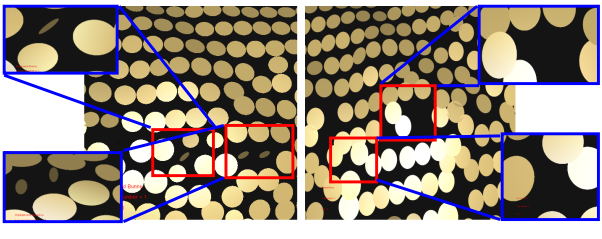

Fig. 3. Comparison between averaging the pairs and retaining one of them. The averaged Bunny (left) have points that move away from the manifold (as shown in the inset), but the retained Bunny (right) has all the points on the original manifold.

3.3 Balanced Quantization and Sampling

It has been established that the quantization and sampling should be matched to each other in point-sampled representations [19]. It is fruitless to represent

coordinates with precision when they are sparsely sampled. Conversely, the quality will be poor if a densely sampled set of points is represented using only a few bits. This can be exploited to gain greater compression ratios at higher levels of decomposition.

The point coordinates are represented using fixed precision integers. The number of bits used should depend on the sampling rate at that level. Twelve to fifteen bits per coordinate will suffice for most practical models used today. Thereafter, when approximation A_{i-1} is split into A_i and D_i, the coordinates of A_i are stripped off the least significant bit. Thus, the coordinates at level i are represented using $p - i$ bits if the original points are represented using p bits. The least significant bits that are collected as an $(N/2^i)$-bit entity E_i of extra information Figure 4.

Hence, our k-level decomposition of points consists of $A_k, D_k, E_k, D_{k-1}, E_{k-1}$ $D_{k-1}, E_{k-2}, \cdots, D_1$, and E_1. The computation of approximation and detail from the point set in our case does not involve the use of local coordinate frames as done in [15].

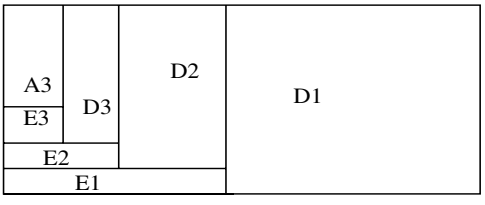

Fig. 4. Point Set after reordering and quantization

3.4 Compressed Representation

The approximation and detail are encoded using arithmetic encoding [20]. The extra detail, obtained during quantization, is packed into a stream of bytes and no encoding is applied as they do not compress much. The encoding is lossless. Given the lower level approximation, detail and extra-detail we can compute the higher level approximation exactly.

3.5 Decompression

The original point set has been reordered so that approximation and detail are matched. Thus decompression is simple and the level up to which we need lossless reconstruction is computed from the same. Hence, in our scheme the data size needed to store the complete progressive point set is same as the original point set. In order to obtain a better approximation, we use the coarsest level of approximation and the encoded detail levels and extra detail levels till we get the required level of approximation. We achieve high compression this way as details and approximation after encoding require less space than the encoded higher approximation level.

Table 1. PSNR and BPP for Different Models. Compression ratio is inversely proportional to BPP.

Level	**Bunny(35k)**	**Santa(75k)**	**GolfBall(122k)**	**Venus(134k)**	**Armadillo(172k)**
0	∞/30.84	∞/41.71	∞/41.29	∞/31.17	∞/42.82
1	45.60/7.65	66.41/12.8	62.38/13.05	49.69/6.96	80.15/12.24
2	44.16/3.74	53.53/5.79	61.30/6.72	48.18/2.99	66.19/5.79
3	43.18/1.74	51.19/2.86	60.68/3.27	47.18/1.41	55.61/2.56
4	42.30/0.91	49.81/1.39	50.17/1.50	46.29/0.67	53.12/1.28
5	41.54/0.47	48.76/0.65	47.46/0.75	45.52/0.30	51.6/0.63
6	40.87/0.23	47.91/0.31	45.86/0.37	44.84/0.16	50.48/0.27
7	40.29/0.14	47.22/0.18	44.71/0.18	44.23/0.08	49.61/0.14
8	-	46.62/0.09	43.74/0.09	43.7/0.04	48.88/0.08
9	-	-	42.93/0.05	43.2/0.02	48.28/0.04

Level	**Lion(183k)**	**Lucy(262k)**	**Heptoroid(286k)**	**Brain(294k)**	**Octopus(465k)**
0	∞/31.57	∞/41.19	∞/22.59	∞/43.01	∞/41.79
1	53.7/8.11	66.26/12.06	65.66/15.03	65.69/13.94	60.33/9.98
2	52.94/3.11	63.37/5.82	61.22/7.05	61.98/6.38	59.36/4.41
3	51.99/1.67	58.11/2.8	56.59/3.27	54.57/3.29	58.35/1.69
4	51.13/0.96	56/1.32	49.5/1.59	50.67/1.56	57.5/0.87
5	50.38/0.47	54.64/0.61	44/0.78	48.82/0.71	56.74/0.48
6	49.72/0.22	53.62/0.27	41.71/0.38	47.60/0.35	56.08/0.21
7	49.15/0.11	52.81/0.13	33.51/0.19	46.72/0.17	55.51/0.10
8	48.65/0.063	52.16/0.07	28.18/0.09	46/0.07	55.02/0.05
9	48.23/0.032	51.65/0.039	25.87/0.04	45.41/0.044	54.57/0.02
10	47.84/0.018	51.22/0.02	24.36/0.021	44.94/0.024	54.19/0.014

3.6 Results

Table 1 shows the PSNR and bits per pixel (BPP) for different point based models after our decomposition. Some of the models at different resolutions are shown in Figure 6. The figure shows the model using a surfel radius that is higher for lower resolution models.

We also give the rate distortion curves for different models (see Figure 5). The compression is lossless hence the reconstructed model from the coarsest approximation and details will match exactly. PSNR is calculated by considering the error induced in the geometry as we go down the levels. The peak signal is the diameter of the bounding sphere of the point set. The mean square error is the cumulative magnitude of the details and the extra-details as we go down the levels. The BPP at a level is calculated as a ratio of the total number of bits in its representation to the total number of points.

We are able to achieve high levels of compression and low BPP values using the scheme. Each lower level has approximately half the number of points as the next higher model. Thus, the levels represent successive approximations using fewer points but at very low bpp numbers. Our method can be thought of as retaining the high level information as we down the lower levels which is more important

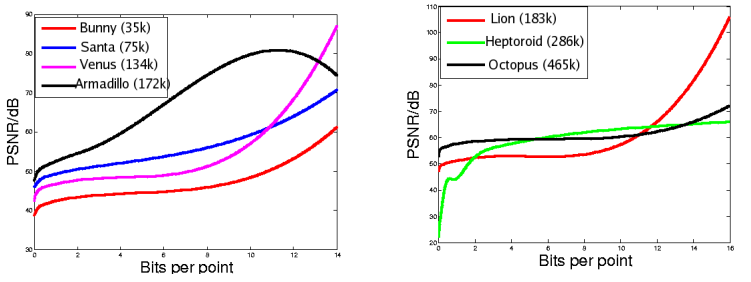

Fig. 5. PSNR/BPP for Bunny, Santa, Venus, Armadillo, Lion, Heptoroid, and Octopus Models

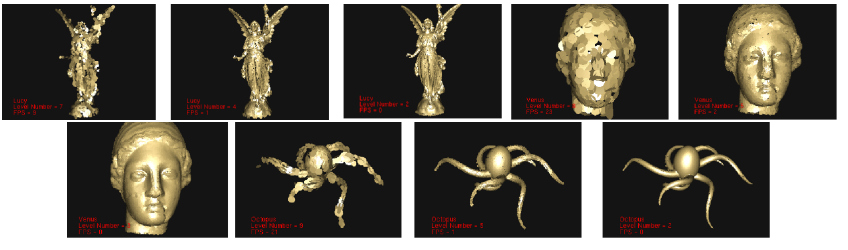

Fig. 6. Left to Right, Top to Bottom: Lucy model at Levels 7 (0.13 BPP), 4 (1.32 BPP) and 2 (5.82 BPP) , Venus at Levels 8 (0.04 BPP), 5 (0.30 BPP) and 3 (1.41 BPP). Octopus at Levels 9 (0.02 BPP), 5 (0.48 BPP) and 2 (4.41 BPP).

for perception and approximating the shape than the low level information which is a result of averaging process. Thus, averaging will lead to low pass version of the signal which is not a true approximation of the original shape. However, retaining the high pass information will give us the important features in the shape and the rest can be filled by changing the surfel radii accordingly.

4 Progressive Representation of the Point Set

The multiresolution decomposition given above differs from a standard wavelet decomposition critically, though there are structural similarities. In a wavelet decomposition, if the detail coefficients are set to zero, an averaged version of the signal is reproduced. This version is approximate but dense, covering the whole domain. In our decomposition, if the detail coefficients are set to zero, the points are repeated. The higher levels do not contain any additional information if a detail value is 0. We need a scheme in which different approximations of D_is can be generated. While a 0 value for D_i repeats points, approximation of D_i can generate a dense representation of the original point set. One way to do this is to approximate the detail values as a $1D$ signal. Different approximations of this signal will contain different details.

4.1 Decomposition

We treat each detail D_i as a one dimensional sequence of slow varying numbers and compress the sequence using DWT. Thus, for each detail D_i, DWT is applied k_i times. k_i is chosen such that the last level has about M points. We used a simple 7 -tap Daubechies wavelet for decomposition and set M as 25. A representation of this is shown in Figure 7 where k_i is $5, 10$, and 15 for D_3, D_2 and D_1 respectively. Let $\mathcal{D}_i$ be the DWT decomposition of D_i.

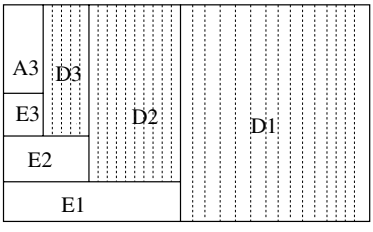

Fig. 7. Point set after reordering, quantization and wavelet decomposition of detail values. This is obtained by decomposing D_3, D_2 and D_1 into $5, 10$, and 15 levels respectively.

4.2 Progressive Representation

Each D_i can be approximated by including only a number of its DWT levels resulting in a smooth version of the D_i sequence. Since, each $D_i[j]$ value acts as a displacement on a point $A_i[j]$, it generates another point in the representation. This results in a better approximation of the point set. A dense representation of the point set with as many points as the original point set can be obtained if we include some of all D_is in the representation. If we set all D_is to 0 for $i < j$, a representation with as many points as the approximation level A_j can be obtained. We give a procedure to generate an approximation of the point set with M points and S number of bits given a model with N original points and k multiresolution decomposition levels.

The above algorithm can produce approximations of the point set with different number of points and total size. The combination of A_k and $\mathcal{D}'_i, E'_i,\ k \leq i < j$ is a compact representation of the point set. They can be used to reconstruct the model in two steps. First, approximate D'_is for each level is found by applying IDWT on each $\mathcal{D}_i$, setting the missing coefficients to 0. Next, an approximation A'_j of the point set is generated using the decompression technique given in Section 3.5 using A_k and D'_is and E_is. In our experiments, we allocate 80% of the bits at every level to $\mathcal{D}_i$ and the rest to E_i.

4.3 Results

We give the rate distortion curves for different models (see Figure 8) using progressive representation. Since the decomposed model has exactly the same

Algorithm 1. Progressive Representation(M, S)

1: Find level $j = \lceil \log_2 N/M \rceil$ with more than M points. Skip D_is for all $i < j$.
2: Include the lowest level A_k. Subtract its size from S. This is the number of bits available for D_is and E_is.
3: Allocate these bits equally among the levels from k to j.
4: **for** $i = k$ to j **do**
5: Let S_i be the size allocated to level i. A fraction r of it is used for D_i and rest for E_i.
6: If S_i is greater than the combined size of $\mathcal{D}_i$ and E_i, set $\mathcal{D}'_i \leftarrow \mathcal{D}_i$ and $E'_i \leftarrow E_i$.
7: Otherwise, construct $\mathcal{D}'_i$ with as many DWT coefficients of $\mathcal{D}_i$, starting with the most approximate level, such that the combined size is rS_i and construct E'_i with $(1-r)S_i$ bits of E_i
8: **end for**
9: Return A_k and $\mathcal{D}'_i, E'_i$ for i= k to j

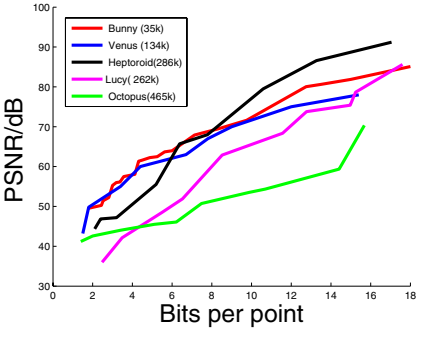

Fig. 8. Rate distortion curve for Bunny, Venus, Heptoroid, Lucy and Octopus models for the progressive representation. These cannot be directly compared directly with Figure 5 as the PSNR is calculated in a different way here as explained in the text.

number of points as the reordered original point-set, the PSNR can be calculated by taking from the error between the corresponding points. This is a better measure of quality unlike those used in Section 3 or by Waschbüsch et al. [15] The peak signal is the diameter of the bounding sphere of the point set. BPP is calculated by taking into account number of bits used to go till the higher most level of approximation using the approximation and reconstructed details. Lucy Model at different BPP is shown in Figure 9 with increased radius for hole free appearance. We compare the quality of models achieved by progressive representation and multiresolution decomposition Figure 10. Note that higher BPP are required in progressive representation but the model has fewer holes. We use the procedure progressive representation for to generate any number of points.

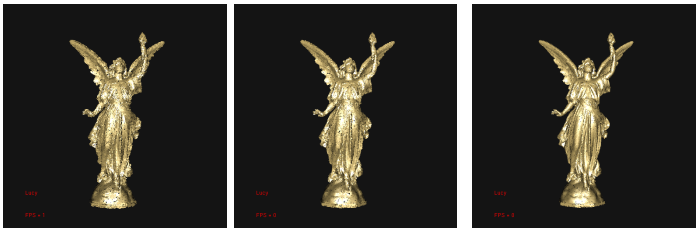

Fig. 9. Lucy Model at 2.27 BPP, 3.16 BPP, and 8.24 BPP respectively using the progressive representation

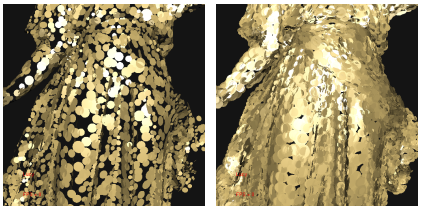

Fig. 10. Lucy Model at 3.11 BPP from the multiresolution decomposition (left) and progressive representation (right) with same radius for both. Progressive representation is visually superior.

5 Conclusions and Future Work

We presented a simple, reordering based algorithm to decompose a point set into multiple resolutions. The algorithm is based on optimal pairing and decomposes the points into a low resolution approximation and a series of detail vectors. The points of all approximation levels fall on the original manifold. We are able to get further compression using balanced quantization at every sampling level. The multiresolution decomposition provides discrete levels of detail to the point set. We also present a progressive representation of the point set by compressing the detail vectors using wavelets. By selectively including different numbers of coefficients of the wavelet decomposition at each detail level, we are able to get a wide range of representations for the point set, ranging from the lowest approximation to a totally lossless representation. The progressive representation scheme can be used to generate a model with the given number of points and a given BPP. Progressive representation results in better visual appearance compared to the multiresolution decomposition.

Currently, the decomposition is performed on the whole point set. This decreases the coherence of the detail vectors. Partitioning the points into different parts of the model and treating each part independently will perform better. The detail vectors will be more coherent and will compress well using DWT. We can also select different progressive levels for different parts of the point set based on proximity or importance. We are exploring these ideas currently.

References

1. Kobbelt, L., Botsch, M.: A survey of point-based techniques in computer graphics. Computer and Graphics **28** (2004) 801–814
2. Pauly, M., Gross, M.: Spectral processing of point-sampled geometry. In: SIGGRAPH '01. (2001) 379–386
3. Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., Silva, C.T.: Computing and rendering point set surfaces. IEEE Transactions on Visualization and Computer Graphics **9** (2003) 3–15
4. Gumhold, S., Strasser, W.: Real time compression of triangle mesh connectivity. In: SIGGRAPH '98. (1998) 133–140
5. Rossignac, J.: Edgebreaker: Connectivity compression for triangle meshes. In: IEEE Transactions on Visualization and Computer Graphics. Volume 5. (1999) 133–140
6. Touma, C., Gotsman, C.: Triangle mesh compression. In: Graphics Interface. (1998) 26–34
7. Hoppe, H.: Progressive meshes. In: SIGGRAPH '96. (1996) 99–108
8. Khodakovsky, A., Schroder, P., Sweldens, W.: Progressive geometry compression. In: SIGGRAPH '00. (2000) 271–278
9. Rusinkiewicz, S., Levoy, M.: Qsplat: a multiresolution point rendering system for large meshes. In: SIGGRAPH '00. (2000) 343–352
10. Levoy, M., Pulli, K., Curless, B., Rusinkiewicz, S., Koller, D., Pereira, L., Ginzton, M., Anderson, S., Davis, J., Ginsberg, J., Shade, J., Fulk, D.: The digital michelangelo project: 3d scanning of large statues. In: SIGGRAPH '00. (2000) 131–144
11. Gobbetti, E., Marton, F.: Layered point clouds. In: Eurographics Symposium on Point Based Graphics. (2004) 113–120, 227
12. Stamminger, M., Drettakis, G.: Interactive sampling and rendering for complex and procedural geometry. In: Rendering Techniques 2001 (Proceedings of the Eurographics Workshop on Rendering 01). (2001)
13. Fleishman, S., Cohen-Or, D., Alexa, M., Silva, C.T.: Progressive point set surfaces. ACM Trans. Graph. **22** (2003) 997–1011
14. Pauly, M., Gross, M., Kobbelt, L.P.: Efficient simplification of point-sampled surfaces. In: VIS '02: Proceedings of the conference on Visualization '02. (2002) 163–170
15. Waschbüsch, M., Gross, M., Eberhard, F., Lamboray, E., Wurmlin., S.: Progressive compression of point-sampled models. In: Proceedings of the Eurographics Symposium on Point-Based Graphics. (2004) 95–102
16. Gumhold, S., Karni, Z., Isenburg, M., Seidel, H.P.: Predictive point-cloud compression. In: Proceedings of the Sixth Israel-Korea Bi-National Conference. (2005) 125–129
17. Cook, W., Rohe, A.: Computing minimum-weight perfect matchings. INFORMS Journal on Computing **11** (1999) 138–148
18. Luebke, D., Reddy, M., Cohen, J.D., Varshney, A., Watson, B., Huebner, R.: Level of Detail for 3D Graphics. Morgan Kaufmann (2003)
19. Botsch, M., Wiratanaya, A., Kobbelt, L.: Efficient high quality rendering of point sampled geometry. In: EGRW '02: Proceedings of the 13th Eurographics workshop on Rendering. (2002) 53–64
20. Moffat, A., Neal, R.M., Witten, I.H.: Arithmetic coding revisited. ACM Trans. Inf. Syst. **16** (1998) 256–294