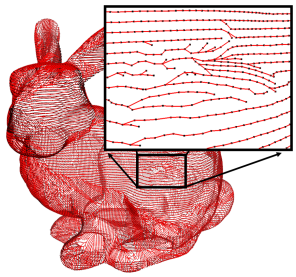
Predictive Point-Cloud Compression

Stefan Gumhold* Zachi Karni Martin Isenburg Hans-Peter Seidel
MPI für Informatik Saarbrücken

a)



b)

c)

model	n	pred	tree	geod	$\min_{x/y/z}$	KD	PC
fandisk	6k	lin	1.24	13.04	12.94z	12.1	20.69
horse	20k	lin	1.80	17.41	17.43y	16.4	21.22
grass14	29k	con	1.54	19.88	19.74z	18.8	21.76
maple01	45k	con	1.57	18.11	17.94x	16.9	16.31
bunny	35k	lin	1.15	14.48	14.31x	14.8	18.52
santa	75k	lin	1.23	12.23	13.17y	-	18.28
chameleon	102k	lin	1.33	6.23	6.35z	-	16.47
igea	134k	lin	1.16	10.83	10.90z	-	14.28
male	148k	lin	0.89	7.29	7.26y	-	13.59

Figure 1: prediction trees built with a) constant prediction (maple01 has no fine-scale structure) and b) linear prediction (bunny is a 3d scan); c) comparison of compression rates in bits per point (bpp) after uniform quantization to 12 bits per coordinate, i.e. 36 bpp uncompressed data

1 Introduction & Related Work

Point clouds have recently become a popular alternative to polygonal meshes for representing three-dimensional geometric models. 3D photography and scanning systems acquire the geometry and appearance of real-world objects in form of point samples. Rendering directly with points eliminates the complex task of reconstructing a surface and allows handling of non-surfaces like models such as trees. With modern acquisition techniques producing larger and larger amounts of points, efficient schemes for compressing such data have become necessary.

Several methods for compressing point clouds have been proposed. Two methods that recursively subdivide the bounding box are the kd-tree approach of Devillers and Gandoin [2000] and the oct-tree approach of Peng and Kuo [2003]. One drawback of these spatial subdivision methods is that they do not generalize to include attribute data such as normals and colors. The method of Waschbüsch et al. [2004] generates a binary tree over the points by pairing close-by points. Each pair is replaced by its centroid to form the next coarser level. This approach generalizes to include attributes. Coding starts at the coarsest level using a predictive scheme and local coordinate systems.

2 Predictive Point Cloud Coding

Our method compresses the points in a spatially sequential order that supports single resolution streaming. Points are predicted from previously coded neighbors with simple prediction rules such that only corrective vectors need to be encoded. However, unlike in mesh compression, where such prediction techniques are commonly used, we do not have connectivity information to determine neighboring points. It is in general not possible to permute the points into a sequential order that allows to guess good neighbors for prediction. Thus we augment the data by a *prediction tree* – a spanning tree over the vertices. Rooting the tree defines a partial order on the points. For prediction of a point all ancestors can be used. Coding is done in a breadth first order after tree construction.

Prediction Rules: The user can choose among two prediction rules: 1) in constant prediction (con) the point $\vec{p}_i$ is predicted from the parent position $\vec{p}_{pred} = \vec{p}_{i-1}$, 2) in linear prediction (lin) also the grandparent point $\vec{p}_{i-2}$ is used according to $\vec{p}_{pred} := 2\vec{p}_{i-1} - \vec{p}_{i-2}$.

Construction of Prediction Tree: Inspired by Kronrod and Gotsman [2002], we build a prediction tree that minimizes the residuals, i.e. the lengths of the corrective vectors. Instead of solving a global optimization problem we construct the tree in a sequential *build order*. After initializing the tree to the first point, each successive point is greedily attached to the node that predicts the new point with the smallest residual. For fast search of the best parent node, the predicted locations of all current tree nodes are hashed into a regular grid over the bounding box. For construction we investigated two build orders: sorted along a coordinate axis or sorted in an approximate geodesic order from the first point. For this the geodesic distance is approximated by the minimal path length in the k-nearest neighbor graph. Figure 1 shows sample trees.

Coding: The tree is coded by entropy coding its valences in breadth first order, which results in bit rates lower than the two bits per point needed by a standard tree coding scheme. The corrective vectors are compressed component wise with arithmetic coding.

3 Results & Conclusion

We compare our results to the bit-rates that are reported in [Devillers and Gandoin 2000](KD) and [Waschbüsch et al. 2004](PC). For the latter we use software from the authors. The table in Figure 1 c) compares the resulting compression rates in bits per point (bpp) after 12 bit quantization. The column "n" gives the number of points, "pred" the used prediction rule, "tree" the bpp used to encode the tree, "geod" the total bpp for the geodesic build order including the cost for the tree and "$\min_{x/y/z}$" the lowest total bpp among the axis aligned build orders. Compression times are about 20 seconds per 100k points and decompression is 500k points per second on a 2GHz PC.

Point clouds from 3d scans and resampling techniques (lower part of table) have a fine-scale structure that is exploited by our technique resulting in significantly better compression rates even without the use of local coordinate systems as proposed in [Waschbüsch et al. 2004]. Furthermore, the method can be easily generalized to streaming compression and handling of attributes like color and normal using the same prediction rules.

References

DEVILLERS, O., AND GANDOIN, P.-M. 2000. Geometric compression for interactive transmission. In *Visualization 2000*, 319–326.

KRONROD, B., AND GOTSMAN, C. 2002. Optimized compression of triangle mesh geometry using prediction trees. In *Proceedings of 3DPVT-02*, 602–608.

PENG, J., AND KUO, C. C. J. 2003. Octree-based progressive geometry encoder. In *Proceedings of the SPIE*, 301–311.

WASCHBÜSCH, M., GROSS, M., EBERHARD, F., LAMBORAY, E., AND WÜRMLIN, S. 2004. Progressive compression of point-sampled models. In *Eurographics Symposium on Point Based Graphics*, 95–102.

*e-mail: sgumhold@mpi-sb.mpg.de