Fast Vertex Transformation for 3D Rendering through Predictive Vector Quantization

Peter H. Chou and Teresa H. Meng
Electrical Engineering Department
Stanford University
Stanford, California 94305
e-mail: pchou@stanford.edu, meng@mojave.stanford.edu

Rendering geometrically detailed 3D polygonal models is both bandwidth and computationally intensive, requiring the transfer and processing of massive amounts of geometry data. In this work, we show vector quantization to be an effective vertex data compression scheme that in addition to reducing geometry bandwidth requirements, can be used to accelerate the computation of vertex transformations.

Compatibility is maintained with most existing connectivity encoding schemes (including hardware-oriented representations designed for direct rendering) by defining the vector process to be coded from the sequence of vertex positions encountered during mesh traversal. Predictive vector quantization is used to exploit the correlation between vectors. Both full-search and product pyramid lattice vector quantizers were tested. A rate of approximately 11 bits/vertex (2048-entry codebook) was observed for tested models to produce only a slight degradation in visual quality.

Our compression scheme not only reduces the number of bits required for vertex representation, it also permits linear transformations to be applied to the vertices with a significantly reduced number of computations. Since a decoded vertex is given by the sum of its prediction and corresponding residual codeword, its linear transformation is given by the sum of its transformed prediction and transformed residual. Using a linear predictor allows the transformed prediction to be computed as a linear combination of previous transformed vertices. The transformed residuals are pre-computed independently of the vertices by applying the linear transformation to the codebook. Since the size of the codebook is fixed and expected to be much smaller than the number of vertices in the mesh, significantly fewer computations are performed.

For example, the modeling transformation in the first stage of the graphics pipeline is traditionally implemented as a 4x4 matrix multiplication. For a mesh of \(N\) vertices and a codebook of size \(M\), the proposed transformation process would require \(16M\) multiplications and \(12M\) additions to transform the codebook, plus \(9N\) additions for computing the predictions (using a third-order linear predictor with coefficients \(\{1, 1, -1\}\)) and adding the codewords. When the number of vertices is large relative to codebook size, the number of arithmetic operations performed is reduced to a fraction of those required by traditional methods. Furthermore, the number of multiplications is fixed and independent of the number of vertices in the model, so the savings in number of multiplications grows proportionally with the number of vertices rendered.