CS7491 - 3D Complexity Techniques for Graphics, Modeling, and Animation
Lecture Notes #23

Jarek Rossignac

April 4th, 2006
Contents

1 Project 3 3

2 Part I: Prediction of Geometry 3

3 Part II: Build a kD tree 3
   3.1 Building the tree 3
   3.2 Compression for the tree 5
   3.3 Other compressions for the tree 6

4 Encoding connectivity 6
1 Project 3

Project 3 is due on April 25th at beginning of class. Teams can be made of 1, 2 or 3 people. It needs to be focused on reliability and trust of the results. For the 3 parts, use all the curves provided for project1 quantize to 16 bits. The purpose of this project is to study lossless compression of those data.

2 Part I : Prediction of Geometry

Extrapolate : using the best prediction(linear or quadratic...). Those corrections will be stored as follow :

- The sign : 1 bit.
- Length of the magnitude (b) : 4 bits.
- The magnitude : the data after throwing away the leading zeros. We can even get rid of the first bit : always '1'.

We will only use entropy encoding for the length of the magnitude. Using it on the sign is useless or on the magnitude which is the thing we cannot predict. Let $E = \text{Entropy}(b)$ and $A$ the average size of the magnitude minus 1. The size of the data will be : $T = E + A + 1$ per coordinates. So the savings will be : $16 - T$.

Report the results for all curves from project 1.

3 Part II : Build a kD tree

3.1 Building the tree

To build a kD tree :

![Figure 1: A kD tree.](image)

Figure 1: A kD tree.
We do as in figure 2 recursively until each cell contains less than 2 vertices (0 or 1).

![Figure 2: Splitting the image into a kD tree.](image)

To deal with “where to cut?” just keep track of indices of begin and end of a cell. In figure 3, the first cut is: \((0, 3) \Rightarrow ((0, 1)(2, 3))\).

![Figure 3: Interval problem.](image)

The kD tree of figure 3 is equivalent to the tree in figure 4.

![Figure 4: The real tree.](image)

We can write the data of each point by interleaving their coordinates: \(x_0y_0x_1y_1 \ldots x_{15}y_{15}\). Each split is equivalent to gathering the data with the same \(x_i\) or \(y_i\). A big cell in the tree is a big blocks of cell with no point. The path in the tree is part of this interleaved data. And the detail at a leaf is the rest of the data: details = 32 – depth.
1. Encode the tree.

2. Then, find the status of a leaf: '0' or '1'.

3. When it is a '1', save the details.

With this compression: the savings are in the tree!

Building the tree:

- \( \text{Split(cell, setOfVertices)} \), cell represented by \((\text{min}, \text{max})\). Do this split recursively.
- Or sort the interleaved data. See figure 5.

\[
\begin{array}{c}
0000010 \\
0000110 \\
0110001 \\
1100010 \\
1100100 \\
1110010 \\
\end{array}
\]

Figure 5: Building the tree.

3.2 Compression for the tree

How to compress the tree: It is a binary tree. It can be compressed by “is a leaf or not?”: this is 1 bit per node. We also need a bit per node for “is the leaf empty of vertices?”. If we have \(v\) vertices and \(z\) empty leaves:

- “is a leaf or not?” \(\Rightarrow 2(v + z) - 1\) bits.
- “is the leaf empty of vertices?” \(\Rightarrow (v + z)\) bits.
- The total cost for the tree and leaf classification is \(3(v + z) - 1\) bits. So with the geometry, it is: \(3(v + z) - 1 + vA\), where \(A\) is the average length of the details.

Statistics on \(v\) and \(z\):

- If \(v = z\), the cost is \((6 + A)v - 1\). So if \(A\) is small enough, this is a good scheme.
3.3 Other compressions for the tree

Try and compare those schemes:

1. Gandoin and Devillers 2002: encode the number of vertices in left-child. It might be a lot more expensive.

2. Botsch Et Al 2002: Encode 1 symbol out of 3 for each node: 0+,++,+0.

3. Lewiner 2005: Encode 1 symbols out of 6 per node ++,+1,11,1+,+0,0+.

For all those schemes, don’t forget to encode the details at each 1-vertex leaf.

4 Encoding connectivity

In this section we focus on how to retrieve the connectivity lost during building the kD tree.

Figure 6: Rebuild connectivity.

Or worse?
For a manifold segment of curve: how to develop a cell?

Figure 7: Rebuild connectivity.