INTERNATIONAL ORGANISATION FOR STANDARDISATION
ORGANISATION INTERNATIONALE DE NORMALISATION
ISO/IEC JTC1/SC29/WG11
CODING OF MOVING PICTURES AND AUDIO

ISO/IEC JCTC1/SC29/WG11 MPEG/m52525
January 2020, Brussels, Belgium

Source:	Apple Inc.	
Status:	Input document	
Title:	G-PCC: Common test conditions for multi-frame sequence coding	
Author(s):	David Flynn	*davidflynn@apple.com*
	Khaled Mammou	*kmammou@apple.com*

Abstract

The current G-PCC reference software supports sequence coding where multiple frames are encoded in a single output sequence. The current common test conditions include multi-frame sequences obtained from LiDaR scanners. However, the common test conditions do not make use of this functionality. This contribution suggests the changes necessary to enable multi-frame sequence coding.

Introduction

Using the frameCount and firstFrameNum configuration options, a single invocation of the TMC13v8 software [1] is able to encode a sequence of source frames to produce a single compressed bitstream. Conversely, the decoder will produce a sequence of reconstructed frames from the compressed bitstream.

The category three test data contains a number of time varying multi-frame LiDaR sequences. However, due to the nature of the data and the default encoder configuration, directly coding these sequences using the multi-frame functionality will fail.

Sequence bounding box and bipolar position data

The sequence parameter set contains an optional bounding box that may be used to indicate the location of the point cloud in a larger scene (or to shift the local origin) and to determine its bounds. While the bounds do not form part of the decoding process, the location (offset) does.

In particular, the geometry octree coder is only able to process positive integer positions. This conflicts with the bipolar nature of 360° LiDaR data. As such, it is necessary to offset the point cloud prior to encoding and to remove the offset afterwards. Currently, the SPS bounding box is used for this purpose.

The current method for determining the bounding box involves scanning all points for the minimum and maximum positions. While this works for a single frame, it is not feasible for a multi frame sequence. In such cases, either the sequence bounding box must be known a priori, or it must be omitted. Since there is a necessary requirement to convert signed positions to unsigned, the only solution is a priori configuration.

For LiDaR data, positions are bounded by the nature of the sensor. For example, the Velodyne VLP-16 sensor used to acquire the QNX sequences has a range of approximately 100m and a ±15° vertical range. This would result in lateral displacements of up to ±100,000mm and vertical displacements up to ±26,000mm from the sensor origin.

Table 1 shows the sequence level bounding box determined for each category three test sequence.

Natural values for the sequence bounding box origin and size are respectively $(-2^{17}, -2^{17}, -2^{17})$ and $(-2^{18}, -2^{18}, -2^{18})$.

Table 1 – Sequence level bounding box sizes for category three content

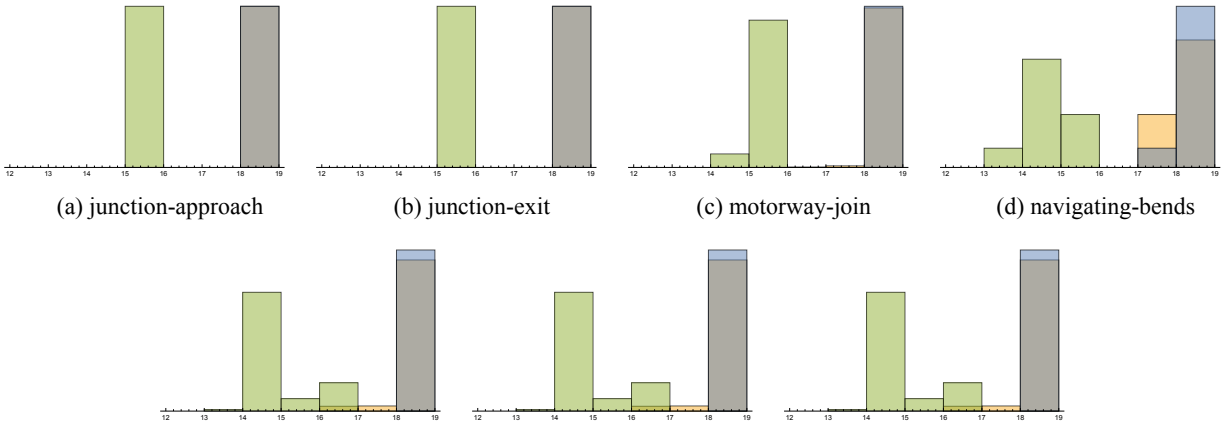
Sequence	x0	y0	z0	x1	y1	z1	w	h	d
ford_01_q1mm	−115448	−115321	−44408	115351	115392	4080	230799	230713	48488
ford_02_q1mm	−115402	−115138	−44366	114804	115467	4078	230206	230605	48444
ford_03_q1mm	−115292	−114954	−44429	115589	115800	4071	230881	230754	48500
qnxadas-junction-approach	−98201	−128497	−6752	124614	124323	19708	222815	252820	26460
qnxadas-junction-exit	−100793	−122169	−11263	125045	128513	21598	225838	250682	32861
qnxadas-motorway-join	−129494	−127322	−11205	128057	125984	27253	257551	253306	38458
qnxadas-navigating-bends	−120709	−116593	−6740	123375	126566	20257	244084	243159	26997

Interaction with non-cubic octrees

Originally, the codec supported only cubic geometry volumes. However, the addition of the so-called QtBt feature permits efficient coding of non-cubic volumes. As a result, using the configuration above results in a significant coding loss, since the effect of QtBt is negated.

Furthermore, using a constant offset for all frames removes the opportunity to exploit variation on a per-frame basis.

Figure 1 illustrates the distribution of bounding box dimensions for all frames in the category three LiDaR sequences.



(a) junction-approach (b) junction-exit (c) motorway-join (d) navigating-bends

(e) ford_01 (f) ford_02 (g) ford_03

Figure 1 – Histograms of per-frame ⌈Log2⌉ bounding box dimensions. x = Blue, y = Yellow, z = Green

To resolve both of these cases, the per-slice offset feature should be used to appropriately offset each slice. The current TMC13v8 software no longer calculates this offset as part of the slice partitioning feature and should be fixed. However, this may be worked around by disabling slice partitioning for these sequences (slice partitioning has no effect) and forcing the slice offset to be appropriately signalled.

```
diff --git a/tmc3/encoder.cpp b/tmc3/encoder.cpp
index a2ea39d..6e8e997 100644
--- a/tmc3/encoder.cpp
+++ b/tmc3/encoder.cpp
@@ -146,6 +146,7 @@ PCCTMC3Encoder3::compress(
       }
     }
   } else {
+    _sliceOrigin = quantizedInputCloud.computeBoundingBox().min;
     tileMaps.emplace_back();
     auto& tile = tileMaps.back();
```

```
     for (int i = 0; i < quantizedInputCloud.getPointCount(); i++)
@@ -154,8 +155,6 @@ PCCTMC3Encoder3::compress(

   // If partitioning is not enabled, encode input as a single "partition"
   if (params->partition.method == PartitionMethod::kNone) {
-    // todo(df): params->gps.geom_box_present_flag = false;
-    _sliceOrigin = Vec3<int>{0};
     compressPartition(
       quantizedInputCloud, inputPointCloud, params, callback,
       reconstructedCloud);
```

Random access (and maintaining experiment compatibility)

In order to truly support random access in a sequential stream, it is necessary to repeat the parameter sets on a per-frame basis. This is currently the behaviour of the TMC13v8 encoder. Maintaining this behaviour has the additional effect of not altering the coding performance relative to single frame coding of multi-frame sequences as has been used on the project so far.

Proposed configuration and results

The following configuration changes are proposed for the category three frame sequences:

```
seq_bounding_box_xyz0: '-131072, -131072, -131072'
seq_bounding_box_whd: '262143, 262143, 262143'
partitionMethod: 0
```

An experiment is performed according to the common test conditions [2, 3] to verify the coding performance of the proposed configuration. Results are presented in Table 2.

Table 2 – Comparison of the proposed configuration against CTC results using octree geometry coding with LoD attribute coding

		BPP Ratio [%]			BD-Rate [Δ%]						Avg. of ratio maxrssk [%]		Ratio of avg. runtime [%]	
Condition	Class	Geometry	Colour	Refl	D1	D2	Y	Cb	Cr	R	Encoder	Decoder	Encoder	Decoder
C1_ai	cat3-frame									0.0	97	100	97	98
C2_ai	cat3-frame				−0.2	−0.1				0.7	96	100	96	101
CW_ai	cat3-frame	100.0		100.0							97	100	100	102
CY_ai	cat3-frame									−0.0	97	100	108	109

References

[1] 3DG, “G-PCC Test Model v8,” ISO/IEC JTC1/SC29/WG11, 128th meeting, Geneva, Tech. Rep. w18882, Oct. 2019.

[2] ——, “G-PCC performance evaluation and anchor results,” ISO/IEC JTC1/SC29/WG11, 128th meeting, Geneva, Tech. Rep. w18885, Oct. 2019.

[3] ——, “G-PCC Future Enhancements,” ISO/IEC JTC1/SC29/WG11, 128th meeting, Geneva, Tech. Rep. w18887, Oct. 2019.