DEPTH CODING BASED ON BINARY TREE DECOMPOSITION

Gonçalo Nuno Silva do Carmo

Instituto de Telecomunicações – Grupo de Imagem
Instituto Superior Técnico
Av. Rovisco Pais, 1049-001 Lisboa, Portugal
E-mail: goncalo.do.carmo@ist.utl.pt

ABSTRACT

We propose in this paper a content adaptive depth coding solution - called IST-Depth - using a binary tree to represent the decompositions of a depth map; where each node of the tree represents a depth triangle (thus classifying this tree as a triangular tree, or simply tritree) and the tree connectivity represents decompositions of the triangles. Each depth triangle is constituted by three depth samples which are differentially encoded and then entropy encoded. The connectivity of the tree is pre-sorted by a breadth based scanning algorithm and then encoded using the 7-zip software. IST-Depth codec was compared to JPEG, H.264/AVC and MVC standards and showing very promising results in terms of PERR quality while showing a worse performance for PSNR quality.

Index Terms — Depth coding, 3D Systems, Binary Tree Decomposition, Depth Assessment.

1. INTRODUCTION

Since the world is a 3D world, there has been a growing interest in 3D video systems, largely motivated by the novel capabilities that acquisition, representation and display technologies are boosting and thus targeting the improvement of the multimedia user experiences. This interest has been reflected in the following markets:

- **3D TV** – Confirm by the fact that the last World Cup (2010), which is the most watched event in the whole world, had twenty five games available in 3D [1] for many countries around the world;
- **3D Cinema** – Confirmed by the fact that Disney announced that, in the future, all its animated movies will be released in 3D [2];
- **3D Displays** – Naturally, there are no 3D experiences without 3D displays; nowadays there are more and more 3D displays in the market, such as the Hyundai 3D TV [3].

In this context, 3D systems must be able to support 3D features and application efficiently, meaning that the tools in 3D systems are responsible for the effectiveness of the resources (namely bits). Thus, the presence of depth data – which provides information regarding the geometry of the scene plays a special role, since it brings two main advantages:

- It strongly enhances the user’s 3D experience by increasing the number of available 3D views by synthesizing new, additional views based on the transmitted views;
- It allows reducing the cost of the 3D experiences for the users and service providers since, in terms of bits, some 3D tools become more efficient.

In this sense, a 3D system using depth information besides the color information is able to more efficiently support the necessary 3D features and capabilities - such as HMPN (Head Motion Parallax Navigation), notably if an efficient depth coding solution is available.

2. REVIEWING DEPTH CODING TECHNIQUES

Before introducing some of the depth coding solutions, this section introduces the depth specific properties, in order to ease some of the depth coding analysis at a later stage.

2.1. About Depth Data

Depth map is a pixel based map where each value measures the distance between the object and the camera, as represented in Figure 1; hence, the depth values range varies from \( z_{\text{near}} \) to \( z_{\text{far}} \) and corresponds respectively, to the minimum and maximum distances between the camera and the objects in the scene. Typically, each depth sample is quantized with 8 bits, i.e. the closest point is associated with the value 255 and the most distant point with the value 0 as shown in Figure 1 [2].

Generally, depth maps have very smooth depth changes for each object and sudden changes in their edges. Figure 1 exemplifies well some of these depth maps main properties, notably:

- Smooth regions in the objects, e.g. in the bottle and in the glasses;
- Sharp edges between smooth regions corresponding to the objects’ boundaries, e.g. between the top of the bottle and the background.
Depth images are not to be seen directly by the user, contrary to what happens with luminance signals. This has a major impact on the type of metrics appropriate to measure the quality of depth images; this should be taken into account in efficient depth coding.

2.2. H.264/AVC Depth Coding

A depth map is a 2D representation of the depth of a scene; however, it may be taken as a texture/luminance map and coded using any available video codecs which are able to code luminance data, such as H.264/AVC standard, which is the most efficient video coding standard available.

Concerning the coding process, one of the main strengths of this approach is the fact that powerful spatial and temporal correlation exploitation tools are used; on the other hand, these tools do not consider in any way the depth specific properties which may mean that more efficient tools may be used. The fact that this codec only exploits texture properties instead of depth properties may result in ringing artifacts along the depth edges, leading to errors in depth positions which may be especially critical in terms of strong visual synthesis artifacts.

2.3. Hierarchical Decomposition Mesh-based Depth Coding

In [5], Kim and Ho have proposed a lossy depth coding solution based on a hierarchical decomposition of the depth map, i.e., decomposing a depth map four different kinds of images. At the decoder’s side, these four images are able to reconstruct a depth map which - through an interpolation technique that was not described - generates a mesh. After the hierarchical decomposition, three of these four images are merged into a single image; then, the two resulting images are coded with H.264/AVC.

The coding performance for the scheme proposed in this section is slightly worse than the coding performance of H.264/AVC, since depth have very high temporal and spatial correlations. Depth maps were decomposed into four new images, each one with different properties, and these new images are coded with an encoder that mainly takes into account visual properties despite these new images are not to be seen by the users, leading to the conclusion that this encoder is not optimized to encode these images. Thus, a better compression ratio can be expected if an encoder taking these image’s properties into account is used, although at the cost of losing compatibility with already implemented 2D systems.

2.4. Platelet-based Depth Coding

In [6], Merkle et al. propose a novel lossy depth coding approach taking into account an important depth property which is the fact that smooth regions are delineated by sharp edges. Therefore, the algorithm models smooth regions with a piecewise constant or a linear function and sharp edges with a straight line.

This depth coding approach seems to consider more the depth properties than the previous ones; interestingly, this is shown through the rendered images and not directly through the depth RD performance which highlights the issue regarding the best way to evaluate the performance of a depth coding solution. However, due to its high complexity, this depth coding solution still cannot be used for real time operation.

2.5. Content Adaptive Mesh-based Depth Coding

In [7], Sarkis, Zia and Diepold propose a lossy image codec based on the content adaptive mesh-based depth coding scheme, which takes the challenge to define the non-uniform sampling process for a depth image, in order to reconstruct the depth content with high quality.

Considering the coded depth maps, the PERR performance is more important than the MSE performance because the first metric may lead to better synthesized views. So, although the proposed performance shows worse MSE performance than JPEG, it shows a better PERR performance which in this case is more relevant.

The adaptive sampling process proves to handle well depth properties. Moreover, using lossless encoding (of the selected depth samples) allows preserving the depth discontinuities which leads to better rendered views.

3. PROPOSED DEPTH CODING SOLUTION

The main objective of the proposed depth coding solution - designated as IST-Depth from now on - is to provide an intra depth coding solution that can compete with the H.264/AVC Intra (coding mode); this objective intends to avoid performing motion estimation for the depth coding since a more detailed study should be made in order to explore the temporal correlation of the data produced by IST-Depth.

3.1. Basic Technical Approach

IST-Depth adjusts the depth samples' rate accordingly to the depth regions and its variations; thus, this implies using the depth samples to obtain the remaining depth data (the non-sampled depth). This is done by creating triangular depth regions – or simply triangles - where the triangle’s vertices corresponds to the depth samples and the remaining depth data is obtained by an estimation/reconstruction function which only uses the triangular depth region’s vertices information. This solution is based on the depth coding solution proposed by Sarkis in [7]; however, it includes significantly algorithmic novelties.
3.2. Architecture

The architecture of IST-Depth is presented in Figure 2 - as mentioned above, this architecture is rather similar to the depth coding solution architecture proposed by Sarkis in [7] - and regards the coding process of a single depth map without exploiting the temporal correlation if a depth map sequence is to be coded, as it would happen while using a H.264/AVC Intra coding solution.

![Diagram of Encoder and Decoder Architecture](image)

Naturally, the encoder and decoder processes are very complementary in terms of the various modules functionalities, except for two architectural modules: Depth Analysis and Depth Map Reconstruction. All the other architectural modules are responsible for the lossless encoding/decoding processes of the associated information, this means the selected depth samples’ values and the binary tree; this implies the amount of depth distortion introduced in the coding process, essentially depends on the Depth Analysis and Depth Map Reconstruction modules.

Since IST-Depth adopts an analysis by synthesis approach, this means the encoder takes decisions after knowing what these decisions will imply at the decoder’s side, the reconstruction tools to be used at the decoder’s side have also to be present at encoder’s side. Therefore, the encoder includes most the tools of IST-Depth architecture: analysis, reconstruction and encoding tools. Thus, there is only described the encoder’s modules, in order to avoid repeating information.

Although IST-Depth architecture only includes lossless encoding processes, overall, this codec may be lossy since the data selection made in the Depth Analysis module, implies the remaining depth data to be approximated/interpolated and therefore making the final depth quality very dependent on the accuracy of the approximation algorithm. Summarizing, IST-Depth coding solution is both lossy and lossless depending on the used coding parameters. In the context of this coding paradigm, the lossless case occurs when all depth regions are perfectly approximated, when all depth regions are just composed by its vertices, or when the two previous conditions happen simultaneously.

3.3. Functional Description by Module

The list of the functions at the encoder’s side by module is the following:

- **Depth Analysis** – Selects from the raw depth map the relevant samples for the coding process accordingly to a set of input parameters related with the desired final quality;
- **Binary Tree Encoding** – Changes the representation of the binary tree, produced in the Depth Analysis module, into another representation which is more efficient in terms of bits and in a lossless fashion;
- **Depth Values Encoding** – Changes the representation of the depth values, produced in the Depth Analysis module, into another representation which is more efficient in terms of bits and in a lossless fashion;
- **Multiplexer** – Aggregates and orders all data produced by the Depth Values Encoding and the Binary Tree Encoding modules.

3.4. Algorithmic Description by Module

This section intends to provide a detailed algorithmic description for each architectural module in of the IST-Depth codec; the goal is not only to detail the implemented solution but also to motivate and justify the solutions adopted and options made.

3.4.1. Depth Analysis

The Depth Analysis module is by far the most complex module in the proposed architecture; throughout this section, besides the algorithmic concepts, some mathematical concepts are also introduced whenever they have a relevant role for the understanding of the algorithmic tools. The overall flowchart for the Depth Analysis module is shown in Figure 3 while the description of each of its modules is made in the four following sub-sections.
The input to this module is a full depth map and the output is a sequence of symbols resulting from the operations performed by this architectural module on the full depth map, notably a set of depth samples and a binary tree with the depth samples associated location in the full depth map.

3.4.1.1. Initial Triangular Decomposition

This module’s flowchart is shown in Figure 4 and its walkthrough is the following:

- **Input**: The rectangular shaped Original Depth Map is a matrix - with the same dimensions of a frame - containing all the original depth information in a frame; this allows to define the location of the triangle’s vertices using their Cartesian coordinates (x,y);
- **Defining the Left/Upper Triangle**: The initial top frame’s corners and bottom-left frame’s corner correspond to the initial left/upper triangle;
- **Defining the Right/Lower Triangle**: The bottom frame’s corners and the top-right frame’s corner correspond to the initial right/lower triangle;
- **Numbering the Triangles Vertices**: The vertices of each of the two triangles are numbered independently using a raster scanning order over those vertices in the Original Depth Map; this means that the first vertex of each triangle to be found using a raster scanning order is numerated as 1, the second as 2 and the third as 3. The numeration is important in order to define which side of the triangle should be divided in case of a further decomposition.
- **Output**: The final output of this module are two sets of three ordered depth samples and positions in which each set defines a triangle.

3.4.1.2. Triangular Decomposition

The Triangular Decomposition module has two main purposes:
- Decompose any given triangle (input) into two triangles (outputs) if required by the quality/distortion control metric;
- Renumber the new two triangles’ vertices using a raster scanning order.

To accomplish these purposes, the operations in the flowchart presented in Figure 5 are described as follows:

- **Input**: A set of three depth samples defining the triangle to be further decomposed;
- **Computing the Triangles Sides Euclidean Distances**: The Euclidean distances between all the triangles vertices are computed as follows to determine the longest side of the triangle:
  \[
  s_1 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \\
  s_2 = \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2} \\
  s_3 = \sqrt{(x_3 - x_1)^2 + (y_3 - y_1)^2}
  \]

  where \((x_1,y_1), (x_2,y_2)\) and \((x_3,y_3)\) are the Cartesian coordinates of the first, second and third triangle vertices, respectively, and \(s_1, s_2\) and \(s_3\) are the length of the first, second and third triangle sides. The first side is defined by the first and second triangle's vertices, the second side is defined by the second and third triangle's vertices and, finally, the third side is defined by the third and first triangle's vertices;
- **Computing the Longest Side Midpoint**: This module computes the longest side midpoint coordinates using Equations 4 and 5, where the vertices \(a\) and \(b\) define the longest side.
  \[
  x_m = \frac{x_a + x_b}{2} \\
  y_m = \frac{y_a + y_b}{2}
  \]

  After this computation, it is necessary to sort the new midpoint vertex among other vertices according to a raster scanning order; therefore, the following tests are made (by omission, go to the next operation after each positive test):

  1. Is the first triangle side the longest? No: Go to 3;
  2. The midpoint vertex is numbered as 2 and the second and third triangle vertices are numbered as 3 and 4, respectively;
  3. Is the second triangle side the longest? No: Go to 5;
4. The midpoint vertex is numbered as 3 and third vertex is numbered as 4;
5. Is the third triangle side the longest? No: Exit;
6. Is the midpoint vertex before (according to the raster scanning order) the second vertex? No: Go to 8;
7. Execute 2 and then exit;
8. Is the midpoint vertex after (according to the raster scanning order) the second vertex? No: Exit;
9. Execute 4 and then exit.
• Output: The final output of this module are two sets of three ordered depth samples and positions in which each set defines a triangle.
An example of this algorithm's input and output is shown in Figure 6.

![Figure 6 - a) Triangle to be decomposed (input) where position 3 (in this case) corresponds to the midpoint in the largest edge; b) Left/upper depth triangle; c) Right/lower depth triangle; d) Renumbered left/upper depth triangle (output); e) Renumbered right/lower depth triangle.](image)

After this decomposition process, each triangle can be independently analyzed, meaning that parallel processing can be explored; therefore, each triangle can be processed by different CPUs and the results can be gathered at the end of each independent process.

### 3.4.1.3. Planar Approximation

The Planar Approximation module has the important target to estimate the depth values inside each intermediate and final triangle generated from the full depth map decomposition, thus strongly determining the final reconstructed depth quality. As mentioned before, the quality of a triangle will be measured both in terms of PERR and MSE/PSNR. The flowchart in Figure 7 shows all operations executed in this module; each operation is described in the following:

• **Input:** The input to this module is sets of three depth samples and their location which it is the minimum necessary to define a triangle. The previous modules always provided two sets of three depth values and their locations – one set representing a left/upper triangle and the other representing a right/lower triangle – therefore, it was defined that the left/upper triangles have priority over the right/lower triangles, this type of information is not relevant here, however will be useful for the Binary Tree Encoding module which follows this same definition when the triangles are decomposed;

• **Defining the Counting Process:** This module’s goal is to avoid useless computations and tests in the next modules over some depth positions (a depth position is any type of coordinates that can provide the location of a depth value). This implies to select the smallest rectangular region (depth window) as possible, instead of the full depth map, which contains the triangle under study in order to be able to efficiently count the number of depth samples inside the triangle and to estimate their reconstructed depth value. With this purpose in mind, two distinct phases are present in this sub-module:

A. **Depth Window Creation:** Based on the Cartesian coordinates, the depth window defines the smallest rectangular region in the Original Depth Data which contains the triangle under processing;

B. **Defining a Scanning Pattern:** Also based on the Cartesian coordinates of the triangle's vertices, a scanning pattern is defined allowing some depth values to be skipped in further computation/tests. A scanning pattern corresponds to a primary direction and a secondary direction; the primary direction is defined as the direction that is never repeated and the secondary direction is defined as the direction that is repeated successively along the primary direction. Four different scanning patterns are illustrated in Figure 8.
The decision on the scanning pattern to be used is only based on the Cartesian coordinates of the triangles’ vertices as follows (by omission, go to the next operation after each positive test):

1. Are there any vertices with a common xx coordinate (meaning that the side defined by those vertices is superimposed to a row of Original Depth Data)? No: Go to 5;
2. Define the primary direction as top-down;
3. Has the vertex with the different xx coordinate a higher xx coordinate than the others which have a common coordinate? No: Define right-left as a secondary direction and exit;
4. Define left-right as a secondary direction and exit;
5. Are there any vertices with a common yy coordinate (meaning that the side defined by those vertices is superimposed to a column of Original Depth Data)? No: Choose default pattern (raster scanning pattern) since is expected to the computation and tests are approximately the same for all patterns. Then exit;
6. Define the primary direction as left-right;
7. Has the vertex with the different yy coordinate a higher yy coordinate than the others which have a common coordinate? No: Define top-down as a secondary direction and exit;
8. Define down-top as a secondary direction and exit.

Now that a scanning pattern has already been defined, the number of depth positions to be tested can be reduced to almost half (for the best case scenario) of the total different depth positions inside the depth window by performing the following operations (by omission, go to the next operation after each positive test):

1. Chose the first depth position, accordingly to the scanning pattern selected, and then execute 2;
3. Did the previous depth position belong to the triangle? No: Execute 5.
4. Is there more depth positions to test accordingly to the scanning pattern and that belong to the depth window? Yes: Select the next depth position and execute 2. No: Exit;
5. Skip the rest of the depth positions in this current pattern’s secondary direction and then execute 4.

All the algorithms present in this module seems to be novelties of IST-Depth codec since in the proposed codec by Sarkis in [7], nothing was mentioned regarding any module or any process that similarly avoids the computations and tests over the full depth map.

- **Barycentric Coordinates Converter**: In order to count the all the depth positions that were inside a triangle, it is essential to know if a certain depth position is inside the triangle or not. By employing the Barycentric coordinates [8], which are homogeneous coordinates defined by the vertices of a ‘simplex’ (in this case a triangle), it is easy to detect if a sample \( a \), with coordinates \( (x_a, y_a) \), belongs to a triangle \( T^{123} \), with vertices \( (x_1, y_1), (x_2, y_2) \) and \( (x_3, y_3) \), just by checking the values of the sample’s Barycentric coordinates (which is done in the next module). The following linear transformation is employed to convert depth sample position \( a \) from Cartesian coordinates to Barycentric coordinates:

\[
\begin{align*}
  w_1 &= \frac{(y_2 - y_3)(x_a - x_3) - (y_3 - y_2)(x_a - x_2)}{(y_1 - y_3)(x_2 - x_3) - (y_3 - y_1)(x_2 - x_1)} \\
  w_2 &= \frac{(y_3 - y_1)(x_a - x_1) - (y_1 - y_3)(x_a - x_3)}{(y_1 - y_3)(x_2 - x_3) - (y_3 - y_1)(x_2 - x_1)} \\
  w_3 &= 1 - w_1 - w_2
\end{align*}
\]

where \((w_1, w_2, w_3)\) are the Barycentric coordinates of depth sample \( a \);

- **Testing Triangle’s Depth Positions**: Using the Barycentric coordinates to check if a sample \( a \) belongs to a triangle \( T^{123} \) boils down to a simple test as follows:

\[
0 \leq w_i \leq 1, \quad \forall i = 1, 2, 3
\]

with a positive answer meaning that the (“mass”) center of sample \( a \) is inside triangle \( T^{123} \). This method is extremely helpful to compute the triangle’s size (TriSize) which is defined as the number of depth positions contained in the triangle under study. The minimum number of depth positions inside a triangle is 3, with each depth position corresponding to a triangle’s vertex and thus achieving 100% reconstruction accuracy since the vertices’ depth values are encoded in a lossless fashion;

- **Updating Triangle Size**: This sub-module just counts the number of depth values which passed the previous test, thus determining the size (TriSize) of the triangle under study;

- **Estimating Samples Depth**: Once again, the Barycentric coordinates eases yet another problem; in this case, the problem to estimate the depth value of a depth position. Being \( d_a \) [9] the depth value estimated for the sample \( a \), \( d_a \) is simply computed as:

\[
\hat{d}_a = w_2d_2 + w_3d_3 + w_5d_5
\]

where \( d_1, d_2 \) and \( d_3 \) are the depth values of the triangle’s vertices and \((w_1, w_2, w_3)\) are the Barycentric coordinates of reconstructed depth sample \( a \);
• **Updating Triangle Quality**: Another novelty of IST-Depth is in the way that the PERR is computed. The PERR proposed by Sarkis represents the percentage of errored pixels in a region relatively this same area’s region (i.e. relatively to the number of pixels inside that region); this means that PERR changes very quickly, up and down, over decompositions since the area is decreasing and thus one errored pixel starts to have a heavy weight in the final PERR. In order to solve this issue, the percentage of errored pixels is computed relatively to the area of the full depth map, i.e. to the total number of pixels in a depth map, which is constant. Therefore, the triangle’s MSE and PERR are computed as follows:

\[
MSE = \sum_{(x_i,y_i)} (d(x_i,y_i) - \hat{d}(x_i,y_i))^2 \quad (11)
\]

\[
PERR = \frac{1}{W \times H} \sum_{i=1}^{TriSize} f(d(x_i,y_i), \hat{d}(x_i,y_i)) \% \quad (12)
\]

where TriSize represents the triangle’s size, \(W\) and \(H\) are the depth map width and height, respectively, \((x_i,y_i)\) are the coordinates of a depth value belonging to the triangle, \(d(x_i,y_i)\) is the original depth value for the corresponding coordinates and \(\hat{d}(x_i,y_i)\) is the estimated depth value for the corresponding coordinates. The function \(f\) returns the value 0 if the original depth value \(d\) and the reconstructed depth value \(\hat{d}\) are the same; otherwise, if these values are different, \(f\) returns the value 1.

3.4.1.4. Assessing the Triangle’s Quality and Size

This sub-module has the target to assess each triangle’s quality to check if the user defined quality targets have been reached or not, meaning that further triangle’s decomposition(s) would be required for the reconstruction function to perform a better approximation. This is clearly an analysis by synthesis solution where the encoder simulates the decoder’s behavior to check if the target performance is being achieved.

As mentioned before, the quality/distortion control parameters adopted are the PERR and the MSE/PSNR simultaneously. To avoid very small triangles from being generated - since this would increase too much the bitrate - a quality/complexity tradeoff is also available: the user may also control the minimum size a triangle may achieve. In summary, two types of controls are possible:

1. **Quality Thresholds**

   - **TriPERR<sub>Threshold</sub>**: This threshold defines the minimum required PERR quality for all triangles, meaning that every triangle must have a PERR quality lower or equal to this PERR threshold;
   - **TriMSE<sub>Threshold</sub>**: This threshold defines the minimum required MSE quality for all triangles, meaning that every triangle must have a MSE quality lower or equal to this MSE threshold.

2. **Complexity Thresholds**

   - **TriSize<sub>Threshold</sub>**: This threshold defines the number of different depth positions inside a triangle after which the triangle cannot be further decomposed. Due to topological reasons, this value must not be lower than three (which is the case where the triangle is composed only by its vertices); any value above that provides a complexity/quality tradeoff.

In this context, this module’s effective function is to assess the triangle’s quality and to check if it has to be further decomposed in order to achieve the target quality and if it can be further decomposed in terms of complexity; therefore, the following tests are performed:

\[
\text{TriMSE} > \text{TriMSE}_{\text{Threshold}} \land \text{TriPERR} > \text{TriPERR}_{\text{Threshold}} \quad (13)
\]

\[
\text{TriSize} > \text{TriSize}_{\text{Threshold}} \quad (14)
\]

A positive answer to both checks means that the triangle must be further decomposed to achieve the target quality and can be further decomposed in terms of complexity; otherwise, the triangle is considered to be a final triangle, this means a triangle part of the final depth map decomposition.

Regardless the outcome of these checks, a label is generated for each triangle; the label ‘0’ is generated whenever the triangle fulfills the both test, this means it is not final, and the label ‘1’ is generated otherwise, this means the triangle is final. Since each decomposition generates two (children) triangles for each (parent) triangle, the tree used to represent the decompositions should be binary; therefore, this binary tree is capable of representing all the decompositions performed in this architectural sub-module.

3.4.2. Binary Tree Encoding

The **Binary Tree Encoding** module has the target to efficiently encode the generated binary tree which is indirectly representing the location of the final triangles vertices.

The binary tree is firstly pre-ordered through a DBS (Depth Based Scanning) algorithm which performs as follows (note that the label’s symbol is immediately written to a file/stream as the node is visited):

1. Push the first left/upper node before pushing the first right/lower node into a LIFO (Last In First Out) list;
2. Pull and visit a node from the LIFO list;
3. If the current visited node has children, push them into the list (always push the left/upper child after the right/lower child);
4. Exit if the FIFO list is empty; otherwise, go to 2.

Then, the binary tree is entropy encoded. No specific entropy coder was developed here and it was decided to use the available 7-Zip software [10]; this entropy coding solution is thus able to create the binary tree bitstream to be provided to the Multiplexer architectural module. To encode a bitstream representing a binary tree with the 7-Zip software, the following parameters were used: Archive Format: 7z; Compression level: Ultra; Compression method: LZMA; Dictionary Size: 64 MB; Word size: 64; Solid Block size: 4 GB; Number of fast bytes: 255.
3.4.3. Depth Values Encoding

The Depth Values Encoding module has the target to efficiently encode the generated depth values which are representing the vertices of the final triangles. A differential representation is proposed for the triangles’ depth values. Since the depth values, in PCM (Pulse-Code Modulation) format, vary from 0 to 255 (8 bit/depth sample), the range of depth’s differences is larger, this means from -255 to +255, corresponding to the transitions from 255 to 0 and from 0 to 255, respectively. Generally, this differential process changes the statistical distribution of the depth values to be entropy coded into a peak-shaped histogram, which is more suitable for entropy encoding processes. Therefore, the depth values are then entropy encoded. A statistical distribution study for the various differential depth values was made in order to find their probabilities; however, their statistical distribution is very dependent on the target quality. Therefore, in order to avoid defining a different entropy encoder, e.g. a Huffman code, for each target quality, the differential depth values are coded as if they have at their disposal an ideal code; this means using their ‘entropy’ computed for the current depth map. Since there are optimal codes that can achieve approximately 98% of the maximum efficiency corresponding to the approach above, coding depth values in this ideal way can be considered to be a realistic approximation since a real code with a very similar representation efficiency may be defined.

3.4.4. Multiplexer

The Multiplexer module has the target to put together the depth values and binary tree corresponding bitstreams. Due to the binary tree’s properties, by keeping track of the bits provided by Binary Tree Encoding module, it is possible to know when all the information regarding the encoding process of a single depth map is complete by the following reasons:

- The binary tree has always one more leaf (corresponding to the number of decomposition labels with value ‘1’) then the total number of nodes (corresponding to the number of decomposition labels with value ‘0’); therefore, when the number of labels ‘1’ is superior by one to the number of labels ‘0’, means the bitstream of the binary tree have ended since (for the previous defined BBS algorithm) this case only occurs when the final node is scanned;
- The number of expected depth values can be indirectly known by accessing the binary tree labels, since each decomposition has an associated depth position.

Due to these two reasons, the bitstream provided by Binary Tree Encoding architectural module is firstly received then the bitstream provided by the Depth Values Encoding architectural module.

Overall, at the decoder’s side, it is necessary to have the binary tree to know the number of depth values to be expected and the binary tree is a type of data which automatically flags its own end, the multiplexing in this architectural module is simply done by firstly gathering all the binary tree’s bits and after all the depth values’ bits for each depth map.

4. PERFORMANCE EVALUATION

The IST-Depth codec performance evaluation study is made by comparing the results obtained for the IST-Depth codec with the results obtained for well know coding standards. To ease the comparison of these results with other previous works, only the mandatory parameters, i.e. the parameters related with direct properties of the sequences, were altered to run the tests for the standards

4.1. Test Conditions and Metrics

This section targets to fully present and motivate the test conditions and metrics used to assess the performance of the developed IST-Depth codec.

- Tested Multiview Sequences: “Breakdancers” and “Newspaper” in Figure 9 and in Figure 10 respectively;

![Figure 9 - “Breakdancers” color and depth example frame.](image)

![Figure 10 - “Newspaper” color and depth example frame.](image)

- Depth Spatial Resolution: Both have 1024x768;
- Depth Temporal Resolution: 25 Hz and 30 Hz, respectively;
- Number of Views: 8 and 3, respectively;
- Number of Frames per View: 100 and 300, respectively;
- Performance Metrics: PERR and PSNR;

4.2. Depth Codecs under Comparison

The codecs under comparison can be grouped into two types: benchmarking and proposed codecs.

4.2.1. Benchmarking Codecs

Four different benchmarking codecs will be used in this performance evaluation.

4.2.1.1. JPEG Standard

Besides the natural changes regarding the test conditions described earlier, no more changes were made in its default
parameters, except for the quality parameter; since the RD points for this codec were obtained by varying the quality parameter from 10 (associated with lower quality) up to 90 (associated with higher quality) with a quality step of 10.

4.2.1.2. H.264/AVC Intra Standard

Besides the inherent changes due to the test conditions previously described, no more changes were made in its the default configuration parameters, except for the quantization parameter; since the RD points for this codec are obtained by varying the QP (quantization parameter) from 10 up to 40 with a quantization step of 5.

4.2.1.2. H.264/AVC Inter Standard

Besides the inherent changes due to the test conditions, the Intra Period was set to 8 (for “Breakdancers”) and 15 (for “Newspaper”) in order to follow the ISO/IEC condition in [11], which states that “The GOP (Group of Pictures) size for each sequence shall not be less than 0.5 seconds”.

By varying the QP (quantization parameter) from 10 up to 40 with a quantization step of 5, the RD points for this codec were obtained.

4.2.1.3. MVC Standard

Besides the inherent changes due to the test conditions, the Intra Period and Anchor Period were both set to 8 (for “Breakdancers”) and 15 (for “Newspaper”) in order to follow the ISO/IEC condition in [11].

By varying the QP (quantization parameter) from 10 up to 40 with a quantization step of 5, the RD points for this codec were obtained.

4.2.2. Proposed Depth Codecs

Regarding complexity parameter, the TriSize\textsubscript{Threshold} was set to 3 (corresponding to its minimum possible value). Note that each threshold targets a triangle and not the full depth map.

4.2.2.1. IST-Depth/PERR Codec

For this codec, the RD points to be evaluated are defined by varying the PERR threshold exponentially as follows:

\[ \text{TriPERR}_{\text{threshold}} = 10^{-x}, x = \{0, 1, 2, 3, 4\} \]

(15)

To minimize the MSE in the decompositions, \( \text{TriMSE}_{\text{threshold}} \) was set to 100.

4.2.2.2. IST-Depth/MSE Codec

For this codec, the RD points to be evaluated are defined by varying the MSE threshold exponentially as follows:

\[ \text{TriMSE}_{\text{threshold}} = 10^{-x}, x = \{0, 1, 2, 3, 4, 5\} \]

(16)

To minimize the PERR threshold effect, in the decompositions, \( \text{TriPERR}_{\text{threshold}} \) was set to 100.

4.3. PERR Performance Analysis

The PERR results obtained for the two multiview test sequences for all the codecs under study are shown in Figure 11 and Figure 12.

![Figure 11 - PERR results for the “Breakdancers” multiview sequence.](image1)

The analysis of the PERR chart in Figure 11 for the “Breakdancers” sequence motivates the following remarks (keep in mind that for the PERR metric, the lower the value, the better the performance):

- IST-Depth/PERR outperforms all other codecs without exception, which is an important fact since this codec just have intra mode and is outperforming codecs with inter mode and interview prediction, i.e., H.264/AVC Inter and MVC respectively;
- The IST-Depth/MSE has a PERR performance much better than the JPEG codec, which is an important conclusion as the very low complexity JPEG codec (also not exploiting the temporal redundancy as the two proposed depth codecs) is still often used for depth maps coding.

![Figure 12 - PERR results for the “Newspaper” multiview sequence.](image2)

The analysis of the PERR chart for the “Newspaper” sequence, in Figure 12, motivates the following remarks (keep in mind that for the PERR metric, the lower the value, the better the performance):

- Both the IST-Depth/PERR and the IST-Depth/MSE codecs outperform the JPEG codec which is an important conclusion since the very low complexity JPEG codec (also not exploiting the temporal redundancy as the two
proposed depth codecs) is still often used for depth maps coding;

- IST-Depth/PERR can compete with the H.264/AVC Intra codec and - for lower and higher bitrates - can even outperform the H.264/AVC Intra codec, which is an important fact since both codecs have only the intra mode.

4.4. PSNR Performance Analysis

The PSNR results obtained for the two multiview sequences for all the codecs under study, according to the previously described test conditions, are shown in, Figure 13 and Figure 14.

![Figure 13 - PSNR results for the “Breakdancers” multiview sequence.](image1)

The analysis of the PSNR chart in Figure 13 for the “Breakdancers” sequence motivates the following remarks:

- IST-Depth/MSE outperforms JPEG codec, which is an important conclusion since the very low complexity JPEG codec (also not exploiting the temporal redundancy as the two proposed depth codecs) is still often used for depth maps coding;
- In terms of PSNR performance, IST-Depth/MSE is similar to H.264/AVC Intra, H.264/AVC Inter and MVC for intermediate bitrates, which is an important conclusion since this codec just have intra mode and is competing codecs with inter mode and interview prediction, i.e., H.264/AVC Inter and MVC respectively.
- IST-Depth/MSE outperforms JPEG codec, which is an important conclusion since the very low complexity JPEG codec (also not exploiting the temporal redundancy as the two proposed depth codecs) is still often used for depth maps coding;
- In terms of PSNR performance, IST-Depth/MSE is similar to H.264/AVC Intra for higher bitrates, which is an important conclusion since both codecs just have intra mode.

4.5. Results

Taking into account all the results, comments and conclusions previously presented, i.e. from a depth coding perspective, and regarding the proposed codecs it can be concluded the following:

- The IST-Depth/MSE solution is a weak solution since it is not the leader for any of the performed tests and it is always rather behind the other codecs;
- The IST-Depth/PERR codec leads some of the performed tests, especially in terms of PERR performance and, thus, it is an interesting solution for the cases where the PERR is considered to be a good metric for depth coding assessment.

Regarding the IST-Depth/PERR codec, it provides a good depth coding solution in some conditions because:

- From a PERR perspective, it always outperforms its direct competitors, the H.264/AVC Intra and the JPEG codecs, since codecs without temporal prediction typically has a lesser associated complexity;
- Also from a PERR perspective, the IST-Depth/PERR codec outperforms some of the its lesser direct competitors - the H.264/AVC Inter and MVC codecs – for the “Breakdancers” sequence which is a very interesting conclusion;
- Although it has an inferior performance than most of the other codecs from a PSNR perspective, the IST-Depth (both PERR and MSE based codecs) provides the location of the lossless coded samples and therefore providing vital information regarding reliable depth data in a depth map at the decoder’s side. For view synthesis purposes, this is a critical issue in order to be able to perform a better reconstruction of the view [12].

Regarding binary tree coding, the compression factor of the 7-zip encoding process varied from 20 up to 70, for higher and lower bitrates respectively. Regardless the success f this encoding process, it has a weak impact (approximately 7% for the best case) regarding the IST-Depth performance; meaning that the encoding process of depth samples has a stronger impact for the IST-Depth performance than the binary tree encoding process.

5. FINAL REMARKS

IST-Depth codec is a depth coding solution based on content adaptive coding, which uses a binary tree to represent the decompositions performed in the depth map. IST-Depth
takes into account depth specific properties, competes with already proposed depth coding solutions and introduces depth coding novel tools that may lead to a better support of 3D features.

By studying depth properties, it was concluded that the depth should be assessed with a PERR based criterion instead of metric which are more focused in the visual effect of the depth map; since depth maps are not seen directly by the human eye.

A comparative study between IST-Depth codec and well know standards, such as JPEG, H.264/AVC an MVC, was made in order to evaluate the performance of the IST-Depth codec.

6. REFERENCES


