# IMPROVED PATCH PACKING SOLUTIONS FOR THE MPEG V-PCC STANDARD

*Afonso Costa*

Instituto Superior Técnico, Universidade de Lisboa - Instituto de Telecomunicações, Lisboa, Portugal
afonsocosta0404@gmail.com

## ABSTRACT

Point cloud is a key representation technology in the 3D experiences paradigm. With the introduction of this type of representation, new applications have emerged in the fields of entertainment, medicine, architecture, robotics, etc. This new type of visual representation urgently needs efficient coding techniques in order the enormous amount of information necessary to store and transmit a point cloud can be reduced. With that in mind, several coding solutions have been recently proposed in the literature, notably the MPEG V-PCC standard. This paper proposes to increase the efficiency of the packing module that is part of the V-PCC standard, with the aim of increasing the overall RD performance. In this context, several patch packing solutions are proposed, including new algorithms and associated metrics. The main novelty regarding the current V-PCC patch packing solution is the adoption of relative packing. In the proposed relative packing solutions, the algorithms test several available positions and decide which one is the best according to some proposed metric(s). These metrics aim at increasing the packing map compactness and color map smoothness. The performance results show BD-Rate improvements of around 0.8% for the best packing solution regarding the V-PCC benchmark. Moreover, the packing map size reductions can go up to 12%, on average.

***Index Terms***— Point Cloud, Patch, Patch Packing, Sorting, Positioning, Video based Point Cloud Coding.

## 1. INTRODUCTION

Multimedia is certainly one of the most impactful technological areas in present days and it is also a field that Humans strongly relate to. The evolution of multimedia technologies has led to an explosion in terms of the number and quality of communication, entertainment, gaming, educational, etc. systems with high impacts also in other economic sectors. In fact, multimedia technology tries to replicate our world in the most suitable way for the human sensory systems. This paper is focused on the representation of visual information, particularly 3D visual representation as it offers more realistic and immersive experiences than 2D visual representation. For this reason, this is one of the hottest multimedia research topics nowadays. Several 3D visual representation models have been studied and one of the most promising is point cloud (PC) representation (see Figure 1).

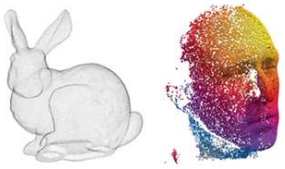

**Figure 1 – PC examples: (left) Rabbit uncolored; (right) Face colored.**

Since 3D visual representations are much richer than 2D visual representations, the amount of associated information and data is also highly superior to the nowadays commonly used 2D visual representations. Consequently, new efficient coding techniques are needed so that it is technically and financially viable to use the new 3D representation technologies, notably for transmission and storage of point clouds. This challenge, this means the efficient coding of point clouds, is precisely the focus of this paper. In this context, the coding advances are thus highly determinant on the viability of point cloud based systems. With that purpose in mind, several point cloud coding solutions have been proposed in the last few years.

Although the state-of-the-art on point cloud coding is still rather immature, since these technologies are recent, there are already some standard coding solutions under development. As it happened for video coding in the past, the MPEG standardization group has decided to design efficient point cloud coding solutions to address the emerging needs from the industry. Within MPEG, two point cloud coding standards are currently under development: a standard addressing the efficient coding of static point clouds, named as Geometry based-Point Cloud Coding (G-PCC), and a standard addressing the efficient coding of dynamic (time varying) point clouds, named as Video based-Point Cloud Compression (V-PCC). The V-PCC standard is an important point cloud codec since it allows to efficiently represent point cloud video sequences, which are extremely heavy in terms of storage and transmission. In this context, this paper targets improving the V-PCC compression efficiency. As will be motivated later, a specific V-PCC module was selected for improvement, the patch packing module.

In this context, the objectives of this paper are to design novel coding tools targeting improving the compression efficiency of state-of-the-art point cloud coding technologies, notably the MPEG V-PCC coding standard. The desired improvement is done by proposing novel patch packing solutions, assess its efficiency under meaningful test conditions and compare them with appropriate benchmarks.

To achieve its objectives, this paper will be organized as follows after this first section: Section II reviews the main point cloud coding solutions in the literature, with more emphasis on the MPEG V-PCC standard. Section III motivates and describes the proposed patch packing solutions. Section IV presents the packing and RD performance results and, finally, Section V closes this paper with relevant conclusions.

## 2. BACKGROUND WORK

Several solutions have already been proposed in the literature for efficient point cloud coding. The main solutions follow octree-based, graph-based and projection-based approaches. Octree-based methods represent the 3D positions by exploiting a tree structure with 8 children for each node, where the points are assigned to leaf nodes. In graph-based methods, the relationship between

neighboring points is exploited by connecting them with a graph. The projection-based solutions are the focus of this work and these coding solutions make use of 3D to 2D transformations or projections to use already available 2D video coding solutions. The MPEG V-PCC standard is one of these solutions.

In [2], Golla and Klein propose a point cloud coding solution involving the definition of 2D patches corresponding to specific parts of the entire point cloud. The coding process consists in dividing the 3D space into voxels and, for each voxel, creating patches by using projections. Each patch contains information related to color, height and occupancy and is a 2D representation (image) of a specific part of the point cloud. All the 2D visual data types are then compressed using state-of-the-art 2D image or video coding solutions. In [3], Lanyi He propose a point cloud coding solution to compress both the color and geometry data based on the creation and coding of 2D maps, which are obtained by the projection of the 3D points onto a set of planes defined by the faces of a cubic bounding box. In [4], Zhang *et al.* propose a static point cloud coding solution to code both geometry and color data where the points are projected onto planes without the need to perform any reprojection at the decoder. To code the point cloud, the proposed solution takes the acquired point cloud data and groups the points into small clusters. After the point cloud is clustered, a plane is fitted to the points in each cluster and these points are projected onto the corresponding plane.

The V-PCC solution is the most relevant patch based solution in the literature and, for this reason, will be presented here with more detail. Initially, the patches are extracted from the point cloud by exploiting surfaces normal vectors. The patch generation module firstly groups the points into clusters according to each point normal, estimated using a normal extraction algorithm [1]. 3D patches are then extracted from each cluster using a connected component analysis. All the patches are then projected into a single 2D grid and images are generated by projecting the color and geometry. It is this packing process that it is the focus of this paper. The packing process adopted by the MPEG V-PCC standard is a rather simple algorithm, so-called *Top-Left packing*. The Top-Left packing algorithm starts by defining the order by which the patches should be packed in the image, in this case the patch height; this is the sorting component of the packing algorithm. After, to restrict the size of the area where the patches are to be packed, the vertical and horizontal resolutions of the packing area are set with user-defined parameters. The packing of the patches can then take place, following an iterative process that searches for an available position for the next patch to be packed, depending on the already packed patches and following the sorting criterion; this is the positioning component of the packing algorithm. The positioning method involves a raster-scan search for empty positions in the projection map. As every patch is projected to the same image/map, this image will naturally contain many empty spaces. The empty spaces are filled using a padding process, which takes into consideration the non-empty space around it. The images/maps can then be encoded using standard video coding solutions, e.g. HEVC. Occupancy maps and patch characteristics are also coded and multiplexed with the coded video stream.

## 3. PROPOSING NOVEL PATCH PACKING METRICS AND ALGORITHMS

The MPEG V-PCC PC coding standard represents the state-of-the-art on dynamic PC coding and, for that reason, it is the most interesting and also challenging solution if an improvement performance perspective is taken. The V-PCC packing algorithm does not effectively take into consideration the need to minimize the overall packing space, nor the color resemblances between neighboring patches, what could also make sense considering that they will be later HEVC coded (after some padding). Because keeping the compatibility/interoperability with V-PCC is important, this paper will only propose non-normative changes to V-PCC, meaning that the proposed improvements on the patch packing module will not alter the bitstream syntax and semantics as well as the decoder behavior. The packing solutions presented in this paper are for static PCs and try to improve the V-PCC static packing solution.

To propose new static packing techniques, a literature research has been made on patch packing techniques. The solution in [5] uses the patch color characteristics to pack the patches; this type of solutions can be seen as puzzle solving techniques. This paper exploits this novel patch packing approach by proposing several ways to consider the patches color and geometry characteristics, individually or together. To achieve this target, the puzzle solving state-of-the-art packing algorithm was taken as starting point [5].

A packing solution, notably a static packing solution, may be decomposed in two main parts: patch sorting and patch positioning. Sorting regards the ordering of the patches to be packed according to some criterion/metric. Positioning regards the placement of the patches in the packed maps according to some criterion/metric.

### 3.1. Sorting Criteria and Metrics

Sorting criteria correspond to patch characteristics or patch metrics used to order the patches in a list of patches before processing their positioning; simple examples are height, width, size, or the color dissimilarity between two patches. Patch sorting criteria can be sub-divided into two classes: absolute sorting and relative sorting. Absolute sorting uses criteria/metrics computed independently for each patch without considering any information regarding the current arrangement of the already packed patches. On the opposite, relative sorting considers the way the already packed patches are organized, thus extracting information from the already packed maps, notably in comparison to each patch to be sorted, and using the measured dissimilarity to define the patch processing order; in this way, both the characteristics of the already packed and to be packed patches are considered.

#### *3.1.1. Absolute Sorting Metrics*

For absolute sorting, only geometry metrics are proposed since the individual patch colors are not very relevant for sorting since no specific colors should get any privilege. The proposed absolute sorting metrics are: height; patch bounding box area (*BBAREA*); patch effective area (*EA*) (number of occupied pixels in the bounding box); and patch wasted space (area) (WS) (number of unoccupied pixels in the bounding box).

#### *3.1.2. Relative Sorting Metrics*

For the relative sorting of patches, it is necessary to take into consideration the packing situation at a certain stage and the already packed patches characteristics as well as the characteristics of the patches yet to be sorted and packed. As it is necessary to extract features from the packing arrangement at a certain stage, relative sorting may only be performed as the positioning proceeds since the sorting is made by comparing the not yet packed patches with the already packed set of patches. The three relative sorting metrics

proposed are: horizontal common boundary length (*HCBL*); extension size (*ES*) and color only based dissimilarity error (*CLDE*). The *HCBL* corresponds to the number of pairs of patch horizontal boundary pixels in two patches under processing that are neighbors along a certain horizontal direction, e.g. left/right, see Figure 2. It is attributed lower dissimilarity to a longer *HCBL*. The *ES* corresponds to the number of blocks associated to the extension of the packing map when two patches are put together. For sorting, lower dissimilarity is attributed for larger *EL,* and the opposite for positioning. The so-called *Color and Length based Dissimilarity Error* (*CLDE*) metric measures the color dissimilarity between two patches [5] (only considering the horizontal common boundary pairs) and is computed as follows, see also Figure 2:

$$CLDE(p_i, p_j) = \frac{1}{(HCBL)^2} \sum_{m=1}^{HCBL} \frac{1}{3} \sum_{c=1}^{3} \left( I_c(x_{p_i}^m, y^m) - I_c\left(x_{p_j}^m, y^m\right) \right)^2 \quad (1)$$

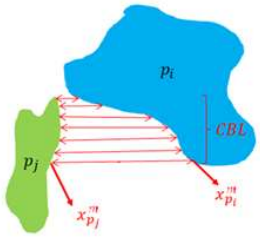


**Figure 2 – Common boundaries example.**

## 3.2. Positioning Criteria and Metrics

After the patch sorting is made, the patches have to be positioned in the packing map using a packing algorithm that may use or not a positioning metric. If a positioning metric is used to position one more patch, the metric is relative since the new patch has to be somehow compared with the already packed patches. Thus, for relative positioning, the selected patch position results from the consideration of both the characteristics of the patch under packing and the already packed map. With this purpose in mind, the relative sorting metrics proposed above for patch sorting may also be used for patch positioning. For patch positioning, the selected dissimilarity metric is assessed over multiple relevant positions within defined boundaries and the position offering the lowest dissimilarity is selected for packing.

## 3.3. Patch Packing Algorithms

This section presents the two novel, proposed patch packing algorithms, which involve some of the previously proposed metrics for sorting and positioning.

### *3.1.1. Bounding Box Close Space Positioning*

The Bounding Box Close Space (BBCSP) packing algorithm follows a similar procedure to the V-PCC packing algorithm (Top-Left). The major difference to the Top-Left algorithm is that the positioning search is performed in the space closer to the patches already packed instead of searching in all the packing map. The Top-Left positioning algorithm applies a raster-scan inside all the map while the BBCSP algorithm applies a raster scan inside the bounding box of the currently packed patches and only after searches in the closest columns/lines around it.

### *3.1.1. Super-Patch based Packing*

The Super-Patch based Packing (SPBP) algorithm is the main packing algorithm proposed in this work and employs a completely different procedure than Top-Left. This packing algorithm is based on the puzzle solving algorithm proposed in [5]. The SPBP packing algorithm starts by choosing a starting patch for the successive and iterative positioning of the remaining patches. At each iteration, a new elementary patch is positioned, thus generating a new super-patch with increasing size and complexity. In this iterative process, the first step is to choose the next best patch to be packed into the current super-patch. The chosen patch is included in the current super-patch and the remaining elementary patches are again processed to identify their best position around the resulting super-patch. When all the elementary patches are packed, a map is defined to hold the final super-patch and proceed to coding. The proposed SPBP algorithm may be organized in six main modules and its architecture is presented in Figure 3.

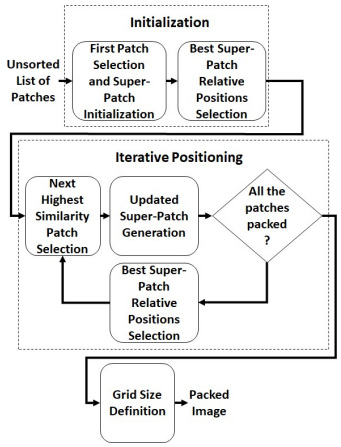


**Figure 3 - Super-patch based packing algorithm architecture.**

The **first patch selection and super-patch initialization** module starts by choosing the first patch according to a chosen absolute sorting metric. After the initial patch is selected, it becomes the initial super-patch to which all other patches will be successively joined. In general, a super-patch is defined by three main components: list of included elementary patches; occupancy map; and texture map. At the initialization phase, the various fields of the super-patch are initialized with only the first chosen patch belonging to the list and the occupancy map and texture map are exact copies of the first patch occupancy and texture map, respectively.

The **best super-patch relative patch positions search** module applies an exhaustive search to assess every positioning of the non-packed patches around the super-patch and identify the best one according to a certain positioning metric. To find the best position for a certain patch around the super-patch, a search must take place at every suitable position, taking only into consideration the closest position to the super-patch in every direction. For each position, it is tested if the position is suitable (if it fits) and a positioning metric/dissimilarity error is computed. After the initialization procedures, an iterative cycle takes place to position the remaining N-1 patches around the initially selected patch and successively added ones.

The **next highest similarity patch selection** module is responsible to choose, from the list of remaining candidate patches, the patch that should be positioned in the next iteration. It is here proposed that the patch to add to the current super-patch at each iteration is

the one with the lowest dissimilarity error regarding the current super-patch.

In the **updated super-patch generation** module an updated super-patch is generated, notably including the recently positioned patch. As input, this module receives the additional patch and its position relatively to the previous super-patch. It is then necessary to update all the fields of the updated super-patch. The list of contained patches, the occupancy map and the texture map should include the new patch. If necessary, the occupancy and texture map may be resized.

After packing all patches, the **map size definition** module fits a rectangular map with an appropriate size to contain all the patches in the selected relative positions. The texture/depth/occupancy maps should be aligned with the final super-patch bounding box orientations and be as small as possible to reduce the coding rate. This is achieved by creating a bounding box around the final super-patch (with all the packed patches) by taking the top most, bottom most, left most and right most blocks on the packing map of the final super-patch.

## 4. PERFORMANCE ASSESSMENT

Since the proposed packing solutions and metrics target improving the MPEG V-PCC packing solution, this later algorithm will take the role of benchmarking, thus meaning that the performance assessment will be always done in comparison to V-PCC.

### 4.1. Test Material and Conditions

The V-PCC reference software version used for benchmarking andalso to integrate the proposed packing algorithms is version v0 [1]. The HEVC and HDRTools software versions used are also those suggested in [1].

The V-PCC dataset defined by MPEG includes four dynamic PCs, notably *Loot, Red and Black, Soldier* and *Longdress*, all representing people. The PCs in Figure 1 correspond to the specific PC frames used in the MPEG static dataset. Since this work focus on static packing, notably for static PCs, it makes sense to also include in the adopted test set some PCs from the MPEG static dataset. The PCs selected from the MPEG static PCs were *Mask* and *Klimt Statue*.

The adopted values for the coding parameters are those defined by MPEG in the so-called *Common Test Conditions* document [6]. To have a fair comparison, the PCs in the dataset are always coded with the same coding parameters for all packing solutions, with the exception of the quantization parameters (QPs). Five RD points are selected for assessment. The precise quantization parameter values used for performance assessment are those recommended by MPEG in the Common Test Conditions, notably [2]: R1: Geometry QP - 32; Texture QP - 42; R2: Geometry QP - 28; Texture QP – 37; R3: Geometry QP - 24; Texture QP - 32; R4: Geometry QP - 20; Texture QP - 27; and R5: Geometry QP - 16; Texture QP - 22;

Naturally, for the performance assessment to be meaningful, appropriate evaluation metrics must be defined, at both packing algorithm level and overall PC coding algorithm level, since at the end the packing should serve to obtain a final better rate-distortion (RD) performance, both for geometry and texture. For the packing performance assessment, it is essential to define metrics providing information on how well the final packing performs in terms of geometry and color. The defined metrics are: map size (MS) and average color difference over the padded discontinuities (ACDPD). The MS is the number of T×T blocks in the occupancy map. The ACDPD is computed over all pairs of pixels in the padded discontinuities and averaged over the total number of padded discontinuities in the final texture map. These padded discontinuities are associated to the pairs of pixels at the end of each line that have been padded in the texture map.

For the RD performance assessment, the adopted objective quality metrics are those used by MPEG [6], notably D1 and D2 for the geometry quality and PSNR over the Y, U and V color components. To better assess the RD performance, it is common to compute the so-called *Bjontegaard Delta-Rate* (BD-Rate) and *Bjontegaard Delta-PSNR* (BD-PSNR) metrics [7], which allow comparing two coding solutions while considering several RD points.

The performance evaluation is done for several packing solutions by combining the proposed metrics and algorithms. More precisely, the results are structured according to the metrics in three phases: absolute sorting and no positioning metric; absolute sorting and relative positioning; and relative sorting and relative positioning. All the solutions have been tested with the proposed absolute sorting metrics: *Height*; *BBAREA*; *EA*; and *WS*. In the tables presented in the next sections, the results are averaged over the 6 selected test PCs.

### 4.2. Absolute Sorting – No Positioning Metric Performance Assessment

This section reports and analyses the results when using several absolute sorting metrics for the Top-Left and BBCSP packing algorithms, which do not use any explicit metric for positioning. The results regarding the Top-Left positioning algorithm are presented in Table 1. With the Top-Left algorithm, none of the absolute sorting metrics alternatives (V-PCC uses the height) allows improving the packing performance; in practice, only *WS* showed a color error reduction but at the cost of a map size increase, what is not desirable. For the most relevant RD performance scores, notably for D1 and PSNR Y, the average variation is towards a poorer RD performance. This was expected since the packing performance is also visually poorer, naturally always in comparison with the benchmark V-PCC.

**Table 1.** Average packing and BD-Rate performance for the Top-Left packing algorithm with different absolute sorting metrics.

| | BBAREA | EA | WS |
|---|---|---|---|
| MS | 4% | 3% | 4% |
| ACDPD | 19% | 1% | -1% |
| D1 | 0,35% | 0,06% | 1,17% |
| D2 | -0,09% | -0,37% | 0,56% |
| PSNR Y | 0,44% | 0,11% | 0,43% |
| PSNR U | -0,23% | -1,98% | 0,26% |
| PSNR V | 0,69% | -0,80% | 0,34% |

The results for the BBCSP algorithm are presented in Table 2. The average packing performance are very promising since the packing map size can be reduced except for the WS criterion. Moreover, the BBCSP packing algorithm with the *BBAREA* sorting metric brings RD performance gains for both geometry and color. It is also relevant to notice that the *BBAREA* sorting metric improvements only occur for 3 out of the 6 test PCs. This is a first clear improvement over the V-PCC packing algorithm benchmark.

**Table 2.** Average packing and BD-Rate performance for the BBCSP packing algorithm with different absolute sorting metrics.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -2% | -4% | -5% | 0% |
| ACDPD | 5% | 13% | 9% | 12% |
| D1 | 0,26% | -0,62% | 0,43% | 0,91% |
| D2 | -0,08% | -0,45% | 0,23% | 0,76% |
| PSNR Y | 0,02% | -0,27% | 0,55% | 0,42% |
| PSNR U | 0,09% | -0,92% | -0,33% | 0,61% |
| PSNR V | 0,86% | -0,41% | -0,02% | -0,07% |

### 4.3. Absolute Sorting – Relative Positioning Performance Assessment

This section presents the performance results when using the SPBP algorithm with several sorting and positioning metrics. In terms of positioning, several metric combinations were tested, notably: *ES*; *ES* with *HCBL* for tie-breaking; and ES with *HCBL* and *CLDE* for tie-breaking in this order.

Table 3 shows the performance results when using the SPBP algorithm with the *ES* positioning metric. The sorting metric offering the best packing performance is the *EA* sorting metric, if the map size is taken as the most relevant criterion. The *ES+HCBL* positioning with *BBAREA* and *EA* sorting are the two best packing combinations so far. The best solution seems to be the one using *BBAREA* since the average RD performance results are more balanced (D1 and PSNR Y have similar performance gains) and significantly better than for the previously presented solutions. With this packing solution, and using *BBAREA* and *EA* as the sorting metrics, the RD performance improves for 4 out of 6 PCs for D1 and PSNR Y. These are the best results so far.

**Table 3.** Average packing and BD-Rate performance for the SPBP packing algorithm when using different absolute sorting metrics for the *ES* positioning metric.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -7% | -10% | -12% | -9% |
| ACDPD | 4% | 9% | 7% | 2% |
| D1 | -0.26% | -0.79% | -0.24% | 0.10% |
| D2 | -0.42% | -1.14% | -0.99% | -0.43% |
| PSNR Y | -0.17% | -0.62% | -1.09% | -0.23% |
| PSNR U | 0.10% | -0.76% | -0.47% | -0.71% |
| PSNR V | -0.14% | 0.38% | -1.00% | -0.07% |

Table 4 describes the performance results when using the SPBP algorithm with the *ES* positioning metric with *HCBL* for tie-breaking. The sorting metric offering the best packing performance is the EA sorting metric. For this *ES* positioning metric, it was expected that the EA sorting metric would produce the best RD performance results due to the previous good packing performance results. However, the best results, on average, happen for the *BBAREA* absolute sorting metric. This *ES+HCBL* positioning with *BBAREA* sorting combination leads to the best RD performance so far since the average RD performance gains are slightly better than the ones obtained for the *ES* positioning solution with BBAREA/EA absolute sorting. With this packing solution, and using *BBAREA* as the sorting metric, the RD performance improves for 5 out of 6 test PCs, both for D1 and PSNR Y. This is better than for the *ES* positioning metric with *BBAREA* sorting and, thus, it is the best so far in percentage of wining PCs. Figure 4 shows the high level of compactness achieved for *Red and Black* with the *ES+HCBL* positioning combination and *EA* sorting, notably a 13% map size reduction regarding V-PCC.

**Table 4.** Average packing and BD-Rate performance for the SPBP packing algorithm when using different absolute sorting metrics for the *ES+HCBL* positioning metric.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -8% | -9% | -11% | -10% |
| ACDPD | 3% | 19% | 34% | -1% |
| D1 | 0.32% | -0.84% | 0.32% | 0.30% |
| D2 | 0.12% | -1.37% | -0.57% | 0.53% |
| PSNR Y | -0.64% | -0.72% | -0.48% | -0.32% |
| PSNR U | 0.12% | -0.16% | -0.43% | -0.91% |
| PSNR V | -0.06% | 0.05% | 0.13% | -0.76% |

**Figure 4 - *Red and Black* PC packing: (left) V-PCC; (right) SPBP with EA sorting metric and *ES+HCBL* positioning metric.**

Table 5 shows the performance results when using the SPBP algorithm with the *ES* positioning metric with *HCBL* and *CLDE* for tie-breaking, in this order. The sorting metric offering the best packing solution for *ES+HCBL+CLDE* positioning is the EA sorting metric, if the map size reduction is taken as the most relevant performance criterion. However, the color error is too high when compared with the other sorting metrics. For this *ES+HCBL+CLDE* positioning solution, the average RD performance results also show positive results, namely for *BBAREA*. However, this packing solution does not offer as good results as the previous positioning metrics combinations. With this solution, and using *WS* as sorting metric, the RD performance improves for 3 out of 6 test PCs for D1 and PSNR Y. This is an obvious worse performance than for the two previous packing solutions.

**Table 5.** Average packing and BD-Rate performance for the SPBP packing algorithm when using different absolute sorting metrics for the *ES+HCBL+CLDE* positioning metric.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -7% | -11% | -11% | -11% |
| ACDPD | 2% | 11% | 42% | 29% |
| D1 | 0.19% | -0.20% | 0.41% | 0.29% |
| D2 | 0.17% | -0.28% | 0.00% | 0.14% |
| PSNR Y | -0.53% | -0.32% | -0.22% | -0.43% |
| PSNR U | -0.59% | -0.43% | -1.53% | -0.74% |
| PSNR V | -0.28% | 0.40% | -0.72% | -0.74% |

### 4.4. Relative Sorting – Relative Positioning Metric Performance Assessment

This section presents the performance results for the SPBP algorithm with some relative sorting metrics and the best positioning metrics combination from the previous section, notably the *ES+HCBL* combination. Regarding the relative sorting metrics, *HCBL+ES* and *ES+HCBL* will be tested where '+' represents a tie-breaking decision. Several absolute sorting metrics are used to choose the first patch to be packed.

Table 6 describes the performance results when using the SPBP algorithm with *HCBL* as the main sorting metric and *ES* for tie-breaking. The absolute sorting metrics offering the best performance are the *BBAREA* and *EA* metrics, with *EA* in advantage since the

color error performance is better. The overall set of results seems to indicate that the packing algorithm is playing a even major role here, as expected, since the absolute sorting metrics only choose the first patch to be packed. From the average performance, it is highly difficult to select one specific absolute sorting metrics as the best in terms of RD performance since the results are very similar. This similarity may be explained by the fact that only the choice of the first patch changes between the various absolute sorting metrics. In fact, all the absolute sorting metrics provide positive RD performance gains except *BBAREA* which shows a minor loss for geometry (0.03%). The sorting metric offering better RD performance results is *WS*. With this packing solution, and using *EA* and *WS* as sorting metrics, there are RD performance improvements for 3 out of 6 test PCs for D1 and PSNR Y. This is worse than for the previous type of packing solutions which obtained gains for 5 out of 6. On average, this packing solution provides the best results so far, with *WS*, but it loses in terms of percentage of gaining PCs. Figure 5 shows the high compactness achieved for *Longdress* with *HCBL+ES* relative sorting combination and *EA* absolute sorting, notably a 25% map size reduction regarding V-PCC.

**Table 6.** Average packing and BD-Rate performance for the SPBP packing algorithm when using *HCBL+ES* relative sorting metrics combination and various absolute sorting metrics for the first patch.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -7% | -8% | -8% | -7% |
| ACDPD | 2% | 3% | -2% | 1% |
| D1 | -0,17% | 0.03% | -0,26% | -0.82% |
| D2 | -0,27% | -0.25% | -0,42% | -0.88% |
| PSNR Y | -0,58% | -0.71% | -0,51% | -0.79% |
| PSNR U | -0,18% | -0.78% | -0,07% | -0.60% |
| PSNR V | -0,31% | -0.27% | 0,20% | -0.88% |

**Figure 5 - *Longdress* PC packing: (left) V-PCC; (right) SPBP packing with *HCBL+ES* relative sorting metrics combination and Height absolute sorting metric for the first patch.**

Table 7 describes the performance results when using the SPBP algorithm with *ES* as the main sorting metric and *HCBL* for tie-breaking. The metric offering the best performance is the *EA* absolute sorting metric, if the map size is taken as the most relevant criterion. In fact, this is the best solution in terms of packing performance since all the metrics have map size reductions higher than 8% and the color error is reduced for all the metrics except *EA* (only loosing 1%). However, the RD performance results are far from being the best. Although it is true that all the absolute sorting metrics provide RD performance gains, the improvements are not as high as expected. With this packing solution, and using *EA* as sorting metric, RD performance improvements occur for 3 out of 6 test PCs, for D1 and PSNR Y.

**Table 7.** Average packing and BD-Rate performance for the SPBP packing algorithm when using *ES+HCBL* relative sorting metrics combination and various absolute sorting metrics for the first patch.

| | Height | BBAREA | EA | WS |
|---|---|---|---|---|
| MS | -9% | -9% | -11% | -9% |
| ACDPD | -3% | -2% | 1% | -1% |
| D1 | -0,31% | 0.08% | -0,08% | -0.23% |
| D2 | -0,74% | -0.07% | -0,57% | -0.25% |
| PSNR Y | -0,53% | -0.31% | -0,48% | -0.10% |
| PSNR U | -0,27% | -1.50% | -0,86% | -0.19% |
| PSNR V | 0,35% | -0.33% | -0,43% | 0.10% |

## 5. CONCLUSIONS

The results obtained show clear RD performance gains against the V-PCC standard benchmark by using the proposed packing metrics and algorithms. In fact, performance improvements are obtained with the three types of proposed packing solutions. With the absolute sorting and no positioning metric solutions, the RD performance gains over the V-PCC benchmark are positive but still do not showing large average improvements, notably with gains around 0.4% for the BBCSP algorithm with the *BBAREA* sorting metric. For the absolute sorting and relative positioning solutions, the RD performance gains are more significant, notably around 0.7%, on average, for both color and geometry. The relative sorting and relative positioning solutions show a residual increase of the RD performance, notably reaching gains around 0.8%. This indicates that the adoption of a relative approach is important, although more critical for the positioning than for the sorting as the gains are residual for the second case. Since the best performance results occur for the packing solutions with a relative approach, it is fair to say that the SPBP packing algorithm is a crucial component for these improvements.

In summary, the absolute sorting and relative positioning solutions offer the best RD performance since they offer performance gains over V-PCC for 5 out of the 6 test PCs. Finally, the solutions proposed in this paper may also bring important contributions to the broad field of patch packing, since the obtained map size reductions seem to be interesting for this research field.

## REFERENCES


[1] MPEG 3DG Subgroup, "PCC Test Model Category 2 v0," ISO/IEC JTC1/SC29/WG11 N17248, Macau, China, Oct. 2017.

[2] T. Golla and R. Klein, "Real-Time Point Cloud Compression," in IEEE/RSJ IROS, Hamburg, Germany, Oct. 2015.

[3] L. He, W. Zhu and Y. Xu, "Best-Effort Projection Based Attribute Compression for 3D Point Cloud," 23rd Asia-Pacific Conference on Communications (APCC), Perth, WA, Australia, Dec. 2017.

[4] X. Zhang, W. Wan and X. An, "Clustering and DCT Based Color Point Cloud Compression," Journal of Signal Processing Systems, vol. 86, no. 1, pp. 41-49, Jan. 2017.

[5] S. T. Cho, S. Avidan and W. T. Freeman, "A Probabilistic Image Jigsaw Puzzle Solver," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, Jun. 2010.

[6] MPEG 3DG Subgroup, "Common Test Conditions for Point Cloud Compression," ISO/IEC JTC1/SC29/WG11 N17345, Gwangju, Korea, Jan. 2018.

[7] Gisle Bjontegaard, "Calculation of average PSNR differences between RD-curves," Video Coding Experts Group, Austin, TX, USA, Mar. 2000