

POINT CLOUD ATTRIBUTE COMPRESSION

[0001] This application claims the benefit of priority of Singapore Patent Application No. 10202008512Q, filed on 2 September 2020, the content of which being hereby incorporated by
5 reference in its entirety for all purposes.

TECHNICAL FIELD

[0002] The present invention generally relates to a method and a system for point cloud attribute compression, and more particularly, for image-based three-dimensional (3D) point
10 cloud attribute compression.

BACKGROUND

[0003] Due to the advancement of light detection and ranging (LiDAR) and photogrammetry technologies, as well as the pervasiveness of more affordable 3D acquisition and digitisation devices, point clouds have been increasingly gaining popularity in a variety of
15 emerging fields, such as but not limited to, localization and pose estimation in an area, virtual and augmented reality, tele-immersive communication, cultural heritage documentation, autonomous driving and so on. For example, to sufficiently represent the shape and appearance of real-world objects or scenes, point clouds can be made up of millions or even billions of
20 points, each of which is associated with a set of numerical coordinates (e.g., 3D coordinates) and possible attribute information (e.g., luminance, color, surface normal, reflectance, and so on). Such a digital representation form could inevitably generate an enormous volume of data. Considering the limitation of network bandwidth and storage capacity, point cloud compression has therefore become a necessity for many 3D-related applications.

[0004] To reduce the information redundancy of point clouds, various compression methods or schemes have been reported in the literature, which may be broadly classified into two categories: geometry compression and attribute compression of static, dynamic and dynamically acquired point clouds. In the context of MPEG point cloud compression standardization, a static point cloud may refer to a point cloud that is a 3D representation of a
30 single object/scene (e.g., a building) and without any temporal information; dynamic point clouds may refer to a group of point cloud frames that capture the locations of a moving 3D object as a function of time; and dynamically acquired point clouds may refer to point cloud

sequences captured by LiDAR sensors, for example, equipped on autonomous driving vehicles for real-time perception of the surrounding environment.

[0005] For the design of effective point cloud attribute codecs, an important factor, akin to 2D visual data compression, is how to better exploit the spatial correlation between adjacent points in 3D space. This is because geometrically closer points may have a higher probability of sharing similar attribute and thus the information redundancy can be reduced using classical coding methodologies. Unfortunately, due to the difference in data structure and dimensionality, well-developed codecs used in other forms of content such as audio, image or video generally cannot be directly applied to 3D point clouds. As a result, a considerable amount of attribute compression algorithms that are specifically tailored for point cloud data have been devised. Among them, a conventional way to exploit the spatial correlation between adjacent points is based on the decomposition of 3D space using octree or kd-tree structure. For instance, a method has been introduced which partitions a point cloud into a layered structure and adopts a block-based intra prediction scheme to improve the coding efficiency. Apart from these, there have also been various studies that treat the attribute as signals over graph and compress the attribute using graph transform (GT). Although GT-based approaches may be effective for point cloud attribute compression, they generally require repeated eigendecompositions and may create isolated sub-graphs when a point cloud is sparse. To tackle this issue, for example, there has been disclosed a method to compress point clouds using region-adaptive hierarchical transform, which has a lower computational complexity but is slightly inferior in terms of rate-distortion performance.

[0006] Beyond the aforementioned conventional methods or schemes, there is another research branch that seeks to bridge the gap between high-dimensional and low-dimensional data compression. This is based on the consideration that 1D compression and 2D compression have been extensively investigated for decades and 3D point cloud attribute compression may benefit from these relatively mature compression techniques if an effective mapping pattern can be found to transform data from high dimension to lower dimension space. Towards this end, several image-based point cloud compression methods have been introduced. For example, there has been disclosed an image-based point cloud compression method whereby the points of a point cloud are structured using octree and then linearized into 1D point sequence in depth-first order. To obtain an attribute image, the points in the 1D point sequence are subsequently mapped to a 8×8 image pixel grid according to a horizontal snake curve pattern. Then, a classic image codec is employed to compress the obtained attribute image. However, it is noted

that such an image-based point cloud compression method may introduce many big jumps during the traversal and mapping process, which may undermine the spatial correlation between adjacent points. More recently, another image-based point cloud attribute compression method was proposed, whereby by performing principle component analysis, each point is projected
5 onto a specific plane of the bounding box of a point cloud. In this regard, 24 or even more projected images corresponding to the depth and RGB values are compressed using PNG and JPEG codecs. Although it may better exploit the spatial correlation between points, the images obtained using global projection are usually not compact enough, which may introduce too much additional information to encode. A panorama-image based approach has also been
10 presented for point cloud attribute compression. However, it is specially designed for point clouds generated by certain 3D laser measurement systems.

[0007] In 2017, MPEG launched a call for proposals that targets the standardization for point cloud compression, and developed three model categories: TMC1 for static point clouds, TMC2 (also known as V-PCC (video-based point cloud compression)) for time-varying point
15 clouds and TMC3 for dynamically acquired point clouds. Recently, TMC1 and TMC3 were merged into TMC13 and referred to as G-PCC (geometry-based point cloud compression). For G-PCC, there are two choices for attribute encoding: the region-adaptive hierarchical transform (RAHT) encoder and the level-of-details (LOD)-based encoder. The RAHT encoder is based on hierarchical transform and arithmetic coding, while the LOD-based encoder adopts an
20 interpolation-based prediction and lifting transform scheme for attribute compression. As for V-PCC, it also takes the advantage of sophisticated video encoding techniques and compresses point cloud attribute by partitioning a point cloud into patches through normal estimation and clustering and then directly projecting these 3D patches onto 2D images. Both of these two codecs have their individual merits, depending on the characteristics of point clouds. According
25 to comparative analysis of recent studies, V-PCC may be more suitable for point clouds with uniform point distribution in 3D space, while for non-uniform point clouds, G-PCC may more likely outperform V-PCC. A possible reason is that the noise and geometrical sparsity exhibited by non-uniform point clouds may affect the accuracy of normal estimation. Besides, V-PCC usually needs a very large projection plane for non-uniform point clouds, which would
30 significantly degrade the coding efficiency. Recently, deep learning based approaches have also been developed for point cloud compression. However, most of these existing methods mainly focus on the coding of the geometry information, which cannot be directly applied to point cloud attribute compression.

[0008] A need therefore exists to provide a method and a system for point cloud attribute compression, that seek to overcome, or at least ameliorate, problem(s) associated with conventional methods and systems for point cloud compression (e.g., point cloud attribute compression), such as but not limited to, improving efficiency and effectiveness in point cloud attribute compression. It is against this background that the present invention has been developed.

SUMMARY

[0009] According to a first aspect of the present invention, there is provided a method of point cloud attribute compression using at least one processor, the method comprising:

obtaining a plurality of three-dimensional (3D) patches of a point cloud, each 3D patch comprising a set of 3D points, each point having associated therewith corresponding attribute information;

generating, for each of the plurality of 3D patches, a two-dimensional (2D) attribute image of the 3D patch, to obtain a plurality of 2D attribute images of the plurality of 3D patches, wherein for at least a first 3D patch of the plurality of 3D patches, the 2D attribute image of the first 3D patch is generated based on a first attribute image generation process;

generating a 2D attribute image of the point cloud based on the plurality of 2D attribute images of the plurality of 3D patches; and

compressing the 2D attribute image of the point cloud based on a 2D image codec to obtain a compressed 2D attribute image of the point cloud,

wherein the first attribute image generation process comprises:

a point linearization stage configured to transform a set of 3D points of an input 3D patch inputted thereto into a one-dimensional (1D) point sequence of the input 3D patch;

and

a first 2D space filling stage configured to map the 1D point sequence of the input 3D patch to a first 2D image pixel grid to generate a 2D attribute image of the input 3D patch, wherein the point linearization stage comprises:

partitioning the set of 3D points of the input 3D patch into a first point subset and a second point subset of the set of 3D points of the input 3D patch; and

partitioning, for each point subset of the first and second point subsets, a set of 3D points of the point subset into a new first point subset and a new second point subset to replace the point subset in the set of 3D points of the input 3D patch, wherein for said

partitioning, for each point subset of the first and second point subsets, the set of 3D points of the point subset, a first 3D point of the second point subset which is nearest to a first pivot point of the first point subset is set as a first pivot point of the second point subset.

5 [0010] According to a second aspect of the present invention, there is provided a system for point cloud attribute compression comprising: a memory; and at least one processor communicatively coupled to the memory and configured to perform the method of point cloud attribute compression according to the above-mentioned first aspect of the present invention.

10 [0011] According to a third aspect of the present invention, there is provided a computer program product, embodied in one or more non-transitory computer-readable storage mediums, comprising instructions executable by at least one processor to perform the method of point cloud attribute compression according to the above-mentioned first aspect of the present invention.

BRIEF DESCRIPTION OF THE DRAWINGS

15 [0012] Embodiments of the present invention will be better understood and readily apparent to one of ordinary skill in the art from the following written description, by way of example only, and in conjunction with the drawings, in which:

FIG. 1 depicts a schematic flow diagram of a method of point cloud attribute compression, according to various embodiments of the present invention;

20 FIG. 2 depicts a schematic block diagram of a system for point cloud attribute compression, according to various embodiments of the present invention;

FIG. 3 depicts a schematic block diagram of an exemplary computer system which may be used to realize or implement the system for point cloud attribute compression, according to various embodiments of the present invention;

25 FIGs. 4A to 4C depict three examples of non-uniform 3D point clouds in public databases;

FIG. 5 depicts a schematic flow diagram of an example method of 3D point cloud attribute compression, according to various example embodiments of the present invention;

30 FIGs. 6A to 6E illustrate an example supervoxel and patch generation, according to various example embodiments of the present invention;

FIGs. 7A to 7C depict bipartite matching and attribute image comparison, according to various example embodiments of the present invention;

FIGs. 8A to 8F depict a BSP-based universal traversal method, according to various example embodiments of the present invention;

FIG. 9 shows several examples of the traversal order visualization of different 3D point linearization methods, according to various example embodiments of the present invention;

5 FIG. 10 depicts a schematic flow diagram of a hybrid 2D space filling pattern method, according to various example embodiments of the present invention;

FIG. 11 illustrates two example 3D patches of a point cloud, namely, a first patch (denoted by 1) and a second patch (denoted by 2), respectively, and the corresponding patch attribute images generated with IsoMap-based and SFC-based attribute image generation processes, respectively, according to various example embodiments of the present invention;

10 FIGs. 12A to 12Q depict example point clouds from a non-uniform 3D point cloud dataset;

FIG. 13 depicts a Table (Table I) showing a comparison of efficiency (the saving of BR and the equivalent PSNR improvement in dB) of the present BSP-based universal traversal method against two conventional traversal methods;

FIGs. 14A to 14C illustrate several examples of the autocorrelation at different lags associated with different traversal methods;

FIG. 15 depicts a Table (Table II) showing a comparison of the efficiency (the saving of BR and the equivalent PSNR improvement in dB) of different space filling patterns;

20 FIGs. 16A to 16Q depict a comparison of the present method with state-of-the-art point cloud attribute compression methods; and

FIG. 17 depicts a Table (Table III) showing an efficiency comparison of state-of-the-art methods with the results of RAHT encoder as the baseline.

25

DETAILED DESCRIPTION

[0013] Various embodiments of the present invention provide a method and a system for point cloud attribute compression, and more particularly, for image-based 3D point cloud attribute compression. As discussed in the background, there are various problems associated with conventional methods and systems for point cloud compression (e.g., point cloud attribute compression), resulting in inefficiencies and/or ineffectiveness thereof, especially in relation to the compression of non-uniform point clouds. Accordingly, various embodiments of the present invention provide a method and a system for point cloud attribute compression, that seek to overcome, or at least ameliorate, problem(s) associated with conventional methods and systems

for point cloud compression (e.g., point cloud attribute compression), such as but not limited to, improving efficiency and effectiveness in point cloud attribute compression, even in relation to non-uniform point clouds.

[0014] FIG. 1 depicts a schematic flow diagram of a method 100 of point cloud attribute compression using at least one processor, according to various embodiments of the present invention. The method 100 comprises: obtaining (at 102) a plurality of 3D patches of a point cloud (i.e., 3D point cloud), each 3D patch comprising a set of 3D points, each point having associated therewith corresponding attribute information; generating (at 104), for each of the plurality of 3D patches, a 2D attribute image of the 3D patch, to obtain a plurality of 2D attribute images of the plurality of 3D patches, whereby for at least a first 3D patch of the plurality of 3D patches, the 2D attribute image of the first 3D patch is generated based on a first attribute image generation process; generating (at 106) a 2D attribute image of the point cloud based on the plurality of 2D attribute images of the plurality of 3D patches; and compressing (at 108) the 2D attribute image of the point cloud based on a 2D image codec to obtain a compressed 2D attribute image of the point cloud. In particular, the first attribute image generation process comprises: a point linearization stage configured to transform a set of 3D points of an input 3D patch inputted thereto into a one-dimensional (1D) point sequence of the input 3D patch; and a first 2D space filling stage configured to map the 1D point sequence of the input 3D patch to a first 2D image pixel grid to generate a 2D attribute image of the input 3D patch. Furthermore, the point linearization stage comprises: partitioning the set of 3D points of the input 3D patch into a first point subset and a second point subset of the set of 3D points of the input 3D patch; and partitioning, for each point subset of the first and second point subsets, a set of 3D points of the point subset into a new first point subset and a new second point subset to replace the point subset in the set of 3D points of the input 3D patch, wherein for the above-mentioned partitioning, for each point subset of the first and second point subsets, the set of 3D points of the point subset, a first 3D point of the second point subset which is nearest to a first pivot point of the first point subset is set as a first pivot point of the second point subset.

[0015] Accordingly, the method 100 of point cloud attribute compression advantageously has improved efficiency and effectiveness in point cloud attribute compression, especially in relation to non-uniform point clouds. These advantages or technical effects, and/or other advantages or technical effects, will become more apparent to a person skilled in the art as the method 100 of point cloud attribute compression, as well as the corresponding system for point cloud attribute compression, is described in more detail according to various embodiments and

example embodiments of the present invention. It will be appreciated by a person skilled in the art that the present invention is not limited to any particular type of attribute information, and various types of attribute information of a point cloud (which may also be referred to as point cloud attribute(s)) are known in the art and are within the scope of the present invention, such as but not limited to, color, luminance, surface normal, reflectance, and so on. It will also be appreciated that a point cloud (and thus, each point thereof) may have associated therewith one or more types of attribute information.

[0016] In various embodiments, for the above-mentioned partitioning the set of 3D points of the input 3D patch, the method 100 further comprises: setting a first 3D point of the set of 3D points of the input 3D patch which is farthest from a centroid of the set of 3D points of the input 3D patch as a first pivot point of the set of 3D points of the input 3D patch; and setting a second 3D point of the set of 3D points of the input 3D patch which is farthest from the first pivot point of the set of 3D points as a second pivot point of the set of 3D points of the input 3D patch. In particular, the above-mentioned partitioning the set of 3D points of the input 3D patch comprises assigning each 3D point of the 3D points of the input 3D patch, except the first and second 3D points, to its nearest pivot point amongst the first and second pivot points of the set of 3D points to form the first point subset and the second point subset, the first point subset comprising the 3D points assigned to the first pivot point of the set of 3D points and the second point subset comprising the 3D points assigned to the second pivot point of the set of 3D points.

[0017] In various embodiments, for the above-mentioned partitioning, for each point subset of the first and second point subsets, the set of 3D points of the point subset, the method 100 further comprises: setting a first 3D point of the first point subset corresponding to the first pivot point of the set of 3D points as a second pivot point of the first point subset; setting a second 3D point of the first point subset which is farthest from the second pivot point of the first point subset as the first pivot point of the first point subset; setting the first 3D point of the second point subset which is nearest to the first pivot point of the first point subset as the first pivot point of the second point subset; and setting a second 3D point of the second point subset which is farthest from the first pivot point of the second point subset as the second pivot point of the second point subset.

[0018] In various embodiments, the above-mentioned partitioning, for each point subset of the first and second point subsets, the set of 3D points of the point subset, comprises: assigning each 3D point of the 3D points of the first point subset, except the first and second 3D points of the first point subset, to its nearest pivot point amongst the first and second pivot points of

the first point subset to form the new first point subset and the new second point subset to replace the first point subset in the set of 3D points of the input 3D patch, the new first point subset comprising the 3D points assigned to the first pivot point of the first point subset and the new second point subset comprising the 3D points assigned to the second pivot point of the first point subset; and assigning each 3D point of the 3D points of the second point subset, except the first and second 3D points of the second point subset, to its nearest pivot point amongst the first and second pivot points of the second point subset to form the new first point subset and the new second point subset to replace the second point subset in the set of 3D points of the input 3D patch, the new first point subset comprising the 3D points assigned to the first pivot point of the second point subset and the new second point subset comprising the 3D points assigned to the second pivot point of the second point subset.

[0019] In various embodiments, the point linearization stage further comprises, for each point subset in the set of 3D points of the input 3D patch iteratively until all point subsets therein have only one 3D point in each, partitioning a set of 3D points of the point subset into a new first point subset and a new second point subset to replace the point subset in the set of 3D points of the input 3D patch, to obtain a processed set of 3D points of the input 3D patch comprising ordered point subsets, each having only one 3D point therein. In other words, as long as there is at least one point subset in the set of 3D points having multiple 3D points therein, each of the at least one point subset is partitioned into a new first point subset and a new second point subset to replace the corresponding point subset in the set of 3D points that was partitioned, until there is no longer any point subset in the set of 3D points having multiple 3D points therein. Thereafter, the 1D point sequence of the input 3D patch is generated based on the processed set of 3D points of the input 3D patch.

[0020] In various embodiments, the first 2D image pixel grid comprises a series of macroblocks. In this regard, the first 2D space filling stage comprises: mapping the 1D point sequence of the input 3D patch to arrays of pixel slots of sub-macroblocks associated with the series of macroblocks according to a sub-macroblock filling pattern; and mapping the sub-macroblocks to the series of macroblocks according to a macroblock filling pattern. In particular, the sub-macroblock filling pattern and the macroblock filling pattern are different space filling patterns.

[0021] In various embodiments, the sub-macroblock filling pattern is a horizontal snake curve filling pattern and the macroblock filling pattern is a Hilbert curve filling pattern.

[0022] In various embodiments, each macroblock of the series of macroblocks has a size of 4×4 sub-macroblocks, and the array of pixel slots of each of the sub-macroblocks has a size of 4×4 pixel slots.

5 [0023] In various embodiments, the above-mentioned generating (at 104), for each of the plurality of 3D patches, the 2D attribute image of the 3D patch comprises: selecting one of a plurality of attribute image generation processes to generate the 2D attribute image of the 3D patch, the plurality of attribute image generation processes comprising the first attribute image generation process and a second attribute image generation process; and generating the 2D attribute image of the 3D patch based on the selected attribute image generation process.

10 [0024] In various embodiments, the above-mentioned selecting one of the plurality of attribute image generation processes to generate the 2D attribute image of the 3D patch is based on a level of uniformity of the set of 3D points of the 3D patch.

[0025] In various embodiments, the above-mentioned selecting one of the plurality of attribute image generation processes to generate the 2D attribute image of the 3D patch
15 comprises: selecting the second attribute image generation process to generate the 2D attribute image of the 3D patch if the set of 3D points of the 3D patch is determined to satisfy a predetermined condition relating to the level of uniformity, and selecting the first attribute image generation process to generate the 2D attribute image of the 3D patch if the set of 3D points of the 3D patch is determined to not satisfy the predetermined condition relating to the
20 level of uniformity.

[0026] In various embodiments, the set of 3D points of the 3D patch is determined to satisfy the predetermined condition relating to the level of uniformity if a reconstruction error associated with embedding the set of 3D points of the 3D patch into a 2D space is less than a predetermined error threshold, and the set of 3D points of the 3D patch is determined to not
25 satisfy the predetermined condition relating to the level of uniformity if the reconstruction error associated with embedding the set of 3D points of the 3D patch into the 2D space is more than the predetermined error threshold.

[0027] In various embodiments, for at least a second 3D patch of the plurality of 3D patches, the 2D attribute image of the second 3D patch is generated based on the second
30 attribute image generation process. In particular, the second attribute image generation process comprises: a dimensionality reduction stage configured to transform a set of 3D points of an input 3D patch inputted thereto into a set of 2D points of the input 3D patch (e.g., into a 2D patch); and a second 2D space filling stage configured to map the set of 2D points of the input

3D patch to a second 2D image pixel grid to generate a 2D attribute image of the input 3D patch.

[0028] In various embodiments, the second 2D space filling stage comprises: mapping each 2D point of the set of 2D points of the input 3D patch to a respective pixel slot of the second 2D image pixel grid based on minimizing error between pairwise distances of the set of 2D points of the input 3D patch and corresponding pairwise distances of the set of 2D points mapped to the second 2D image pixel grid.

[0029] In various embodiments, the second 2D space filling stage further comprises: adding one or more extra 2D points to one or more unfilled pixel slots in the second 2D image pixel grid remaining after said mapping each 2D point of the set of 2D points of the input 3D patch to the respective pixel slot of the second 2D image pixel grid.

[0030] In various embodiments, the second 2D space filling stage further comprises: determining a first 2D point of the set of 2D points of the input 3D patch which is farthest from a center of the set of 2D points of the input 3D patch; and configuring the one or more extra 2D points to respectively have associated therewith attribute information that is the same as the attribute information associated with the first 2D point.

[0031] In various embodiments, the method 100 further comprises combining the compressed 2D attribute image of the point cloud and auxiliary information for the compressed 2D attribute image of the point cloud, the auxiliary information comprising attribute image generation type information indicating, for each of the plurality of 3D patches, the type of attribute image generation process applied to generate the 2D attribute image of the 3D patch.

[0032] FIG. 2 depicts a schematic block diagram of a system 200 for point cloud attribute compression, according to various embodiments of the present invention, corresponding to the method 100 of point cloud attribute compression as described hereinbefore with reference to FIG. 1 according to various embodiments of the present invention. The system 200 comprises: a memory 202; and at least one processor 204 communicatively coupled to the memory 202 and configured to perform the method 100 of point cloud attribute compression as described herein according to various embodiments of the present invention. Accordingly, in various embodiments, the at least one processor 204 is configured to: obtain a plurality of 3D patches of a point cloud, each 3D patch comprising a set of 3D points, each point having associated therewith corresponding attribute information; generate, for each of the plurality of 3D patches, a 2D attribute image of the 3D patch to obtain a plurality of 2D attribute images of the plurality of 3D patches, whereby for at least a first 3D patch of the plurality of 3D patches, the 2D

attribute image of the first 3D patch is generated based on a first attribute image generation process; generate a 2D attribute image of the point cloud based on the plurality of 2D attribute images of the plurality of 3D patches; and compress the 2D attribute image of the point cloud based on a 2D image codec to obtain a compressed 2D attribute image of the point cloud. As
5 described hereinbefore, the first attribute image generation process comprises: a point linearization stage configured to transform a set of 3D points of an input 3D patch inputted thereto into a 1D point sequence of the input 3D patch; and a first 2D space filling stage configured to map the 1D point sequence of the input 3D patch to a first 2D image pixel grid to generate a 2D attribute image of the input 3D patch. In particular, the point linearization stage
10 comprises: partitioning the set of 3D points of the input 3D patch into a first point subset and a second point subset of the set of 3D points of the input 3D patch; and partitioning, for each point subset of the first and second point subsets, a set of 3D points of the point subset into a new first point subset and a new second point subset to replace the point subset in the set of 3D points of the input 3D patch, wherein for the above-mentioned partitioning, for each point
15 subset of the first and second point subsets, the set of 3D points of the point subset, a first 3D point of the second point subset which is nearest to a first pivot point of the first point subset is set as a first pivot point of the second point subset.

[0033] It will be appreciated by a person skilled in the art that the at least one processor 204 may be configured to perform various functions or operations through set(s) of instructions
20 (e.g., software modules) executable by the at least one processor 204 to perform various functions or operations. Accordingly, as shown in FIG. 2, the system 200 may comprise a point cloud patch module (or a point cloud patch circuit) 206 configured to perform the above-mentioned obtaining a plurality of 3D patches of a point cloud, each 3D patch comprising a set of 3D points, each point having associated therewith corresponding attribute information; a first
25 2D attribute image generating module (or a first 2D attribute image generating circuit) 208 configured to generate, for each of the plurality of 3D patches, a 2D attribute image of the 3D patch to obtain a plurality of 2D attribute images of the plurality of 3D patches, whereby for at least a first 3D patch of the plurality of 3D patches, the 2D attribute image of the first 3D patch is generated based on a first attribute image generation process; a second 2D attribute image
30 generating module (or a second 2D attribute image generating circuit) 210 configured to generate a 2D attribute image of the point cloud based on the plurality of 2D attribute images of the plurality of 3D patches; and a 2D attribute image compressing module (or a 2D attribute

image compressing circuit) 212 configured to compress the 2D attribute image of the point cloud based on a 2D image codec to obtain a compressed 2D attribute image of the point cloud.

[0034] It will be appreciated by a person skilled in the art that the above-mentioned modules are not necessarily separate modules, and one or more modules may be realized by or implemented as one functional module (e.g., a circuit or a software program) as desired or as appropriate without deviating from the scope of the present invention. For example, two or more of the point cloud patch module 206, the first 2D attribute image generating module 208, the second 2D attribute image generating module 210 and the 2D attribute image compressing module 212 may be realized (e.g., compiled together) as one executable software program (e.g., software application or simply referred to as an “app”), which for example may be stored in the memory 202 and executable by the at least one processor 204 to perform various functions/operations as described herein according to various embodiments of the present invention.

[0035] In various embodiments, the system 200 for point cloud attribute compression corresponds to the method 100 of point cloud attribute compression as described hereinbefore with reference to FIG. 1, therefore, various functions or operations configured to be performed by the least one processor 204 may correspond to various steps or operations of the method 100 of point cloud attribute compression as described herein according to various embodiments, and thus need not be repeated with respect to the system 200 for point cloud attribute compression for clarity and conciseness. In other words, various embodiments described herein in context of the methods are analogously valid for the corresponding systems, and vice versa.

[0036] For example, in various embodiments, the memory 202 may have stored therein the point cloud patch module 206, the first 2D attribute image generating module 208, the second 2D attribute image generating module 210 and/or the 2D attribute image compressing module 212, which respectively correspond to various steps (or operations or functions) of the method 100 of point cloud attribute compression as described herein according to various embodiments, which are executable by the at least one processor 204 to perform the corresponding functions or operations as described herein.

[0037] A computing system, a controller, a microcontroller or any other system providing a processing capability may be provided according to various embodiments in the present disclosure. Such a system may be taken to include one or more processors and one or more computer-readable storage mediums. For example, the system 200 for point cloud attribute compression described hereinbefore may include a processor (or controller) 204 and a

computer-readable storage medium (or memory) 202 which are for example used in various processing carried out therein as described herein. A memory or computer-readable storage medium used in various embodiments may be a volatile memory, for example a DRAM (Dynamic Random Access Memory) or a non-volatile memory, for example a PROM (Programmable Read Only Memory), an EPROM (Erasable PROM), EEPROM (Electrically Erasable PROM), or a flash memory, e.g., a floating gate memory, a charge trapping memory, an MRAM (Magnetoresistive Random Access Memory) or a PCRAM (Phase Change Random Access Memory).

[0038] In various embodiments, a “circuit” may be understood as any kind of a logic implementing entity, which may be special purpose circuitry or a processor executing software stored in a memory, firmware, or any combination thereof. Thus, in an embodiment, a “circuit” may be a hard-wired logic circuit or a programmable logic circuit such as a programmable processor, e.g., a microprocessor (e.g., a Complex Instruction Set Computer (CISC) processor or a Reduced Instruction Set Computer (RISC) processor). A “circuit” may also be a processor executing software, e.g., any kind of computer program, e.g., a computer program using a virtual machine code, e.g., Java. Any other kind of implementation of the respective functions may also be understood as a “circuit” in accordance with various embodiments. Similarly, a “module” may be a portion of a system according to various embodiments and may encompass a “circuit” as described above, or may be understood to be any kind of a logic-implementing entity.

[0039] Some portions of the present disclosure are explicitly or implicitly presented in terms of algorithms and functional or symbolic representations of operations on data within a computer memory. These algorithmic descriptions and functional or symbolic representations are the means used by those skilled in the data processing arts to convey most effectively the substance of their work to others skilled in the art. An algorithm is here, and generally, conceived to be a self-consistent sequence of steps leading to a desired result. The steps are those requiring physical manipulations of physical quantities, such as electrical, magnetic or optical signals capable of being stored, transferred, combined, compared, and otherwise manipulated.

[0040] Unless specifically stated otherwise, and as apparent from the following, it will be appreciated that throughout the present specification, description or discussions utilizing terms such as “obtaining”, “producing”, “generating”, “compressing”, “transforming”, “mapping”, “partitioning”, “setting”, “assigning”, “selecting”, “determining”, “adding”, “configuring”,

“encoding” or the like, refer to the actions and processes of a computer system, or similar electronic device, that manipulates and transforms data represented as physical quantities within the computer system into other data similarly represented as physical quantities within the computer system or other information storage, transmission or display devices.

5 [0041] The present specification also discloses a system (e.g., which may also be embodied as a device or an apparatus), such as the system 200 for point cloud attribute compression, for performing various operations/functions of various methods described herein. Such a system may be specially constructed for the required purposes, or may comprise a general purpose computer or other device selectively activated or reconfigured by a computer program stored
10 in the computer. The algorithms presented herein are not inherently related to any particular computer or other apparatus. Various general-purpose machines may be used with computer programs in accordance with the teachings herein. Alternatively, the construction of more specialized apparatus to perform various method steps may be appropriate.

[0042] In addition, the present specification also at least implicitly discloses a computer
15 program or software/functional module, in that it would be apparent to the person skilled in the art that individual steps of various methods described herein may be put into effect by computer code. The computer program is not intended to be limited to any particular programming language and implementation thereof. It will be appreciated that a variety of programming languages and coding thereof may be used to implement the teachings of the disclosure
20 contained herein. Moreover, the computer program is not intended to be limited to any particular control flow. There are many other variants of the computer program, which can use different control flows without departing from the scope of the invention. It will be appreciated by a person skilled in the art that various modules described herein (e.g., the point cloud patch module 206, the first 2D attribute image generating module 208, the second 2D attribute
25 image generating module 210 and/or the 2D attribute image compressing module 212) may be software module(s) realized by computer program(s) or set(s) of instructions executable by a computer processor to perform the required functions, or may be hardware module(s) being functional hardware unit(s) designed to perform the required functions. It will also be appreciated that a combination of hardware and software modules may be implemented.

30 [0043] Furthermore, one or more of the steps of a computer program/module or method described herein may be performed in parallel rather than sequentially. Such a computer program may be stored on any computer readable medium. The computer readable medium may include storage devices such as magnetic or optical disks, memory chips, or other storage

devices suitable for interfacing with a general purpose computer. The computer program when loaded and executed on such a general-purpose computer effectively results in an apparatus that implements the steps of the methods described herein.

5 [0044] In various embodiments, there is provided a computer program product, embodied in one or more computer-readable storage mediums (non-transitory computer-readable storage medium(s)), comprising instructions (e.g., the point cloud patch module 206, the first 2D attribute image generating module 208, the second 2D attribute image generating module 210 and/or the 2D attribute image compressing module 212) executable by one or more computer processors to perform the method 100 of point cloud attribute compression, as described herein
10 with reference to FIG. 1 according to various embodiments. Accordingly, various computer programs or modules described herein may be stored in a computer program product receivable by a system therein, such as the system 200 for point cloud attribute compression as shown in FIG. 2, for execution by at least one processor 204 of the system 200 to perform various functions.

15 [0045] Software or functional modules described herein may also be implemented as hardware modules. More particularly, in the hardware sense, a module is a functional hardware unit designed for use with other components or modules. For example, a module may be implemented using discrete electronic components, or it can form a portion of an entire electronic circuit such as an Application Specific Integrated Circuit (ASIC). Numerous other possibilities exist. Those skilled in the art will appreciate that the software or functional
20 module(s) described herein can also be implemented as a combination of hardware and software modules.

[0046] In various embodiments, the system 200 for point cloud attribute compression may be realized by any computer system (e.g., desktop or portable computer system) including at
25 least one processor and a memory, such as a computer system 300 as schematically shown in FIG. 3 as an example only and without limitation. Various methods/steps or functional modules may be implemented as software, such as a computer program being executed within the computer system 300, and instructing the computer system 300 (in particular, one or more processors therein) to conduct various functions or operations as described herein according to
30 various embodiments. The computer system 300 may comprise a computer module 302, input modules, such as a keyboard and/or a touchscreen 304 and a mouse 306, and a plurality of output devices such as a display 308, and a printer 310. The computer module 302 may be connected to a computer network 312 via a suitable transceiver device 314, to enable access to

e.g., the Internet or other network systems such as Local Area Network (LAN) or Wide Area Network (WAN). The computer module 302 in the example may include a processor 318 for executing various instructions, a Random Access Memory (RAM) 320 and a Read Only Memory (ROM) 322. The computer module 302 may also include a number of Input/Output (I/O) interfaces, for example I/O interface 324 to the display 308, and I/O interface 326 to the keyboard 304. The components of the computer module 302 typically communicate via an interconnected bus 328 and in a manner known to the person skilled in the relevant art.

[0047] It will be appreciated by a person skilled in the art that the terminology used herein is for the purpose of describing various embodiments only and is not intended to be limiting of the present invention. As used herein, the singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. It will be further understood that the terms "comprises" and/or "comprising," when used in this specification, specify the presence of stated features, integers, steps, operations, elements, and/or components, but do not preclude the presence or addition of one or more other features, integers, steps, operations, elements, components, and/or groups thereof.

[0048] Any reference to an element or a feature herein using a designation such as "first", "second" and so forth does not limit the quantity or order of such elements or features, unless stated or the context requires otherwise. For example, such designations may be used herein as a convenient way of distinguishing between two or more elements or instances of an element. Thus, a reference to first and second elements does not necessarily mean that only two elements can be employed, or that the first element must precede the second element. In addition, a phrase referring to "at least one of" a list of items refers to any single item therein or any combination of two or more items therein.

[0049] In order that the present invention may be readily understood and put into practical effect, various example embodiments of the present invention will be described hereinafter by way of examples only and not limitations. It will be appreciated by a person skilled in the art that the present invention may, however, be embodied in various different forms or configurations and should not be construed as limited to the example embodiments set forth hereinafter. Rather, these example embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the present invention to those skilled in the art.

[0050] Various example embodiments of the present invention provide a method of image-based 3D point cloud attribute compression using two-stage dimensionality transformation

(which may herein be referred to as the present method, e.g., corresponding to the method 100 of point cloud attribute compression as described hereinbefore according to various embodiments), according to various example embodiments of the present invention.

5 [0051] For example, the wide availability of 3D scanning equipment and ever-growing 3D applications are generating more and more point cloud data at an unprecedented rate, and this poses great challenges for efficient and economic data storage, transmission and processing. To alleviate this situation, various codecs have been tailored for point cloud data. However, their compression efficiency is still far from being satisfactory due to the difficulties caused by structural irregularity and higher space dimensionality of data points. Accordingly, various
10 example embodiments of the present invention provide an image-based approach for static point cloud attribute compression. The present method provides two attribute image generation processes (or schemes), namely, a first attribute image generation process (or simply referred to herein as a first process or scheme, e.g., corresponding to the first attribute image generation process as described hereinbefore according to various embodiments) and a second attribute
15 image generation process (or simply referred to herein a second process or scheme, e.g., corresponding to the second attribute image generation process as described hereinbefore according to various embodiments) to map the attribute (i.e., attribute information) of a 3D point cloud into image pixel grids while preserving the spatial correlation between adjacent points.

20 [0052] In various example embodiments, for the first attribute image generation process, unordered 3D points are linearized into a 1D point sequence using a binary space partition (BSP) based universal traversal method (or algorithm) (e.g., corresponding to the point linearization stage of the first attribute image generation process as described hereinbefore according to various embodiments), and a synthetic attribute image is obtained by mapping the 1D point
25 sequence onto a 2D grid structure layout according to a hybrid space-filling pattern (e.g., corresponding to the first 2D space filling stage of the first attribute image generation process as described hereinbefore according to various embodiments). In various example embodiments, for the second attribute image generation process, points in 3D space are first transformed into 2D using an IsoMap-based dimensionality reduction method (e.g., corresponding to the
30 dimensionality reduction stage of the second attribute image generation process as described hereinbefore according to various embodiments) and then compactly arranges the obtained 2D point cloud into image pixel grids (e.g., corresponding to the second 2D space filling stage of the second attribute image generation process as described hereinbefore according to various

embodiments). In various example embodiments, a mode selection module is provided to adaptively choose (or select) the most or more suitable attribute image generation process for each patch (3D patch) of a point cloud. Thereafter, effective point cloud attribute compression can be achieved by taking advantage of well-developed or conventional image codecs in the art (e.g., corresponding to the 2D image codec as described hereinbefore according to various embodiments), such as but not limited to, JPEG and WebP. In this regard, experimental results on a standard public dataset demonstrate the efficiency and effectiveness of the present method in point cloud attribute compression, even in handling non-uniform point clouds, which will be discussed later below according to various example embodiments of the present invention.

10 [0053] As discussed in the background, there are various problems associated with conventional methods and systems for point cloud compression (e.g., point cloud attribute compression), resulting in inefficiencies and/or ineffectiveness thereof, especially in relation to the compression of non-uniform point clouds.

[0054] For example, for image-based point cloud attribute compression, conventional methods may either use tree structure based two-stage mapping paradigm or single-stage 3D to 2D mapping paradigm. For the tree structure based two-stage mapping paradigm, 3D points of a point cloud are first linearized into 1D point sequence using octree-based depth-first traversal and then mapped into image pixel grids following a specific space-filling pattern. However, the performance of methods based on such a paradigm lags significantly behind state-of-the-art point cloud attribute codecs such as G-PCC. For the single-stage 3D to 2D mapping paradigm, 3D points are directly projected to their corresponding projection planes by performing normal estimation and these planes are then processed and packed together to generate a synthetic image for compression. Codecs based on this paradigm, such as V-PCC, may perform well on uniform point clouds but experience noticeable performance degradation on non-uniform ones. In various example embodiments, to alleviate this situation, the present method adopts a two-stage mapping paradigm with a number of components that are superior than conventional methods in handling non-uniform 3D point clouds.

[0055] Although traditional depth-first traversal schemes based on tree structures (e.g. octree, k-d tree) can well retain a considerable portion of spatial correlation, they may also introduce many inevitable big jumps caused by the intrinsic structure of a point cloud and the traversal patterns. In contrast, various example embodiments provide a BSP-based universal traversal process for 3D points linearization, which adopts a heuristic strategy to preserve the inherent coherence of points by iteratively partitioning a given point set into 1D point sequence.

Compared with octree-based depth-first traversal and 3D Hilbert space-filling curve (SFC) based traversal, the BSP based universal traversal process of the present method advantageously minimises the number of big jumps (e.g., introduces significantly fewer big jumps) during the point linearization process and thus achieves significant compression performance gains.

5 [0056] To obtain a compact image for attribute compression, the 1D point sequence obtained with point linearization method is mapped onto a 2D layout (2D image pixel grid) using a space-filling pattern. A simple way to achieve this may be to sequentially assign each point in the 1D stream to its corresponding pixel grid of a pre-defined image canvas according to one of the patterns of common SFCs, such as zigzag, Z-order, Peano, and so on. However,
10 various example embodiments note that these simple schemes either easily destroy the point coherence during the mapping process or are incapable of fully exerting the block intra prediction merits of advanced image codecs. In various example embodiments, to address this issue, the above-mentioned hybrid space-filling curve for point cloud attribute image generation is provided, which has been advantageously found to well retain the attribute correlation in sub-
15 macroblocks, as well as that between adjacent macroblocks, compared to existing space-filling schemes.

[0057] Various example embodiments note that point cloud codecs using the above-mentioned two-stage transformation paradigm (i.e., 3D-1D-2D) generally work well on a point cloud with high attribute variance since their inherent coherence is relatively weak. However,
20 for those with specific texture patterns, various example embodiments note that they may not be able to well maintain the spatial correlation during the linearization and space-filling processes. In this regard, for a point cloud with high attribute variance or is deemed sufficiently uniform, and in particular, for 3D patches thereof with high attribute variance or are deemed sufficiently uniform, various example embodiments provide the above-mentioned IsoMap-
25 based point cloud attribute image generation method to reduce the dimensionality of each of such 3D patches from 3D to 2D and then compactly arranges the transformed 2D points into a respective 2D image pixel grid with a distance-preserving placement method, which has been advantageously found to better maintain the spatial correlation among adjacent points.

[0058] Various example embodiments note that how to segment a point cloud into fine-grained patches is an important factor that may affect the coding efficacy. For example, V-PCC
30 segments point cloud into patches according to the estimated normals of points and packs all the obtained 2D patches into one image for compression. However, it usually introduces too much extra information that needs to be encoded and is very likely to destroy the coherence

between adjacent patches. Besides, the method is very inefficient for non-uniform point cloud attribute compression because it usually needs a very large projection plane for such point clouds and noise and sparsity may affect the accuracy of normal estimation. Various example embodiments seek to address this issue by developing a uniform 3D patch generation and patch attribute image assembling method. With this unified setting or configuration, the 2D attribute images of a point cloud can be efficiently generated and the spatial correlation between adjacent patches can also be well retained. In various example embodiments, an attribute image generation process mode (or type) selection module is provided to adaptively choose the most or more suitable attribute image generation process for each 3D patch of a point cloud, which has been found to improve the attribute coding efficacy of non-uniform point clouds.

[0059] Accordingly, various example embodiments of the present invention focus on the attribute compression of static point clouds, especially those that exhibit geometrical sparsity, noise and irregular point distribution (which may herein be collectively referred to as non-uniform point clouds, as shown in FIGs. 4A to 4C by way of examples only), which poses unique challenges and has been inadequately addressed by conventional studies. In particular, for illustration purpose, FIGs. 4A to 4C depict three examples of non-uniform 3D point clouds in public databases, each example showing an enlarged non-uniform section of a point cloud. In this regard, various example embodiments note that effective compression of non-uniform point clouds is very practical since raw point clouds are usually non-uniform in practice. According to various example embodiments, whether a set of 3D points of a 3D patch of a point cloud is considered to be uniform or non-uniform may be determined based on whether the set of 3D points satisfy a predetermined condition or criterion, such as whether a reconstruction error associated with embedding the set of 3D points of the 3D patch into a 2D space is less than or more than a predetermined error threshold, as will be described later below.

[0060] Accordingly, various example embodiments leverage existing advanced 2D visual data compression techniques and implements an image-based attribute compression of non-uniform 3D point clouds. Instead of directly mapping 3D point patches to 2D attribute images in all situations, various example embodiments adopt a two-stage dimensionality transformation paradigm (which may be referred to herein as Stage I and Stage II) and introduce the above-mentioned two possible types of attribute image generation processes (or schemes) to generate a synthetic attribute image for a given point cloud by mapping its points into 2D image pixel grids. For the first attribute image generation process, 3D points are transformed to 1D point sequence in Stage I and the 1D point sequence is then mapped to a 2D image pixel

grid (i.e., a 2D grid structure layout, each grid slot corresponding to a pixel) in Stage II. For the second attribute image generation process, 3D points are converted into a 2D point cloud in Stage I using IsoMap, which is a non-linear dimensionality reduction technique that estimates the intrinsic dimension of a set of data points based on the geodesic distances imposed by a weighted neighbourhood graph, and the obtained 2D points are subsequently assigned to their corresponding pixel locations based on a distance-preserving point placement technique in Stage II. In various example embodiments, these two types of attribute image generation processes are adaptively selected for each 3D patch of the point cloud according to the geometry characteristics of the 3D patch (e.g., based on a level of uniformity of the set of 3D points of the 3D patch).

[0061] Accordingly, in various example embodiments, an IsoMap-based attribute image generation technique, a BSP-based universal traversal technique and a hybrid 2D space-filling pattern technique are advantageously provided in the method of point cloud attribute compression. For example, the IsoMap-based attribute image generation technique is provided to preserve the inherent coherence between points during the 3D to 2D transformation process and improve the efficiency of point cloud attribute compression. For example, the BSP-based universal traversal technique is provided to linearize 3D points of a point cloud which better maintaining their spatial correlation than using traditional traversal methods. The hybrid 2D space-filling pattern technique is provided to map the obtained 1D point sequence to 2D image pixel grids, which is more effective in eliminating redundancy and thus achieves better coding efficiency than existing space-filling techniques for attribute image synthesis.

[0062] According to various example embodiments, for better understanding, an example methodological overview of the present method, along with an example process of point cloud supervoxel and patch generation and examples of the above-mentioned two types (e.g., modes) of attribute image generation processes will now be described below, followed by experimental results and analysis.

METHODOLOGICAL OVERVIEW AND POINT CLOUD PREPROCESSING

Example Overview of the Present Method

[0063] FIG. 5 depicts a schematic flow diagram of an example method of 3D point cloud attribute compression according to various example embodiments (e.g., corresponding to the method 100 of point cloud attribute compression as described hereinbefore according to various

embodiments). As shown in FIG. 5, the method may comprise three components or main stages, namely, (1) point cloud preprocessing, (2) two-stage dimensionality transformation and (3) auxiliary information and image compression.

5 [0064] In the point cloud preprocessing, a given 3D point cloud is processed or segmented into a series of basic units, i.e., patches (3D patches), using a patch generation method according to various example embodiments of the present invention. In this regard, each patch may be classified into a transformable patch or an untransformable patch for further processing according to the complexity of their geometric structures. In various example embodiments, a patch may be classified as transformable or untransformable (or non-transformable) based on a level of uniformity of a set of 3D points of the patch. For example, the patch may be classified as transformable if the set of 3D points of the patch is determined to be uniform (or sufficiently uniform), and the patch may be classified as untransformable if the set of 3D points of the patch is determined to be non-uniform (or not sufficiently uniform).

15 [0065] In the two-stage dimensionality transformation, a two-stage dimensionality transformation technique is applied based on two possible types (or modes) of attribute image generation (AIG) processes or schemes, namely, the above-mentioned IsoMap-based attribute image generation process and the above-mentioned hybrid 2D space-filling curve (SFC) based attribute image generation process, to synthesize attribute images for the obtained transformable and untransformable patches, respectively. For IsoMap-based attribute image generation process, a transformable 3D patch is converted or transformed into a 2D point cloud using IsoMap-based dimensionality reduction technique in Stage I, and then an attribute image may be synthesized by assigning the transformed 2D points to an image pixel grid (a 2D image pixel grid) based on a bipartite matching technique (or algorithm) in Stage II. On the other hand, for the SFC-based attribute image generation process, the 3D points of an untransformable patch are linearized into a 1D point sequence using the BSP-based universal traversal algorithm according to various example embodiments in Stage I, and then the 1D point sequence of the untransformable patch is mapped to an image pixel grid using a hybrid space-filling pattern according to various example embodiments in Stage II.

25 [0066] In the auxiliary information and image compression, the 2D attribute image of the whole point cloud is harvested or generated by assembling all the 2D attribute images of patches of the point cloud together while maintaining the spatial correlation among adjacent patches. In this regard, in various example embodiments, auxiliary information associated with the patch generation of the point cloud (i.e., generation of the 3D patches of the point cloud) may be

stored to facilitate the decoding process of the 2D attribute image of the point cloud. In various example embodiments, in the patch generation process, as will be described in more detail later below, a point cloud may be segmented into supervoxels, and if a reconstruction error associated with embedding a set of 3D points of a supervoxel into a 2D space is more than (or equal to or more than) a predetermined error threshold, the supervoxel may be recursively divided into two sub-clusters until a predetermined criteria for stopping is met. Therefore, for a point cloud, each supervoxel may be associated with a binary tree, in which the leaf nodes are the final patches (3D patches) of the point cloud. For example, in the auxiliary information, various example embodiments may allocate one bit for each non-leaf node to indicate whether the non-leaf node satisfies the stopping criterion, and one bit for each leaf node to indicate whether the leaf node (i.e., the 3D patch) is transformable or untransformable. For the obtained attribute image of the point cloud, it may be compressed using conventional image codecs such as JPEG and WebP. The final compressed bit stream may be obtained by combining or appending the compressed attribute image of the point cloud to the auxiliary information.

15

Supervoxel and Patch Generation

[0067] In various example embodiments, as many point clouds are geometrically and texturally complex, each point cloud is segmented them into basic structural units, such as supervoxels. The supervoxels may then be processed into 3D patches to facilitate further processing including point linearization and dimensionality reduction as described hereinbefore. In this regard, various example embodiments provide a simple yet effective method for point cloud supervoxel generation. However, it will be appreciated by a person skilled in the art that the present invention is not limited to this example method for point cloud supervoxel generation, and other point cloud supervoxel segmentation methods in the art may also be applied to generate point cloud supervoxel as desired or as appropriate. Initially, a constrained Poisson-disk sampling method (or algorithm) (e.g., as described in Corsini *et al.*, “Efficient and flexible sampling with blue noise properties of triangular meshes,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 6, pp. 914–924, 2012) may be employed to obtain two simplified versions of an original point cloud by setting two different numbers of samples N_1 and N_2 , where N_1 is a relatively larger number that would harvest a more granular simplified point cloud, while N_2 is a smaller parameter with which the sampling algorithm would generate a coarse representation of the original point cloud, which may be referred to as a coarse simplified point cloud. With the points of the coarse simplified point cloud serving as

20

25

30

seed points, supervoxels of a point cloud may be generated by assigning the points of the granular simplified point cloud obtained based on N_1 to their respective nearest seed point of the coarse simplified point cloud. In various example embodiments, a ball-tree structure is constructed (e.g., as described in Omohundro, *Five balltree construction algorithm*,
5 International Computer Science Institute Berkeley, 1989) for nearest neighbour search, and N_1 and N_2 are empirically estimated by dividing the total number of points of a point cloud by 32 and 8192, respectively, which was found to achieve a balance between compression ratio and computational complexity. FIGs. 6A to 6E illustrate an example supervoxel and patch generation according to various example embodiments of the present invention, whereby FIG.
10 6A depicts an example original point cloud (3D point cloud) 602; FIG. 6B depicts an example granular simplified point cloud 604 based on the number of samples being set to N_1 ; FIG. 6C depicts example supervoxels 606 generated; FIG. 6D depicts example simplified 3D patches 608 generated (untransformable patches are marked with the darkest shade of black, while transformable patches are marked with lighter shades of black); and FIG. 6E depicts example
15 3D patches 610 of the example original point cloud generated (similarly, untransformable patches are marked with the darkest shade of black, while transformable patches are marked with lighter shades of black).

[0068] Various example embodiments note that although the given point cloud may be segmented into supervoxels having relatively simple geometric structure, it cannot be ensured
20 that each supervoxel can be embedded into a lower dimensional space (3D space to 2D space) with a sufficiently or acceptably small reconstruction error. To address this issue, various example embodiments further examine the transformability of each supervoxel and segment the supervoxel into sub-clusters (or point clusters) if a reconstruction error associated with embedding a set of 3D points of the supervoxel into a 2D space is more than a predetermined
25 error threshold. In various example embodiments, an IsoMap-based method (e.g., as described in Tenenbaum *et al.*, “A global geometric framework for nonlinear dimensionality reduction,” *science*, vol. 290, no. 5500, pp. 2319–2323, 2000) is employed to reduce the dimensionality of a supervoxel from 3D to 2D, and the reconstruction error for the embedding is evaluated, which will be described in further details later below. The IsoMap dimensionality reduction and
30 reconstruction error evaluation will also be described later below in more details according to various example embodiments of the present invention. In various example embodiments, if the reconstruction error is larger than a predetermined or pre-defined threshold (e.g., larger than 5 when the reconstruction error is as defined in Equation 4 (whereby σ_e is set to 0.75) to be

described later below), agglomerative clustering is performed to divide the supervoxel into two sub-clusters (or point clusters) and repeat the process until a predetermined criterion (e.g., one or more predetermined constraint conditions) for stopping the process is met. In various example embodiments, a threshold for the minimum number of points (e.g., the minimum number may be set to 32 or any number as deemed appropriate) of a patch may also be set to avoid generating too many small patches, since they may destroy the spatial correlation during the assembling process. Accordingly, by way of an example only and without limitation, the above-mentioned predetermined criterion may be set or defined as the number of points in a supervoxel is less than a predetermined number (e.g., 32) or the reconstruction error (e.g., as defined in Equation 4 below whereby σ_e is set to 0.75) is smaller than or equal to a predetermined threshold (e.g., 5). It will be appreciated by a person skilled in the art that the present invention is not limited to the above-mentioned exemplary predetermined criterion or the above-mentioned exemplary values thereof, and that they may be modified as desired or as appropriate without going outside the scope of the present invention.

15 **[0069]** For example, FIG. 6D depicts example simplified 3D patches 608 generated including the obtained fine-grained clusters. Subsequently, 3D patches of the original point cloud can be generated by assigning each point of the original point cloud to its nearest cluster, resulting in the plurality of 3D patches 610 of the original point clouds as shown in FIG. 6E. In various example embodiments, 3D patches that have been determined to not be able to be embedded into 2D space may be referred to as untransformable patches (marked with the darkest shade of black in FIG. 6E) and the remaining patches may be referred to as transformable patches (marked with lighter shades of black), as shown in FIG. 6E. In various example embodiments, in the above-mentioned agglomerative clustering of a supervoxel, a point cluster that meets the above-mentioned predetermined criterion for stopping the process and has a reconstruction error that is larger than the predetermined threshold (e.g., larger than 5 when the reconstruction error is as defined in Equation 4 (whereby σ_e is set to 0.75)) may be determined as not be able to be embedded into 2D space, and may thus be determined or defined as an untransformable 3D patch. By way of an example only and without limitation, the above-mentioned predetermined criterion may be set or defined as the number of points in a supervoxel is less than a predetermined number (e.g., 32) or the reconstruction error (e.g., as defined in Equation 4 below whereby σ_e is set to 0.75), is smaller than or equal to a predetermined threshold (e.g., 5). According to various example embodiments, these two

different types (or classes) of 3D patches are converted into 2D attribute images using different attribute image generation processes.

ISOMAP-BASED METHOD FOR POINT CLOUD ATTRIBUTE IMAGE GENERATION

- 5 [0070] To obtain a compact image for point cloud attribute compression, various example embodiments provide an IsoMap-based attribute image generation process (e.g., corresponding to the second attribute image generation process as described hereinbefore according to various embodiments) for transformable 3D patches. In this process, the transformable 3D patches are each first transformed into 2D point clouds by performing dimensionality reduction in the first stage (Stage I). In the second stage (stage II), attribute image generation is configured as a bipartite matching problem and the obtained 2D points of the 2D point clouds from Stage I are assigned to their corresponding image pixel grids while preserving their inherent coherence.

IsoMap-based Dimensionality Reduction

- 15 [0071] The IsoMap-based dimensionality reduction stage (Stage I, e.g., corresponding to the dimensionality reduction stage of the second attribute image generation process as described hereinbefore according to various embodiments) is configured to reduce the dimensionality of 3D patches from 3D to 2D for further processing by the 2D-based bipartite matching in Stage II. It will be appreciated by a person skilled in the art that the present invention is not limited to the IsoMap-based dimensionality reduction method and other dimensionality reduction methods (or algorithms) in the art may also be applied to reduce transform a 3D patch to a 2D point clouds as desired or as appropriate. However, in various example embodiments, the IsoMap-based dimensionality reduction method is preferred because of its high efficiency. Given a 3D patch with N points, the steps of IsoMap embedding include:

- 25 • Construct a neighbour graph by connecting each point to its K (K is set to 8 in our implementation) nearest neighbours. The edge length $d(i, j)$ between two neighbour points p_i and p_j equals to their Euclidean distance in 3D space;
- Compute the shortest path for each pair of points to obtain a symmetric squared geodesic distance matrix $\mathbf{G}_N$;
- 30 • Calculate the doubly centered geodesic distance matrix $\widehat{\mathbf{G}}_N$ with the operator below:

$$\widehat{\mathbf{G}}_N = -\frac{1}{2}\mathbf{C}_N\mathbf{G}_N\mathbf{C}_N$$

(Equation 1)

where $\mathbf{C}_N$ is the centering matrix, defined by:

$$\mathbf{C}_N = \mathbf{I}_N - \frac{1}{N} \mathbf{e}_N \mathbf{e}_N^T \quad (\text{Equation 2})$$

where $\mathbf{I}_N$ is the identity matrix of size N and $\mathbf{e}_N$ is a column vector of N ones;

- 5 • Compute the M largest eigenvalues $\lambda_1, \lambda_2, \dots, \lambda_M$ and corresponding eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_M$. For each point p_i of the supervoxel, the m -th component of the vector $\mathbf{q}_i$ in M dimension space may be calculated by $\mathbf{q}_i^m = \sqrt{\lambda_m} \mathbf{v}_m^i$ ($m = 1, 2, \dots, M$).

[0072] To evaluate the reconstruction error for an IsoMap embedding, for example, the following metric may be used:

$$10 \quad \xi = \frac{1}{N} \|\hat{\mathbf{G}}_N - \tilde{\mathbf{G}}_N\|_F \quad (\text{Equation 3})$$

where N is the number of points of a patch, $\hat{\mathbf{G}}_N$ and $\tilde{\mathbf{G}}_N$ represent the distance matrix for the original points and embedded points, respectively, and $\|\cdot\|_F$ is the Frobenius norm. This metric calculates the total loss for the embedding, but various example embodiments note that it does not evaluate the reconstruction error in a local region. In this regard, various example embodiment introduce another cost function that takes into account the local continuity (e.g., as described in *Najim et al.*, “Trustworthy dimension reduction for visualization different data sets,” *Information Sciences*, vol. 278, pp. 206–220, 2014), which is more suitable for evaluating the change of correlation among adjacent points than the above-mentioned metric. For a point $p_i \in \mathbb{R}^{D_o}$ in the original space and its corresponding transformed point $\hat{p}_i \in \mathbb{R}^{D_e}$ in the embedding space, let $\mathcal{N}_k^{D_o}(p_i)$ and $\mathcal{N}_k^{D_e}(\hat{p}_i)$ be the sets of their k -nearest neighbours, respectively, the cost function may be defined as follow:

$$\hat{\xi} = \sum_{i=1}^N I \left[\frac{|\mathcal{N}_k^{D_o}(p_i) \cap \mathcal{N}_k^{D_e}(\hat{p}_i)|}{k} > \sigma_e \right] \quad (\text{Equation 4})$$

25 where $I[\cdot]$ denotes an indicator function, whose value is 1 if the condition is true, and 0 otherwise. Accordingly, in various example embodiments, the metric ($\hat{\xi}$) in Equation (4) is used to determine whether a cluster is to be further partitioned during the above-mentioned patch generation process. In various example embodiments, the metric ($\hat{\xi}$) may also be used to determine the level of uniformity of a set of 3D points of a 3D patch of a point cloud, such as to determine or classify whether the 3D patch is transformable or untransformable for selecting

the particular attribute image generation process to generate the 2D attribute image of the 3D patch.

Alignment of 2D Points and Pixel Grids

5 [0073] To generate a 2D attribute image for a transformed 2D patch, various example embodiments formulate the alignment of 2D points of the 2D patch and image pixel slots of an image pixel grid as a bipartite matching problem and then determine an optimized placement solution by minimising the error of pairwise Euclidean distances between points (Stage II, e.g., corresponding to the second 2D space filling stage of the second attribute image generation process as described hereinbefore according to various embodiments). For example, similar to
10 the function used in nonlinear mapping described in Lee *et al.*, *Nonlinear dimensionality reduction*, Springer Science & Business Media, 2007, the cost for a placement according to various example embodiments may be defined as:

$$c = \frac{1}{k} \sum_{i=1, i < j}^N \frac{(d_p(i, j) - d_g(i, j))^2}{d_p(i, j)}$$

15 (Equation 5)

where $d_p(i, j)$ and $d_g(i, j)$ represent the Euclidean distances between the i -th and j -th points in the transformed 2D patch and image pixel grids, respectively, and k is a normalizing factor and is defined as $k = \sum_{i=1, i < j}^N d_p(i, j)$. As an illustrative example, the bipartite matching process and alignment result of a transformed 2D patch are shown in FIGs. 7A to 7C. In particular,
20 FIGs. 7A to 7C depict bipartite matching and attribute image comparison, whereby FIG. 7A depicts a transformed 2D patch 702 (the point farthest from the center is used to generate virtual points (corresponding to the extra 2D points as described hereinbefore according to various embodiments)), FIG. 7B depicts the bipartite matching process, whereby points are assigned to pixel slots of an image pixel grid 704 by minimizing the error of pairwise Euclidean distances
25 between points, and FIG. 7C depicts result of 2D points and pixel slots alignment (virtual points are represented as hollow circles). Accordingly, in the bipartite matching process, each 2D point of the transformed 2D patch 702 is mapped to a respective pixel slot of the image pixel grid 704 based on minimizing error between pairwise distances of the set of 2D points at the transformed 2D patch 702 and corresponding pairwise distances of the set of 2D points mapped
30 to the image pixel grid 704.

[0074] Various example embodiments note that the number of points of a patch may not be always exactly the same as the number of pixels of an image canvas (an image pixel grid). To

address this issue, various example embodiments add several virtual points 708 to the set of 2D points mapped to the image pixel grid 704 as shown in FIG. 7C to generate a compact image for compression. In various example embodiments, the farthest point 714 from the center 712 of transformed 2D patch 702 is determined and the geometry and attribute information of this farthest point 714 is used as that of virtual points 708, as shown in FIGs. 7A to 7C. Various example embodiments note that different sizes of image canvases can generate very different placements and may further have an impact on the compression performance. To address this issue, various example embodiments seek to determine an optimal placement from the solution space. In view of searching efficiency, various example embodiments create a solution space as $\ell = \{(W_i, H_i) | H_i = 2^i h, H_i \leq W_i, i \in \mathbb{Z}^{0+}\}$, where h is set to 16, which is in line with the height of the macroblock described later below. By designating the height of the image canvas, its width and the number of virtual points can be calculated. Accordingly, various example embodiments define an energy function to evaluate the correlation among adjacent pixels of different placements as follows:

$$\mathcal{E} = \sum_{x=1}^W \sum_{y=1}^H \sum_{(x', y') \in \Omega_{x,y}} |Y_{x,y} - Y_{x',y'}| \quad (\text{Equation 6})$$

where $Y_{x,y}$ is the luminance component of the pixel at (x, y) , and $\Omega_{x,y} = \{(x \pm 1, y), (x, y \pm 1)\}$ is the set of the locations of its 4-connected neighbour pixels. In various example embodiments, the placement with smallest energy is used to encode the patch.

SFC-BASED METHOD FOR POINT CLOUD ATTRIBUTE IMAGE GENERATION

[0075] Various example embodiments provide a SFC-based attribute image generation method (e.g., corresponding to the first attribute image generation process as described hereinbefore according to various embodiments) for untransformable patches. The SFC-based method maps 3D points of a 3D patch to 1D continuous array in Stage I (e.g., corresponding to the point linearization stage of the first attribute image generation process as described hereinbefore according to various embodiments) and arranges each point in the 1D continuous array into pixel grids of a 2D attribute image in Stage II (e.g., corresponding to the first 2D space filling stage of the first attribute image generation process as described hereinbefore according to various embodiments). In various example embodiments, two methods for these two stages are provided, i.e., a BSP-based universal traversal stage and a hybrid space-filling pattern stage.

BSP-based Universal Traversal

[0076] Various example embodiments note that although traditional depth-first traversal schemes based on tree structures (e.g., octree, k-d tree) may well retain a considerable portion of spatial correlation, they may also introduce many inevitable big jumps caused by the intrinsic structure of a point cloud and the traversal patterns. Therefore, various example embodiments improve point cloud attribute coding efficiency by reducing the number of big jumps when converting 3D points of a 3D patch into a 1D ordered sequence. In this regard, various example embodiments provide a BSP-based universal traversal method based on a ball-tree space partition data structure. Before describing the technical details of the BSP-based universal traversal method, the main idea of the construction of a ball-tree is first described below. Given a set of points $\mathcal{S}$, the ball-tree construction method or procedure can be summarized as follows:

- (1) Calculate the centroid O of the point set;
- (2) Find the farthest point from O and set it as the left pivot point P_{pivot}^L ;
- 15 (3) Find the farthest point from P_{pivot}^L and set it as the right pivot point P_{pivot}^R ;
- (4) Partition the points in $\mathcal{S}$ into two subsets by assigning each of them to its nearest pivot point; and
- (5) Perform the above steps on each obtained subset until there is only a single point in it.

20 [0077] Various example embodiments note that if the leaf nodes of the constructed ball tree is traversed in depth-first order, it will suffer from the big jump problem such as in octree-based methods. This is because each iteration will recalculate the centroid as well as the left and right pivot points, which cannot ensure that the right pivot point P_{pivot}^R of the first subset is close to the left pivot point P_{pivot}^L of the second sub-set. To address this issue, the BSP-based universal traversal method according to various example embodiments adopts a heuristic strategy to preserve the inherent coherence of points as described below.

[0078] According to the above-mentioned ball-tree construction method, two sub-sets of points are obtained after the first iteration (i.e., the above-mentioned step 1 to step 4). Let $\mathbb{S}^{(1)} = \{S_1^L, S_2^R\}$ and $P_{pivot}^{L,*}$ be the obtained point set (e.g., corresponding to the set of 3D points of the input 3D patch as described hereinbefore according to various embodiments) and the left pivot point (e.g., corresponding to the first pivot point of the set of 3D points of the input 3D patch as described hereinbefore according to various embodiments) used for partition in this iteration,

respectively. In the second iteration, BSP is performed on each subset of $\mathbb{S}^{(1)}$ (e.g., corresponding to the first and second point subsets of the set of 3D points of the input 3D patch as described hereinbefore according to various embodiments) sequentially. Firstly, for S_1^L , we set $P_{pivot}^{L,*}$ pivot as its left pivot point (e.g., corresponding to the second pivot point of the first point subset as described hereinbefore according to various embodiments) and use it to find the right pivot point $P_{pivot}^{R,1}$ (e.g., corresponding to the first pivot point of the first point subset as described hereinbefore according to various embodiments) of S_1^L . Then, S_1^L is divided into two subsets (e.g., corresponding to the new first and second point subsets as described hereinbefore according to various embodiments) with $S_{1,1}^L$ and $S_{1,2}^R$ as the two pivot points. It should be noted that $P_{pivot}^{L,*}$ is fixed as the left pivot point of the first subset in $\mathbb{S}$, which is different from traditional ball construction process. To reduce the number of big jumps, for S_2^R , the nearest neighbour of $P_{pivot}^{R,1}$ from it is determined as its left pivot point $P_{pivot}^{L,2}$ (e.g., corresponding to the first pivot point of the second point subset as described hereinbefore according to various embodiments) and further use $P_{pivot}^{L,2}$ to find its right pivot point $P_{pivot}^{R,2}$ (e.g., corresponding to the second pivot point of the second point subset as described hereinbefore according to various embodiments). Similarly, S_2^R is also partitioned into two subsets $S_{2,1}^L$ and $S_{2,2}^R$ (e.g., corresponding to the new first and second point subsets as described hereinbefore according to various embodiments). Replacing S_1^L and S_2^R with the obtained new subsets will harvest a new or updated set $\mathbb{S}^{(2)} = \{S_{1,1}^L, S_{1,2}^R, S_{2,1}^L, S_{2,2}^R\}$. The above operations are then repeated until there is only one point (3D point) in each subset of $\mathbb{S}$.

[0079] An illustrative example will now be described to further illustrate the BSP-based universal traversal method according to various example embodiments for better understanding. For simplicity but without loss of generality, a 2D point set (instead of a 3D set) is used in the illustrative example as shown in FIGs. 8A to 8F. Initially, the point set is divided into two subsets with p_1 and p_9 as the left and right pivot points (e.g., corresponding to the first and second pivot points of the set of 3D points of the input 3D patch, respectively, as described hereinbefore according to various embodiments), respectively, and $\mathbb{S}^{(1)}$ will be $\{\{p_1, p_2, p_3, p_4\}, \{p_5, p_6, p_7, p_8, p_9\}\}$ as shown in FIG. 8C. In second iteration, p_1 and p_4 will be the pivot points (e.g., corresponding to the second and first pivot points of the first point subset, respectively, as described hereinbefore according to various embodiments) of the first subset. Then, p_5 will be the left pivot point (e.g., corresponding to the first pivot point of the second point subset as described hereinbefore according to various embodiments) of the $\{p_5, p_6, p_7, p_8, p_9\}$

since it is the nearest point to p_4 . Then, $\mathbb{S}^{(2)}$ will be $\{\{p_1, p_2\}, \{p_3, p_4\}, \{p_5, p_7, p_6\}, \{p_8, p_9\}\}$ after performing BSP, as shown in FIG. 8D. Accordingly, by repeating the above operations iteratively for each point subset in the 2D point set until all point subsets therein have only one point in each, the final traversal order obtained in this illustrative example is $\{p_1, p_2, p_3, p_4, p_5, p_7, p_6, p_8, p_9\}$, as shown in FIG. 8F. In particular, FIGs. 8A to 8F depict the BSP-based universal traversal method according to various example embodiments, whereby FIG. 8A depicts an example 2D point set, and FIGs. 8B to 8F depict the point linearization procedure using the BSP-based universal traversal method. In various example embodiments, the leftmost pivot point $P_{pivot}^{L,*} = p_1$ is fixed during iteration.

- 10 **[0080]** Various example embodiments also configured the point linearization of a point cloud as a travelling salesman problem (TSP), that is, given the start point and end point of a point set, the task is to find the shortest path that visits each point exactly once. Using the points in FIG. 8A as an illustrative example, if p_1 and p_9 are fixed as the start point and end point respectively, the traversal order shown in FIG. 8F will be a possible solution for this task.
- 15 However, due to the complexity of calculating the optimal path, various example embodiments note that it may not be feasible to apply the TSP on the whole point cloud. Therefore, according to various example embodiments, the BSP-based traversal method improves the linearization efficiency by applying TSP on supervoxels of the point cloud. For example, the start and end points can be found in the same or similar manner as determining pivot points described above.
- 20 The point sequences of each supervoxel may then be assembled together according to the order of seed points obtained with the above-mentioned BSP-based traversal method. For illustration purpose, FIG. 9 shows several examples of the traversal order visualization of different 3D point linearization methods, namely, the conventional octree-based depth-first traversal, the conventional 3D Hilbert SFC based traversal and the present BSP-based traversal method.
- 25 Compared with the conventional octree-based depth-first traversal and 3D Hilbert SFC based traversal methods, it can be observed that the present method introduces significantly fewer big jumps during the linearization process. The grayscale bar on the right indicates the point traversal order of a point cloud.

30 *Hybrid 2D Space-filling Pattern*

[0081] To obtain a compact image for attribute compression, the 1D point sequence obtained in the above-mentioned BSP-based traversal method is mapped onto a 2D image pixel grid (a 2D grid structure layout) using a space-filling pattern. A simple way to achieve this

mapping is to sequentially assign each point in the 1D stream to its corresponding pixel slot of a pre-defined image canvas (a 2D image pixel grid) according to one of the patterns of common SFCs, such as zigzag, Z-order, Peano, and so on. Various example embodiments note that both the canvas size and filling pattern are important factors for the development of high-efficiency point cloud attribute codecs, since mapping schemes without elaborate design may destroy the point coherence. Various example embodiments note that these issues are inadequately addressed by previous studies.

[0082] Accordingly, various example embodiments provide a hybrid 2D SFC method (e.g., corresponding to the first 2D space filling stage of the first attribute image generation process as described hereinbefore according to various embodiments) to address the above issues associated with conventional space filling methods. FIG. 10 depicts a schematic flow diagram of a hybrid 2D space filling pattern method according to various example embodiments of the present invention. As shown in FIG. 10, the image canvas 1004 comprises a series of macroblocks 1008 with a size of 16×16 pixels (or pixel slots). Each macroblock 1004 is further divided into 16 sub-macroblocks 1012 that are 4×4 pixels (or pixel slots) in size. Such a structure or configuration is advantageously compatible with the design of classic image codecs such as JPEG and WebP, which generally adopt a block-based coding technique.

[0083] With regards to the selection of the size of macroblock 1004, a number of other possible candidates can also be considered, such as but not limited to, 8×8 , 32×32 , 64×64 pixels and so on. Accordingly, it will be appreciated by a person skilled in the art that the present invention is not limited to the specific size of the macroblocks and sub-macroblocks as described above. Nevertheless, various example embodiments found that both smaller and larger size usually cannot fully exert the block intra prediction merits of advanced image codecs. Therefore, a modest size is adopted in the present framework, which was discovered to be more effective in taking the advantages of image compression techniques. As for the type of sub-macroblocks, various example embodiments select 4×4 pixels since it is also used as a basic block unit in many block-based codecs and can well preserve the attribute coherence in sub-macroblocks.

[0084] To map the ordered 1D point sequence 1016 into the 2D grid structure layout described above, various example embodiments present a hybrid pattern that can advantageously retain the attribute correlation in sub-macroblocks 1012, as well as that between adjacent macroblocks 1004. For each sub-macroblock 1012, a horizontal snake curve is employed as its filling pattern, which was found to above big jumps and is more likely to

preserve the color coherence. According to the horizontal snake curve filling pattern, as shown in FIG. 10, pixel slots are filled from a first row (e.g., the top row) of pixel slots until a last row (e.g., the bottom row) of pixel slots of the sub-macroblock 1012, row by row consecutively, whereby at each row of pixel slots, pixel slots are filled from a first end to a second end of the row, and the filling directions of immediately adjacent rows are opposite. For example, let N_s be the size of sub-macroblock 1012 ($N_s = 4$ according to various example embodiments), the location in the sub-macroblock 1012 (i.e., the x -th column, y -th row of the i -th sub-macroblock) of the t -th point in the 1D sequence 1016 can be determined as:

$$i = \lfloor t / (T \cdot N_s) \rfloor$$

10 (Equation 7)

$$y = \lfloor (t \bmod (N_s \cdot N_s)) / N_s \rfloor$$

(Equation 8)

$$x = (-1)^{y \bmod 2} \cdot (t \bmod N_s) + (N_s - 1) \cdot (y \bmod 2)$$

(Equation 9)

15 **[0085]** Note that all the above indexes are numbered starting from 0. With the obtained sub-macroblocks 1012, another mapping is performed by using the Hilbert SFC to better utilize the intra prediction mode of image codecs (e.g., see FIG. 10). Similar to horizontal snake curve, the Hilbert curve also does not have big jumps but was found according to various example embodiments to be more suitable for exploiting intra prediction in both horizontal and vertical directions. Accordingly, various example embodiments use the Hilbert function, such as
20 introduced in Bader *et al.*, *Space-filling curves: an introduction with applications in scientific computing*, Springer Science & Business Media, 2012, vol. 9, to map the sub-macroblocks 1012 to macroblocks 1004. FIG. 10 illustrates a segment of the obtained 2D attribute image 1004 of the 3D patch (one 3D patch) using the hybrid 2D SFC method according to various example
25 embodiments of the present invention.

[0086] Accordingly, with the IsoMap-based and hybrid 2D SFC-based attribute image generation processes, a 2D attribute image for each obtained 3D patch can be generated. For example, FIG. 11 illustrates two example 3D patches 1104a, 1104b of a point cloud 1108, namely, a first patch 1104a (denoted as 1) and a second patch 1104b (denoted as 2), respectively,
30 and the corresponding patch attribute images generated with IsoMap-based and SFC-based attribute image generation processes, respectively. For illustration purposes, the first patch 1104a was processed using both the IsoMap-based attribute image generation process and the hybrid 2D SFC-based attribute image process to generate a IsoMap-based attribute image and

a hybrid 2D SFC-based attribute image, respectively. Similarly, the second patch 1104b was processed using both the IsoMap-based attribute image generation process and the hybrid 2D SFC-based attribute image process to generate a IsoMap-based attribute image and a hybrid 2D SFC-based attribute image, respectively. It can be observed that the IsoMap-based image generation process is better than the hybrid 2D SFC-based attribute image process for the first patch while for the second patch is the opposite, for reasons as described hereinbefore according to various example embodiments of the present invention.

[0087] Thereafter, all attribute images of 3D patches of a point cloud are assembled together and a 2D attribute image of the whole point cloud is harvested or generated for compression.

As described hereinbefore, the height of each attribute image of 3D patch obtained with IsoMap-based attribute image generation process may vary greatly. Accordingly, to assemble them together, various example embodiments may slice those attribute images into several segments whose height is equal to that of macroblocks. With this unified height setting, all the segments can be horizontally or sequentially stacked together. In various example embodiments, to maintain the correlation among adjacent 3D patches, the geometrical centers of all 3D patches are traversed using the present BSP-based traversal method and follow the traversal order to stack the segments sequentially. Thereafter, the 2D attribute image of the point cloud may be compressed using well-developed image codecs.

EXPERIMENTAL RESULTS AND ANALYSIS

Experiment Setup

[0088] To evaluate the effectiveness of the present method of point cloud attribute compression, a series of experiments were conducted by comparing the present method with conventional methods of point cloud compression on a public point cloud dataset, which is widely adopted by many existing studies. As shown in FIGs. 12A to 12Q, seventeen 3D point cloud models in total were collected. In particular, FIGs. 12A to 12Q depict example point clouds from a non-uniform 3D point cloud dataset, whereby FIGs. 12A to 12L show a mixture of omni-directional and semi-directional point cloud models, and FIGs. 12M to 12Q show a uniform omni-directional 3D point cloud dataset. More specifically, FIGs. 12A to 12E are five point clouds with noise and uneven point distributions in 3D space, which were selected from MICROSOFT voxelized upper bodies dataset; FIGs. 12F to 12I are seven relatively sparse and irregular point clouds that were chosen from the common test conditions for point cloud

compression. To further evaluate the efficiency of the present method, five dense and uniform point clouds were also collected from the 8i full-body dataset, as shown in FIGs. 12M to 12Q. The number of points of these point clouds ranges from 200K to 5 million. The attribute images were compressed using classic image codecs including JPEG, WebP and versatile video coding (VVC) at different quality scales and obtain their rate-distortion (RD) curves. Bjontegaard metric (e.g., as described in Bjontegaard, "Calculation of average psnr differences between rd-curves," *VCEG-M33*, 2001) was also employed to compare the performance in terms of the BD bit rate (BD-BR) and BD-PSNR.

10 *Efficiency Evaluation of BSP-based Universal Traversal*

[0089] 1) *Comparison in terms of bit rate saving and PSNR gain*: To verify the effectiveness of the present BSP-based universal traversal method, it was compared with Octree depth-first traversal (Oct) (as described in Mekuria *et al.*, "Design, implementation, and evaluation of a point cloud codec for tele-immersive video," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 27, no. 4, pp. 828–842, 2016, which is herein referred to as the Mekuria reference) and 3D Hilbert traversal (3DH) (as described in Bader, *Space-filling curves: an introduction with applications in scientific computing*, Springer Science & Business Media, 2012, vol. 9, which is herein referred to as the Bader reference) schemes. For 3DH, the bounding box of a point cloud is divided into $n \times n \times n$ voxels, in which there is no more than one point. Then the voxels are numbered with the 3DH curve construction algorithm (see the Bader reference). The 1D sequence can be obtained by traversing these voxels sequentially. To obtain the attribute images for compression, the image canvas setting introduced in the Mekuria reference was used. In the Octree depth-first traversal method disclosed in the Mekuria reference, the macroblock size is set to 8×8 pixels and horizontal snake space-filling pattern is adopted to generate the attribute images. In the experiments, JPEG and WebP codecs were employed to compress these images for fair comparison (because both are well-recognized existing codecs). As shown in Table I in FIG. 13, the present method using WebP as the codec saves more bit rate than Octree-based method (Oct_WebP) by 11.29% and 3DH-based method (3DH_WebP) by 6.00%, on average. In particular, Table I shows a comparison of efficiency (the saving of BR and the equivalent PSNR improvement in dB) of the present BSP-based universal traversal method against the above-mentioned two conventional traversal methods.

[0090] For example, the significance of the present BSP-based universal traversal method can also be observed by using JPEG codec, especially for point clouds with strong inherent color coherence like *ricardo* and *Facade*. For those with very weak inherent color coherence like *Arco*, the present method achieves almost comparable PSNR gain compared with 3DH. However, the coding efficiency of such point clouds can still be improved by using the present hybrid 2D space-filling pattern method, which will be demonstrated below.

[0091] 2) *Comparison in terms of autocorrelation*: As autocorrelation is generally closely related to the compression performance, it has been widely used for the coherence evaluation of 1D pixel sequence in the sector of image/video compression. In the experiments, this metric was used to further investigate the effectiveness of different traversal algorithms with respect to preserving the spatial correlation of points during the 3D to 1D mapping process. FIGs. 14A to 14C illustrate several examples of the autocorrelation at different lags. As can be seen from FIGs. 14A to 14C, compared with 3D Hilbert and Octree based traversal methods, the present method achieves a remarkable improvement of point attribute correlation for point clouds *phil* (shown in FIG. 12D), *Facade* (shown in FIG. 12I) and *Frog* (shown in FIG. 12K). Besides, by comparing FIGs. 14A to 14C with Table I in FIG. 13, it can be observed that autocorrelation gain is positively correlated with bit rate saving. These results indicate that the present method can better preserve the point coherence than conventional or peer methods.

Efficiency Evaluation of Hybrid SFC

[0092] To verify the effectiveness of the hybrid 2D space-filling pattern, the 1D point sequence obtained by using the present universal traversal method was compared with the horizontal snake (HS) space-filling curve as described in the Mekuria reference. Beyond the attribute image obtained in HS space-filling curve, another image was generated with 2D Hilbert curve (Hil) for comparison. Likewise, both JPEG and WebP were employed for the comparison. The experimental results, which are illustrated in Table II in FIG. 15, demonstrate the effectiveness of the present hybrid 2D space-filling pattern. In particular, Table II shows a comparison of the efficiency (the saving of BR and the equivalent PSNR improvement in dB) of different space filling patterns. Specifically, with JPEG codec, it respectively reduces the bit rate than HS (HS_JPEG) and Hil (Hil_JPEG) by 9.07% and 15.08%, on average. While for WebP codec, its superiority can also be verified in exploiting point attribute correlation than HS (HS_WebP) and Hil (Hil_WebP), which saves the bit rate by 26.30% and 27.94%, respectively.

Overall Comparison with State-of-the-art Methods

[0093] In this part, the relevant state-of-the-art compression methods that are specially designed for point cloud data were used to further investigate the efficiency of the present method. The peer methods used for comparison are geometry-guided sparse representation codec (GSR) (as described in Gu *et al.*, “3d point cloud attribute compression using geometry-guided sparse representation,” *IEEE Transactions on Image Processing*, vol. 29, pp. 796–808, 2019, which is herein referred to as the Gu reference), region-adaptive hierarchical transform codec (RAHT) (as described in de Queiroz *et al.*, “Compression of 3d point clouds using a region-adaptive hierarchical transform,” *IEEE Transactions on Image Processing*, vol. 25, no. 8, pp. 3947–3956, 2016, which is herein referred to as the de Queiroz reference) and MPEG G-PCC level-of-detail (LoD) encoder with lift transform (LoD-LT) (as described in Schwarz *et al.*, “Common test conditions for point cloud compression,” *ISO/IEC JTC1/SC29/WG11 MPEG, output document N*, vol. 17995, 2019, which is herein referred to as the Schwarz reference).

Various example embodiments note that MPEG V-PCC is more suitable for uniform point cloud compression, which has been demonstrated by existing studies, and thus is not selected for comparison in the experiments conducted. FIGs. 16A to 16Q and Table III in FIG. 17 show the rate-PSNR curves and BD-PSNR (in dB) improvement of state-of-art methods against baseline method RAHT (as described in the de Queiroz reference), respectively. In particular, FIGs. 16A to 16Q depict a comparison with state-of-the-art point cloud attribute compression methods. Table III in FIG. 17 depicts an efficiency comparison of state-of-the-art methods with the results of RAHT encoder as the baseline. The results of the present method are obtained by using the above-mentioned hybrid 2D space-filling patterns and VVC codec. Overall, compared to the baseline method, the present method improves the BD-PSNR by an average 5.43 dB, which is far better than state-of-art methods LoD-LT (0.60 dB) and GSR (1.35 dB). According to the Gu reference, GSR achieves better performance on point clouds with less color variance and relatively simple geometrical structure, for example, those from the Microsoft voxelized upper bodies dataset such as *ricardo9*, *David* and *sarah9*. However, various example embodiments note that it is not able to show its advantage compared with LoD-LT and RAHT on point clouds such as *Shiva*, *House* and *Frog*. As illustrated in FIGs. 16A to 16Q, the present method outperforms RAHT and MPEG LoD-LT on all of the test point clouds. Notably, it achieves a significant performance improvement than state-of-the-art methods on point clouds such as *David*, *Frog* and *loot*. More specifically, compared with baseline method, it improves

BD-PSNR by 5.99 dB, 6.65 dB and 6.43 dB (the equivalent BD-rate gain by 82.11%, 84.42% and 84.59%), while the state-of-the-art method LoD-LT just achieves BD-PSNR gain by 0.37 dB, 0.30 dB and 0.44 dB (the equivalent BD-rate gain by 8.88%, 9.32% and 9.95%), respectively. It is worth pointing out that *Arco* (as shown in FIG. 12L) is one of the most
5 challenging non-uniform point clouds for effective attribute compression since it simultaneously exhibits several characteristics including geometrical sparsity, noise, irregular point distribution and high color variance, which is very difficult to achieve satisfied compression performance using existing codecs. Even the performance of advanced point cloud codec LoD-LT is inferior to the baseline method RAHT. With the present method, the BD-
10 PSNR was improved by 4.31 dB (the equivalent BD-rate gain by 37.77%) than RAHT, which verifies the superiority of the present method in handling non-uniform point clouds.

[0094] Accordingly, various example embodiments have introduced an image-based method to compress point cloud attribute. In particular, largely scattered, unordered 3D point data are transformed into synthetic images most suitable to be compressed by the available
15 image coding strategies that have been already developed during the past decades. This is in line with the major current effort of the research community in this area, to make full use of the existing infrastructure to solve the new problems. For non-uniform point clouds, the sparse or irregular geometrical structure poses great challenges since it is usually very difficult to have satisfactory coding performance using image-based compression approaches under the single-
20 stage 3D to 2D mapping paradigm. In this regard, various example embodiments provide an adaptive two-stage dimensionality transformation strategy to map the attribute (attribute information) of 3D points of a point cloud into 2D grid structure layout to obtain a compact attribute image for compression. Accordingly, two 3D point cloud attribute image generation methods described hereinbefore according to various example embodiments are designed to
25 better exploit the spatial correlation among adjacent points. The experimental results demonstrate the effectiveness of the present method in point cloud attribute compression and its superiority over relevant state-of-the-art codecs in handling non-uniform point clouds.

[0095] With the wide availability of 3D scanning equipment and ever-growing 3D applications, point clouds are being used in many emerging fields in smart cities, digital transformation and Industry 4.0, such as VR/AR, BIM, autonomous driving, robot navigation,
30 e-commerce, aids for senior citizens, intelligent manufacturing, urban planning, biomedical modeling, architecture, engineering and construction (AEC), cultural heritage preservation and so on. For example and without limitation, the present point cloud attribute compression method

can be used to facilitate efficient and economic storage, transmission and processing of 3D point cloud data in these areas, making full use of the existing, well established image and video coding standards and infrastructure. Besides, the hybrid space-filling scheme can also work with other efficient 3D point linearization methods to develop real-time 3D applications such as tele-immersive communication systems.

[0096] While embodiments of the invention have been particularly shown and described with reference to specific embodiments, it should be understood by those skilled in the art that various changes in form and detail may be made therein without departing from the scope of the invention as defined by the appended claims. The scope of the invention is thus indicated by the appended claims and all changes which come within the meaning and range of equivalency of the claims are therefore intended to be embraced.