DESCRIPTION

THREE-DIMENSIONAL DATA ENCODING METHOD, THREE-DIMENSIONAL DATA DECODING METHOD, THREE-DIMENSIONAL DATA ENCODING DEVICE, AND THREE-DIMENSIONAL DATA DECODING DEVICE

TECHNICAL FIELD

[0001]

The present disclosure relates to a three-dimensional data encoding method, a three-dimensional data decoding method, a three-dimensional data encoding device, and a three-dimensional data decoding device.

BACKGROUND ART

[0002]

Devices or services utilizing three-dimensional data are expected to find their widespread use in a wide range of fields, such as computer vision that enables autonomous operations of cars or robots, map information, monitoring, infrastructure inspection, and video distribution. Three-dimensional data is obtained through various means including a distance sensor such as a rangefinder, as well as a stereo camera and a combination of a plurality of monocular cameras.

[0003]

Methods of representing three-dimensional data include a method known as a point cloud scheme that represents the shape of a three-dimensional structure by a point group in a three-dimensional space. In the point cloud scheme, the positions and colors of a point group are stored. While point cloud

is expected to be a mainstream method of representing three-dimensional data, a massive amount of data of a point group necessitates compression of the amount of three-dimensional data by encoding for accumulation and transmission, as in the case of a two-dimensional moving picture (examples include MPEG-4 AVC and HEVC standardized by MPEG).

[0004]

Meanwhile, point cloud compression is partially supported by, for example, an open-source library (Point Cloud Library) for point cloud-related processing.

[0005]

Furthermore, a technique for searching for and displaying a facility located in the surroundings of the vehicle is known (for example, see Patent Literature (PTL) 1).

Citation List

Patent Literature

[0006]

PTL 1: International Publication WO 2014/020663

SUMMARY OF THE INVENTION

TECHNICAL PROBLEM

[0007]

There has been a demand for reducing an amount of processing in encoding and decoding of three-dimensional data.

[0008]

Embodiments of the present disclosure provide a three-dimensional data encoding method, a three-dimensional data decoding method, a three-dimensional data encoding device, or a three-dimensional data decoding device that is capable of reducing the amount of processing.

SOLUTIONS TO PROBLEM

[0009]

According to a first aspect of the present disclosure, there is provided a three-dimensional data encoding method, including: generating an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2; encoding a first parameter; determining a value indicated by the first parameter; when the first parameter is determined as indicating a first value, (i) selecting a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic encoding on information of the current node using the selected contexts; and when the first parameter is determined as indicating a second value different from the

first value, (i) selecting a group from M groups according to the occupancy states of neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic encoding on information of the current node using the selected contexts.

[0009a]

According to a second aspect of the present disclosure, there is provided a three-dimensional data encoding device that encodes three-dimensional points each including attribute information, the three-dimensional data encoding device including: a processor; and memory, wherein using the memory, the processor: generates an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2; encodes a first parameter; determines a value indicated by the first parameter; when the first parameter is determined as indicating a first value, (i) selects a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic encoding on information of the current node using the selected contexts; and when the first parameter is determined as indicating a second value different from the first value, (i) selects a group from M groups according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic encoding on information of the current node using the selected contexts.

[0010]

According to a third aspect of the present disclosure, there is provided a three-dimensional data decoding method, including: obtaining an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2; obtaining a first parameter; determining a value indicated by the first parameter; when the first parameter is determined as indicating a first value, (i) selecting a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic decoding on information of the current node using the selected contexts; when the first parameter is determined as indicating a second value different from the first value, (i) selecting a group from M groups according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic decoding on information of the current node using the selected contexts.

[0010a]

According to a fourth aspect of the present disclosure, there is provided a three-dimensional data decoding device that decodes three-dimensional points each including attribute information, the three-dimensional data decoding device including: a processor; and memory, wherein using the memory, the processor: obtains an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2; obtains a first parameter; determines a value indicated by the first parameter; when the first parameter is determined as indicating a first value, (i) selects a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic decoding on information of the current node using the selected contexts; when the first parameter is determined as indicating a second value different from the first value, (i) selects a group from M groups according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic decoding on information of the current node using the selected contexts.

ADVANTAGEOUS EFFECT OF INVENTION

[0011]

Embodiments of the present disclosure provide a three-dimensional data encoding method, a three-dimensional data decoding method, a three-dimensional data encoding device, or a three-dimensional data decoding device that is capable of reducing an amount of processing.

BRIEF DESCRIPTION OF DRAWINGS

[0012]

FIG. 1 is a diagram showing the structure of encoded three-dimensional data according to Embodiment 1.

FIG. 2 is a diagram showing an example of prediction structures among SPCs that belong to the lowermost layer in a GOS according to Embodiment 1.

FIG. 3 is a diagram showing an example of prediction structures among layers according to Embodiment 1.

FIG. 4 is a diagram showing an example order of encoding GOSs according to Embodiment 1.

FIG. 5 is a diagram showing an example order of encoding GOSs according to Embodiment 1.

FIG. 6 is a block diagram of a three-dimensional data encoding device according to Embodiment 1.

FIG. 7 is a flowchart of encoding processes according to Embodiment 1.

FIG. 8 is a block diagram of a three-dimensional data decoding device according to Embodiment 1.

FIG. 9 is a flowchart of decoding processes according to Embodiment 1.

FIG. 10 is a diagram showing an example of meta information according to Embodiment 1.

FIG. 11 is a diagram showing an example structure of a SWLD according to Embodiment 2.

FIG. 12 is a diagram showing example operations performed by a server and a client according to Embodiment 2.

FIG. 13 is a diagram showing example operations performed by the server and a client according to Embodiment 2.

FIG. 14 is a diagram showing example operations performed by the server and the clients according to Embodiment 2.

FIG. 15 is a diagram showing example operations performed by the server and the clients according to Embodiment 2.

FIG. 16 is a block diagram of a three-dimensional data encoding device according to Embodiment 2.

FIG. 17 is a flowchart of encoding processes according to Embodiment 2.

FIG. 18 is a block diagram of a three-dimensional data decoding device according to Embodiment 2.

FIG. 19 is a flowchart of decoding processes according to Embodiment 2.

FIG. 20 is a diagram showing an example structure of a WLD according to Embodiment 2.

FIG. 21 is a diagram showing an example octree structure of the WLD according to Embodiment 2.

FIG. 22 is a diagram showing an example structure of a SWLD according to Embodiment 2.

FIG. 23 is a diagram showing an example octree structure of the SWLD according to Embodiment 2.

FIG. 24 is a block diagram of a three-dimensional data creation device according to Embodiment 3.

FIG. 25 is a block diagram of a three-dimensional data transmission device according to Embodiment 3.

FIG. 26 is a block diagram of a three-dimensional information processing device according to Embodiment 4.

FIG. 27 is a block diagram of a three-dimensional data creation device according to Embodiment 5.

FIG. 28 is a diagram showing a structure of a system according to Embodiment 6.

FIG. 29 is a block diagram of a client device according to Embodiment 6.

FIG. 30 is a block diagram of a server according to Embodiment 6.

FIG. 31 is a flowchart of a three-dimensional data creation process performed by the client device according to Embodiment 6.

FIG. 32 is a flowchart of a sensor information transmission process performed by the client device according to Embodiment 6.

FIG. 33 is a flowchart of a three-dimensional data creation process performed by the server according to Embodiment 6.

FIG. 34 is a flowchart of a three-dimensional map transmission process performed by the server according to Embodiment 6.

FIG. 35 is a diagram showing a structure of a variation of the system according to Embodiment 6.

FIG. 36 is a diagram showing a structure of the server and client devices according to Embodiment 6.

FIG. 37 is a block diagram of a three-dimensional data encoding device according to Embodiment 7.

FIG. 38 is a diagram showing an example of a prediction residual

according to Embodiment 7.

FIG. 39 is a diagram showing an example of a volume according to Embodiment 7.

FIG. 40 is a diagram showing an example of an octree representation of the volume according to Embodiment 7.

FIG. 41 is a diagram showing an example of bit sequences of the volume according to Embodiment 7.

FIG. 42 is a diagram showing an example of an octree representation of a volume according to Embodiment 7.

FIG. 43 is a diagram showing an example of the volume according to Embodiment 7.

FIG. 44 is a diagram for describing an intra prediction process according to Embodiment 7.

FIG. 45 is a diagram for describing a rotation and translation process according to Embodiment 7.

FIG. 46 is a diagram showing an example syntax of an RT flag and RT information according to Embodiment 7.

FIG. 47 is a diagram for describing an inter prediction process according to Embodiment 7.

FIG. 48 is a block diagram of a three-dimensional data decoding device according to Embodiment 7.

FIG. 49 is a flowchart of a three-dimensional data encoding process performed by the three-dimensional data encoding device according to Embodiment 7.

FIG. 50 is a flowchart of a three-dimensional data decoding process performed by the three-dimensional data decoding device according to Embodiment 7.

FIG. 51 is a diagram illustrating an example of a tree structure according to Embodiment 8.

FIG. 52 is a diagram illustrating an example of occupancy codes according to Embodiment 8.

FIG. 53 is a diagram schematically illustrating an operation performed by a three-dimensional data encoding device according to Embodiment 8.

FIG. 54 is a diagram illustrating an example of geometry information according to Embodiment 8.

FIG. 55 is a diagram illustrating an example of selecting a coding table using geometry information according to Embodiment 8.

FIG. 56 is a diagram illustrating an example of selecting a coding table using structure information according to Embodiment 8.

FIG. 57 is a diagram illustrating an example of selecting a coding table using attribute information according to Embodiment 8.

FIG. 58 is a diagram illustrating an example of selecting a coding table using attribute information according to Embodiment 8.

FIG. 59 is a diagram illustrating an example of a structure of a bitstream according to Embodiment 8.

FIG. 60 is a diagram illustrating an example of a coding table according to Embodiment 8.

FIG. 61 is a diagram illustrating an example of a coding table according to Embodiment 8.

FIG. 62 is a diagram illustrating an example of a structure of a bitstream according to Embodiment 8.

FIG. 63 is a diagram illustrating an example of a coding table according to Embodiment 8.

FIG. 64 is a diagram illustrating an example of a coding table according

to Embodiment 8.

FIG. 65 is a diagram illustrating an example of bit numbers of an occupancy code according to Embodiment 8.

FIG. 66 is a flowchart of an encoding process using geometry information according to Embodiment 8.

FIG. 67 is a flowchart of a decoding process using geometry information according to Embodiment 8.

FIG. 68 is a flowchart of an encoding process using structure information according to Embodiment 8.

FIG. 69 is a flowchart of a decoding process using structure information according to Embodiment 8.

FIG. 70 is a flowchart of an encoding process using attribute information according to Embodiment 8.

FIG. 71 is a flowchart of a decoding process using attribute information according to Embodiment 8.

FIG. 72 is a flowchart of a process of selecting a coding table using geometry information according to Embodiment 8.

FIG. 73 is a flowchart of a process of selecting a coding table using structure information according to Embodiment 8.

FIG. 74 is a flowchart of a process of selecting a coding table using attribute information according to Embodiment 8.

FIG. 75 is a block diagram of a three-dimensional data encoding device according to Embodiment 8.

FIG. 76 is a block diagram of a three-dimensional data decoding device according to Embodiment 8.

FIG. 77 is a diagram illustrating a reference relationship in an octree structure according to Embodiment 9.

FIG. 78 is a diagram illustrating a reference relationship in a spatial region according to Embodiment 9.

FIG. 79 is a diagram illustrating an example of neighboring reference nodes according to Embodiment 9.

FIG. 80 is a diagram illustrating a relationship between a parent node and nodes according to Embodiment 9.

FIG. 81 is a diagram illustrating an example of an occupancy code of the parent node according to Embodiment 9.

FIG. 82 is a block diagram of a three-dimensional data encoding device according to Embodiment 9.

FIG. 83 is a block diagram of a three-dimensional data decoding device according to Embodiment 9.

FIG. 84 is a flowchart of a three-dimensional data encoding process according to Embodiment 9.

FIG. 85 is a flowchart of a three-dimensional data decoding process according to Embodiment 9.

FIG. 86 is a diagram illustrating an example of selecting a coding table according to Embodiment 9.

FIG. 87 is a diagram illustrating a reference relationship in a spatial region according to Variation 1 of Embodiment 9.

FIG. 88 is a diagram illustrating an example of a syntax of header information according to Variation 1 of Embodiment 9.

FIG. 89 is a diagram illustrating an example of a syntax of header information according to Variation 1 of Embodiment 9.

FIG. 90 is a diagram illustrating an example of neighboring reference nodes according to Variation 2 of Embodiment 9.

FIG. 91 is a diagram illustrating an example of a current node and

neighboring nodes according to Variation 2 of Embodiment 9.

FIG. 92 is a diagram illustrating a reference relationship in an octree structure according to Variation 3 of Embodiment 9.

FIG. 93 is a diagram illustrating a reference relationship in a spatial region according to Variation 3 of Embodiment 9.

FIG. 94 is a diagram illustrating an example of translation according to Embodiment 10.

FIG. 95 is a diagram illustrating an example of rotation according to Embodiment 10.

FIG. 96 is a diagram illustrating an example of horizontality and verticality according to Embodiment 10.

FIG. 97 is a diagram illustrating an example of an adjacent surface according to Embodiment 10.

FIG. 98 is a diagram illustrating examples of translation according to Embodiment 10.

FIG. 99 is a diagram illustrating examples of x-axis rotation according to Embodiment 10.

FIG. 100 is a diagram illustrating examples of y-axis rotation according to Embodiment 10.

FIG. 101 is a diagram illustrating examples of z-axis rotation according to Embodiment 10.

FIG. 102 is a diagram illustrating examples of horizontality and verticality according to Embodiment 10.

FIG. 103 is a diagram illustrating examples of an adjacent surface according to Embodiment 10.

FIG. 104 is a diagram illustrating an example of grouping neighbor occupancy patterns according to Embodiment 10.

FIG. 105 is a diagram illustrating an example of grouping neighbor occupancy patterns according to Embodiment 10.

FIG. 106 is a diagram illustrating an example of a conversion table according to Embodiment 10.

FIG. 107 is a diagram illustrating an example of a conversion table according to Embodiment 10.

FIG. 108 is a diagram illustrating an outline of a mapping process according to Embodiment 10.

FIG. 109 is a diagram illustrating an outline of a mapping process according to Embodiment 10.

FIG. 110 is a block diagram of a three-dimensional data encoding device according to Embodiment 10.

FIG. 111 is a block diagram of a three-dimensional data decoding device according to Embodiment 10.

FIG. 112 is a flowchart of a three-dimensional data encoding process according to Embodiment 10.

FIG. 113 is a flowchart of a three-dimensional data decoding process according to Embodiment 10.

FIG. 114 is a flowchart of a three-dimensional data encoding process according to Embodiment 10.

FIG. 115 is a flowchart of a three-dimensional data decoding process according to Embodiment 10.

FIG. 116 is a flowchart of a three-dimensional data encoding process according to Embodiment 10.

FIG. 117 is a flowchart of a three-dimensional data decoding process according to Embodiment 10.

FIG. 118 is a diagram illustrating an example of grouping neighbor

occupancy patterns according to Embodiment 11.

FIG. 119 is a flowchart of a coding table selection process according to Embodiment 11.

FIG. 120 is a flowchart of a three-dimensional data encoding process according to Embodiment 11.

FIG. 121 is a flowchart of a three-dimensional data decoding process according to Embodiment 11.

FIG. 122 is a diagram for illustrating redundant coding tables according to Embodiment 12.

FIG. 123 is a diagram illustrating static and dynamic coding tables according to Embodiment 12.

FIG. 124 is a diagram illustrating an example of a current node according to Embodiment 12.

FIG. 125 is a diagram illustrating operations in case 1 according to Embodiment 12.

FIG. 126 is a diagram illustrating operations in case 2 according to Embodiment 12.

FIG. 127 is a diagram illustrating an example of neighboring nodes according to Embodiment 12.

FIG. 128 is a diagram illustrating specific examples of the number of redundant tables according to Embodiment 12.

FIG. 129 is a diagram illustrating specific examples of the number of redundant tables according to Embodiment 12.

FIG. 130 is a diagram illustrating a specific example of a redundant table according to Embodiment 12.

FIG. 131 is a diagram illustrating specific examples of a redundant table according to Embodiment 12.

FIG. 132 is a diagram illustrating an operation of not removing a redundant table according to Embodiment 12.

FIG. 133 is a diagram illustrating examples of coding tables from which redundant tables are removed according to Embodiment 12.

FIG. 134 is a diagram illustrating examples of the number of coding tables when redundant tables are removed according to Embodiment 12.

FIG. 135 is a diagram illustrating a process of creating a coding table having a dynamic size according to Embodiment 12.

FIG. 136 is a diagram illustrating examples of the size of a table according to Embodiment 12.

FIG. 137 is a flowchart of a process of creating a coding table having a dynamic size according to Embodiment 12.

FIG. 138 is a diagram illustrating examples of the size of a table according to Embodiment 12.

FIG. 139 is a flowchart of a coding table selection process according to Embodiment 12.

DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0013]

A three-dimensional data encoding method according to one aspect of the present disclosure includes: encoding a first flag indicating whether a node having a parent node different from a parent node of a current node is to be referred to in encoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2; selecting a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performing arithmetic encoding on information of the current node using the coding table selected, when the first flag indicates that the node is to be referred

to; and selecting a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performing arithmetic encoding on information of the current node using the coding table selected, when the first flag indicates that the node is not to be referred to, M being an integer different from N.

[0014]

According to this configuration, since it is possible to reduce the number of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0015]

For example, N may be greater than M.

[0016]

For example, when a coding table is selected from the M coding tables, the coding table may be selected from the M coding tables by reference to a correspondence table according to the occupancy states of the neighboring nodes, the correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and the M coding tables, L being an integer greater than M.

[0017]

For example, when a coding table is selected from the M coding tables, the coding table may be selected from the M coding tables by reference to a first correspondence table and a second correspondence table, according to the occupancy states of the neighboring nodes, the first correspondence table indicating a correspondence relationship between L occupancy patterns

indicating the occupancy states of the neighboring nodes and I coding tables, the second correspondence table indicating a correspondence relationship between the I coding tables and the M coding tables, L being an integer greater than I, I being an integer greater than M.

[0018]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction horizontal to an x-y plane.

[0019]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction vertical to an x-y plane.

[0020]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node,

and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is horizontal to an x-y plane.

[0021]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is vertical to an x-y plane.

[0022]

A three-dimensional data decoding method according to one aspect of the present disclosure includes: decoding a first flag indicating whether a node having a parent node different from a parent node of a current node is to be referred to in decoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2; selecting a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performing arithmetic decoding on information of the current node using the coding table selected, when the first flag indicates that the node is to be referred to; and selecting a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performing arithmetic decoding on information of the current node using the coding table selected, when the first flag indicates that the node is not to be referred to, M being an

integer different from N.

[0023]

According to this configuration, since it is possible to reduce the number of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0024]

For example, N may be greater than M.

[0025]

For example, when a coding table is selected from the M coding tables, the coding table may be selected from the M coding tables by reference to a correspondence table, according to the occupancy states of the neighboring nodes, the correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and the M coding tables, L being an integer greater than M.

[0026]

For example, when a coding table is selected from the M coding tables, the coding table may be selected from the M coding tables by reference to a first correspondence table and a second correspondence table, according to the occupancy states of the neighboring nodes, the first correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and I coding tables, the second correspondence table indicating a correspondence relationship between the I coding tables and the M coding tables, L being an integer greater than I, I being an integer greater than M.

[0027]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction horizontal to an x-y plane.

[0028]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction vertical to an x-y plane.

[0029]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is horizontal to an x-y plane.

[0030]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to may be occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node, and an identical coding table among the M coding tables may be assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is vertical to an x-y plane.

[0031]

A three-dimensional data encoding device according to one aspect of the present disclosure is a three-dimensional data encoding device that encodes three-dimensional points each including attribute information, the three-dimensional data encoding device including a processor and memory. Using the memory, the processor: encodes a first flag indicating whether a node having a parent node different from a parent node of a current node is to be referred to in encoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2; selects a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performs arithmetic encoding on information of the current node using the coding table selected, when the first flag indicates that the node is to be referred to; and selects a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performs arithmetic encoding on information of the current node using the coding table selected, when the first flag indicates that the node is not to be referred to, M being an integer different from N.

[0032]

According to this configuration, since it is possible to reduce the number of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0033]

A three-dimensional data decoding device according to one aspect of the present disclosure is a three-dimensional data decoding device that decode three-dimensional points each having attribute information, the three-dimensional data decoding device including a processor and memory. Using the memory, the processor: decodes a first flag indicating whether a node having a parent node different from a parent node of a current node is to be referred to in decoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2; selects a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performs arithmetic decoding on information of the current node using the coding table selected, when the first flag indicates that the node is to be referred to; and selects a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performs arithmetic decoding on information of the current node using the coding table selected, when the first flag indicates that the node is not to be referred to, M being an integer different from N.

[0034]

According to this configuration, since it is possible to reduce the number

of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0035]

Note that these general or specific aspects may be implemented as a system, a method, an integrated circuit, a computer program, or a computer-readable recording medium such as a CD-ROM, or may be implemented as any combination of a system, a method, an integrated circuit, a computer program, and a recording medium.

[0036]

The following describes embodiments with reference to the drawings. Note that the following embodiments show exemplary embodiments of the present disclosure. The numerical values, shapes, materials, structural components, the arrangement and connection of the structural components, steps, the processing order of the steps, etc. shown in the following embodiments are mere examples, and thus are not intended to limit the present disclosure. Of the structural components described in the following embodiments, structural components not recited in any one of the independent claims that indicate the broadest concepts will be described as optional structural components.

[0037]

EMBODIMENT 1

First, the data structure of encoded three-dimensional data (hereinafter also referred to as encoded data) according to the present embodiment will be described. FIG. 1 is a diagram showing the structure of encoded three-

dimensional data according to the present embodiment.

[0038]

In the present embodiment, a three-dimensional space is divided into spaces (SPCs), which correspond to pictures in moving picture encoding, and the three-dimensional data is encoded on a SPC-by-SPC basis. Each SPC is further divided into volumes (VLMs), which correspond to macroblocks, etc. in moving picture encoding, and predictions and transforms are performed on a VLM-by-VLM basis. Each volume includes a plurality of voxels (VXLs), each being a minimum unit in which position coordinates are associated. Note that prediction is a process of generating predictive three-dimensional data analogous to a current processing unit by referring to another processing unit, and encoding a differential between the predictive three-dimensional data and the current processing unit, as in the case of predictions performed on two-dimensional images. Such prediction includes not only spatial prediction in which another prediction unit corresponding to the same time is referred to, but also temporal prediction in which a prediction unit corresponding to a different time is referred to.

[0039]

When encoding a three-dimensional space represented by point group data such as a point cloud, for example, the three-dimensional data encoding device (hereinafter also referred to as the encoding device) encodes the points in the point group or points included in the respective voxels in a collective manner, in accordance with a voxel size. Finer voxels enable a highly-precise representation of the three-dimensional shape of a point group, while larger voxels enable a rough representation of the three-dimensional shape of a point group.

[0040]

Note that the following describes the case where three-dimensional data is a point cloud, but three-dimensional data is not limited to a point cloud, and thus three-dimensional data of any format may be employed.

[0041]

Also note that voxels with a hierarchical structure may be used. In such a case, when the hierarchy includes n levels, whether a sampling point is included in the n-1th level or its lower levels (the lower levels of the n-th level) may be sequentially indicated. For example, when only the n-th level is decoded, and the n-1th level or its lower levels include a sampling point, the n-th level can be decoded on the assumption that a sampling point is included at the center of a voxel in the n-th level.

[0042]

Also, the encoding device obtains point group data, using, for example, a distance sensor, a stereo camera, a monocular camera, a gyroscope sensor, or an inertial sensor.

[0043]

As in the case of moving picture encoding, each SPC is classified into one of at least the three prediction structures that include: intra SPC (I-SPC), which is individually decodable; predictive SPC (P-SPC) capable of only a unidirectional reference; and bidirectional SPC (B-SPC) capable of bidirectional references. Each SPC includes two types of time information: decoding time and display time.

[0044]

Furthermore, as shown in FIG. 1, a processing unit that includes a plurality of SPCs is a group of spaces (GOS), which is a random access unit. Also, a processing unit that includes a plurality of GOSs is a world (WLD).

[0045]

The spatial region occupied by each world is associated with an absolute position on earth, by use of, for example, GPS, or latitude and longitude information. Such position information is stored as meta-information. Note that meta-information may be included in encoded data, or may be transmitted separately from the encoded data.

[0046]

Also, inside a GOS, all SPCs may be three-dimensionally adjacent to one another, or there may be a SPC that is not three-dimensionally adjacent to another SPC.

[0047]

Note that the following also describes processes such as encoding, decoding, and reference to be performed on three-dimensional data included in processing units such as GOS, SPC, and VLM, simply as performing encoding/to encode, decoding/to decode, referring to, etc. on a processing unit. Also note that three-dimensional data included in a processing unit includes, for example, at least one pair of a spatial position such as three-dimensional coordinates and an attribute value such as color information.

[0048]

Next, the prediction structures among SPCs in a GOS will be described. A plurality of SPCs in the same GOS or a plurality of VLMs in the same SPC occupy mutually different spaces, while having the same time information (the decoding time and the display time).

[0049]

A SPC in a GOS that comes first in the decoding order is an I-SPC. GOSs come in two types: closed GOS and open GOS. A closed GOS is a GOS in which all SPCs in the GOS are decodable when decoding starts from the first I-SPC. Meanwhile, an open GOS is a GOS in which a different GOS is referred

to in one or more SPCs preceding the first I-SPC in the GOS in the display time, and thus cannot be singly decoded.

[0050]

Note that in the case of encoded data of map information, for example, a WLD is sometimes decoded in the backward direction, which is opposite to the encoding order, and thus backward reproduction is difficult when GOSs are interdependent. In such a case, a closed GOS is basically used.

[0051]

Each GOS has a layer structure in height direction, and SPCs are sequentially encoded or decoded from SPCs in the bottom layer.

[0052]

FIG. 2 is a diagram showing an example of prediction structures among SPCs that belong to the lowermost layer in a GOS. FIG. 3 is a diagram showing an example of prediction structures among layers.

[0053]

A GOS includes at least one I-SPC. Of the objects in a three-dimensional space, such as a person, an animal, a car, a bicycle, a signal, and a building serving as a landmark, a small-sized object is especially effective when encoded as an I-SPC. When decoding a GOS at a low throughput or at a high speed, for example, the three-dimensional data decoding device (hereinafter also referred to as the decoding device) decodes only I-SPC(s) in the GOS.

[0054]

The encoding device may also change the encoding interval or the appearance frequency of I-SPCs, depending on the degree of sparseness and denseness of the objects in a WLD.

[0055]

In the structure shown in FIG. 3, the encoding device or the decoding

device encodes or decodes a plurality of layers sequentially from the bottom layer (layer 1). This increases the priority of data on the ground and its vicinity, which involve a larger amount of information, when, for example, a self-driving car is concerned.

[0056]

Regarding encoded data used for a drone, for example, encoding or decoding may be performed sequentially from SPCs in the top layer in a GOS in height direction.

[0057]

The encoding device or the decoding device may also encode or decode a plurality of layers in a manner that the decoding device can have a rough grasp of a GOS first, and then the resolution is gradually increased. The encoding device or the decoding device may perform encoding or decoding in the order of layers 3, 8, 1, 9..., for example.

[0058]

Next, the handling of static objects and dynamic objects will be described.

[0059]

A three-dimensional space includes scenes or still objects such as a building and a road (hereinafter collectively referred to as static objects), and objects with motion such as a car and a person (hereinafter collectively referred to as dynamic objects). Object detection is separately performed by, for example, extracting keypoints from point cloud data, or from video of a camera such as a stereo camera. In this description, an example method of encoding a dynamic object will be described.

[0060]

A first method is a method in which a static object and a dynamic object are encoded without distinction. A second method is a method in which a

distinction is made between a static object and a dynamic object on the basis of identification information.

[0061]

For example, a GOS is used as an identification unit. In such a case, a distinction is made between a GOS that includes SPCs constituting a static object and a GOS that includes SPCs constituting a dynamic object, on the basis of identification information stored in the encoded data or stored separately from the encoded data.

[0062]

Alternatively, a SPC may be used as an identification unit. In such a case, a distinction is made between a SPC that includes VLMs constituting a static object and a SPC that includes VLMs constituting a dynamic object, on the basis of the identification information thus described.

[0063]

Alternatively, a VLM or a VXL may be used as an identification unit. In such a case, a distinction is made between a VLM or a VXL that includes a static object and a VLM or a VXL that includes a dynamic object, on the basis of the identification information thus described.

[0064]

The encoding device may also encode a dynamic object as at least one VLM or SPC, and may encode a VLM or a SPC including a static object and a SPC including a dynamic object as mutually different GOSs. When the GOS size is variable depending on the size of a dynamic object, the encoding device separately stores the GOS size as meta-information.

[0065]

The encoding device may also encode a static object and a dynamic object separately from each other, and may superimpose the dynamic object onto a

world constituted by static objects. In such a case, the dynamic object is constituted by at least one SPC, and each SPC is associated with at least one SPC constituting the static object onto which the each SPC is to be superimposed. Note that a dynamic object may be represented not by SPC(s) but by at least one VLM or VXL.

[0066]

The encoding device may also encode a static object and a dynamic object as mutually different streams.

[0067]

The encoding device may also generate a GOS that includes at least one SPC constituting a dynamic object. The encoding device may further set the size of a GOS including a dynamic object (GOS_M) and the size of a GOS including a static object corresponding to the spatial region of GOS_M at the same size (such that the same spatial region is occupied). This enables superimposition to be performed on a GOS-by-GOS basis.

[0068]

SPC(s) included in another encoded GOS may be referred to in a P-SPC or a B-SPC constituting a dynamic object. In the case where the position of a dynamic object temporally changes, and the same dynamic object is encoded as an object in a GOS corresponding to a different time, referring to SPC(s) across GOSs is effective in terms of compression rate.

[0069]

The first method and the second method may be selected in accordance with the intended use of encoded data. When encoded three-dimensional data is used as a map, for example, a dynamic object is desired to be separated, and thus the encoding device uses the second method. Meanwhile, the encoding device uses the first method when the separation of a dynamic object is not

required such as in the case where three-dimensional data of an event such as a concert and a sports event is encoded.

[0070]

The decoding time and the display time of a GOS or a SPC are storable in encoded data or as meta-information. All static objects may have the same time information. In such a case, the decoding device may determine the actual decoding time and display time. Alternatively, a different value may be assigned to each GOS or SPC as the decoding time, and the same value may be assigned as the display time. Furthermore, as in the case of the decoder model in moving picture encoding such as Hypothetical Reference Decoder (HRD) compliant with HEVC, a model may be employed that ensures that a decoder can perform decoding without fail by having a buffer of a predetermined size and by reading a bitstream at a predetermined bit rate in accordance with the decoding times.

[0071]

Next, the topology of GOSs in a world will be described. The coordinates of the three-dimensional space in a world are represented by the three coordinate axes (x axis, y axis, and z axis) that are orthogonal to one another. A predetermined rule set for the encoding order of GOSs enables encoding to be performed such that spatially adjacent GOSs are contiguous in the encoded data. In an example shown in FIG. 4, for example, GOSs in the x and z planes are successively encoded. After the completion of encoding all GOSs in certain x and z planes, the value of the y axis is updated. Stated differently, the world expands in the y axis direction as the encoding progresses. The GOS index numbers are set in accordance with the encoding order.

[0072]

Here, the three-dimensional spaces in the respective worlds are

previously associated one-to-one with absolute geographical coordinates such as GPS coordinates or latitude/longitude coordinates. Alternatively, each three-dimensional space may be represented as a position relative to a previously set reference position. The directions of the x axis, the y axis, and the z axis in the three-dimensional space are represented by directional vectors that are determined on the basis of the latitudes and the longitudes, etc. Such directional vectors are stored together with the encoded data as meta-information.

[0073]

GOSs have a fixed size, and the encoding device stores such size as meta-information. The GOS size may be changed depending on, for example, whether it is an urban area or not, or whether it is inside or outside of a room. Stated differently, the GOS size may be changed in accordance with the amount or the attributes of objects with information values. Alternatively, in the same world, the encoding device may adaptively change the GOS size or the interval between I-SPCs in GOSs in accordance with the object density, etc. For example, the encoding device sets the GOS size to smaller and the interval between I-SPCs in GOSs to shorter, as the object density is higher.

[0074]

In an example shown in FIG. 5, to enable random access with a finer granularity, a GOS with a high object density is partitioned into the regions of the third to tenth GOSs. Note that the seventh to tenth GOSs are located behind the third to sixth GOSs.

[0075]

Next, the structure and the operation flow of the three-dimensional data encoding device according to the present embodiment will be described. FIG. 6 is a block diagram of three-dimensional data encoding device 100 according to

the present embodiment. FIG. 7 is a flowchart of an example operation performed by three-dimensional data encoding device 100.

[0076]

Three-dimensional data encoding device 100 shown in FIG. 6 encodes three-dimensional data 111, thereby generating encoded three-dimensional data 112. Such three-dimensional data encoding device 100 includes obtainer 101, encoding region determiner 102, divider 103, and encoder 104.

[0077]

As shown in FIG. 7, first, obtainer 101 obtains three-dimensional data 111, which is point group data (S101).

[0078]

Next, encoding region determiner 102 determines a current region for encoding from among spatial regions corresponding to the obtained point group data (S102). For example, in accordance with the position of a user or a vehicle, encoding region determiner 102 determines, as the current region, a spatial region around such position.

[0079]

Next, divider 103 divides the point group data included in the current region into processing units. The processing units here means units such as GOSs and SPCs described above. The current region here corresponds to, for example, a world described above. More specifically, divider 103 divides the point group data into processing units on the basis of a predetermined GOS size, or the presence/absence/size of a dynamic object (S103). Divider 103 further determines the starting position of the SPC that comes first in the encoding order in each GOS.

[0080]

Next, encoder 104 sequentially encodes a plurality of SPCs in each GOS,

thereby generating encoded three-dimensional data 112 (S104).

[0081]

Note that although an example is described here in which the current region is divided into GOSs and SPCs, after which each GOS is encoded, the processing steps are not limited to this order. For example, steps may be employed in which the structure of a single GOS is determined, which is followed by the encoding of such GOS, and then the structure of the subsequent GOS is determined.

[0082]

As thus described, three-dimensional data encoding device 100 encodes three-dimensional data 111, thereby generating encoded three-dimensional data 112. More specifically, three-dimensional data encoding device 100 divides three-dimensional data into first processing units (GOSs), each being a random access unit and being associated with three-dimensional coordinates, divides each of the first processing units (GOSs) into second processing units (SPCs), and divides each of the second processing units (SPCs) into third processing units (VLMs). Each of the third processing units (VLMs) includes at least one voxel (VXL), which is the minimum unit in which position information is associated.

[0083]

Next, three-dimensional data encoding device 100 encodes each of the first processing units (GOSs), thereby generating encoded three-dimensional data 112. More specifically, three-dimensional data encoding device 100 encodes each of the second processing units (SPCs) in each of the first processing units (GOSs). Three-dimensional data encoding device 100 further encodes each of the third processing units (VLMs) in each of the second processing units (SPCs).

[0084]

When a current first processing unit (GOS) is a closed GOS, for example, three-dimensional data encoding device 100 encodes a current second processing unit (SPC) included in such current first processing unit (GOS) by referring to another second processing unit (SPC) included in the current first processing unit (GOS). Stated differently, three-dimensional data encoding device 100 refers to no second processing unit (SPC) included in a first processing unit (GOS) that is different from the current first processing unit (GOS).

[0085]

Meanwhile, when a current first processing unit (GOS) is an open GOS, three-dimensional data encoding device 100 encodes a current second processing unit (SPC) included in such current first processing unit (GOS) by referring to another second processing unit (SPC) included in the current first processing unit (GOS) or a second processing unit (SPC) included in a first processing unit (GOS) that is different from the current first processing unit (GOS).

[0086]

Also, three-dimensional data encoding device 100 selects, as the type of a current second processing unit (SPC), one of the following: a first type (I-SPC) in which another second processing unit (SPC) is not referred to; a second type (P-SPC) in which another single second processing unit (SPC) is referred to; and a third type in which other two second processing units (SPC) are referred to. Three-dimensional data encoding device 100 encodes the current second processing unit (SPC) in accordance with the selected type.

[0087]

Next, the structure and the operation flow of the three-dimensional data decoding device according to the present embodiment will be described. FIG. 8 is a block diagram of three-dimensional data decoding device 200 according to

the present embodiment. FIG. 9 is a flowchart of an example operation performed by three-dimensional data decoding device 200.

[0088]

Three-dimensional data decoding device 200 shown in FIG. 8 decodes encoded three-dimensional data 211, thereby generating decoded three-dimensional data 212. Encoded three-dimensional data 211 here is, for example, encoded three-dimensional data 112 generated by three-dimensional data encoding device 100. Such three-dimensional data decoding device 200 includes obtainer 201, decoding start GOS determiner 202, decoding SPC determiner 203, and decoder 204.

[0089]

First, obtainer 201 obtains encoded three-dimensional data 211 (S201). Next, decoding start GOS determiner 202 determines a current GOS for decoding (S202). More specifically, decoding start GOS determiner 202 refers to meta-information stored in encoded three-dimensional data 211 or stored separately from the encoded three-dimensional data to determine, as the current GOS, a GOS that includes a SPC corresponding to the spatial position, the object, or the time from which decoding is to start.

[0090]

Next, decoding SPC determiner 203 determines the type(s) (I, P, and/or B) of SPCs to be decoded in the GOS (S203). For example, decoding SPC determiner 203 determines whether to (1) decode only I-SPC(s), (2) to decode I-SPC(s) and P-SPCs, or (3) to decode SPCs of all types. Note that the present step may not be performed, when the type(s) of SPCs to be decoded are previously determined such as when all SPCs are previously determined to be decoded.

[0091]

Next, decoder 204 obtains an address location within encoded three-dimensional data 211 from which a SPC that comes first in the GOS in the decoding order (the same as the encoding order) starts. Decoder 204 obtains the encoded data of the first SPC from the address location, and sequentially decodes the SPCs from such first SPC (S204). Note that the address location is stored in the meta-information, etc.

[0092]

Three-dimensional data decoding device 200 decodes decoded three-dimensional data 212 as thus described. More specifically, three-dimensional data decoding device 200 decodes each encoded three-dimensional data 211 of the first processing units (GOSs), each being a random access unit and being associated with three-dimensional coordinates, thereby generating decoded three-dimensional data 212 of the first processing units (GOSs). Even more specifically, three-dimensional data decoding device 200 decodes each of the second processing units (SPCs) in each of the first processing units (GOSs). Three-dimensional data decoding device 200 further decodes each of the third processing units (VLMs) in each of the second processing units (SPCs).

[0093]

The following describes meta-information for random access. Such meta-information is generated by three-dimensional data encoding device 100, and included in encoded three-dimensional data 112 (211).

[0094]

In the conventional random access for a two-dimensional moving picture, decoding starts from the first frame in a random access unit that is close to a specified time. Meanwhile, in addition to times, random access to spaces (coordinates, objects, etc.) is assumed to be performed in a world.

[0095]

To enable random access to at least three elements of coordinates, objects, and times, tables are prepared that associate the respective elements with the GOS index numbers. Furthermore, the GOS index numbers are associated with the addresses of the respective first I-SPCs in the GOSs. FIG. 10 is a diagram showing example tables included in the meta-information. Note that not all the tables shown in FIG. 10 are required to be used, and thus at least one of the tables is used.

[0096]

The following describes an example in which random access is performed from coordinates as a starting point. To access the coordinates (x2, y2, and z2), the coordinates-GOS table is first referred to, which indicates that the point corresponding to the coordinates (x2, y2, and z2) is included in the second GOS. Next, the GOS-address table is referred to, which indicates that the address of the first I-SPC in the second GOS is addr(2). As such, decoder 204 obtains data from this address to start decoding.

[0097]

Note that the addresses may either be logical addresses or physical addresses of an HDD or a memory. Alternatively, information that identifies file segments may be used instead of addresses. File segments are, for example, units obtained by segmenting at least one GOS, etc.

[0098]

When an object spans across a plurality of GOSs, the object-GOS table may show a plurality of GOSs to which such object belongs. When such plurality of GOSs are closed GOSs, the encoding device and the decoding device can perform encoding or decoding in parallel. Meanwhile, when such plurality of GOSs are open GOSs, a higher compression efficiency is achieved by the plurality of GOSs referring to each other.

[0099]

Example objects include a person, an animal, a car, a bicycle, a signal, and a building serving as a landmark. For example, three-dimensional data encoding device 100 extracts keypoints specific to an object from a three-dimensional point cloud, etc., when encoding a world, and detects the object on the basis of such keypoints to set the detected object as a random access point.

[0100]

As thus described, three-dimensional data encoding device 100 generates first information indicating a plurality of first processing units (GOSs) and the three-dimensional coordinates associated with the respective first processing units (GOSs). Encoded three-dimensional data 112 (211) includes such first information. The first information further indicates at least one of objects, times, and data storage locations that are associated with the respective first processing units (GOSs).

[0101]

Three-dimensional data decoding device 200 obtains the first information from encoded three-dimensional data 211. Using such first information, three-dimensional data decoding device 200 identifies encoded three-dimensional data 211 of the first processing unit that corresponds to the specified three-dimensional coordinates, object, or time, and decodes encoded three-dimensional data 211.

[0102]

The following describes an example of other meta-information. In addition to the meta-information for random access, three-dimensional data encoding device 100 may also generate and store meta-information as described below, and three-dimensional data decoding device 200 may use such meta-information at the time of decoding.

[0103]

When three-dimensional data is used as map information, for example, a profile is defined in accordance with the intended use, and information indicating such profile may be included in meta-information. For example, a profile is defined for an urban or a suburban area, or for a flying object, and the maximum or minimum size, etc. of a world, a SPC or a VLM, etc. is defined in each profile. For example, more detailed information is required for an urban area than for a suburban area, and thus the minimum VLM size is set to small.

[0104]

The meta-information may include tag values indicating object types. Each of such tag values is associated with VLMs, SPCs, or GOSs that constitute an object. For example, a tag value may be set for each object type in a manner, for example, that the tag value "0" indicates "person," the tag value "1" indicates "car," and the tag value "2" indicates "signal." Alternatively, when an object type is hard to judge, or such judgment is not required, a tag value may be used that indicates the size or the attribute indicating, for example, whether an object is a dynamic object or a static object.

[0105]

The meta-information may also include information indicating a range of the spatial region occupied by a world.

[0106]

The meta-information may also store the SPC or VXL size as header information common to the whole stream of the encoded data or to a plurality of SPCs, such as SPCs in a GOS.

[0107]

The meta-information may also include identification information on a distance sensor or a camera that has been used to generate a point cloud, or

information indicating the positional accuracy of a point group in the point cloud.

[0108]

The meta-information may also include information indicating whether a world is made only of static objects or includes a dynamic object.

[0109]

The following describes variations of the present embodiment.

[0110]

The encoding device or the decoding device may encode or decode two or more mutually different SPCs or GOSs in parallel. GOSs to be encoded or decoded in parallel can be determined on the basis of meta-information, etc. indicating the spatial positions of the GOSs.

[0111]

When three-dimensional data is used as a spatial map for use by a car or a flying object, etc. in traveling, or for creation of such a spatial map, for example, the encoding device or the decoding device may encode or decode GOSs or SPCs included in a space that is identified on the basis of GPS information, the route information, the zoom magnification, etc.

[0112]

The decoding device may also start decoding sequentially from a space that is close to the self-location or the traveling route. The encoding device or the decoding device may give a lower priority to a space distant from the self-location or the traveling route than the priority of a nearby space to encode or decode such distant place. To "give a lower priority" means here, for example, to lower the priority in the processing sequence, to decrease the resolution (to apply decimation in the processing), or to lower the image quality (to increase the encoding efficiency by, for example, setting the quantization step to larger).

[0113]

When decoding encoded data that is hierarchically encoded in a space, the decoding device may decode only the bottom level in the hierarchy.

[0114]

The decoding device may also start decoding preferentially from the bottom level of the hierarchy in accordance with the zoom magnification or the intended use of the map.

[0115]

For self-location estimation or object recognition, etc. involved in the self-driving of a car or a robot, the encoding device or the decoding device may encode or decode regions at a lower resolution, except for a region that is lower than or at a specified height from the ground (the region to be recognized).

[0116]

The encoding device may also encode point clouds representing the spatial shapes of a room interior and a room exterior separately. For example, the separation of a GOS representing a room interior (interior GOS) and a GOS representing a room exterior (exterior GOS) enables the decoding device to select a GOS to be decoded in accordance with a viewpoint location, when using the encoded data.

[0117]

The encoding device may also encode an interior GOS and an exterior GOS having close coordinates so that such GOSs come adjacent to each other in an encoded stream. For example, the encoding device associates the identifiers of such GOSs with each other, and stores information indicating the associated identifiers into the meta-information that is stored in the encoded stream or stored separately. This enables the decoding device to refer to the information in the meta-information to identify an interior GOS and an exterior GOS having close coordinates.

[0118]

The encoding device may also change the GOS size or the SPC size depending on whether a GOS is an interior GOS or an exterior GOS. For example, the encoding device sets the size of an interior GOS to smaller than the size of an exterior GOS. The encoding device may also change the accuracy of extracting keypoints from a point cloud, or the accuracy of detecting objects, for example, depending on whether a GOS is an interior GOS or an exterior GOS.

[0119]

The encoding device may also add, to encoded data, information by which the decoding device displays objects with a distinction between a dynamic object and a static object. This enables the decoding device to display a dynamic object together with, for example, a red box or letters for explanation. Note that the decoding device may display only a red box or letters for explanation, instead of a dynamic object. The decoding device may also display more particular object types. For example, a red box may be used for a car, and a yellow box may be used for a person.

[0120]

The encoding device or the decoding device may also determine whether to encode or decode a dynamic object and a static object as a different SPC or GOS, in accordance with, for example, the appearance frequency of dynamic objects or a ratio between static objects and dynamic objects. For example, when the appearance frequency or the ratio of dynamic objects exceeds a threshold, a SPC or a GOS including a mixture of a dynamic object and a static object is accepted, while when the appearance frequency or the ratio of dynamic objects is below a threshold, a SPC or GOS including a mixture of a dynamic object and a static object is unaccepted.

[0121]

When detecting a dynamic object not from a point cloud but from two-dimensional image information of a camera, the encoding device may separately obtain information for identifying a detection result (box or letters) and the object position, and encode these items of information as part of the encoded three-dimensional data. In such a case, the decoding device superimposes auxiliary information (box or letters) indicating the dynamic object onto a resultant of decoding a static object to display it.

[0122]

The encoding device may also change the sparseness and denseness of VXLs or VLMs in a SPC in accordance with the degree of complexity of the shape of a static object. For example, the encoding device sets VXLs or VLMs at a higher density as the shape of a static object is more complex. The encoding device may further determine a quantization step, etc. for quantizing spatial positions or color information in accordance with the sparseness and denseness of VXLs or VLMs. For example, the encoding device sets the quantization step to smaller as the density of VXLs or VLMs is higher.

[0123]

As described above, the encoding device or the decoding device according to the present embodiment encodes or decodes a space on a SPC-by-SPC basis that includes coordinate information.

[0124]

Furthermore, the encoding device and the decoding device perform encoding or decoding on a volume-by-volume basis in a SPC. Each volume includes a voxel, which is the minimum unit in which position information is associated.

[0125]

Also, using a table that associates the respective elements of spatial

information including coordinates, objects, and times with GOSs or using a table that associates these elements with each other, the encoding device and the decoding device associate any ones of the elements with each other to perform encoding or decoding. The decoding device uses the values of the selected elements to determine the coordinates, and identifies a volume, a voxel, or a SPC from such coordinates to decode a SPC including such volume or voxel, or the identified SPC.

[0126]

Furthermore, the encoding device determines a volume, a voxel, or a SPC that is selectable in accordance with the elements, through extraction of keypoints and object recognition, and encodes the determined volume, voxel, or SPC, as a volume, a voxel, or a SPC to which random access is possible.

[0127]

SPCs are classified into three types: I-SPC that is singly encodable or decodable; P-SPC that is encoded or decoded by referring to any one of the processed SPCs; and B-SPC that is encoded or decoded by referring to any two of the processed SPCs.

[0128]

At least one volume corresponds to a static object or a dynamic object. A SPC including a static object and a SPC including a dynamic object are encoded or decoded as mutually different GOSs. Stated differently, a SPC including a static object and a SPC including a dynamic object are assigned to different GOSs.

[0129]

Dynamic objects are encoded or decoded on an object-by-object basis, and are associated with at least one SPC including a static object. Stated differently, a plurality of dynamic objects are individually encoded, and the

obtained encoded data of the dynamic objects is associated with a SPC including a static object.

[0130]

The encoding device and the decoding device give an increased priority to I-SPC(s) in a GOS to perform encoding or decoding. For example, the encoding device performs encoding in a manner that prevents the degradation of I-SPCs (in a manner that enables the original three-dimensional data to be reproduced with a higher fidelity after decoded). The decoding device decodes, for example, only I-SPCs.

[0131]

The encoding device may change the frequency of using I-SPCs depending on the sparseness and denseness or the number (amount) of the objects in a world to perform encoding. Stated differently, the encoding device changes the frequency of selecting I-SPCs depending on the number or the sparseness and denseness of the objects included in the three-dimensional data. For example, the encoding device uses I-SPCs at a higher frequency as the density of the objects in a world is higher.

[0132]

The encoding device also sets random access points on a GOS-by-GOS basis, and stores information indicating the spatial regions corresponding to the GOSs into the header information.

[0133]

The encoding device uses, for example, a default value as the spatial size of a GOS. Note that the encoding device may change the GOS size depending on the number (amount) or the sparseness and denseness of objects or dynamic objects. For example, the encoding device sets the spatial size of a GOS to smaller as the density of objects or dynamic objects is higher or the number of

objects or dynamic objects is greater.

[0134]

Also, each SPC or volume includes a keypoint group that is derived by use of information obtained by a sensor such as a depth sensor, a gyroscope sensor, or a camera sensor. The coordinates of the keypoints are set at the central positions of the respective voxels. Furthermore, finer voxels enable highly accurate position information.

[0135]

The keypoint group is derived by use of a plurality of pictures. A plurality of pictures include at least two types of time information: the actual time information and the same time information common to a plurality of pictures that are associated with SPCs (for example, the encoding time used for rate control, etc.).

[0136]

Also, encoding or decoding is performed on a GOS-by-GOS basis that includes at least one SPC.

[0137]

The encoding device and the decoding device predict P-SPCs or B-SPCs in a current GOS by referring to SPCs in a processed GOS.

[0138]

Alternatively, the encoding device and the decoding device predict P-SPCs or B-SPCs in a current GOS, using the processed SPCs in the current GOS, without referring to a different GOS.

[0139]

Furthermore, the encoding device and the decoding device transmit or receive an encoded stream on a world-by-world basis that includes at least one GOS.

[0140]

Also, a GOS has a layer structure in one direction at least in a world, and the encoding device and the decoding device start encoding or decoding from the bottom layer. For example, a random accessible GOS belongs to the lowermost layer. A GOS that belongs to the same layer or a lower layer is referred to in a GOS that belongs to an upper layer. Stated differently, a GOS is spatially divided in a predetermined direction in advance to have a plurality of layers, each including at least one SPC. The encoding device and the decoding device encode or decode each SPC by referring to a SPC included in the same layer as the each SPC or a SPC included in a layer lower than that of the each SPC.

[0141]

Also, the encoding device and the decoding device successively encode or decode GOSs on a world-by-world basis that includes such GOSs. In so doing, the encoding device and the decoding device write or read out information indicating the order (direction) of encoding or decoding as metadata. Stated differently, the encoded data includes information indicating the order of encoding a plurality of GOSs.

[0142]

The encoding device and the decoding device also encode or decode mutually different two or more SPCs or GOSs in parallel.

[0143]

Furthermore, the encoding device and the decoding device encode or decode the spatial information (coordinates, size, etc.) on a SPC or a GOS.

[0144]

The encoding device and the decoding device encode or decode SPCs or GOSs included in an identified space that is identified on the basis of external

information on the self-location or/and region size, such as GPS information, route information, or magnification.

[0145]

The encoding device or the decoding device gives a lower priority to a space distant from the self-location than the priority of a nearby space to perform encoding or decoding.

[0146]

The encoding device sets a direction at one of the directions in a world, in accordance with the magnification or the intended use, to encode a GOS having a layer structure in such direction. Also, the decoding device decodes a GOS having a layer structure in one of the directions in a world that has been set in accordance with the magnification or the intended use, preferentially from the bottom layer.

[0147]

The encoding device changes the accuracy of extracting keypoints, the accuracy of recognizing objects, or the size of spatial regions, etc. included in a SPC, depending on whether an object is an interior object or an exterior object. Note that the encoding device and the decoding device encode or decode an interior GOS and an exterior GOS having close coordinates in a manner that these GOSs come adjacent to each other in a world, and associates their identifiers with each other for encoding and decoding.

[0148]

EMBODIMENT 2

When using encoded data of a point cloud in an actual device or service, it is desirable that necessary information be transmitted/received in accordance with the intended use to reduce the network bandwidth. However, there has been no such functionality in the structure of encoding three-dimensional data,

nor an encoding method therefor.

[0149]

The present embodiment describes a three-dimensional data encoding method and a three-dimensional data encoding device for providing the functionality of transmitting/receiving only necessary information in encoded data of a three-dimensional point cloud in accordance with the intended use, as well as a three-dimensional data decoding method and a three-dimensional data decoding device for decoding such encoded data.

[0150]

A voxel (VXL) with a feature greater than or equal to a given amount is defined as a feature voxel (FVXL), and a world (WLD) constituted by FVXLs is defined as a sparse world (SWLD). FIG. 11 is a diagram showing example structures of a sparse world and a world. A SWLD includes: FGOSs, each being a GOS constituted by FVXLs; FSPCs, each being a SPC constituted by FVXLs; and FVLMs, each being a VLM constituted by FVXLs. The data structure and prediction structure of a FGOS, a FSPC, and a FVLM may be the same as those of a GOS, a SPC, and a VLM.

[0151]

A feature represents the three-dimensional position information on a VXL or the visible-light information on the position of a VXL. A large number of features are detected especially at a corner, an edge, etc. of a three-dimensional object. More specifically, such a feature is a three-dimensional feature or a visible-light feature as described below, but may be any feature that represents the position, luminance, or color information, etc. on a VXL.

[0152]

Used as three-dimensional features are signature of histograms of orientations (SHOT) features, point feature histograms (PFH) features, or point

pair feature (PPF) features.

[0153]

SHOT features are obtained by dividing the periphery of a VXL, and calculating an inner product of the reference point and the normal vector of each divided region to represent the calculation result as a histogram. SHOT features are characterized by a large number of dimensions and high-level feature representation.

[0154]

PFH features are obtained by selecting a large number of two point pairs in the vicinity of a VXL, and calculating the normal vector, etc. from each two point pair to represent the calculation result as a histogram. PFH features are histogram features, and thus are characterized by robustness against a certain extent of disturbance and also high-level feature representation.

[0155]

PPF features are obtained by using a normal vector, etc. for each two points of VXLs. PPF features, for which all VXLs are used, has robustness against occlusion.

[0156]

Used as visible-light features are scale-invariant feature transform (SIFT), speeded up robust features (SURF), or histogram of oriented gradients (HOG), etc. that use information on an image such as luminance gradient information.

[0157]

A SWLD is generated by calculating the above-described features of the respective VXLs in a WLD to extract FVXLs. Here, the SWLD may be updated every time the WLD is updated, or may be regularly updated after the elapse of a certain period of time, regardless of the timing at which the WLD is updated.

[0158]

A SWLD may be generated for each type of features. For example, different SWLDs may be generated for the respective types of features, such as SWLD1 based on SHOT features and SWLD2 based on SIFT features so that SWLDs are selectively used in accordance with the intended use. Also, the calculated feature of each FVXL may be held in each FVXL as feature information.

[0159]

Next, the usage of a sparse world (SWLD) will be described. A SWLD includes only feature voxels (FVXLs), and thus its data size is smaller in general than that of a WLD that includes all VXLs.

[0160]

In an application that utilizes features for a certain purpose, the use of information on a SWLD instead of a WLD reduces the time required to read data from a hard disk, as well as the bandwidth and the time required for data transfer over a network. For example, a WLD and a SWLD are held in a server as map information so that map information to be sent is selected between the WLD and the SWLD in accordance with a request from a client. This reduces the network bandwidth and the time required for data transfer. More specific examples will be described below.

[0161]

FIG. 12 and FIG. 13 are diagrams showing usage examples of a SWLD and a WLD. As FIG. 12 shows, when client 1, which is a vehicle-mounted device, requires map information to use it for self-location determination, client 1 sends to a server a request for obtaining map data for self-location estimation (S301). The server sends to client 1 the SWLD in response to the obtainment request (S302). Client 1 uses the received SWLD to determine the self-location

(S303). In so doing, client 1 obtains VXL information on the periphery of client 1 through various means including a distance sensor such as a rangefinder, as well as a stereo camera and a combination of a plurality of monocular cameras. Client 1 then estimates the self-location information from the obtained VXL information and the SWLD. Here, the self-location information includes three-dimensional position information, orientation, etc. of client 1.

[0162]

As FIG. 13 shows, when client 2, which is a vehicle-mounted device, requires map information to use it for rendering a map such as a three-dimensional map, client 2 sends to the server a request for obtaining map data for map rendering (S311). The server sends to client 2 the WLD in response to the obtainment request (S312). Client 2 uses the received WLD to render a map (S313). In so doing, client 2 uses, for example, an image client 2 has captured by a visible-light camera, etc. and the WLD obtained from the server to create a rendering image, and renders such created image onto a screen of a car navigation system, etc.

[0163]

As described above, the server sends to a client a SWLD when the features of the respective VXLs are mainly required such as in the case of self-location estimation, and sends to a client a WLD when detailed VXL information is required such as in the case of map rendering. This allows for an efficient sending/receiving of map data.

[0164]

Note that a client may self-judge which one of a SWLD and a WLD is necessary, and request the server to send a SWLD or a WLD. Also, the server may judge which one of a SWLD and a WLD to send in accordance with the status of the client or a network.

[0165]

Next, a method will be described of switching the sending/receiving between a sparse world (SWLD) and a world (WLD).

[0166]

Whether to receive a WLD or a SWLD may be switched in accordance with the network bandwidth. FIG. 14 is a diagram showing an example operation in such case. For example, when a low-speed network is used that limits the usable network bandwidth, such as in a Long-Term Evolution (LTE) environment, a client accesses the server over a low-speed network (S321), and obtains the SWLD from the server as map information (S322). Meanwhile, when a high-speed network is used that has an adequately broad network bandwidth, such as in a WiFi environment, a client accesses the server over a high-speed network (S323), and obtains the WLD from the server (S324). This enables the client to obtain appropriate map information in accordance with the network bandwidth such client is using.

[0167]

More specifically, a client receives the SWLD over an LTE network when in outdoors, and obtains the WLD over a WiFi network when in indoors such as in a facility. This enables the client to obtain more detailed map information on indoor environment.

[0168]

As described above, a client may request for a WLD or a SWLD in accordance with the bandwidth of a network such client is using. Alternatively, the client may send to the server information indicating the bandwidth of a network such client is using, and the server may send to the client data (the WLD or the SWLD) suitable for such client in accordance with the information. Alternatively, the server may identify the network bandwidth the client is using,

and send to the client data (the WLD or the SWLD) suitable for such client.

[0169]

Also, whether to receive a WLD or a SWLD may be switched in accordance with the speed of traveling. FIG. 15 is a diagram showing an example operation in such case. For example, when traveling at a high speed (S331), a client receives the SWLD from the server (S332). Meanwhile, when traveling at a low speed (S333), the client receives the WLD from the server (S334). This enables the client to obtain map information suitable to the speed, while reducing the network bandwidth. More specifically, when traveling on an expressway, the client receives the SWLD with a small data amount, which enables the update of rough map information at an appropriate speed. Meanwhile, when traveling on a general road, the client receives the WLD, which enables the obtainment of more detailed map information.

[0170]

As described above, the client may request the server for a WLD or a SWLD in accordance with the traveling speed of such client. Alternatively, the client may send to the server information indicating the traveling speed of such client, and the server may send to the client data (the WLD or the SWLD) suitable to such client in accordance with the information. Alternatively, the server may identify the traveling speed of the client to send data (the WLD or the SWLD) suitable to such client.

[0171]

Also, the client may obtain, from the server, a SWLD first, from which the client may obtain a WLD of an important region. For example, when obtaining map information, the client first obtains a SWLD for rough map information, from which the client narrows to a region in which features such as buildings, signals, or persons appear at high frequency so that the client can

later obtain a WLD of such narrowed region. This enables the client to obtain detailed information on a necessary region, while reducing the amount of data received from the server.

[0172]

The server may also create from a WLD different SWLDs for the respective objects, and the client may receive SWLDs in accordance with the intended use. This reduces the network bandwidth. For example, the server recognizes persons or cars in a WLD in advance, and creates a SWLD of persons and a SWLD of cars. The client, when wishing to obtain information on persons around the client, receives the SWLD of persons, and when wising to obtain information on cars, receives the SWLD of cars. Such types of SWLDs may be distinguished by information (flag, or type, etc.) added to the header, etc.

[0173]

Next, the structure and the operation flow of the three-dimensional data encoding device (e.g., a server) according to the present embodiment will be described. FIG. 16 is a block diagram of three-dimensional data encoding device 400 according to the present embodiment. FIG. 17 is a flowchart of three-dimensional data encoding processes performed by three-dimensional data encoding device 400.

[0174]

Three-dimensional data encoding device 400 shown in FIG. 16 encodes input three-dimensional data 411, thereby generating encoded three-dimensional data 413 and encoded three-dimensional data 414, each being an encoded stream. Here, encoded three-dimensional data 413 is encoded three-dimensional data corresponding to a WLD, and encoded three-dimensional data 414 is encoded three-dimensional data corresponding to a SWLD. Such three-dimensional data encoding device 400 includes, obtainer 401, encoding region

determiner 402, SWLD extractor 403, WLD encoder 404, and SWLD encoder 405.

[0175]

First, as FIG. 17 shows, obtainer 401 obtains input three-dimensional data 411, which is point group data in a three-dimensional space (S401).

[0176]

Next, encoding region determiner 402 determines a current spatial region for encoding on the basis of a spatial region in which the point cloud data is present (S402).

[0177]

Next, SWLD extractor 403 defines the current spatial region as a WLD, and calculates the feature from each VXL included in the WLD. Then, SWLD extractor 403 extracts VXLs having an amount of features greater than or equal to a predetermined threshold, defines the extracted VXLs as FVXLs, and adds such FVXLs to a SWLD, thereby generating extracted three-dimensional data 412 (S403). Stated differently, extracted three-dimensional data 412 having an amount of features greater than or equal to the threshold is extracted from input three-dimensional data 411.

[0178]

Next, WLD encoder 404 encodes input three-dimensional data 411 corresponding to the WLD, thereby generating encoded three-dimensional data 413 corresponding to the WLD (S404). In so doing, WLD encoder 404 adds to the header of encoded three-dimensional data 413 information that distinguishes that such encoded three-dimensional data 413 is a stream including a WLD.

[0179]

SWLD encoder 405 encodes extracted three-dimensional data 412

corresponding to the SWLD, thereby generating encoded three-dimensional data 414 corresponding to the SWLD (S405). In so doing, SWLD encoder 405 adds to the header of encoded three-dimensional data 414 information that distinguishes that such encoded three-dimensional data 414 is a stream including a SWLD.

[0180]

Note that the process of generating encoded three-dimensional data 413 and the process of generating encoded three-dimensional data 414 may be performed in the reverse order. Also note that a part or all of these processes may be performed in parallel.

[0181]

A parameter "world_type" is defined, for example, as information added to each header of encoded three-dimensional data 413 and encoded three-dimensional data 414. world_type=0 indicates that a stream includes a WLD, and world_type=1 indicates that a stream includes a SWLD. An increased number of values may be further assigned to define a larger number of types, e.g., world_type=2. Also, one of encoded three-dimensional data 413 and encoded three-dimensional data 414 may include a specified flag. For example, encoded three-dimensional data 414 may be assigned with a flag indicating that such stream includes a SWLD. In such a case, the decoding device can distinguish whether such stream is a stream including a WLD or a stream including a SWLD in accordance with the presence/absence of the flag.

[0182]

Also, an encoding method used by WLD encoder 404 to encode a WLD may be different from an encoding method used by SWLD encoder 405 to encode a SWLD.

[0183]

For example, data of a SWLD is decimated, and thus can have a lower correlation with the neighboring data than that of a WLD. For this reason, of intra prediction and inter prediction, inter prediction may be more preferentially performed in an encoding method used for a SWLD than in an encoding method used for a WLD.

[0184]

Also, an encoding method used for a SWLD and an encoding method used for a WLD may represent three-dimensional positions differently. For example, three-dimensional coordinates may be used to represent the three-dimensional positions of FVXLs in a SWLD and an octree described below may be used to represent three-dimensional positions in a WLD, and vice versa.

[0185]

Also, SWLD encoder 405 performs encoding in a manner that encoded three-dimensional data 414 of a SWLD has a smaller data size than the data size of encoded three-dimensional data 413 of a WLD. A SWLD can have a lower inter-data correlation, for example, than that of a WLD as described above. This can lead to a decreased encoding efficiency, and thus to encoded three-dimensional data 414 having a larger data size than the data size of encoded three-dimensional data 413 of a WLD. When the data size of the resulting encoded three-dimensional data 414 is larger than the data size of encoded three-dimensional data 413 of a WLD, SWLD encoder 405 performs encoding again to re-generate encoded three-dimensional data 414 having a reduced data size.

[0186]

For example, SWLD extractor 403 re-generates extracted three-dimensional data 412 having a reduced number of keypoints to be extracted, and SWLD encoder 405 encodes such extracted three-dimensional data 412.

Alternatively, SWLD encoder 405 may perform more coarse quantization. More coarse quantization is achieved, for example, by rounding the data in the lowermost level in an octree structure described below.

[0187]

When failing to decrease the data size of encoded three-dimensional data 414 of the SWLD to smaller than the data size of encoded three-dimensional data 413 of the WLD, SWLD encoder 405 may not generate encoded three-dimensional data 414 of the SWLD. Alternatively, encoded three-dimensional data 413 of the WLD may be copied as encoded three-dimensional data 414 of the SWLD. Stated differently, encoded three-dimensional data 413 of the WLD may be used as it is as encoded three-dimensional data 414 of the SWLD.

[0188]

Next, the structure and the operation flow of the three-dimensional data decoding device (e.g., a client) according to the present embodiment will be described. FIG. 18 is a block diagram of three-dimensional data decoding device 500 according to the present embodiment. FIG. 19 is a flowchart of three-dimensional data decoding processes performed by three-dimensional data decoding device 500.

[0189]

Three-dimensional data decoding device 500 shown in FIG. 18 decodes encoded three-dimensional data 511, thereby generating decoded three-dimensional data 512 or decoded three-dimensional data 513. Encoded three-dimensional data 511 here is, for example, encoded three-dimensional data 413 or encoded three-dimensional data 414 generated by three-dimensional data encoding device 400.

[0190]

Such three-dimensional data decoding device 500 includes obtainer 501,

header analyzer 502, WLD decoder 503, and SWLD decoder 504.

[0191]

First, as FIG. 19 shows, obtainer 501 obtains encoded three-dimensional data 511 (S501). Next, header analyzer 502 analyzes the header of encoded three-dimensional data 511 to identify whether encoded three-dimensional data 511 is a stream including a WLD or a stream including a SWLD (S502). For example, the above-described parameter world_type is referred to in making such identification.

[0192]

When encoded three-dimensional data 511 is a stream including a WLD (Yes in S503), WLD decoder 503 decodes encoded three-dimensional data 511, thereby generating decoded three-dimensional data 512 of the WLD (S504). Meanwhile, when encoded three-dimensional data 511 is a stream including a SWLD (No in S503), SWLD decoder 504 decodes encoded three-dimensional data 511, thereby generating decoded three-dimensional data 513 of the SWLD (S505).

[0193]

Also, as in the case of the encoding device, a decoding method used by WLD decoder 503 to decode a WLD may be different from a decoding method used by SWLD decoder 504 to decode a SWLD. For example, of intra prediction and inter prediction, inter prediction may be more preferentially performed in a decoding method used for a SWLD than in a decoding method used for a WLD.

[0194]

Also, a decoding method used for a SWLD and a decoding method used for a WLD may represent three-dimensional positions differently. For example, three-dimensional coordinates may be used to represent the three-dimensional positions of FVXLs in a SWLD and an octree described below may be used to

represent three-dimensional positions in a WLD, and vice versa.

[0195]

Next, an octree representation will be described, which is a method of representing three-dimensional positions. VXL data included in three-dimensional data is converted into an octree structure before encoded. FIG. 20 is a diagram showing example VXLs in a WLD. FIG. 21 is a diagram showing an octree structure of the WLD shown in FIG. 20. An example shown in FIG. 20 illustrates three VXLs 1 to 3 that include point groups (hereinafter referred to as effective VXLs). As FIG. 21 shows, the octree structure is made of nodes and leaves. Each node has a maximum of eight nodes or leaves. Each leaf has VXL information. Here, of the leaves shown in FIG. 21, leaf 1, leaf 2, and leaf 3 represent VXL1, VXL2, and VXL3 shown in FIG. 20, respectively.

[0196]

More specifically, each node and each leaf correspond to a three-dimensional position. Node 1 corresponds to the entire block shown in FIG. 20. The block that corresponds to node 1 is divided into eight blocks. Of these eight blocks, blocks including effective VXLs are set as nodes, while the other blocks are set as leaves. Each block that corresponds to a node is further divided into eight nodes or leaves. These processes are repeated by the number of times that is equal to the number of levels in the octree structure. All blocks in the lowermost level are set as leaves.

[0197]

FIG. 22 is a diagram showing an example SWLD generated from the WLD shown in FIG. 20. VXL1 and VXL2 shown in FIG. 20 are judged as FVXL1 and FVXL2 as a result of feature extraction, and thus are added to the SWLD. Meanwhile, VXL3 is not judged as a FVXL, and thus is not added to the SWLD. FIG. 23 is a diagram showing an octree structure of the SWLD

shown in FIG. 22. In the octree structure shown in FIG. 23, leaf 3 corresponding to VXL3 shown in FIG. 21 is deleted. Consequently, node 3 shown in FIG. 21 has lost an effective VXL, and has changed to a leaf. As described above, a SWLD has a smaller number of leaves in general than a WLD does, and thus the encoded three-dimensional data of the SWLD is smaller than the encoded three-dimensional data of the WLD.

[0198]

The following describes variations of the present embodiment.

[0199]

For self-location estimation, for example, a client, being a vehicle-mounted device, etc., may receive a SWLD from the server to use such SWLD to estimate the self-location. Meanwhile, for obstacle detection, the client may detect obstacles by use of three-dimensional information on the periphery obtained by such client through various means including a distance sensor such as a rangefinder, as well as a stereo camera and a combination of a plurality of monocular cameras.

[0200]

In general, a SWLD is less likely to include VXL data on a flat region. As such, the server may hold a subsample world (subWLD) obtained by subsampling a WLD for detection of static obstacles, and send to the client the SWLD and the subWLD. This enables the client to perform self-location estimation and obstacle detection on the client's part, while reducing the network bandwidth.

[0201]

When the client renders three-dimensional map data at a high speed, map information having a mesh structure is more useful in some cases. As such, the server may generate a mesh from a WLD to hold it beforehand as a

mesh world (MWLD). For example, when wishing to perform coarse three-dimensional rendering, the client receives a MWLD, and when wishing to perform detailed three-dimensional rendering, the client receives a WLD. This reduces the network bandwidth.

[0202]

In the above description, the server sets, as FVXLs, VXLs having an amount of features greater than or equal to the threshold, but the server may calculate FVXLs by a different method. For example, the server may judge that a VXL, a VLM, a SPC, or a GOS that constitutes a signal, or an intersection, etc. as necessary for self-location estimation, driving assist, or self-driving, etc., and incorporate such VXL, VLM, SPC, or GOS into a SWLD as a FVXL, a FVLM, a FSPC, or a FGOS. Such judgment may be made manually. Also, FVXLs, etc. that have been set on the basis of an amount of features may be added to FVXLs, etc. obtained by the above method. Stated differently, SWLD extractor 403 may further extract, from input three-dimensional data 411, data corresponding to an object having a predetermined attribute as extracted three-dimensional data 412.

[0203]

Also, that a VXL, a VLM, a SPC, or a GOS is necessary for such intended usage may be labeled separately from the features. The server may separately hold, as an upper layer of a SWLD (e.g., a lane world), FVXLs of a signal or an intersection, etc. necessary for self-location estimation, driving assist, or self-driving, etc.

[0204]

The server may also add an attribute to VXLs in a WLD on a random access basis or on a predetermined unit basis. An attribute, for example, includes information indicating whether VXLs are necessary for self-location

estimation, or information indicating whether VXLs are important as traffic information such as a signal, or an intersection, etc. An attribute may also include a correspondence between VXLs and features (intersection, or road, etc.) in lane information (geographic data files (GDF), etc.).

[0205]

A method as described below may be used to update a WLD or a SWLD.

[0206]

Update information indicating changes, etc. in a person, a roadwork, or a tree line (for trucks) is uploaded to the server as point groups or meta data. The server updates a WLD on the basis of such uploaded information, and then updates a SWLD by use of the updated WLD.

[0207]

The client, when detecting a mismatch between the three-dimensional information such client has generated at the time of self-location estimation and the three-dimensional information received from the server, may send to the server the three-dimensional information such client has generated, together with an update notification. In such a case, the server updates the SWLD by use of the WLD. When the SWLD is not to be updated, the server judges that the WLD itself is old.

[0208]

In the above description, information that distinguishes whether an encoded stream is that of a WLD or a SWLD is added as header information of the encoded stream. However, when there are many types of worlds such as a mesh world and a lane world, information that distinguishes these types of the worlds may be added to header information. Also, when there are many SWLDs with different amounts of features, information that distinguishes the respective SWLDs may be added to header information.

[0209]

In the above description, a SWLD is constituted by FVXLs, but a SWLD may include VXLs that have not been judged as FVXLs. For example, a SWLD may include an adjacent VXL used to calculate the feature of a FVXL. This enables the client to calculate the feature of a FVXL when receiving a SWLD, even in the case where feature information is not added to each FVXL of the SWLD. In such a case, the SWLD may include information that distinguishes whether each VXL is a FVXL or a VXL.

[0210]

As described above, three-dimensional data encoding device 400 extracts, from input three-dimensional data 411 (first three-dimensional data), extracted three-dimensional data 412 (second three-dimensional data) having an amount of a feature greater than or equal to a threshold, and encodes extracted three-dimensional data 412 to generate encoded three-dimensional data 414 (first encoded three-dimensional data).

[0211]

This three-dimensional data encoding device 400 generates encoded three-dimensional data 414 that is obtained by encoding data having an amount of a feature greater than or equal to the threshold. This reduces the amount of data compared to the case where input three-dimensional data 411 is encoded as it is. Three-dimensional data encoding device 400 is thus capable of reducing the amount of data to be transmitted.

[0212]

Three-dimensional data encoding device 400 further encodes input three-dimensional data 411 to generate encoded three-dimensional data 413 (second encoded three-dimensional data).

[0213]

This three-dimensional data encoding device 400 enables selective transmission of encoded three-dimensional data 413 and encoded three-dimensional data 414, in accordance, for example, with the intended use, etc.

[0214]

Also, extracted three-dimensional data 412 is encoded by a first encoding method, and input three-dimensional data 411 is encoded by a second encoding method different from the first encoding method.

[0215]

This three-dimensional data encoding device 400 enables the use of an encoding method suitable for each of input three-dimensional data 411 and extracted three-dimensional data 412.

[0216]

Also, of intra prediction and inter prediction, the inter prediction is more preferentially performed in the first encoding method than in the second encoding method.

[0217]

This three-dimensional data encoding device 400 enables inter prediction to be more preferentially performed on extracted three-dimensional data 412 in which adjacent data items are likely to have low correlation.

[0218]

Also, the first encoding method and the second encoding method represent three-dimensional positions differently. For example, the second encoding method represents three-dimensional positions by octree, and the first encoding method represents three-dimensional positions by three-dimensional coordinates.

[0219]

This three-dimensional data encoding device 400 enables the use of a

more suitable method to represent the three-dimensional positions of three-dimensional data in consideration of the difference in the number of data items (the number of VXLs or FVXLs) included.

[0220]

Also, at least one of encoded three-dimensional data 413 and encoded three-dimensional data 414 includes an identifier indicating whether the encoded three-dimensional data is encoded three-dimensional data obtained by encoding input three-dimensional data 411 or encoded three-dimensional data obtained by encoding part of input three-dimensional data 411. Stated differently, such identifier indicates whether the encoded three-dimensional data is encoded three-dimensional data 413 of a WLD or encoded three-dimensional data 414 of a SWLD.

[0221]

This enables the decoding device to readily judge whether the obtained encoded three-dimensional data is encoded three-dimensional data 413 or encoded three-dimensional data 414.

[0222]

Also, three-dimensional data encoding device 400 encodes extracted three-dimensional data 412 in a manner that encoded three-dimensional data 414 has a smaller data amount than a data amount of encoded three-dimensional data 413.

[0223]

This three-dimensional data encoding device 400 enables encoded three-dimensional data 414 to have a smaller data amount than the data amount of encoded three-dimensional data 413.

[0224]

Also, three-dimensional data encoding device 400 further extracts data

corresponding to an object having a predetermined attribute from input three-dimensional data 411 as extracted three-dimensional data 412. The object having a predetermined attribute is, for example, an object necessary for self-location estimation, driving assist, or self-driving, etc., or more specifically, a signal, an intersection, etc.

[0225]

This three-dimensional data encoding device 400 is capable of generating encoded three-dimensional data 414 that includes data required by the decoding device.

[0226]

Also, three-dimensional data encoding device 400 (server) further sends, to a client, one of encoded three-dimensional data 413 and encoded three-dimensional data 414 in accordance with a status of the client.

[0227]

This three-dimensional data encoding device 400 is capable of sending appropriate data in accordance with the status of the client.

[0228]

Also, the status of the client includes one of a communication condition (e.g., network bandwidth) of the client and a traveling speed of the client.

[0229]

Also, three-dimensional data encoding device 400 further sends, to a client, one of encoded three-dimensional data 413 and encoded three-dimensional data 414 in accordance with a request from the client.

[0230]

This three-dimensional data encoding device 400 is capable of sending appropriate data in accordance with the request from the client.

[0231]

Also, three-dimensional data decoding device 500 according to the present embodiment decodes encoded three-dimensional data 413 or encoded three-dimensional data 414 generated by three-dimensional data encoding device 400 described above.

[0232]

Stated differently, three-dimensional data decoding device 500 decodes, by a first decoding method, encoded three-dimensional data 414 obtained by encoding extracted three-dimensional data 412 having an amount of a feature greater than or equal to a threshold, extracted three-dimensional data 412 having been extracted from input three-dimensional data 411. Three-dimensional data decoding device 500 also decodes, by a second decoding method, encoded three-dimensional data 413 obtained by encoding input three-dimensional data 411, the second decoding method being different from the first decoding method.

[0233]

This three-dimensional data decoding device 500 enables selective reception of encoded three-dimensional data 414 obtained by encoding data having an amount of a feature greater than or equal to the threshold and encoded three-dimensional data 413, in accordance, for example, with the intended use, etc. Three-dimensional data decoding device 500 is thus capable of reducing the amount of data to be transmitted. Such three-dimensional data decoding device 500 further enables the use of a decoding method suitable for each of input three-dimensional data 411 and extracted three-dimensional data 412.

[0234]

Also, of intra prediction and inter prediction, the inter prediction is more preferentially performed in the first decoding method than in the second

decoding method.

[0235]

This three-dimensional data decoding device 500 enables inter prediction to be more preferentially performed on the extracted three-dimensional data in which adjacent data items are likely to have low correlation.

[0236]

Also, the first decoding method and the second decoding method represent three-dimensional positions differently. For example, the second decoding method represents three-dimensional positions by octree, and the first decoding method represents three-dimensional positions by three-dimensional coordinates.

[0237]

This three-dimensional data decoding device 500 enables the use of a more suitable method to represent the three-dimensional positions of three-dimensional data in consideration of the difference in the number of data items (the number of VXLs or FVXLs) included.

[0238]

Also, at least one of encoded three-dimensional data 413 and encoded three-dimensional data 414 includes an identifier indicating whether the encoded three-dimensional data is encoded three-dimensional data obtained by encoding input three-dimensional data 411 or encoded three-dimensional data obtained by encoding part of input three-dimensional data 411. Three-dimensional data decoding device 500 refers to such identifier in identifying between encoded three-dimensional data 413 and encoded three-dimensional data 414.

[0239]

This three-dimensional data decoding device 500 is capable of readily

judging whether the obtained encoded three-dimensional data is encoded three-dimensional data 413 or encoded three-dimensional data 414.

[0240]

Three-dimensional data decoding device 500 further notifies a server of a status of the client (three-dimensional data decoding device 500). Three-dimensional data decoding device 500 receives one of encoded three-dimensional data 413 and encoded three-dimensional data 414 from the server, in accordance with the status of the client.

[0241]

This three-dimensional data decoding device 500 is capable of receiving appropriate data in accordance with the status of the client.

[0242]

Also, the status of the client includes one of a communication condition (e.g., network bandwidth) of the client and a traveling speed of the client.

[0243]

Three-dimensional data decoding device 500 further makes a request of the server for one of encoded three-dimensional data 413 and encoded three-dimensional data 414, and receives one of encoded three-dimensional data 413 and encoded three-dimensional data 414 from the server, in accordance with the request.

[0244]

This three-dimensional data decoding device 500 is capable of receiving appropriate data in accordance with the intended use.

[0245]

EMBODIMENT 3

The present embodiment will describe a method of transmitting/receiving three-dimensional data between vehicles. For example,

the three-dimensional data is transmitted/received between the own vehicle and the nearby vehicle.

[0246]

FIG. 24 is a block diagram of three-dimensional data creation device 620 according to the present embodiment. Such three-dimensional data creation device 620, which is included, for example, in the own vehicle, mergers first three-dimensional data 632 created by three-dimensional data creation device 620 with the received second three-dimensional data 635, thereby creating third three-dimensional data 636 having a higher density.

[0247]

Such three-dimensional data creation device 620 includes three-dimensional data creator 621, request range determiner 622, searcher 623, receiver 624, decoder 625, and merger 626.

[0248]

First, three-dimensional data creator 621 creates first three-dimensional data 632 by use of sensor information 631 detected by the sensor included in the own vehicle. Next, request range determiner 622 determines a request range, which is the range of a three-dimensional space, the data on which is insufficient in the created first three-dimensional data 632.

[0249]

Next, searcher 623 searches for the nearby vehicle having the three-dimensional data of the request range, and sends request range information 633 indicating the request range to nearby vehicle 601 having been searched out (S623). Next, receiver 624 receives encoded three-dimensional data 634, which is an encoded stream of the request range, from nearby vehicle 601 (S624). Note that searcher 623 may indiscriminately send requests to all vehicles included in a specified range to receive encoded three-dimensional data 634 from

a vehicle that has responded to the request. Searcher 623 may send a request not only to vehicles but also to an object such as a signal and a sign, and receive encoded three-dimensional data 634 from the object.

[0250]

Next, decoder 625 decodes the received encoded three-dimensional data 634, thereby obtaining second three-dimensional data 635. Next, merger 626 merges first three-dimensional data 632 with second three-dimensional data 635, thereby creating three-dimensional data 636 having a higher density.

[0251]

Next, the structure and operations of three-dimensional data transmission device 640 according to the present embodiment will be described. FIG. 25 is a block diagram of three-dimensional data transmission device 640.

[0252]

Three-dimensional data transmission device 640 is included, for example, in the above-described nearby vehicle. Three-dimensional data transmission device 640 processes fifth three-dimensional data 652 created by the nearby vehicle into sixth three-dimensional data 654 requested by the own vehicle, encodes sixth three-dimensional data 654 to generate encoded three-dimensional data 634, and sends encoded three-dimensional data 634 to the own vehicle.

[0253]

Three-dimensional data transmission device 640 includes three-dimensional data creator 641, receiver 642, extractor 643, encoder 644, and transmitter 645.

[0254]

First, three-dimensional data creator 641 creates fifth three-dimensional data 652 by use of sensor information 651 detected by the sensor

included in the nearby vehicle. Next, receiver 642 receives request range information 633 from the own vehicle.

[0255]

Next, extractor 643 extracts from fifth three-dimensional data 652 the three-dimensional data of the request range indicated by request range information 633, thereby processing fifth three-dimensional data 652 into sixth three-dimensional data 654. Next, encoder 644 encodes sixth three-dimensional data 654 to generate encoded three-dimensional data 643, which is an encoded stream. Then, transmitter 645 sends encoded three-dimensional data 634 to the own vehicle.

[0256]

Note that although an example case is described here in which the own vehicle includes three-dimensional data creation device 620 and the nearby vehicle includes three-dimensional data transmission device 640, each of the vehicles may include the functionality of both three-dimensional data creation device 620 and three-dimensional data transmission device 640.

[0257]

EMBODIMENT 4

The present embodiment describes operations performed in abnormal cases when self-location estimation is performed on the basis of a three-dimensional map.

[0258]

A three-dimensional map is expected to find its expanded use in self-driving of a vehicle and autonomous movement, etc. of a mobile object such as a robot and a flying object (e.g., a drone). Example means for enabling such autonomous movement include a method in which a mobile object travels in accordance with a three-dimensional map, while estimating its self-location on

the map (self-location estimation).

[0259]

The self-location estimation is enabled by matching a three-dimensional map with three-dimensional information on the surrounding of the own vehicle (hereinafter referred to as self-detected three-dimensional data) obtained by a sensor equipped in the own vehicle, such as a rangefinder (e.g., a LiDAR) and a stereo camera to estimate the location of the own vehicle on the three-dimensional map.

[0260]

As in the case of an HD map suggested by HERE Technologies, for example, a three-dimensional map may include not only a three-dimensional point cloud, but also two-dimensional map data such as information on the shapes of roads and intersections, or information that changes in real-time such as information on a traffic jam and an accident. A three-dimensional map includes a plurality of layers such as layers of three-dimensional data, two-dimensional data, and meta-data that changes in real-time, from among which the device can obtain or refer to only necessary data.

[0261]

Point cloud data may be a SWLD as described above, or may include point group data that is different from keypoints. The transmission/reception of point cloud data is basically carried out in one or more random access units.

[0262]

A method described below is used as a method of matching a three-dimensional map with self-detected three-dimensional data. For example, the device compares the shapes of the point groups in each other's point clouds, and determines that portions having a high degree of similarity among keypoints correspond to the same position. When the three-dimensional map is formed

by a SWLD, the device also performs matching by comparing the keypoints that form the SWLD with three-dimensional keypoints extracted from the self-detected three-dimensional data.

[0263]

Here, to enable highly accurate self-location estimation, the following needs to be satisfied: (A) the three-dimensional map and the self-detected three-dimensional data have been already obtained; and (B) their accuracies satisfy a predetermined requirement. However, one of (A) and (B) cannot be satisfied in abnormal cases such as ones described below.

[0264]

1. A three-dimensional map is unobtainable over communication.

[0265]

2. A three-dimensional map is not present, or a three-dimensional map having been obtained is corrupt.

[0266]

3. A sensor of the own vehicle has trouble, or the accuracy of the generated self-detected three-dimensional data is inadequate due to bad weather.

[0267]

The following describes operations to cope with such abnormal cases. The following description illustrates an example case of a vehicle, but the method described below is applicable to mobile objects on the whole that are capable of autonomous movement, such as a robot and a drone.

[0268]

The following describes the structure of the three-dimensional information processing device and its operation according to the present embodiment capable of coping with abnormal cases regarding a three-

dimensional map or self-detected three-dimensional data. FIG. 26 is a block diagram of an example structure of three-dimensional information processing device 700 according to the present embodiment.

[0269]

Three-dimensional information processing device 700 is equipped, for example, in a mobile object such as a car. As shown in FIG. 26, three-dimensional information processing device 700 includes three-dimensional map obtainer 701, self-detected data obtainer 702, abnormal case judgment unit 703, coping operation determiner 704, and operation controller 705.

[0270]

Note that three-dimensional information processing device 700 may include a non-illustrated two-dimensional or one-dimensional sensor that detects a structural object or a mobile object around the own vehicle, such as a camera capable of obtaining two-dimensional images and a sensor for one-dimensional data utilizing ultrasonic or laser. Three-dimensional information processing device 700 may also include a non-illustrated communication unit that obtains a three-dimensional map over a mobile communication network, such as 4G and 5G, or via inter-vehicle communication or road-to-vehicle communication.

[0271]

Three-dimensional map obtainer 701 obtains three-dimensional map 711 of the surroundings of the traveling route. For example, three-dimensional map obtainer 701 obtains three-dimensional map 711 over a mobile communication network, or via inter-vehicle communication or road-to-vehicle communication.

[0272]

Next, self-detected data obtainer 702 obtains self-detected three-

dimensional data 712 on the basis of sensor information. For example, self-detected data obtainer 702 generates self-detected three-dimensional data 712 on the basis of the sensor information obtained by a sensor equipped in the own vehicle.

[0273]

Next, abnormal case judgment unit 703 conducts a predetermined check of at least one of obtained three-dimensional map 711 and self-detected three-dimensional data 712 to detect an abnormal case. Stated differently, abnormal case judgment unit 703 judges whether at least one of obtained three-dimensional map 711 and self-detected three-dimensional data 712 is abnormal.

[0274]

When the abnormal case is detected, coping operation determiner 704 determines a coping operation to cope with such abnormal case. Next, operation controller 705 controls the operation of each of the processing units necessary to perform the coping operation.

[0275]

Meanwhile, when no abnormal case is detected, three-dimensional information processing device 700 terminates the process.

[0276]

Also, three-dimensional information processing device 700 estimates the location of the vehicle equipped with three-dimensional information processing device 700, using three-dimensional map 711 and self-detected three-dimensional data 712. Next, three-dimensional information processing device 700 performs the automatic operation of the vehicle by use of the estimated location of the vehicle.

[0277]

As described above, three-dimensional information processing device

700 obtains, via a communication channel, map data (three-dimensional map 711) that includes first three-dimensional position information. The first three-dimensional position information includes, for example, a plurality of random access units, each of which is an assembly of at least one subspace and is individually decodable, the at least one subspace having three-dimensional coordinates information and serving as a unit in which each of the plurality of random access units is encoded. The first three-dimensional position information is, for example, data (SWLD) obtained by encoding keypoints, each of which has an amount of a three-dimensional feature greater than or equal to a predetermined threshold.

[0278]

Three-dimensional information processing device 700 also generates second three-dimensional position information (self-detected three-dimensional data 712) from information detected by a sensor. Three-dimensional information processing device 700 then judges whether one of the first three-dimensional position information and the second three-dimensional position information is abnormal by performing, on one of the first three-dimensional position information and the second three-dimensional position information, a process of judging whether an abnormality is present.

[0279]

Three-dimensional information processing device 700 determines a coping operation to cope with the abnormality when one of the first three-dimensional position information and the second three-dimensional position information is judged to be abnormal. Three-dimensional information processing device 700 then executes a control that is required to perform the coping operation.

[0280]

This structure enables three-dimensional information processing device 700 to detect an abnormality regarding one of the first three-dimensional position information and the second three-dimensional position information, and to perform a coping operation therefor.

[0281]

EMBODIMENT 5

The present embodiment describes a method, etc. of transmitting three-dimensional data to a following vehicle.

[0282]

FIG. 27 is a block diagram of an exemplary structure of three-dimensional data creation device 810 according to the present embodiment. Such three-dimensional data creation device 810 is equipped, for example, in a vehicle. Three-dimensional data creation device 810 transmits and receives three-dimensional data to and from an external cloud-based traffic monitoring system, a preceding vehicle, or a following vehicle, and creates and stores three-dimensional data.

[0283]

Three-dimensional data creation device 810 includes data receiver 811, communication unit 812, reception controller 813, format converter 814, a plurality of sensors 815, three-dimensional data creator 816, three-dimensional data synthesizer 817, three-dimensional data storage 818, communication unit 819, transmission controller 820, format converter 821, and data transmitter 822.

[0284]

Data receiver 811 receives three-dimensional data 831 from a cloud-based traffic monitoring system or a preceding vehicle. Three-dimensional data 831 includes, for example, information on a region undetectable by sensors

815 of the own vehicle, such as a point cloud, visible light video, depth information, sensor position information, and speed information.

[0285]

Communication unit 812 communicates with the cloud-based traffic monitoring system or the preceding vehicle to transmit a data transmission request, etc. to the cloud-based traffic monitoring system or the preceding vehicle.

[0286]

Reception controller 813 exchanges information, such as information on supported formats, with a communications partner via communication unit 812 to establish communication with the communications partner.

[0287]

Format converter 814 applies format conversion, etc. on three-dimensional data 831 received by data receiver 811 to generate three-dimensional data 832. Format converter 814 also decompresses or decodes three-dimensional data 831 when three-dimensional data 831 is compressed or encoded.

[0288]

A plurality of sensors 815 are a group of sensors, such as visible light cameras and infrared cameras, that obtain information on the outside of the vehicle and generate sensor information 833. Sensor information 833 is, for example, three-dimensional data such as a point cloud (point group data), when sensors 815 are laser sensors such as LiDARs. Note that a single sensor may serve as a plurality of sensors 815.

[0289]

Three-dimensional data creator 816 generates three-dimensional data 834 from sensor information 833. Three-dimensional data 834 includes, for

example, information such as a point cloud, visible light video, depth information, sensor position information, and speed information.

[0290]

Three-dimensional data synthesizer 817 synthesizes three-dimensional data 834 created on the basis of sensor information 833 of the own vehicle with three-dimensional data 832 created by the cloud-based traffic monitoring system or the preceding vehicle, etc., thereby forming three-dimensional data 835 of a space that includes the space ahead of the preceding vehicle undetectable by sensors 815 of the own vehicle.

[0291]

Three-dimensional data storage 818 stores generated three-dimensional data 835, etc.

[0292]

Communication unit 819 communicates with the cloud-based traffic monitoring system or the following vehicle to transmit a data transmission request, etc. to the cloud-based traffic monitoring system or the following vehicle.

[0293]

Transmission controller 820 exchanges information such as information on supported formats with a communications partner via communication unit 819 to establish communication with the communications partner. Transmission controller 820 also determines a transmission region, which is a space of the three-dimensional data to be transmitted, on the basis of three-dimensional data formation information on three-dimensional data 832 generated by three-dimensional data synthesizer 817 and the data transmission request from the communications partner.

[0294]

More specifically, transmission controller 820 determines a transmission

region that includes the space ahead of the own vehicle undetectable by a sensor of the following vehicle, in response to the data transmission request from the cloud-based traffic monitoring system or the following vehicle. Transmission controller 820 judges, for example, whether a space is transmittable or whether the already transmitted space includes an update, on the basis of the three-dimensional data formation information to determine a transmission region. For example, transmission controller 820 determines, as a transmission region, a region that is: a region specified by the data transmission request; and a region, corresponding three-dimensional data 835 of which is present. Transmission controller 820 then notifies format converter 821 of the format supported by the communications partner and the transmission region.

[0295]

Of three-dimensional data 835 stored in three-dimensional data storage 818, format converter 821 converts three-dimensional data 836 of the transmission region into the format supported by the receiver end to generate three-dimensional data 837. Note that format converter 821 may compress or encode three-dimensional data 837 to reduce the data amount.

[0296]

Data transmitter 822 transmits three-dimensional data 837 to the cloud-based traffic monitoring system or the following vehicle. Such three-dimensional data 837 includes, for example, information on a blind spot, which is a region hidden from view of the following vehicle, such as a point cloud ahead of the own vehicle, visible light video, depth information, and sensor position information.

[0297]

Note that an example has been described in which format converter 814 and format converter 821 perform format conversion, etc., but format conversion

may not be performed.

[0298]

With the above structure, three-dimensional data creation device 810 obtains, from an external device, three-dimensional data 831 of a region undetectable by sensors 815 of the own vehicle, and synthesizes three-dimensional data 831 with three-dimensional data 834 that is based on sensor information 833 detected by sensors 815 of the own vehicle, thereby generating three-dimensional data 835. Three-dimensional data creation device 810 is thus capable of generating three-dimensional data of a range undetectable by sensors 815 of the own vehicle.

[0299]

Three-dimensional data creation device 810 is also capable of transmitting, to the cloud-based traffic monitoring system or the following vehicle, etc., three-dimensional data of a space that includes the space ahead of the own vehicle undetectable by a sensor of the following vehicle, in response to the data transmission request from the cloud-based traffic monitoring system or the following vehicle.

[0300]

EMBODIMENT 6

In embodiment 5, an example is described in which a client device of a vehicle or the like transmits three-dimensional data to another vehicle or a server such as a cloud-based traffic monitoring system. In the present embodiment, a client device transmits sensor information obtained through a sensor to a server or a client device.

[0301]

A structure of a system according to the present embodiment will first be described. FIG. 28 is a diagram showing the structure of a

transmission/reception system of a three-dimensional map and sensor information according to the present embodiment. This system includes server 901, and client devices 902A and 902B. Note that client devices 902A and 902B are also referred to as client device 902 when no particular distinction is made therebetween.

[0302]

Client device 902 is, for example, a vehicle-mounted device equipped in a mobile object such as a vehicle. Server 901 is, for example, a cloud-based traffic monitoring system, and is capable of communicating with the plurality of client devices 902.

[0303]

Server 901 transmits the three-dimensional map formed by a point cloud to client device 902. Note that a structure of the three-dimensional map is not limited to a point cloud, and may also be another structure expressing three-dimensional data such as a mesh structure.

[0304]

Client device 902 transmits the sensor information obtained by client device 902 to server 901. The sensor information includes, for example, at least one of information obtained by LiDAR, a visible light image, an infrared image, a depth image, sensor position information, or sensor speed information.

[0305]

The data to be transmitted and received between server 901 and client device 902 may be compressed in order to reduce data volume, and may also be transmitted uncompressed in order to maintain data precision. When compressing the data, it is possible to use a three-dimensional compression method on the point cloud based on, for example, an octree structure. It is possible to use a two-dimensional image compression method on the visible light

image, the infrared image, and the depth image. The two-dimensional image compression method is, for example, MPEG-4 AVC or HEVC standardized by MPEG.

[0306]

Server 901 transmits the three-dimensional map managed by server 901 to client device 902 in response to a transmission request for the three-dimensional map from client device 902. Note that server 901 may also transmit the three-dimensional map without waiting for the transmission request for the three-dimensional map from client device 902. For example, server 901 may broadcast the three-dimensional map to at least one client device 902 located in a predetermined space. Server 901 may also transmit the three-dimensional map suited to a position of client device 902 at fixed time intervals to client device 902 that has received the transmission request once. Server 901 may also transmit the three-dimensional map managed by server 901 to client device 902 every time the three-dimensional map is updated.

[0307]

Client device 902 sends the transmission request for the three-dimensional map to server 901. For example, when client device 902 wants to perform the self-location estimation during traveling, client device 902 transmits the transmission request for the three-dimensional map to server 901.

[0308]

Note that in the following cases, client device 902 may send the transmission request for the three-dimensional map to server 901. Client device 902 may send the transmission request for the three-dimensional map to server 901 when the three-dimensional map stored by client device 902 is old. For example, client device 902 may send the transmission request for the three-dimensional map to server 901 when a fixed period has passed since the three-

dimensional map is obtained by client device 902.

[0309]

Client device 902 may also send the transmission request for the three-dimensional map to server 901 before a fixed time when client device 902 exits a space shown in the three-dimensional map stored by client device 902. For example, client device 902 may send the transmission request for the three-dimensional map to server 901 when client device 902 is located within a predetermined distance from a boundary of the space shown in the three-dimensional map stored by client device 902. When a movement path and a movement speed of client device 902 are understood, a time when client device 902 exits the space shown in the three-dimensional map stored by client device 902 may be predicted based on the movement path and the movement speed of client device 902.

[0310]

Client device 902 may also send the transmission request for the three-dimensional map to server 901 when an error during alignment of the three-dimensional data and the three-dimensional map created from the sensor information by client device 902 is at least at a fixed level.

[0311]

Client device 902 transmits the sensor information to server 901 in response to a transmission request for the sensor information from server 901. Note that client device 902 may transmit the sensor information to server 901 without waiting for the transmission request for the sensor information from server 901. For example, client device 902 may periodically transmit the sensor information during a fixed period when client device 902 has received the transmission request for the sensor information from server 901 once. Client device 902 may determine that there is a possibility of a change in the three-

dimensional map of a surrounding area of client device 902 having occurred, and transmit this information and the sensor information to server 901, when the error during alignment of the three-dimensional data created by client device 902 based on the sensor information and the three-dimensional map obtained from server 901 is at least at the fixed level.

[0312]

Server 901 sends a transmission request for the sensor information to client device 902. For example, server 901 receives position information, such as GPS information, about client device 902 from client device 902. Server 901 sends the transmission request for the sensor information to client device 902 in order to generate a new three-dimensional map, when it is determined that client device 902 is approaching a space in which the three-dimensional map managed by server 901 contains little information, based on the position information about client device 902. Server 901 may also send the transmission request for the sensor information, when wanting to (i) update the three-dimensional map, (ii) check road conditions during snowfall, a disaster, or the like, or (iii) check traffic congestion conditions, accident/incident conditions, or the like.

[0313]

Client device 902 may set an amount of data of the sensor information to be transmitted to server 901 in accordance with communication conditions or bandwidth during reception of the transmission request for the sensor information to be received from server 901. Setting the amount of data of the sensor information to be transmitted to server 901 is, for example, increasing/reducing the data itself or appropriately selecting a compression method.

[0314]

FIG. 29 is a block diagram showing an example structure of client device 902. Client device 902 receives the three-dimensional map formed by a point cloud and the like from server 901, and estimates a self-location of client device 902 using the three-dimensional map created based on the sensor information of client device 902. Client device 902 transmits the obtained sensor information to server 901.

[0315]

Client device 902 includes data receiver 1011, communication unit 1012, reception controller 1013, format converter 1014, sensors 1015, three-dimensional data creator 1016, three-dimensional image processor 1017, three-dimensional data storage 1018, format converter 1019, communication unit 1020, transmission controller 1021, and data transmitter 1022.

[0316]

Data receiver 1011 receives three-dimensional map 1031 from server 901. Three-dimensional map 1031 is data that includes a point cloud such as a WLD or a SWLD. Three-dimensional map 1031 may include compressed data or uncompressed data.

[0317]

Communication unit 1012 communicates with server 901 and transmits a data transmission request (e.g. transmission request for three-dimensional map) to server 901.

[0318]

Reception controller 1013 exchanges information, such as information on supported formats, with a communications partner via communication unit 1012 to establish communication with the communications partner.

[0319]

Format converter 1014 performs a format conversion and the like on

three-dimensional map 1031 received by data receiver 1011 to generate three-dimensional map 1032. Format converter 1014 also performs a decompression or decoding process when three-dimensional map 1031 is compressed or encoded. Note that format converter 1014 does not perform the decompression or decoding process when three-dimensional map 1031 is uncompressed data.

[0320]

Sensors 815 are a group of sensors, such as LiDARs, visible light cameras, infrared cameras, or depth sensors that obtain information about the outside of a vehicle equipped with client device 902, and generate sensor information 1033. Sensor information 1033 is, for example, three-dimensional data such as a point cloud (point group data) when sensors 1015 are laser sensors such as LiDARs. Note that a single sensor may serve as sensors 1015.

[0321]

Three-dimensional data creator 1016 generates three-dimensional data 1034 of a surrounding area of the own vehicle based on sensor information 1033. For example, three-dimensional data creator 1016 generates point cloud data with color information on the surrounding area of the own vehicle using information obtained by LiDAR and visible light video obtained by a visible light camera.

[0322]

Three-dimensional image processor 1017 performs a self-location estimation process and the like of the own vehicle, using (i) the received three-dimensional map 1032 such as a point cloud, and (ii) three-dimensional data 1034 of the surrounding area of the own vehicle generated using sensor information 1033. Note that three-dimensional image processor 1017 may generate three-dimensional data 1035 about the surroundings of the own vehicle by merging three-dimensional map 1032 and three-dimensional data 1034, and

may perform the self-location estimation process using the created three-dimensional data 1035.

[0323]

Three-dimensional data storage 1018 stores three-dimensional map 1032, three-dimensional data 1034, three-dimensional data 1035, and the like.

[0324]

Format converter 1019 generates sensor information 1037 by converting sensor information 1033 to a format supported by a receiver end. Note that format converter 1019 may reduce the amount of data by compressing or encoding sensor information 1037. Format converter 1019 may omit this process when format conversion is not necessary. Format converter 1019 may also control the amount of data to be transmitted in accordance with a specified transmission range.

[0325]

Communication unit 1020 communicates with server 901 and receives a data transmission request (transmission request for sensor information) and the like from server 901.

[0326]

Transmission controller 1021 exchanges information, such as information on supported formats, with a communications partner via communication unit 1020 to establish communication with the communications partner.

[0327]

Data transmitter 1022 transmits sensor information 1037 to server 901. Sensor information 1037 includes, for example, information obtained through sensors 1015, such as information obtained by LiDAR, a luminance image obtained by a visible light camera, an infrared image obtained by an infrared

camera, a depth image obtained by a depth sensor, sensor position information, and sensor speed information.

[0328]

A structure of server 901 will be described next. FIG. 30 is a block diagram showing an example structure of server 901. Server 901 transmits sensor information from client device 902 and creates three-dimensional data based on the received sensor information. Server 901 updates the three-dimensional map managed by server 901 using the created three-dimensional data. Server 901 transmits the updated three-dimensional map to client device 902 in response to a transmission request for the three-dimensional map from client device 902.

[0329]

Server 901 includes data receiver 1111, communication unit 1112, reception controller 1113, format converter 1114, three-dimensional data creator 1116, three-dimensional data merger 1117, three-dimensional data storage 1118, format converter 1119, communication unit 1120, transmission controller 1121, and data transmitter 1122.

[0330]

Data receiver 1111 receives sensor information 1037 from client device 902. Sensor information 1037 includes, for example, information obtained by LiDAR, a luminance image obtained by a visible light camera, an infrared image obtained by an infrared camera, a depth image obtained by a depth sensor, sensor position information, sensor speed information, and the like.

[0331]

Communication unit 1112 communicates with client device 902 and transmits a data transmission request (e.g. transmission request for sensor information) and the like to client device 902.

[0332]

Reception controller 1113 exchanges information, such as information on supported formats, with a communications partner via communication unit 1112 to establish communication with the communications partner.

[0333]

Format converter 1114 generates sensor information 1132 by performing a decompression or decoding process when the received sensor information 1037 is compressed or encoded. Note that format converter 1114 does not perform the decompression or decoding process when sensor information 1037 is uncompressed data.

[0334]

Three-dimensional data creator 1116 generates three-dimensional data 1134 of a surrounding area of client device 902 based on sensor information 1132. For example, three-dimensional data creator 1116 generates point cloud data with color information on the surrounding area of client device 902 using information obtained by LiDAR and visible light video obtained by a visible light camera.

[0335]

Three-dimensional data merger 1117 updates three-dimensional map 1135 by merging three-dimensional data 1134 created based on sensor information 1132 with three-dimensional map 1135 managed by server 901.

[0336]

Three-dimensional data storage 1118 stores three-dimensional map 1135 and the like.

[0337]

Format converter 1119 generates three-dimensional map 1031 by converting three-dimensional map 1135 to a format supported by the receiver

end. Note that format converter 1119 may reduce the amount of data by compressing or encoding three-dimensional map 1135. Format converter 1119 may omit this process when format conversion is not necessary. Format converter 1119 may also control the amount of data to be transmitted in accordance with a specified transmission range.

[0338]

Communication unit 1120 communicates with client device 902 and receives a data transmission request (transmission request for three-dimensional map) and the like from client device 902.

[0339]

Transmission controller 1121 exchanges information, such as information on supported formats, with a communications partner via communication unit 1120 to establish communication with the communications partner.

[0340]

Data transmitter 1122 transmits three-dimensional map 1031 to client device 902. Three-dimensional map 1031 is data that includes a point cloud such as a WLD or a SWLD. Three-dimensional map 1031 may include one of compressed data and uncompressed data.

[0341]

An operational flow of client device 902 will be described next. FIG. 31 is a flowchart of an operation when client device 902 obtains the three-dimensional map.

[0342]

Client device 902 first requests server 901 to transmit the three-dimensional map (point cloud, etc.) (S1001). At this point, by also transmitting the position information about client device 902 obtained through GPS and the

like, client device 902 may also request server 901 to transmit a three-dimensional map relating to this position information.

[0343]

Client device 902 next receives the three-dimensional map from server 901 (S1002). When the received three-dimensional map is compressed data, client device 902 decodes the received three-dimensional map and generates an uncompressed three-dimensional map (S1003).

[0344]

Client device 902 next creates three-dimensional data 1034 of the surrounding area of client device 902 using sensor information 1033 obtained by sensors 1015 (S1004). Client device 902 next estimates the self-location of client device 902 using three-dimensional map 1032 received from server 901 and three-dimensional data 1034 created using sensor information 1033 (S1005).

[0345]

FIG. 32 is a flowchart of an operation when client device 902 transmits the sensor information. Client device 902 first receives a transmission request for the sensor information from server 901 (S1011). Client device 902 that has received the transmission request transmits sensor information 1037 to server 901 (S1012). Note that client device 902 may generate sensor information 1037 by compressing each piece of information using a compression method suited to each piece of information, when sensor information 1033 includes a plurality of pieces of information obtained by sensors 1015.

[0346]

An operational flow of server 901 will be described next. FIG. 33 is a flowchart of an operation when server 901 obtains the sensor information. Server 901 first requests client device 902 to transmit the sensor information (S1021). Server 901 next receives sensor information 1037 transmitted from

client device 902 in accordance with the request (S1022). Server 901 next creates three-dimensional data 1134 using the received sensor information 1037 (S1023). Server 901 next reflects the created three-dimensional data 1134 in three-dimensional map 1135 (S1024).

[0347]

FIG. 34 is a flowchart of an operation when server 901 transmits the three-dimensional map. Server 901 first receives a transmission request for the three-dimensional map from client device 902 (S1031). Server 901 that has received the transmission request for the three-dimensional map transmits the three-dimensional map to client device 902 (S1032). At this point, server 901 may extract a three-dimensional map of a vicinity of client device 902 along with the position information about client device 902, and transmit the extracted three-dimensional map. Server 901 may compress the three-dimensional map formed by a point cloud using, for example, an octree structure compression method, and transmit the compressed three-dimensional map.

[0348]

Hereinafter, variations of the present embodiment will be described.

[0349]

Server 901 creates three-dimensional data 1134 of a vicinity of a position of client device 902 using sensor information 1037 received from client device 902. Server 901 next calculates a difference between three-dimensional data 1134 and three-dimensional map 1135, by matching the created three-dimensional data 1134 with three-dimensional map 1135 of the same area managed by server 901. Server 901 determines that a type of anomaly has occurred in the surrounding area of client device 902, when the difference is greater than or equal to a predetermined threshold. For example, it is conceivable that a large difference occurs between three-dimensional map 1135

managed by server 901 and three-dimensional data 1134 created based on sensor information 1037, when land subsidence and the like occurs due to a natural disaster such as an earthquake.

[0350]

Sensor information 1037 may include information indicating at least one of a sensor type, a sensor performance, and a sensor model number. Sensor information 1037 may also be appended with a class ID and the like in accordance with the sensor performance. For example, when sensor information 1037 is obtained by LiDAR, it is conceivable to assign identifiers to the sensor performance. A sensor capable of obtaining information with precision in units of several millimeters is class 1, a sensor capable of obtaining information with precision in units of several centimeters is class 2, and a sensor capable of obtaining information with precision in units of several meters is class 3. Server 901 may estimate sensor performance information and the like from a model number of client device 902. For example, when client device 902 is equipped in a vehicle, server 901 may determine sensor specification information from a type of the vehicle. In this case, server 901 may obtain information on the type of the vehicle in advance, and the information may also be included in the sensor information. Server 901 may change a degree of correction with respect to three-dimensional data 1134 created using sensor information 1037, using the obtained sensor information 1037. For example, when the sensor performance is high in precision (class 1), server 901 does not correct three-dimensional data 1134. When the sensor performance is low in precision (class 3), server 901 corrects three-dimensional data 1134 in accordance with the precision of the sensor. For example, server 901 increases the degree (intensity) of correction with a decrease in the precision of the sensor.

[0351]

Server 901 may simultaneously send the transmission request for the sensor information to the plurality of client devices 902 in a certain space. Server 901 does not need to use all of the sensor information for creating three-dimensional data 1134 and may, for example, select sensor information to be used in accordance with the sensor performance, when having received a plurality of pieces of sensor information from the plurality of client devices 902. For example, when updating three-dimensional map 1135, server 901 may select high-precision sensor information (class 1) from among the received plurality of pieces of sensor information, and create three-dimensional data 1134 using the selected sensor information.

[0352]

Server 901 is not limited to only being a server such as a cloud-based traffic monitoring system, and may also be another (vehicle-mounted) client device. FIG. 35 is a diagram of a system structure in this case.

[0353]

For example, client device 902C sends a transmission request for sensor information to client device 902A located nearby, and obtains the sensor information from client device 902A. Client device 902C then creates three-dimensional data using the obtained sensor information of client device 902A, and updates a three-dimensional map of client device 902C. This enables client device 902C to generate a three-dimensional map of a space that can be obtained from client device 902A, and fully utilize the performance of client device 902C. For example, such a case is conceivable when client device 902C has high performance.

[0354]

In this case, client device 902A that has provided the sensor information is given rights to obtain the high-precision three-dimensional map generated by

client device 902C. Client device 902A receives the high-precision three-dimensional map from client device 902C in accordance with these rights.

[0355]

Server 901 may send the transmission request for the sensor information to the plurality of client devices 902 (client device 902A and client device 902B) located nearby client device 902C. When a sensor of client device 902A or client device 902B has high performance, client device 902C is capable of creating the three-dimensional data using the sensor information obtained by this high-performance sensor.

[0356]

FIG. 36 is a block diagram showing a functionality structure of server 901 and client device 902. Server 901 includes, for example, three-dimensional map compression/decoding processor 1201 that compresses and decodes the three-dimensional map and sensor information compression/decoding processor 1202 that compresses and decodes the sensor information.

[0357]

Client device 902 includes three-dimensional map decoding processor 1211 and sensor information compression processor 1212. Three-dimensional map decoding processor 1211 receives encoded data of the compressed three-dimensional map, decodes the encoded data, and obtains the three-dimensional map. Sensor information compression processor 1212 compresses the sensor information itself instead of the three-dimensional data created using the obtained sensor information, and transmits the encoded data of the compressed sensor information to server 901. With this structure, client device 902 does not need to internally store a processor that performs a process for compressing the three-dimensional data of the three-dimensional map (point cloud, etc.), as long as client device 902 internally stores a processor that performs a process

for decoding the three-dimensional map (point cloud, etc.). This makes it possible to limit costs, power consumption, and the like of client device 902.

[0358]

As stated above, client device 902 according to the present embodiment is equipped in the mobile object, and creates three-dimensional data 1034 of a surrounding area of the mobile object using sensor information 1033 that is obtained through sensor 1015 equipped in the mobile object and indicates a surrounding condition of the mobile object. Client device 902 estimates a self-location of the mobile object using the created three-dimensional data 1034. Client device 902 transmits the obtained sensor information 1033 to server 901 or another mobile object.

[0359]

This enables client device 902 to transmit sensor information 1033 to server 901 or the like. This makes it possible to further reduce the amount of transmission data compared to when transmitting the three-dimensional data. Since there is no need for client device 902 to perform processes such as compressing or encoding the three-dimensional data, it is possible to reduce the processing amount of client device 902. As such, client device 902 is capable of reducing the amount of data to be transmitted or simplifying the structure of the device.

[0360]

Client device 902 further transmits the transmission request for the three-dimensional map to server 901 and receives three-dimensional map 1031 from server 901. In the estimating of the self-location, client device 902 estimates the self-location using three-dimensional data 1034 and three-dimensional map 1032.

[0361]

Sensor information 1034 includes at least one of information obtained by a laser sensor, a luminance image, an infrared image, a depth image, sensor position information, or sensor speed information.

[0362]

Sensor information 1033 includes information that indicates a performance of the sensor.

[0363]

Client device 902 encodes or compresses sensor information 1033, and in the transmitting of the sensor information, transmits sensor information 1037 that has been encoded or compressed to server 901 or another mobile object 902. This enables client device 902 to reduce the amount of data to be transmitted.

[0364]

For example, client device 902 includes a processor and memory. The processor performs the above processes using the memory.

[0365]

Server 901 according to the present embodiment is capable of communicating with client device 902 equipped in the mobile object, and receives sensor information 1037 that is obtained through sensor 1015 equipped in the mobile object and indicates a surrounding condition of the mobile object. Server 901 creates three-dimensional data 1134 of a surrounding area of the mobile object using the received sensor information 1037.

[0366]

With this, server 901 creates three-dimensional data 1134 using sensor information 1037 transmitted from client device 902. This makes it possible to further reduce the amount of transmission data compared to when client device 902 transmits the three-dimensional data. Since there is no need for client

device 902 to perform processes such as compressing or encoding the three-dimensional data, it is possible to reduce the processing amount of client device 902. As such, server 901 is capable of reducing the amount of data to be transmitted or simplifying the structure of the device.

[0367]

Server 901 further transmits a transmission request for the sensor information to client device 902.

[0368]

Server 901 further updates three-dimensional map 1135 using the created three-dimensional data 1134, and transmits three-dimensional map 1135 to client device 902 in response to the transmission request for three-dimensional map 1135 from client device 902.

[0369]

Sensor information 1037 includes at least one of information obtained by a laser sensor, a luminance image, an infrared image, a depth image, sensor position information, or sensor speed information.

[0370]

Sensor information 1037 includes information that indicates a performance of the sensor.

[0371]

Server 901 further corrects the three-dimensional data in accordance with the performance of the sensor. This enables the three-dimensional data creation method to improve the quality of the three-dimensional data.

[0372]

In the receiving of the sensor information, server 901 receives a plurality of pieces of sensor information 1037 received from a plurality of client devices 902, and selects sensor information 1037 to be used in the creating of three-

dimensional data 1134, based on a plurality of pieces of information that each indicates the performance of the sensor included in the plurality of pieces of sensor information 1037. This enables server 901 to improve the quality of three-dimensional data 1134.

[0373]

Server 901 decodes or decompresses the received sensor information 1037, and creates three-dimensional data 1134 using sensor information 1132 that has been decoded or decompressed. This enables server 901 to reduce the amount of data to be transmitted.

[0374]

For example, server 901 includes a processor and memory. The processor performs the above processes using the memory.

[0375]

EMBODIMENT 7

In the present embodiment, three-dimensional data encoding and decoding methods using an inter prediction process will be described.

[0376]

FIG. 37 is a block diagram of three-dimensional data encoding device 1300 according to the present embodiment. This three-dimensional data encoding device 1300 generates an encoded bitstream (hereinafter, also simply referred to as bitstream) that is an encoded signal, by encoding three-dimensional data. As illustrated in FIG. 37, three-dimensional data encoding device 1300 includes divider 1301, subtractor 1302, transformer 1303, quantizer 1304, inverse quantizer 1305, inverse transformer 1306, adder 1307, reference volume memory 1308, intra predictor 1309, reference space memory 1310, inter predictor 1311, prediction controller 1312, and entropy encoder 1313.

[0377]

Divider 1301 divides a plurality of volumes (VLMs) that are encoding units of each space (SPC) included in the three-dimensional data. Divider 1301 makes an octree representation (make into an octree) of voxels in each volume. Note that divider 1301 may make the spaces into an octree representation with the spaces having the same size as the volumes. Divider 1301 may also append information (depth information, etc.) necessary for making the octree representation to a header and the like of a bitstream.

[0378]

Subtractor 1302 calculates a difference between a volume (encoding target volume) outputted by divider 1301 and a predicted volume generated through intra prediction or inter prediction, which will be described later, and outputs the calculated difference to transformer 1303 as a prediction residual. FIG. 38 is a diagram showing an example calculation of the prediction residual. Note that bit sequences of the encoding target volume and the predicted volume shown here are, for example, position information indicating positions of three-dimensional points included in the volumes.

[0379]

Hereinafter, a scan order of an octree representation and voxels will be described. A volume is encoded after being converted into an octree structure (made into an octree). The octree structure includes nodes and leaves. Each node has eight nodes or leaves, and each leaf has voxel (VXL) information. FIG. 39 is a diagram showing an example structure of a volume including voxels. FIG. 40 is a diagram showing an example of the volume shown in FIG. 39 having been converted into the octree structure. Among the leaves shown in FIG. 40, leaves 1, 2, and 3 respectively represent VXL 1, VXL 2, and VXL 3, and represent VXLs including a point group (hereinafter, active VXLs).

[0380]

An octree is represented by, for example, binary sequences of 1s and 0s. For example, when giving the nodes or the active VXLs a value of 1 and everything else a value of 0, each node and leaf is assigned with the binary sequence shown in FIG. 40. Thus, this binary sequence is scanned in accordance with a breadth-first or a depth-first scan order. For example, when scanning breadth-first, the binary sequence shown in A of FIG. 41 is obtained. When scanning depth-first, the binary sequence shown in B of FIG. 41 is obtained. The binary sequences obtained through this scanning are encoded through entropy encoding, which reduces an amount of information.

[0381]

Depth information in the octree representation will be described next. Depth in the octree representation is used in order to control up to how fine a granularity point cloud information included in a volume is stored. Upon setting a great depth, it is possible to reproduce the point cloud information to a more precise level, but an amount of data for representing the nodes and leaves increases. Upon setting a small depth, however, the amount of data decreases, but some information that the point cloud information originally held is lost, since pieces of point cloud information including different positions and different colors are now considered as pieces of point cloud information including the same position and the same color.

[0382]

For example, FIG. 42 is a diagram showing an example in which the octree with a depth of 2 shown in FIG. 40 is represented with a depth of 1. The octree shown in FIG. 42 has a lower amount of data than the octree shown in FIG. 40. In other words, the binarized octree shown in FIG. 42 has a lower bit count than the octree shown in FIG. 40. Leaf 1 and leaf 2 shown in FIG. 40 are represented by leaf 1 shown in FIG. 41. In other words, the information on leaf

1 and leaf 2 being in different positions is lost.

[0383]

FIG. 43 is a diagram showing a volume corresponding to the octree shown in FIG. 42. VXL 1 and VXL 2 shown in FIG. 39 correspond to VXL 12 shown in FIG. 43. In this case, three-dimensional data encoding device 1300 generates color information of VXL 12 shown in FIG. 43 using color information of VXL 1 and VXL 2 shown in FIG. 39. For example, three-dimensional data encoding device 1300 calculates an average value, a median, a weighted average value, or the like of the color information of VXL 1 and VXL 2 as the color information of VXL 12. In this manner, three-dimensional data encoding device 1300 may control a reduction of the amount of data by changing the depth of the octree.

[0384]

Three-dimensional data encoding device 1300 may set the depth information of the octree to units of worlds, units of spaces, or units of volumes. In this case, three-dimensional data encoding device 1300 may append the depth information to header information of the world, header information of the space, or header information of the volume. In all worlds, spaces, and volumes associated with different times, the same value may be used as the depth information. In this case, three-dimensional data encoding device 1300 may append the depth information to header information managing the worlds associated with all times.

[0385]

When the color information is included in the voxels, transformer 1303 applies frequency transformation, e.g. orthogonal transformation, to a prediction residual of the color information of the voxels in the volume. For example, transformer 1303 creates a one-dimensional array by scanning the

prediction residual in a certain scan order. Subsequently, transformer 1303 transforms the one-dimensional array to a frequency domain by applying one-dimensional orthogonal transformation to the created one-dimensional array. With this, when a value of the prediction residual in the volume is similar, a value of a low-frequency component increases and a value of a high-frequency component decreases. As such, it is possible to more efficiently reduce an encoding amount in quantizer 1304.

[0386]

Transformer 1303 does not need to use orthogonal transformation in one dimension, but may also use orthogonal transformation in two or more dimensions. For example, transformer 1303 maps the prediction residual to a two-dimensional array in a certain scan order, and applies two-dimensional orthogonal transformation to the obtained two-dimensional array. Transformer 1303 may select an orthogonal transformation method to be used from a plurality of orthogonal transformation methods. In this case, three-dimensional data encoding device 1300 appends, to the bitstream, information indicating which orthogonal transformation method is used. Transformer 1303 may select an orthogonal transformation method to be used from a plurality of orthogonal transformation methods in different dimensions. In this case, three-dimensional data encoding device 1300 appends, to the bitstream, in how many dimensions the orthogonal transformation method is used.

[0387]

For example, transformer 1303 matches the scan order of the prediction residual to a scan order (breadth-first, depth-first, or the like) in the octree in the volume. This makes it possible to reduce overhead, since information indicating the scan order of the prediction residual does not need to be appended to the bitstream. Transformer 1303 may apply a scan order different from the

scan order of the octree. In this case, three-dimensional data encoding device 1300 appends, to the bitstream, information indicating the scan order of the prediction residual. This enables three-dimensional data encoding device 1300 to efficiently encode the prediction residual. Three-dimensional data encoding device 1300 may append, to the bitstream, information (flag, etc.) indicating whether to apply the scan order of the octree, and may also append, to the bitstream, information indicating the scan order of the prediction residual when the scan order of the octree is not applied.

[0388]

Transformer 1303 does not only transform the prediction residual of the color information, and may also transform other attribute information included in the voxels. For example, transformer 1303 may transform and encode information, such as reflectance information, obtained when obtaining a point cloud through LiDAR and the like.

[0389]

Transformer 1303 may skip these processes when the spaces do not include attribute information such as color information. Three-dimensional data encoding device 1300 may append, to the bitstream, information (flag) indicating whether to skip the processes of transformer 1303.

[0390]

Quantizer 1304 generates a quantized coefficient by performing quantization using a quantization control parameter on a frequency component of the prediction residual generated by transformer 1303. With this, the amount of information is further reduced. The generated quantized coefficient is outputted to entropy encoder 1313. Quantizer 1304 may control the quantization control parameter in units of worlds, units of spaces, or units of volumes. In this case, three-dimensional data encoding device 1300 appends

the quantization control parameter to each header information and the like. Quantizer 1304 may perform quantization control by changing a weight per frequency component of the prediction residual. For example, quantizer 1304 may precisely quantize a low-frequency component and roughly quantize a high-frequency component. In this case, three-dimensional data encoding device 1300 may append, to a header, a parameter expressing a weight of each frequency component.

[0391]

Quantizer 1304 may skip these processes when the spaces do not include attribute information such as color information. Three-dimensional data encoding device 1300 may append, to the bitstream, information (flag) indicating whether to skip the processes of quantizer 1304.

[0392]

Inverse quantizer 1305 generates an inverse quantized coefficient of the prediction residual by performing inverse quantization on the quantized coefficient generated by quantizer 1304 using the quantization control parameter, and outputs the generated inverse quantized coefficient to inverse transformer 1306.

[0393]

Inverse transformer 1306 generates an inverse transformation-applied prediction residual by applying inverse transformation on the inverse quantized coefficient generated by inverse quantizer 1305. This inverse transformation-applied prediction residual does not need to completely coincide with the prediction residual outputted by transformer 1303, since the inverse transformation-applied prediction residual is a prediction residual that is generated after the quantization.

[0394]

Adder 1307 adds, to generate a reconstructed volume, (i) the inverse transformation-applied prediction residual generated by inverse transformer 1306 to (ii) a predicted volume that is generated through intra prediction or intra prediction, which will be described later, and is used to generate a pre-quantized prediction residual. This reconstructed volume is stored in reference volume memory 1308 or reference space memory 1310.

[0395]

Intra predictor 1309 generates a predicted volume of an encoding target volume using attribute information of a neighboring volume stored in reference volume memory 1308. The attribute information includes color information or a reflectance of the voxels. Intra predictor 1309 generates a predicted value of color information or a reflectance of the encoding target volume.

[0396]

FIG. 44 is a diagram for describing an operation of intra predictor 1309. For example, intra predictor 1309 generates the predicted volume of the encoding target volume (volume idx = 3) shown in FIG. 44, using a neighboring volume (volume idx = 0). Volume idx here is identifier information that is appended to a volume in a space, and a different value is assigned to each volume. An order of assigning volume idx may be the same as an encoding order, and may also be different from the encoding order. For example, intra predictor 1309 uses an average value of color information of voxels included in volume idx = 0, which is a neighboring volume, as the predicted value of the color information of the encoding target volume shown in FIG. 44. In this case, a prediction residual is generated by deducting the predicted value of the color information from the color information of each voxel included in the encoding target volume. The following processes are performed by transformer 1303 and subsequent processors with respect to this prediction residual. In this case,

three-dimensional data encoding device 1300 appends, to the bitstream, neighboring volume information and prediction mode information. The neighboring volume information here is information indicating a neighboring volume used in the prediction, and indicates, for example, volume idx of the neighboring volume used in the prediction. The prediction mode information here indicates a mode used to generate the predicted volume. The mode is, for example, an average value mode in which the predicted value is generated using an average value of the voxels in the neighboring volume, or a median mode in which the predicted value is generated using the median of the voxels in the neighboring volume.

[0397]

Intra predictor 1309 may generate the predicted volume using a plurality of neighboring volumes. For example, in the structure shown in FIG. 44, intra predictor 1309 generates predicted volume 0 using a volume with volume idx = 0, and generates predicted volume 1 using a volume with volume idx = 1. Intra predictor 1309 then generates an average of predicted volume 0 and predicted volume 1 as a final predicted volume. In this case, three-dimensional data encoding device 1300 may append, to the bitstream, a plurality of volumes idx of a plurality of volumes used to generate the predicted volume.

[0398]

FIG. 45 is a diagram schematically showing the inter prediction process according to the present embodiment. Inter predictor 1311 encodes (inter predicts) a space (SPC) associated with certain time T_Cur using an encoded space associated with different time T_LX. In this case, inter predictor 1311 performs an encoding process by applying a rotation and translation process to the encoded space associated with different time T_LX.

[0399]

Three-dimensional data encoding device 1300 appends, to the bitstream, RT information relating to a rotation and translation process suited to the space associated with different time T_LX. Different time T_LX is, for example, time T_L0 before certain time T_Cur. At this point, three-dimensional data encoding device 1300 may append, to the bitstream, RT information RT_L0 relating to a rotation and translation process suited to a space associated with time T_L0.

[0400]

Alternatively, different time T_LX is, for example, time T_L1 after certain time T_Cur. At this point, three-dimensional data encoding device 1300 may append, to the bitstream, RT information RT_L1 relating to a rotation and translation process suited to a space associated with time T_L1.

[0401]

Alternatively, inter predictor 1311 encodes (bidirectional prediction) with reference to the spaces associated with time T_L0 and time T_L1 that differ from each other. In this case, three-dimensional data encoding device 1300 may append, to the bitstream, both RT information RT_L0 and RT information RT_L1 relating to the rotation and translation process suited to the spaces thereof.

[0402]

Note that T_L0 has been described as being before T_Cur and T_L1 as being after T_Cur, but are not necessarily limited thereto. For example, T_L0 and T_L1 may both be before T_Cur. T_L0 and T_L1 may also both be after T_Cur.

[0403]

Three-dimensional data encoding device 1300 may append, to the

bitstream, RT information relating to a rotation and translation process suited to spaces associated with different times, when encoding with reference to each of the spaces. For example, three-dimensional data encoding device 1300 manages a plurality of encoded spaces to be referred to, using two reference lists (list L0 and list L1). When a first reference space in list L0 is L0R0, a second reference space in list L0 is L0R1, a first reference space in list L1 is L1R0, and a second reference space in list L1 is L1R1, three-dimensional data encoding device 1300 appends, to the bitstream, RT information RT_L0R0 of L0R0, RT information RT_L0R1 of L0R1, RT information RT_L1R0 of L1R0, and RT information RT_L1R1 of L1R1. For example, three-dimensional data encoding device 1300 appends these pieces of RT information to a header and the like of the bitstream.

[0404]

Three-dimensional data encoding device 1300 determines whether to apply rotation and translation per reference space, when encoding with reference to reference spaces associated with different times. In this case, three-dimensional data encoding device 1300 may append, to header information and the like of the bitstream, information (RT flag, etc.) indicating whether rotation and translation are applied per reference space. For example, three-dimensional data encoding device 1300 calculates the RT information and an Iterative Closest Point (ICP) error value, using an ICP algorithm per reference space to be referred to from the encoding target space. Three-dimensional data encoding device 1300 determines that rotation and translation do not need to be performed and sets the RT flag to OFF, when the ICP error value is lower than or equal to a predetermined fixed value. In contrast, three-dimensional data encoding device 1300 sets the RT flag to ON and appends the RT information to the bitstream, when the ICP error value exceeds the above

fixed value.

[0405]

FIG. 46 is a diagram showing an example syntax to be appended to a header of the RT information and the RT flag. Note that a bit count assigned to each syntax may be decided based on a range of this syntax. For example, when eight reference spaces are included in reference list L0, 3 bits may be assigned to MaxRefSpc_l0. The bit count to be assigned may be variable in accordance with a value each syntax can be, and may also be fixed regardless of the value each syntax can be. When the bit count to be assigned is fixed, three-dimensional data encoding device 1300 may append this fixed bit count to other header information.

[0406]

MaxRefSpc_l0 shown in FIG. 46 indicates a number of reference spaces included in reference list L0. RT_flag_l0[i] is an RT flag of reference space i in reference list L0. When RT_flag_l0[i] is 1, rotation and translation are applied to reference space i. When RT_flag_l0[i] is 0, rotation and translation are not applied to reference space i.

[0407]

R_l0[i] and T_l0[i] are RT information of reference space i in reference list L0. R_l0[i] is rotation information of reference space i in reference list L0. The rotation information indicates contents of the applied rotation process, and is, for example, a rotation matrix or a quaternion. T_l0[i] is translation information of reference space i in reference list L0. The translation information indicates contents of the applied translation process, and is, for example, a translation vector.

[0408]

MaxRefSpc_l1 indicates a number of reference spaces included in

reference list L1. RT_flag_l1[i] is an RT flag of reference space i in reference list L1. When RT_flag_l1[i] is 1, rotation and translation are applied to reference space i. When RT_flag_l1[i] is 0, rotation and translation are not applied to reference space i.

[0409]

R_l1[i] and T_l1[i] are RT information of reference space i in reference list L1. R_l1[i] is rotation information of reference space i in reference list L1. The rotation information indicates contents of the applied rotation process, and is, for example, a rotation matrix or a quaternion. T_l1[i] is translation information of reference space i in reference list L1. The translation information indicates contents of the applied translation process, and is, for example, a translation vector.

[0410]

Inter predictor 1311 generates the predicted volume of the encoding target volume using information on an encoded reference space stored in reference space memory 1310. As stated above, before generating the predicted volume of the encoding target volume, inter predictor 1311 calculates RT information at an encoding target space and a reference space using an ICP algorithm, in order to approach an overall positional relationship between the encoding target space and the reference space. Inter predictor 1311 then obtains reference space B by applying a rotation and translation process to the reference space using the calculated RT information. Subsequently, inter predictor 1311 generates the predicted volume of the encoding target volume in the encoding target space using information in reference space B. Three-dimensional data encoding device 1300 appends, to header information and the like of the encoding target space, the RT information used to obtain reference space B.

[0411]

In this manner, inter predictor 1311 is capable of improving precision of the predicted volume by generating the predicted volume using the information of the reference space, after approaching the overall positional relationship between the encoding target space and the reference space, by applying a rotation and translation process to the reference space. It is possible to reduce the encoding amount since it is possible to limit the prediction residual. Note that an example has been described in which ICP is performed using the encoding target space and the reference space, but is not necessarily limited thereto. For example, inter predictor 1311 may calculate the RT information by performing ICP using at least one of (i) an encoding target space in which a voxel or point cloud count is pruned, or (ii) a reference space in which a voxel or point cloud count is pruned, in order to reduce the processing amount.

[0412]

When the ICP error value obtained as a result of the ICP is smaller than a predetermined first threshold, i.e., when for example the positional relationship between the encoding target space and the reference space is similar, inter predictor 1311 determines that a rotation and translation process is not necessary, and the rotation and translation process does not need to be performed. In this case, three-dimensional data encoding device 1300 may control the overhead by not appending the RT information to the bitstream.

[0413]

When the ICP error value is greater than a predetermined second threshold, inter predictor 1311 determines that a shape change between the spaces is large, and intra prediction may be applied on all volumes of the encoding target space. Hereinafter, spaces to which intra prediction is applied will be referred to as intra spaces. The second threshold is greater than the

above first threshold. The present embodiment is not limited to ICP, and any type of method may be used as long as the method calculates the RT information using two voxel sets or two point cloud sets.

[0414]

When attribute information, e.g. shape or color information, is included in the three-dimensional data, inter predictor 1311 searches, for example, a volume whose attribute information, e.g. shape or color information, is the most similar to the encoding target volume in the reference space, as the predicted volume of the encoding target volume in the encoding target space. This reference space is, for example, a reference space on which the above rotation and translation process has been performed. Inter predictor 1311 generates the predicted volume using the volume (reference volume) obtained through the search. FIG. 47 is a diagram for describing a generating operation of the predicted volume. When encoding the encoding target volume (volume idx = 0) shown in FIG. 47 using inter prediction, inter predictor 1311 searches a volume with a smallest prediction residual, which is the difference between the encoding target volume and the reference volume, while sequentially scanning the reference volume in the reference space. Inter predictor 1311 selects the volume with the smallest prediction residual as the predicted volume. The prediction residuals of the encoding target volume and the predicted volume are encoded through the processes performed by transformer 1303 and subsequent processors. The prediction residual here is a difference between the attribute information of the encoding target volume and the attribute information of the predicted volume. Three-dimensional data encoding device 1300 appends, to the header and the like of the bitstream, volume idx of the reference volume in the reference space, as the predicted volume.

[0415]

In the example shown in FIG. 47, the reference volume with volume idx = 4 of reference space L0R0 is selected as the predicted volume of the encoding target volume. The prediction residuals of the encoding target volume and the reference volume, and reference volume idx = 4 are then encoded and appended to the bitstream.

[0416]

Note that an example has been described in which the predicted volume of the attribute information is generated, but the same process may be applied to the predicted volume of the position information.

[0417]

Prediction controller 1312 controls whether to encode the encoding target volume using intra prediction or inter prediction. A mode including intra prediction and inter prediction is referred to here as a prediction mode. For example, prediction controller 1312 calculates the prediction residual when the encoding target volume is predicted using intra prediction and the prediction residual when the encoding target volume is predicted using inter prediction as evaluation values, and selects the prediction mode whose evaluation value is smaller. Note that prediction controller 1312 may calculate an actual encoding amount by applying orthogonal transformation, quantization, and entropy encoding to the prediction residual of the intra prediction and the prediction residual of the inter prediction, and select a prediction mode using the calculated encoding amount as the evaluation value. Overhead information (reference volume idx information, etc.) aside from the prediction residual may be added to the evaluation value. Prediction controller 1312 may continuously select intra prediction when it has been decided in advance to encode the encoding target space using intra space.

[0418]

Entropy encoder 1313 generates an encoded signal (encoded bitstream) by variable-length encoding the quantized coefficient, which is an input from quantizer 1304. To be specific, entropy encoder 1313, for example, binarizes the quantized coefficient and arithmetically encodes the obtained binary signal.

[0419]

A three-dimensional data decoding device that decodes the encoded signal generated by three-dimensional data encoding device 1300 will be described next. FIG. 48 is a block diagram of three-dimensional data decoding device 1400 according to the present embodiment. This three-dimensional data decoding device 1400 includes entropy decoder 1401, inverse quantizer 1402, inverse transformer 1403, adder 1404, reference volume memory 1405, intra predictor 1406, reference space memory 1407, inter predictor 1408, and prediction controller 1409.

[0420]

Entropy decoder 1401 variable-length decodes the encoded signal (encoded bitstream). For example, entropy decoder 1401 generates a binary signal by arithmetically decoding the encoded signal, and generates a quantized coefficient using the generated binary signal.

[0421]

Inverse quantizer 1402 generates an inverse quantized coefficient by inverse quantizing the quantized coefficient inputted from entropy decoder 1401, using a quantization parameter appended to the bitstream and the like.

[0422]

Inverse transformer 1403 generates a prediction residual by inverse transforming the inverse quantized coefficient inputted from inverse quantizer 1402. For example, inverse transformer 1403 generates the prediction residual by inverse orthogonally transforming the inverse quantized coefficient, based on

information appended to the bitstream.

[0423]

Adder 1404 adds, to generate a reconstructed volume, (i) the prediction residual generated by inverse transformer 1403 to (ii) a predicted volume generated through intra prediction or intra prediction. This reconstructed volume is outputted as decoded three-dimensional data and is stored in reference volume memory 1405 or reference space memory 1407.

[0424]

Intra predictor 1406 generates a predicted volume through intra prediction using a reference volume in reference volume memory 1405 and information appended to the bitstream. To be specific, intra predictor 1406 obtains neighboring volume information (e.g. volume idx) appended to the bitstream and prediction mode information, and generates the predicted volume through a mode indicated by the prediction mode information, using a neighboring volume indicated in the neighboring volume information. Note that the specifics of these processes are the same as the above-mentioned processes performed by intra predictor 1309, except for which information appended to the bitstream is used.

[0425]

Inter predictor 1408 generates a predicted volume through inter prediction using a reference space in reference space memory 1407 and information appended to the bitstream. To be specific, inter predictor 1408 applies a rotation and translation process to the reference space using the RT information per reference space appended to the bitstream, and generates the predicted volume using the rotated and translated reference space. Note that when an RT flag is present in the bitstream per reference space, inter predictor 1408 applies a rotation and translation process to the reference space in

accordance with the RT flag. Note that the specifics of these processes are the same as the above-mentioned processes performed by inter predictor 1311, except for which information appended to the bitstream is used.

[0426]

Prediction controller 1409 controls whether to decode a decoding target volume using intra prediction or inter prediction. For example, prediction controller 1409 selects intra prediction or inter prediction in accordance with information that is appended to the bitstream and indicates the prediction mode to be used. Note that prediction controller 1409 may continuously select intra prediction when it has been decided in advance to decode the decoding target space using intra space.

[0427]

Hereinafter, variations of the present embodiment will be described. In the present embodiment, an example has been described in which rotation and translation is applied in units of spaces, but rotation and translation may also be applied in smaller units. For example, three-dimensional data encoding device 1300 may divide a space into subspaces, and apply rotation and translation in units of subspaces. In this case, three-dimensional data encoding device 1300 generates RT information per subspace, and appends the generated RT information to a header and the like of the bitstream. Three-dimensional data encoding device 1300 may apply rotation and translation in units of volumes, which is an encoding unit. In this case, three-dimensional data encoding device 1300 generates RT information in units of encoded volumes, and appends the generated RT information to a header and the like of the bitstream. The above may also be combined. In other words, three-dimensional data encoding device 1300 may apply rotation and translation in large units and subsequently apply rotation and translation in small units. For

example, three-dimensional data encoding device 1300 may apply rotation and translation in units of spaces, and may also apply different rotations and translations to each of a plurality of volumes included in the obtained spaces.

[0428]

In the present embodiment, an example has been described in which rotation and translation is applied to the reference space, but is not necessarily limited thereto. For example, three-dimensional data encoding device 1300 may apply a scaling process and change a size of the three-dimensional data. Three-dimensional data encoding device 1300 may also apply one or two of the rotation, translation, and scaling. When applying the processes in multiple stages and different units as stated above, a type of the processes applied in each unit may differ. For example, rotation and translation may be applied in units of spaces, and translation may be applied in units of volumes.

[0429]

Note that these variations are also applicable to three-dimensional data decoding device 1400.

[0430]

As stated above, three-dimensional data encoding device 1300 according to the present embodiment performs the following processes. FIG. 48 is a flowchart of the inter prediction process performed by three-dimensional data encoding device 1300.

[0431]

Three-dimensional data encoding device 1300 generates predicted position information (e.g. predicted volume) using position information on three-dimensional points included in three-dimensional reference data (e.g. reference space) associated with a time different from a time associated with current three-dimensional data (e.g. encoding target space) (S1301). To be specific,

three-dimensional data encoding device 1300 generates the predicted position information by applying a rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data.

[0432]

Note that three-dimensional data encoding device 1300 may perform a rotation and translation process using a first unit (e.g. spaces), and may perform the generating of the predicted position information using a second unit (e.g. volumes) that is smaller than the first unit. For example, three-dimensional data encoding device 1300 searches a volume among a plurality of volumes included in the rotated and translated reference space, whose position information differs the least from the position information of the encoding target volume included in the encoding target space. Note that three-dimensional data encoding device 1300 may perform the rotation and translation process, and the generating of the predicted position information in the same unit.

[0433]

Three-dimensional data encoding device 1300 may generate the predicted position information by applying (i) a first rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data, and (ii) a second rotation and translation process to the position information on the three-dimensional points obtained through the first rotation and translation process, the first rotation and translation process using a first unit (e.g. spaces) and the second rotation and translation process using a second unit (e.g. volumes) that is smaller than the first unit.

[0434]

For example, as illustrated in FIG. 41, the position information on the

three-dimensional points and the predicted position information is represented using an octree structure. For example, the position information on the three-dimensional points and the predicted position information is expressed in a scan order that prioritizes a breadth over a depth in the octree structure. For example, the position information on the three-dimensional points and the predicted position information is expressed in a scan order that prioritizes a depth over a breadth in the octree structure.

[0435]

As illustrated in FIG. 46, three-dimensional data encoding device 1300 encodes an RT flag that indicates whether to apply the rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data. In other words, three-dimensional data encoding device 1300 generates the encoded signal (encoded bitstream) including the RT flag. Three-dimensional data encoding device 1300 encodes RT information that indicates contents of the rotation and translation process. In other words, three-dimensional data encoding device 1300 generates the encoded signal (encoded bitstream) including the RT information. Note that three-dimensional data encoding device 1300 may encode the RT information when the RT flag indicates to apply the rotation and translation process, and does not need to encode the RT information when the RT flag indicates not to apply the rotation and translation process.

[0436]

The three-dimensional data includes, for example, the position information on the three-dimensional points and the attribute information (color information, etc.) of each three-dimensional point. Three-dimensional data encoding device 1300 generates predicted attribute information using the attribute information of the three-dimensional points included in the three-

dimensional reference data (S1302).

[0437]

Three-dimensional data encoding device 1300 next encodes the position information on the three-dimensional points included in the current three-dimensional data, using the predicted position information. For example, as illustrated in FIG. 38, three-dimensional data encoding device 1300 calculates differential position information, the differential position information being a difference between the predicted position information and the position information on the three-dimensional points included in the current three-dimensional data (S1303).

[0438]

Three-dimensional data encoding device 1300 encodes the attribute information of the three-dimensional points included in the current three-dimensional data, using the predicted attribute information. For example, three-dimensional data encoding device 1300 calculates differential attribute information, the differential attribute information being a difference between the predicted attribute information and the attribute information on the three-dimensional points included in the current three-dimensional data (S1304). Three-dimensional data encoding device 1300 next performs transformation and quantization on the calculated differential attribute information (S1305).

[0439]

Lastly, three-dimensional data encoding device 1300 encodes (e.g. entropy encodes) the differential position information and the quantized differential attribute information (S1036). In other words, three-dimensional data encoding device 1300 generates the encoded signal (encoded bitstream) including the differential position information and the differential attribute information.

[0440]

Note that when the attribute information is not included in the three-dimensional data, three-dimensional data encoding device 1300 does not need to perform steps S1302, S1304, and S1305. Three-dimensional data encoding device 1300 may also perform only one of the encoding of the position information on the three-dimensional points and the encoding of the attribute information of the three-dimensional points.

[0441]

An order of the processes shown in FIG. 49 is merely an example and is not limited thereto. For example, since the processes with respect to the position information (S1301 and S1303) and the processes with respect to the attribute information (S1302, S1304, and S1305) are separate from one another, they may be performed in an order of choice, and a portion thereof may also be performed in parallel.

[0442]

With the above, three-dimensional data encoding device 1300 according to the present embodiment generates predicted position information using position information on three-dimensional points included in three-dimensional reference data associated with a time different from a time associated with current three-dimensional data; and encodes differential position information, which is a difference between the predicted position information and the position information on the three-dimensional points included in the current three-dimensional data. This makes it possible to improve encoding efficiency since it is possible to reduce the amount of data of the encoded signal.

[0443]

Three-dimensional data encoding device 1300 according to the present embodiment generates predicted attribute information using attribute

information on three-dimensional points included in three-dimensional reference data; and encodes differential attribute information, which is a difference between the predicted attribute information and the attribute information on the three-dimensional points included in the current three-dimensional data. This makes it possible to improve encoding efficiency since it is possible to reduce the amount of data of the encoded signal.

[0444]

For example, three-dimensional data encoding device 1300 includes a processor and memory. The processor uses the memory to perform the above processes.

[0445]

FIG. 48 is a flowchart of the inter prediction process performed by three-dimensional data decoding device 1400.

[0446]

Three-dimensional data decoding device 1400 decodes (e.g. entropy decodes) the differential position information and the differential attribute information from the encoded signal (encoded bitstream) (S1401).

[0447]

Three-dimensional data decoding device 1400 decodes, from the encoded signal, an RT flag that indicates whether to apply the rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data. Three-dimensional data decoding device 1400 encodes RT information that indicates contents of the rotation and translation process. Note that three-dimensional data decoding device 1400 may decode the RT information when the RT flag indicates to apply the rotation and translation process, and does not need to decode the RT information when the RT flag indicates not to apply the rotation and translation process.

[0448]

Three-dimensional data decoding device 1400 next performs inverse transformation and inverse quantization on the decoded differential attribute information (S1402).

[0449]

Three-dimensional data decoding device 1400 next generates predicted position information (e.g. predicted volume) using the position information on the three-dimensional points included in the three-dimensional reference data (e.g. reference space) associated with a time different from a time associated with the current three-dimensional data (e.g. decoding target space) (S1403). To be specific, three-dimensional data decoding device 1400 generates the predicted position information by applying a rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data.

[0450]

More specifically, when the RT flag indicates to apply the rotation and translation process, three-dimensional data decoding device 1400 applies the rotation and translation process on the position information on the three-dimensional points included in the three-dimensional reference data indicated in the RT information. In contrast, when the RT flag indicates not to apply the rotation and translation process, three-dimensional data decoding device 1400 does not apply the rotation and translation process on the position information on the three-dimensional points included in the three-dimensional reference data.

[0451]

Note that three-dimensional data decoding device 1400 may perform the rotation and translation process using a first unit (e.g. spaces), and may perform

the generating of the predicted position information using a second unit (e.g. volumes) that is smaller than the first unit. Note that three-dimensional data decoding device 1400 may perform the rotation and translation process, and the generating of the predicted position information in the same unit.

[0452]

Three-dimensional data decoding device 1400 may generate the predicted position information by applying (i) a first rotation and translation process to the position information on the three-dimensional points included in the three-dimensional reference data, and (ii) a second rotation and translation process to the position information on the three-dimensional points obtained through the first rotation and translation process, the first rotation and translation process using a first unit (e.g. spaces) and the second rotation and translation process using a second unit (e.g. volumes) that is smaller than the first unit.

[0453]

For example, as illustrated in FIG. 41, the position information on the three-dimensional points and the predicted position information is represented using an octree structure. For example, the position information on the three-dimensional points and the predicted position information is expressed in a scan order that prioritizes a breadth over a depth in the octree structure. For example, the position information on the three-dimensional points and the predicted position information is expressed in a scan order that prioritizes a depth over a breadth in the octree structure.

[0454]

Three-dimensional data decoding device 1400 generates predicted attribute information using the attribute information of the three-dimensional points included in the three-dimensional reference data (S1404).

[0455]

Three-dimensional data decoding device 1400 next restores the position information on the three-dimensional points included in the current three-dimensional data, by decoding encoded position information included in an encoded signal, using the predicted position information. The encoded position information here is the differential position information. Three-dimensional data decoding device 1400 restores the position information on the three-dimensional points included in the current three-dimensional data, by adding the differential position information to the predicted position information (S1405).

[0456]

Three-dimensional data decoding device 1400 restores the attribute information of the three-dimensional points included in the current three-dimensional data, by decoding encoded attribute information included in an encoded signal, using the predicted attribute information. The encoded attribute information here is the differential position information. Three-dimensional data decoding device 1400 restores the attribute information on the three-dimensional points included in the current three-dimensional data, by adding the differential attribute information to the predicted attribute information (S1406).

[0457]

Note that when the attribute information is not included in the three-dimensional data, three-dimensional data decoding device 1400 does not need to perform steps S1402, S1404, and S1406. Three-dimensional data decoding device 1400 may also perform only one of the decoding of the position information on the three-dimensional points and the decoding of the attribute information of the three-dimensional points.

[0458]

An order of the processes shown in FIG. 50 is merely an example and is not limited thereto. For example, since the processes with respect to the position information (S1403 and S1405) and the processes with respect to the attribute information (S1402, S1404, and S1406) are separate from one another, they may be performed in an order of choice, and a portion thereof may also be performed in parallel.

[0459]

EMBODIMENT 8

In the present embodiment, adaptive entropy encoding (arithmetic coding) performed on occupancy codes of an octree will be described.

[0460]

FIG. 51 is a diagram illustrating an example of a quadtree structure. FIG. 52 is a diagram illustrating occupancy codes of the tree structure illustrated in FIG. 51. FIG. 53 is a diagram schematically illustrating an operation performed by a three-dimensional data encoding device according to the present embodiment.

[0461]

The three-dimensional data encoding device according to the present embodiment entropy encodes an 8-bit occupancy code in an octree. The three-dimensional data encoding device also updates a coding table in an entropy encoding process for occupancy code. Additionally, the three-dimensional data encoding device does not use a single coding table but uses an adaptive coding table in order to use similarity information of three-dimensional points. In other words, the three-dimensional data encoding device uses coding tables.

[0462]

Similarity information is, for example, geometry information of a three-

dimensional point, structure information of an octree, or attribute information of a three-dimensional point.

[0463]

It should be noted that although the quadtree is shown as the example in FIG. 51 to FIG. 53, the same method may be applied to an N-ary tree such as a binary tree, an octree, and a hexadecatree. For example, the three-dimensional data encoding device entropy encodes an 8-bit occupancy code in the case of an octree, a 4-bit occupancy code in the case of a quadtree, and a 16-bit occupancy code in the case of a hexadecatree, using an adaptive table (also referred to as a coding table).

[0464]

The following describes an adaptive entropy encoding process using geometry information of a three-dimensional point.

[0465]

When local geometries of two nodes in a tree structure are similar to each other, there is a chance that occupancy states (i.e., states each indicating whether a three-dimensional point is included) of child nodes are similar to each other. As a result, the three-dimensional data encoding device performs grouping using a local geometry of a parent node. This enables the three-dimensional data encoding device to group together the occupancy states of the child nodes, and use a different coding table for each group. Accordingly, it is possible to improve the entropy encoding efficiency.

[0466]

FIG. 54 is a diagram illustrating an example of geometry information. Geometry information includes information indicating whether each of neighboring nodes of a current node is occupied (i.e., includes a three-dimensional point). For example, the three-dimensional data encoding device

calculates a local geometry of the current node using information indicating whether a neighboring node includes a three-dimensional point (is occupied or non-occupied). A neighboring node is, for example, a node spatially located around a current node, or a node located in the same position in a different time as the current node or spatially located around the position.

[0467]

In FIG. 54, a hatched cube indicates a current node. A white cube is a neighboring node, and indicates a node including a three-dimensional point. In FIG. 54, the geometry pattern indicated in (2) is obtained by rotating the geometry pattern indicated in (1). Accordingly, the three-dimensional data encoding device determines that these geometry patterns have a high geometry similarity, and entropy encodes the geometry patterns using the same coding table. In addition, the three-dimensional data encoding device determines that the geometry patterns indicated in (3) and (4) have a low geometry similarity, and entropy encodes the geometry patterns using other coding tables.

[0468]

FIG. 55 is a diagram illustrating an example of occupancy codes of current nodes in the geometry patterns of (1) to (4) illustrated in FIG. 54, and coding tables used for entropy encoding. As illustrated above, the three-dimensional data encoding device determines that the geometry patterns of (1) and (2) are included in the same geometry group, and uses same coding table A for the geometry patterns of (1) and (2). The three-dimensional data encoding device uses coding table B and coding table C for the geometry patterns of (3) and (4), respectively.

[0469]

As illustrated in FIG. 55, there is a case in which the occupancy codes of the current nodes in the geometry patterns of (1) and (2) included in the same

geometry group are identical to each other.

[0470]

Next, the following describes an adaptive entropy encoding process using structure information of a tree structure. For example, structure information includes information indicating a layer to which a current node belongs.

[0471]

FIG. 56 is a diagram illustrating an example of a tree structure. Generally speaking, a local shape of an object depends on a search criterion. For example, a tree structure tends to be sparser in a lower layer than in an upper layer. Accordingly, the three-dimensional data encoding device uses different coding tables for upper layers and lower layers as illustrated in FIG. 56, which makes it possible to improve the entropy encoding efficiency.

[0472]

In other words, when the three-dimensional data encoding device encodes an occupancy code of each layer, the three-dimensional data encoding device may use a different coding table for each layer. For example, when the three-dimensional data encoding device encodes an occupancy code of layer N (N = 0 to 6), the three-dimensional data encoding device may perform entropy encoding on the tree structure illustrated in FIG. 56 using a coding table for layer N. Since this enables the three-dimensional data encoding device to select a coding table in accordance with an appearance pattern of an occupancy code of each layer, the three-dimensional data encoding device can improve the coding efficiency.

[0473]

Moreover, as illustrated in FIG. 56, the three-dimensional data encoding device may use coding table A for the occupancy codes of layer 0 to layer 2, and

may use coding table B for the occupancy codes of layer 3 to layer 6. Since this enables the three-dimensional data encoding device to select a coding table in accordance with an appearance pattern of the occupancy code for each group of layers, the three-dimensional data encoding device can improve the coding efficiency. The three-dimensional data encoding device may append information of the coding table used for each layer, to a header of a bitstream. Alternatively, the coding table used for each layer may be predefined by standards etc.

[0474]

Next, the following describes an adaptive entropy encoding process using attribute information (property information) of a three-dimensional point. For example, attribute information includes information about an object including a current node, or information about a normal vector of the current node.

[0475]

It is possible to group together three-dimensional points having a similar geometry, using pieces of attribute information of the three-dimensional points. For example, a normal vector indicating a direction of each of the three-dimensional points may be used as common attribute information of the three-dimensional points. It is possible to find a geometry relating to a similar occupancy code in a tree structure by using the normal vector.

[0476]

Moreover, a color or a degree of reflection (reflectance) may be used as attribute information. For example, the three-dimensional data encoding device groups together three-dimensional points having a similar geometry, using the colors or reflectances of the three-dimensional points, and performs a process such as switching between coding tables for each of the groups.

[0477]

FIG. 57 is a diagram for describing switching between coding tables based on a normal vector. As illustrated in FIG. 57, when normal vector groups to which normal vectors of current nodes belong are different, different coding tables are used. For example, a normal vector included in a predetermined range is categorized into one normal vector group.

[0478]

When objects belong in different categories, there is a high possibility that occupancy codes are different. Accordingly, the three-dimensional data encoding device may select a coding table in accordance with a category of an object to which a current node belongs. FIG. 58 is a diagram for describing switching between coding tables based on a category of an object. As illustrated in FIG. 58, when objects belong in different categories, different coding tables are used.

[0479]

The following describes an example of a structure of a bitstream according to the present embodiment. FIG. 59 is a diagram illustrating an example of a structure of a bitstream generated by the three-dimensional data encoding device according to the present embodiment. As illustrated in FIG. 59, the bitstream includes a coding table group, table indexes, and encoded occupancy codes. The coding table group includes coding tables.

[0480]

A table index indicates a coding table used for entropy encoding of a subsequent encoded occupancy code. An encoded occupancy code is an occupancy code that has been entropy encoded. As illustrated in FIG. 59, the bitstream also includes combinations of a table index and an encoded occupancy code.

[0481]

For example, in the example illustrated in FIG. 59, encoded occupancy code 0 is data that has been entropy encoded using a context model (also referred to as a context) indicated by table index 0. Encoded occupancy code 1 is data that has been entropy encoded using a context indicated by table index 1. A context for encoding encoded occupancy code 0 may be predefined by standards etc., and a three-dimensional data decoding device may use this context when decoding encoded occupancy code 0. Since this eliminates the need for appending the table index to the bitstream, it is possible to reduce overhead.

[0482]

Moreover, the three-dimensional data encoding device may append, in the header, information for resetting each context.

[0483]

The three-dimensional data encoding device determines a coding table using geometry information, structure information, or attribute information of a current node, and encodes an occupancy code using the determined coding table. The three-dimensional data encoding device appends a result of the encoding and information (e.g., a table index) of the coding table used for the encoding to a bitstream, and transmits the bitstream to the three-dimensional data decoding device. This enables the three-dimensional data decoding device to decode the occupancy code using the information of the coding table appended to the header.

[0484]

Moreover, the three-dimensional data encoding device need not append information of a coding table used for encoding to a bitstream, and the three-dimensional data decoding device may determine a coding table using geometry information, structure information, or attribute information of a current node

that has been decoded, using the same method as the three-dimensional data encoding device, and decode an occupancy code using the determined coding table. Since this eliminates the need for appending the information of the coding table to the bitstream, it is possible to reduce overhead.

[0485]

FIG. 60 and FIG. 61 each are a diagram illustrating an example of a coding table. As illustrated in FIG. 60 and FIG. 61, one coding table shows, for each value of an 8-bit occupancy code, a context model and a context model type associated with the value.

[0486]

As with the coding table illustrated in FIG. 60, the same context model (context) may be applied to occupancy codes. In addition, a different context model may be assigned to each occupancy code. Since this enables assignment of a context model in accordance with a probability of appearance of an occupancy code, it is possible to improve the coding efficiency.

[0487]

A context model type indicates, for example, whether a context model is a context model that updates a probability table in accordance with an appearance frequency of an occupancy code, or is a context model having a fixed probability table.

[0488]

Next, the following gives another example of a bitstream and a coding table. FIG. 61 is a diagram illustrating a variation of a structure of a bitstream. As illustrated in FIG. 61, the bitstream includes a coding table group and an encoded occupancy code. The coding table group includes coding tables.

[0489]

FIG. 63 and FIG. 64 each are a diagram illustrating an example of a

coding table. As illustrated in FIG. 63 and FIG. 64, one coding table shows, for each 1 bit included in an occupancy code, a context model and a context model type associated with the 1 bit.

[0490]

FIG. 65 is a diagram illustrating an example of a relationship between an occupancy code and bit numbers of the occupancy code.

[0491]

As stated above, the three-dimensional data encoding device may handle an occupancy code as binary data, assign a different context model for each bit, and entropy encode the occupancy code. Since this enables assignment of a context model in accordance with a probability of appearance of each bit of the occupancy code, it is possible to improve the coding efficiency.

[0492]

Specifically, each bit of the occupancy code corresponds to a sub-block obtained by dividing a spatial block corresponding to a current node. Accordingly, when sub-blocks in the same spatial position in a block have the same tendency, it is possible to improve the coding efficiency. For example, when a ground surface or a road surface crosses through a block, in an octree, four lower blocks include three-dimensional points, and four upper blocks include no three-dimensional point. Additionally, the same pattern appears in blocks horizontally arranged. Accordingly, it is possible to improve the coding efficiency by switching between contexts for each bit as described above.

[0493]

A context model that updates a probability table in accordance with an appearance frequency of each bit of an occupancy code may also be used. In addition, a context model having a fixed probability table may be used.

[0494]

Next, the following describes procedures for a three-dimensional data encoding process and a three-dimensional data decoding process according to the present embodiment.

[0495]

FIG. 66 is a flowchart of a three-dimensional data encoding process including an adaptive entropy encoding process using geometry information.

[0496]

In a decomposition process, an octree is generated from an initial bounding box of three-dimensional points. A bounding box is divided in accordance with the position of a three-dimensional point in the bounding box. Specifically, a non-empty sub-space is further divided. Next, information indicating whether a sub-space includes a three-dimensional point is encoded into an occupancy code. It should be noted that the same process is performed in the processes illustrated in FIG. 68 and FIG. 70.

[0497]

First, the three-dimensional data encoding device obtains inputted three-dimensional points (S1901). Next, the three-dimensional data encoding device determines whether a decomposition process per unit length is completed (S1902).

[0498]

When the decomposition process per unit length is not completed (NO in S1902), the three-dimensional data encoding device generates an octree by performing the decomposition process on a current node (S1903).

[0499]

Then, the three-dimensional data encoding device obtains geometry information (S1904), and selects a coding table based on the obtained geometry information (S1905). Here, as stated above, the geometry information is

information indicating, for example, a geometry of occupancy states of neighboring blocks of a current node.

[0500]

After that, the three-dimensional data encoding device entropy encodes an occupancy code of the current node using the selected coding table (S1906).

[0501]

Steps S1903 to S1906 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is completed (YES in S1902), the three-dimensional data encoding device outputs a bitstream including generated information (S1907).

[0502]

The three-dimensional data encoding device determines a coding table using geometry information, structure information, or attribute information of a current node, and encodes a bit sequence of an occupancy code using the determined coding table. The three-dimensional data encoding device appends a result of the encoding and information (e.g., a table index) of the coding table used for the encoding to a bitstream, and transmits the bitstream to the three-dimensional data decoding device. This enables the three-dimensional data decoding device to decode the occupancy code using the information of the coding table appended to the header.

[0503]

Moreover, the three-dimensional data encoding device need not append information of a coding table used for encoding to a bitstream, and the three-dimensional data decoding device may determine a coding table using geometry information, structure information, or attribute information of a current node that has been decoded, using the same method as the three-dimensional data encoding device, and decode an occupancy code using the determined coding

table. Since this eliminates the need for appending the information of the coding table to the bitstream, it is possible to reduce overhead.

[0504]

FIG. 67 is a flowchart of a three-dimensional data decoding process including an adaptive entropy decoding process using geometry information.

[0505]

A decomposition process included in the decoding process is similar to the decomposition process included in the above-described encoding process, they differ in the following point. The three-dimensional data decoding device divides an initial bounding box using a decoded occupancy code. When the three-dimensional data decoding device completes a process per unit length, the three-dimensional data decoding device stores the position of a bounding box as the position of a three-dimensional point. It should be noted that the same process is performed in the processes illustrated in FIG. 69 and FIG. 71.

[0506]

First, the three-dimensional data decoding device obtains an inputted bitstream (S1911). Next, the three-dimensional data decoding device determines whether a decomposition process per unit length is completed (S1912).

[0507]

When the decomposition process per unit length is not completed (NO in S1912), the three-dimensional data decoding device generates an octree by performing the decomposition process on a current node (S1913).

[0508]

Then, the three-dimensional data decoding device obtains geometry information (S1914), and selects a coding table based on the obtained geometry information (S1915). Here, as stated above, the geometry information is

information indicating, for example, a geometry of occupancy states of neighboring blocks of a current node.

[0509]

After that, the three-dimensional data decoding device entropy decodes an occupancy code of the current node using the selected coding table (S1916).

[0510]

Steps S1913 to S1916 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is completed (YES in S1912), the three-dimensional data decoding device outputs three-dimensional points (S1917).

[0511]

FIG. 68 is a flowchart of a three-dimensional data encoding process including an adaptive entropy encoding process using structure information.

[0512]

First, the three-dimensional data encoding device obtains inputted three-dimensional points (S1921). Next, the three-dimensional data encoding device determines whether a decomposition process per unit length is completed (S1922).

[0513]

When the decomposition process per unit length is not completed (NO in S1922), the three-dimensional data encoding device generates an octree by performing the decomposition process on a current node (S1923).

[0514]

Then, the three-dimensional data encoding device obtains structure information (S1924), and selects a coding table based on the obtained structure information (S1925). Here, as stated above, the structure information is information indicating, for example, a layer to which a current node belongs.

[0515]

After that, the three-dimensional data encoding device entropy encodes an occupancy code of the current node using the selected coding table (S1926).

[0516]

Steps S1923 to S1926 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is completed (YES in S1922), the three-dimensional data encoding device outputs a bitstream including generated information (S1927).

[0517]

FIG. 69 is a flowchart of a three-dimensional data decoding process including an adaptive entropy decoding process using structure information.

[0518]

First, the three-dimensional data decoding device obtains an inputted bitstream (S1931). Next, the three-dimensional data decoding device determines whether a decomposition process per unit length is completed (S1932).

[0519]

When the decomposition process per unit length is not completed (NO in S1932), the three-dimensional data decoding device generates an octree by performing the decomposition process on a current node (S1933).

[0520]

Then, the three-dimensional data decoding device obtains structure information (S1934), and selects a coding table based on the obtained structure information (S1935). Here, as stated above, the structure information is information indicating, for example, a layer to which a current node belongs.

[0521]

After that, the three-dimensional data decoding device entropy decodes

an occupancy code of the current node using the selected coding table (S1936).

[0522]

Steps S1933 to S1936 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is completed (YES in S1932), the three-dimensional data decoding device outputs three-dimensional points (S1937).

[0523]

FIG. 70 is a flowchart of a three-dimensional data encoding process including an adaptive entropy encoding process using attribute information.

[0524]

First, the three-dimensional data encoding device obtains inputted three-dimensional points (S1941). Next, the three-dimensional data encoding device determines whether a decomposition process per unit length is completed (S1942).

[0525]

When the decomposition process per unit length is not completed (NO in S1942), the three-dimensional data encoding device generates an octree by performing the decomposition process on a current node (S1943).

[0526]

Then, the three-dimensional data encoding device obtains attribute information (S1944), and selects a coding table based on the obtained attribute information (S1945). Here, as stated above, the attribute information is information indicating, for example, a normal vector of a current node.

[0527]

After that, the three-dimensional data encoding device entropy encodes an occupancy code of the current node using the selected coding table (S1946).

[0528]

Steps S1943 to S1946 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is completed (YES in S1942), the three-dimensional data encoding device outputs a bitstream including generated information (S1947).

[0529]

FIG. 71 is a flowchart of a three-dimensional data decoding process including an adaptive entropy decoding process using attribute information.

[0530]

First, the three-dimensional data decoding device obtains an inputted bitstream (S1951). Next, the three-dimensional data decoding device determines whether a decomposition process per unit length is completed (S1952).

[0531]

When the decomposition process per unit length is not completed (NO in S1952), the three-dimensional data decoding device generates an octree by performing the decomposition process on a current node (S1953).

[0532]

Then, the three-dimensional data encoding device obtains attribute information (S1954), and selects a coding table based on the obtained attribute information (S1955). Here, as stated above, the attribute information is information indicating, for example, a normal vector of a current node.

[0533]

After that, the three-dimensional data decoding device entropy decodes an occupancy code of the current node using the selected coding table (S1956).

[0534]

Steps S1953 to S1956 are repeated until the decomposition process per unit length is completed. When the decomposition process per unit length is

completed (YES in S1952), the three-dimensional data decoding device outputs three-dimensional points (S1957).

[0535]

FIG. 72 is a flowchart of the process of selecting a coding table using geometry information (S1905).

[0536]

The three-dimensional data encoding device may select a coding table to be used for entropy encoding of an occupancy code, using, as geometry information, information of a geometry group of a tree structure, for example. Here, information of a geometry group is information indicating a geometry group including a geometry pattern of a current node.

[0537]

As illustrated in FIG. 72, when a geometry group indicated by geometry information is geometry group 0 (YES in S1961), the three-dimensional data encoding device selects coding table 0 (S1962). When the geometry group indicated by the geometry information is geometry group 1 (YES in S1963), the three-dimensional data encoding device selects coding table 1 (S1964). In any other case (NO in S1963), the three-dimensional data encoding device selects coding table 2 (S1965).

[0538]

It should be noted that a method of selecting a coding table is not limited to the above. For example, when a geometry group indicated by geometry information is geometry group 2, the three-dimensional data encoding device may further select a coding table according to a value of the geometry group, such as using coding table 2.

[0539]

For example, a geometry group is determined using occupancy

information indicating whether a node neighboring a current node includes a point cloud. Geometry patterns that become the same shape by transform such as rotation being applied to may be included in the same geometry group. The three-dimensional data encoding device may select a geometry group using occupancy information of a node that neighbors a current node or is located around the current node, and belongs to the same layer as the current node. In addition, the three-dimensional data encoding device may select a geometry group using occupancy information of a node that belongs to a layer different from that of a current node. For example, the three-dimensional data encoding device may select a geometry group using occupancy information of a parent node, a node neighboring the parent node, or a node located around the parent node.

[0540]

It should be noted that the same applies to the process of selecting a coding table using geometry information (S1915) in the three-dimensional data decoding device.

[0541]

FIG. 73 is a flowchart of the process of selecting a coding table using structure information (S1925).

[0542]

The three-dimensional data encoding device may select a coding table to be used for entropy encoding of an occupancy code, using, as structure information, layer information of a tree structure, for example. Here, the layer information indicates, for example, a layer to which a current node belongs.

[0543]

As illustrated in FIG. 73, when a current node belongs to layer 0 (YES in S1971), the three-dimensional data encoding device selects coding table 0

(S1972). When the current node belongs to layer 1 (YES in S1973), the three-dimensional data encoding device selects coding table 1 (S1974). In any other case (NO in S1973), the three-dimensional data encoding device selects coding table 2 (S1975).

[0544]

It should be noted that a method of selecting a coding table is not limited to the above. For example, when a current node belongs to layer 2, the three-dimensional data encoding device may further select a coding table in accordance with the layer to which the current node belongs, such as using coding table 2.

[0545]

The same applies to the process of selecting a coding table using structure information (S1935) in the three-dimensional data decoding device.

[0546]

FIG. 74 is a flowchart of the process of selecting a coding table using attribute information (S1945).

[0547]

The three-dimensional data encoding device may select a coding table to be used for entropy encoding of an occupancy code, using, as attribute information, information about an object to which a current node belongs or information about a normal vector of the current node.

[0548]

As illustrated in FIG. 74, when a normal vector of a current node belongs to normal vector group 0 (YES in S1981), the three-dimensional data encoding device selects coding table 0 (S1982). When the normal vector of the current node belongs to normal vector group 1 (YES in S1983), the three-dimensional data encoding device selects coding table 1 (S1984). In any other case (NO in

S1983), the three-dimensional data encoding device selects coding table 2 (S1985).

[0549]

It should be noted that a method of selecting a coding table is not limited to the above. For example, when a normal vector of a current node belongs to normal vector group 2, the three-dimensional data encoding device may further select a coding table in accordance with a normal vector group to which the normal vector of the current belongs, such as using coding table 2.

[0550]

For example, the three-dimensional data encoding device selects a normal vector group using information about a normal vector of a current node. For example, the three-dimensional data encoding device determines, as the same normal vector group, normal vectors having a distance between normal vectors that is less than or equal to a predetermined threshold value.

[0551]

The information about the object to which the current node belongs may be information about, for example, a person, a vehicle, or a building.

[0552]

The following describes configurations of three-dimensional data encoding device 1900 and three-dimensional data decoding device 1910 according to the present embodiment. FIG. 75 is a block diagram of three-dimensional data encoding device 1900 according to the present embodiment. Three-dimensional data encoding device 1900 illustrated in FIG. 75 includes octree generator 1901, similarity information calculator 1902, coding table selector 1903, and entropy encoder 1904.

[0553]

Octree generator 1901 generates, for example, an octree from inputted

three-dimensional points, and generates an occupancy code for each node included in the octree. Similarity information calculator 1902 obtains, for example, similarity information that is geometry information, structure information, or attribute information of a current node. Coding table selector 1903 selects a context to be used for entropy encoding of an occupancy code, according to the similarity information of the current node. Entropy encoder 1904 generates a bitstream by entropy encoding the occupancy code using the selected context. It should be noted that entropy encoder 1904 may append, to the bitstream, information indicating the selected context.

[0554]

FIG. 76 is a block diagram of three-dimensional data decoding device 1910 according to the present embodiment. Three-dimensional data decoding device 1910 illustrated in FIG. 76 includes octree generator 1911, similarity information calculator 1912, coding table selector 1913, and entropy decoder 1914.

[0555]

Octree generator 1911 generates an octree in order from, for example, a lower layer to an upper layer using information obtained from entropy decoder 1914. Similarity information calculator 1912 obtains similarity information that is geometry information, structure information, or attribute information of a current node. Coding table selector 1913 selects a context to be used for entropy encoding of an occupancy code, according to the similarity information of the current node. Entropy decoder 1914 generates three-dimensional points by entropy decoding the occupancy code using the selected context. It should be noted that entropy decoder 1914 may obtain, by performing decoding, information of the selected context appended to a bitstream, and use the context indicated by the information.

[0556]

As illustrated in FIG. 63 to FIG. 65 above, the contexts are provided to the respective bits of the occupancy code. In other words, the three-dimensional data encoding device entropy encodes a bit sequence representing an N-ary (N is an integer greater than or equal to 2) tree structure of three-dimensional points included in three-dimensional data, using a coding table selected from coding tables. The bit sequence includes N-bit information for each node in the N-ary tree structure. The N-bit information includes N pieces of 1-bit information each indicating whether a three-dimensional point is present in a corresponding one of N child nodes of a corresponding node. In each of the coding tables, a context is provided to each bit of the N-bit information. The three-dimensional data encoding device entropy encodes each bit of the N-bit information using the context provided to the bit in the selected coding table.

[0557]

This enables the three-dimensional data encoding device to improve the coding efficiency by selecting a context for each bit.

[0558]

For example, in the entropy encoding, the three-dimensional data encoding device selects a coding table to be used from coding tables, based on whether a three-dimensional point is present in each of neighboring nodes of a current node. This enables the three-dimensional data encoding device to improve the coding efficiency by selecting a coding table based on whether the three-dimensional point is present in the neighboring node.

[0559]

For example, in the entropy encoding, the three-dimensional data encoding device (i) selects a coding table based on an arrangement pattern

indicating an arranged position of a neighboring node in which a three-dimensional point is present, among neighboring nodes, and (ii) selects the same coding table for arrangement patterns that become identical by rotation, among arrangement patterns. This enables the three-dimensional data encoding device to reduce an increase in the number of coding tables.

[0560]

For example, in the entropy encoding, the three-dimensional data encoding device selects a coding table to be used from coding tables, based on a layer to which a current node belongs. This enables the three-dimensional data encoding device to improve the coding efficiency by selecting a coding table based on the layer to which the current node belongs.

[0561]

For example, in the entropy encoding, the three-dimensional data encoding device selects a coding table to be used from coding tables, based on a normal vector of a current node. This enables the three-dimensional data encoding device to improve the coding efficiency by selecting a coding table based on the normal vector.

[0562]

For example, the three-dimensional data encoding device includes a processor and memory, and the processor performs the above process using the memory.

[0563]

The three-dimensional data decoding device entropy decodes a bit sequence representing an N-ary (N is an integer greater than or equal to 2) tree structure of three-dimensional points included in three-dimensional data, using a coding table selected from coding tables. The bit sequence includes N-bit information for each node in the N-ary tree structure. The N-bit information

includes N pieces of 1-bit information each indicating whether a three-dimensional point is present in a corresponding one of N child nodes of a corresponding node. In each of the coding tables, a context is provided to each bit of the N-bit information. The three-dimensional data decoding device entropy decodes each bit of the N-bit information using the context provided to the bit in the selected coding table.

[0564]

This enables the three-dimensional data decoding device to improve the coding efficiency by selecting a context for each bit.

[0565]

For example, in the entropy decoding, the three-dimensional data decoding device selects a coding table to be used from coding tables, based on whether a three-dimensional point is present in each of neighboring nodes of a current node. This enables the three-dimensional data decoding device to improve the coding efficiency by selecting a coding table based on whether the three-dimensional point is present in the neighboring node.

[0566]

For example, in the entropy decoding, the three-dimensional data decoding device (i) selects a coding table based on an arrangement pattern indicating an arranged position of a neighboring node in which a three-dimensional point is present, among neighboring nodes, and (ii) selects the same coding table for arrangement patterns that become identical by rotation, among arrangement patterns. This enables the three-dimensional data decoding device to reduce an increase in the number of coding tables.

[0567]

For example, in the entropy decoding, the three-dimensional data decoding device selects a coding table to be used from coding tables, based on a

layer to which a current node belongs. This enables the three-dimensional data decoding device to improve the coding efficiency by selecting a coding table based on the layer to which the current node belongs.

[0568]

For example, in the entropy decoding, the three-dimensional data decoding device selects a coding table to be used from coding tables, based on a normal vector of a current node. This enables the three-dimensional data decoding device to improve the coding efficiency by selecting a coding table based on the normal vector.

[0569]

For example, the three-dimensional data decoding device includes a processor and memory, and the processor performs the above process using the memory.

[0570]

EMBODIMENT 9

In the present embodiment, a method of controlling reference when an occupancy code is encoded will be described. It should be noted that although the following mainly describes an operation of a three-dimensional data encoding device, a three-dimensional data decoding device may perform the same process.

[0571]

FIG. 77 and FIG. 78 each are a diagram illustrating a reference relationship according to the present embodiment. Specifically, FIG. 77 is a diagram illustrating a reference relationship in an octree structure, and FIG. 78 is a diagram illustrating a reference relationship in a spatial region.

[0572]

In the present embodiment, when the three-dimensional data encoding

device encodes encoding information of a current node to be encoded (hereinafter referred to as a current node), the three-dimensional data encoding device refers to encoding information of each node in a parent node to which the current node belongs. In this regard, however, the three-dimensional data encoding device does not refer to encoding information of each node in another node (hereinafter referred to as a parent neighbor node) that is in the same layer as the parent node. In other words, the three-dimensional data encoding device disables or prohibits reference to a parent neighbor node.

[0573]

It should be noted that the three-dimensional data encoding device may permit reference to encoding information of a parent node (hereinafter also referred to as a grandparent node) of the parent node. In other words, the three-dimensional data encoding device may encode the encoding information of the current node by reference to the encoding information of each of the grandparent node and the parent node to which the current node belongs.

[0574]

Here, encoding information is, for example, an occupancy code. When the three-dimensional data encoding device encodes the occupancy code of the current node, the three-dimensional data encoding device refers to information (hereinafter referred to as occupancy information) indicating whether a point cloud is included in each node in the parent node to which the current node belongs. To put it in another way, when the three-dimensional data encoding device encodes the occupancy code of the current node, the three-dimensional data encoding device refers to an occupancy code of the parent node. On the other hand, the three-dimensional data encoding device does not refer to occupancy information of each node in a parent neighbor node. In other words, the three-dimensional data encoding device does not refer to an occupancy code

of the parent neighbor node. Moreover, the three-dimensional data encoding device may refer to occupancy information of each node in the grandparent node. In other words, the three-dimensional data encoding device may refer to the occupancy information of each of the parent node and the parent neighbor node.

[0575]

For example, when the three-dimensional data encoding device encodes the occupancy code of the current node, the three-dimensional data encoding device selects a coding table to be used for entropy encoding of the occupancy code of the current node, using the occupancy code of the grandparent node or the parent node to which the current node belongs. It should be noted that the details will be described later. At this time, the three-dimensional data encoding device need not refer to the occupancy code of the parent neighbor node. Since this enables the three-dimensional data encoding device to, when encoding the occupancy code of the current node, appropriately select a coding table according to information of the occupancy code of the parent node or the grandparent node, the three-dimensional data encoding device can improve the coding efficiency. Moreover, by not referring to the parent neighbor node, the three-dimensional data encoding device can suppress a process of checking the information of the parent neighbor node and reduce a memory capacity for storing the information. Furthermore, scanning the occupancy code of each node of the octree in a depth-first order makes encoding easy.

[0576]

The following describes an example of selecting a coding table using an occupancy code of a parent node. FIG. 79 is a diagram illustrating an example of a current node and neighboring reference nodes. FIG. 80 is a diagram illustrating a relationship between a parent node and nodes. FIG. 81 is a diagram illustrating an example of an occupancy code of the parent node. Here,

a neighboring reference node is a node referred to when a current node is encoded, among nodes spatially neighboring the current node. In the example shown in FIG. 79, the neighboring nodes belong to the same layer as the current node. Moreover, node X neighboring the current node in the x direction, node Y neighboring the current block in the y direction, and node Z neighboring the current block in the z direction are used as the reference neighboring nodes. In other words, one neighboring node is set as a reference neighboring node in each of the x, y, and z directions.

[0577]

It should be noted that the node numbers shown in FIG. 80 are one example, and a relationship between node numbers and node positions is not limited to the relationship shown in FIG. 80. Although node 0 is assigned to the lowest-order bit and node 7 is assigned to the highest-order bit in FIG. 81, assignments may be made in reverse order. In addition, each node may be assigned to any bit.

[0578]

The three-dimensional data encoding device determines a coding table to be used when the three-dimensional data encoding device entropy encodes an occupancy code of a current node, using the following equation, for example.

[0579]

$$CodingTable = (FlagX << 2) + (FlagY << 1) + (FlagZ)$$

[0580]

Here, CodingTable indicates a coding table for an occupancy code of a current node, and indicates one of values ranging from 0 to 7. FlagX is occupancy information of neighboring node X. FlagX indicates 1 when neighboring node X includes a point cloud (is occupied), and indicates 0 when it does not. FlagY is occupancy information of neighboring node Y. FlagY

indicates 1 when neighboring node Y includes a point cloud (is occupied), and indicates 0 when it does not. FlagZ is occupancy information of neighboring node Z. FlagZ indicates 1 when neighboring node Z includes a point cloud (is occupied), and indicates 0 when it does not.

[0581]

It should be noted that since information indicating whether a neighboring node is occupied is included in an occupancy code of a parent node, the three-dimensional data encoding device may select a coding table using a value indicated by the occupancy code of the parent node.

[0582]

From the foregoing, the three-dimensional data encoding device can improve the coding efficiency by selecting a coding table using the information indicating whether the neighboring node of the current node includes a point cloud.

[0583]

Moreover, as illustrated in FIG. 79, the three-dimensional data encoding device may select a neighboring reference node according to a spatial position of the current node in the parent node. In other words, the three-dimensional data encoding device may select a neighboring node to be referred to from the neighboring nodes, according to the spatial position of the current node in the parent node.

[0584]

Next, the following describes examples of configurations of the three-dimensional data encoding device and the three-dimensional data decoding device. FIG. 82 is a block diagram of three-dimensional data encoding device 2100 according to the present embodiment. Three-dimensional data encoding device 2100 illustrated in FIG. 82 includes octree generator 2101, geometry

information calculator 2102, coding table selector 2103, and entropy encoder 2104.

[0585]

Octree generator 2101 generates, for example, an octree from inputted three-dimensional points (a point cloud), and generates an occupancy code for each node included in the octree. Geometry information calculator 2102 obtains occupancy information indicating whether a neighboring reference node of a current node is occupied. For example, geometry information calculator 2102 obtains the occupancy information of the neighboring reference node from an occupancy code of a parent node to which the current node belongs. It should be noted that, as illustrated in FIG. 79, geometry information calculator 2102 may select a neighboring reference node according to a position of the current node in the parent node. In addition, geometry information calculator 2102 does not refer to occupancy information of each node in a parent neighbor node.

[0586]

Coding table selector 2103 selects a coding table to be used for entropy encoding of an occupancy code of the current node, using the occupancy information of the neighboring reference node calculated by geometry information calculator 2102. Entropy encoder 2104 generates a bitstream by entropy encoding the occupancy code using the selected coding table. It should be noted that entropy encoder 2104 may append, to the bitstream, information indicating the selected coding table.

[0587]

FIG. 83 is a block diagram of three-dimensional data decoding device 2110 according to the present embodiment. Three-dimensional data decoding device 2110 illustrated in FIG. 83 includes octree generator 2111, geometry

information calculator 2112, coding table selector 2113, and entropy decoder 2114.

[0588]

Octree generator 2111 generates an octree of a space (nodes) using header information of a bitstream etc. Octree generator 2111 generates an octree by, for example, generating a large space (a root node) using the size of a space along the x-axis, y-axis, and z-axis directions appended to the header information, and generating eight small spaces A (nodes A0 to A7) by dividing the space into two along each of the x-axis, y-axis, and z-axis directions. Nodes A0 to A7 are set as a current node in sequence.

[0589]

Geometry information calculator 2112 obtains occupancy information indicating whether a neighboring reference node of a current node is occupied. For example, geometry information calculator 2112 obtains the occupancy information of the neighboring reference node from an occupancy code of a parent node to which the current node belongs. It should be noted that, as illustrated in FIG. 79, geometry information calculator 2112 may select a neighboring reference node according to a position of the current node in the parent node. In addition, geometry information calculator 2112 does not refer to occupancy information of each node in a parent neighboring node.

[0590]

Coding table selector 2113 selects a coding table (a decoding table) to be used for entropy decoding of the occupancy code of the current node, using the occupancy information of the neighboring reference node calculated by geometry information calculator 2112. Entropy decoder 2114 generates three-dimensional points by entropy decoding the occupancy code using the selected coding table. It should be noted that coding table selector 2113 may obtain, by

performing decoding, information of the selected coding table appended to the bitstream, and entropy decoder 2114 may use a coding table indicated by the obtained information.

[0591]

Each bit of the occupancy code (8 bits) included in the bitstream indicates whether a corresponding one of eight small spaces A (nodes A0 to A7) includes a point cloud. Furthermore, the three-dimensional data decoding device generates an octree by dividing small space node A0 into eight small spaces B (nodes B0 to B7), and obtains information indicating whether each node of small space B includes a point cloud, by decoding the occupancy code. In this manner, the three-dimensional data decoding device decodes the occupancy code of each node while generating an octree by dividing a large space into small spaces.

[0592]

The following describes procedures for processes performed by the three-dimensional data encoding device and the three-dimensional data decoding device. FIG. 84 is a flowchart of a three-dimensional data encoding process in the three-dimensional data encoding device. First, the three-dimensional data encoding device determines (defines) a space (a current node) including part or whole of an inputted three-dimensional point cloud (S2101). Next, the three-dimensional data encoding device generates eight small spaces (nodes) by dividing the current node into eight (S2102). Then, the three-dimensional data encoding device generates an occupancy code for the current node according to whether each node includes a point cloud (S2103).

[0593]

After that, the three-dimensional data encoding device calculates (obtains) occupancy information of a neighboring reference node of the current

node from an occupancy code of a parent node of the current node (S2104). Next, the three-dimensional data encoding device selects a coding table to be used for entropy encoding, based on the calculated occupancy information of the neighboring reference node of the current node (S2105). Then, the three-dimensional data encoding device entropy encodes the occupancy code of the current node using the selected coding table (S2106).

[0594]

Finally, the three-dimensional data encoding device repeats a process of dividing each node into eight and encoding an occupancy code of the node, until the node cannot be divided (S2107). In other words, steps S2102 to S2106 are recursively repeated.

[0595]

FIG. 85 is a flowchart of a three-dimensional data decoding process in the three-dimensional data decoding device. First, the three-dimensional data decoding device determines (defines) a space (a current node) to be decoded, using header information of a bitstream (S2111). Next, the three-dimensional data decoding device generates eight small spaces (nodes) by dividing the current node into eight (S2112). Then, the three-dimensional data decoding device calculates (obtains) occupancy information of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S2113).

[0596]

After that, the three-dimensional data decoding device selects a coding table to be used for entropy decoding, based on the occupancy information of the neighboring reference node (S2114). Next, the three-dimensional data decoding device entropy decodes the occupancy code of the current node using the selected coding table (S2115).

[0597]

Finally, the three-dimensional data decoding device repeats a process of dividing each node into eight and decoding an occupancy code of the node, until the node cannot be divided (S2116). In other words, steps S2112 to S2115 are recursively repeated.

[0598]

Next, the following describes an example of selecting a coding table. FIG. 86 is a diagram illustrating an example of selecting a coding table. For example, as in coding table 0 shown in FIG. 86, the same context mode may be applied to occupancy codes. Moreover, a different context model may be assigned to each occupancy code. Since this enables assignment of a context model in accordance with a probability of appearance of an occupancy code, it is possible to improve the coding efficiency. Furthermore, a context mode that updates a probability table in accordance with an appearance frequency of an occupancy code may be used. Alternatively, a context model having a fixed probability table may be used.

[0599]

It should be noted that although the coding tables illustrated in FIG. 60 and FIG. 61 are used in the example shown in FIG. 86, the coding tables illustrated in FIG. 63 and FIG. 64 may be used instead.

[0600]

Hereinafter, Variation 1 of the present embodiment will be described. FIG. 87 is a diagram illustrating a reference relationship in the present variation. Although the three-dimensional data encoding device does not refer to the occupancy code of the parent neighbor node in the above-described embodiment, the three-dimensional data encoding device may switch whether to refer to an occupancy code of a parent neighbor node, according to a specific

condition.

[0601]

For example, when the three-dimensional data encoding device encodes an octree while scanning the octree breadth-first, the three-dimensional data encoding device encodes an occupancy code of a current node by reference to occupancy information of a node in a parent neighbor node. In contrast, when the three-dimensional data encoding device encodes the octree while scanning the octree depth-first, the three-dimensional data encoding device prohibits reference to the occupancy information of the node in the parent neighbor node. By appropriately selecting a referable node according to the scan order (encoding order) of nodes of the octree in the above manner, it is possible to improve the coding efficiency and reduce the processing load.

[0602]

It should be noted that the three-dimensional data encoding device may append, to a header of a bitstream, information indicating, for example, whether an octree is encoded breadth-first or depth-first. FIG. 88 is a diagram illustrating an example of a syntax of the header information in this case. octree_scan_order shown in FIG. 88 is encoding order information (an encoding order flag) indicating an encoding order for an octree. For example, when octree_scan_order is 0, breadth-first is indicated, and when octree_scan_order is 1, depth-first is indicated. Since this enables the three-dimensional data decoding device to determine whether a bitstream has been encoded breadth-first or depth-first by reference to octree_scan_order, the three-dimensional data decoding device can appropriately decode the bitstream

[0603]

Moreover, the three-dimensional data encoding device may append, to header information of a bitstream, information indicating whether to prohibit

reference to a parent neighbor node. FIG. 89 is a diagram illustrating an example of a syntax of the header information in this case. limit_refer_flag is prohibition switch information (a prohibition switch flag) indicating whether to prohibit reference to a parent neighbor node. For example, when limit_refer_flag is 1, prohibition of reference to the parent neighbor node is indicated, and when limit_refer_flag is 0, no reference limitation (permission of reference to the parent neighbor node) is indicated.

[0604]

In other words, the three-dimensional data encoding device determines whether to prohibit the reference to the parent neighbor node, and selects whether to prohibit or permit the reference to the parent neighbor node, based on a result of the above determination. In addition, the three-dimensional data encoding device generates a bitstream including prohibition switch information that indicates the result of the determination and indicates whether to prohibit the reference to the parent neighbor node.

[0605]

The three-dimensional data decoding device obtains, from a bitstream, prohibition switch information indicating whether to prohibit reference to a parent neighbor node, and selects whether to prohibit or permit the reference to the parent neighbor node, based on the prohibition switch information.

[0606]

This enables the three-dimensional data encoding device to control the reference to the parent neighbor node and generate the bitstream. That also enables the three-dimensional data decoding device to obtain, from the header of the bitstream, the information indicating whether to prohibit the reference to the parent neighbor node.

[0607]

Although the process of encoding an occupancy code has been described as an example of an encoding process in which reference to a parent neighbor node is prohibited in the present embodiment, the present disclosure is not necessarily limited to this. For example, the same method can be applied when other information of a node of an octree is encoded. For example, the method of the present embodiment may be applied when other attribute information, such as a color, a normal vector, or a degree of reflection, added to a node is encoded. Additionally, the same method can be applied when a coding table or a predicted value is encoded.

[0608]

Hereinafter, Variation 2 of the present embodiment will be described. In the above description, as illustrated in FIG. 79, the example in which the three reference neighboring nodes are used is given, but four or more reference neighboring nodes may be used. FIG. 90 is a diagram illustrating an example of a current node and neighboring reference nodes.

[0609]

For example, the three-dimensional data encoding device calculates a coding table to be used when the three-dimensional data encoding device entropy encodes an occupancy code of the current node shown in FIG. 90, using the following equation.

[0610]

CodingTable = (FlagX0 << 3) + (FlagX1 << 2) + (FlagY << 1) + (FlagZ)

[0611]

Here, CodingTable indicates a coding table for an occupancy code of a current node, and indicates one of values ranging from 0 to 15. FlagXN is occupancy information of neighboring node XN (N = 0.. 1). FlaxXN indicates 1 when neighboring node XN includes a point cloud (is occupied), and indicates 0

when it does not. FlagY is occupancy information of neighboring node Y. FlagY indicates 1 when neighboring node Y includes a point cloud (is occupied), and indicates 0 when it does not. FlagZ is occupancy information of neighboring node Z. FlagZ indicates 1 when neighboring node Z includes a point cloud (is occupied), and indicates 0 when it does not.

[0612]

At this time, when a neighboring node, for example, neighboring node X0 in FIG. 90, is unreferable (prohibited from being referred to), the three-dimensional data encoding device may use, as a substitute value, a fixed value such as 1 (occupied) or 0 (unoccupied).

[0613]

FIG. 91 is a diagram illustrating an example of a current node and neighboring reference nodes. As illustrated in FIG. 91, when a neighboring node is unreferable (prohibited from being referred to), occupancy information of the neighboring node may be calculated by reference to an occupancy code of a grandparent node of the current node. For example, the three-dimensional data encoding device may calculate FlagX0 in the above equation using occupancy information of neighboring node G0 instead of neighboring node X0 illustrated in FIG. 91, and may determine a value of a coding table using calculated FlagX0. It should be noted that neighboring node G0 illustrated in FIG. 91 is a neighboring node occupancy or unoccupancy of which can be determined using the occupancy code of the grandparent node. Neighboring node X1 is a neighboring node occupancy or unoccupancy of which can be determined using an occupancy code of a parent node.

[0614]

Hereinafter, Variation 3 of the present embodiment will be described. FIG. 92 and FIG. 93 each are a diagram illustrating a reference relationship

according to the present variation. Specifically, FIG. 92 is a diagram illustrating a reference relationship in an octree structure, and FIG. 93 is a diagram illustrating a reference relationship in a spatial region.

[0615]

In the present variation, when the three-dimensional data encoding device encodes encoding information of a current node to be encoded (hereinafter referred to as current node 2), the three-dimensional data encoding device refers to encoding information of each node in a parent node to which current node 2 belongs. In other words, the three-dimensional data encoding device permits reference to information (e.g., occupancy information) of a child node of a first node, among neighboring nodes, that has the same parent node as a current node. For example, when the three-dimensional data encoding device encodes an occupancy code of current node 2 illustrated in FIG. 92, the three-dimensional data encoding device refers to an occupancy code of a node in the parent node to which current node 2 belongs, for example, the current node illustrated in FIG. 92. As illustrated in FIG. 93, the occupancy code of the current node illustrated in FIG. 92 indicates, for example, whether each node in the current node neighboring current node 2 is occupied. Accordingly, since the three-dimensional data encoding device can select a coding table for the occupancy code of current node 2 in accordance with a more particular shape of the current node, the three-dimensional data encoding device can improve the coding efficiency.

[0616]

The three-dimensional data encoding device may calculate a coding table to be used when the three-dimensional data encoding device entropy encodes the occupancy code of current node 2, using the following equation, for example.

[0617]

CodingTable = (FlagX1 << 5) + (FlagX2 << 4) + (FlagX3 << 3) + (FlagX4 << 2) + (FlagY << 1) + (FlagZ)

[0618]

Here, CodingTable indicates a coding table for an occupancy code of current node 2, and indicates one of values ranging from 0 to 63. FlagXN is occupancy information of neighboring node XN (N = 1.. 4). FlagXN indicates 1 when neighboring node XN includes a point cloud (is occupied), and indicates 0 when it does not. FlagY is occupancy information of neighboring node Y. FlagY indicates 1 when neighboring node Y includes a point cloud (is occupied), and indicates 0 when it does not. FlagZ is occupancy information of neighboring node Z. FlagZ indicates 1 when neighboring node Z includes a point cloud (is occupied), and indicates 0 when it does not.

[0619]

It should be noted that the three-dimensional data encoding device may change a method of calculating a coding table, according to a node position of current node 2 in the parent node.

[0620]

When reference to a parent neighbor node is not prohibited, the three-dimensional data encoding device may refer to encoding information of each node in the parent neighbor node. For example, when the reference to the parent neighbor node is not prohibited, reference to information (e.g., occupancy information) of a child node of a third node having a different parent node from that of a current node. In the example illustrated in FIG. 91, for example, the three-dimensional data encoding device obtains occupancy information of a child node of neighboring node X0 by reference to an occupancy code of neighboring node X0 having a different parent node from that of the current node. The

three-dimensional data encoding device selects a coding table to be used for entropy encoding of an occupancy code of the current node, based on the obtained occupancy information of the child node of neighboring node X0.

[0621]

As stated above, the three-dimensional data encoding device according to the present embodiment encodes information (e.g., an occupancy code) of a current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, where N is an integer greater than or equal to 2. As illustrated in FIG. 77 and FIG. 78, in the encoding, the three-dimensional data encoding device permits reference to information (e.g., occupancy information) of a first node included in neighboring nodes spatially neighboring the current node, and prohibits reference to information of a second node included in the neighboring nodes, the first node having a same parent node as the current node, the second node having a different parent node from the parent node of the current node. To put it another way, in the encoding, the three-dimensional data encoding device permits reference to information (e.g., an occupancy code) of the parent node, and prohibits reference to information (e.g., an occupancy code) of another node (a parent neighbor node) in the same layer as the parent node.

[0622]

With this, the three-dimensional data encoding device can improve coding efficiency by reference to the information of the first node included in the neighboring nodes spatially neighboring the current node, the first node having the same parent node as the current node. Besides, the three-dimensional data encoding device can reduce a processing amount by not reference to the information of the second node included in the neighboring nodes, the second node having a different parent node from the parent node of the current node.

In this manner, the three-dimensional data encoding device can not only improve the coding efficiency but also reduce the processing amount.

[0623]

For example, the three-dimensional data encoding device further determines whether to prohibit the reference to the information of the second node. In the encoding, the three-dimensional data encoding device selects whether to prohibit or permit the reference to the information of the second node, based on a result of the determining. Moreover, the three-dimensional data encoding device generates a bit stream including prohibition switch information (e.g., limit_refer_flag shown in FIG. 89) that indicates the result of the determining and indicates whether to prohibit the reference to the information of the second node.

[0624]

With this, the three-dimensional data encoding device can select whether to prohibit the reference to the information of the second node. In addition, a three-dimensional data decoding device can appropriately perform a decoding process using the prohibition switch information.

[0625]

For example, the information of the current node is information (e.g., an occupancy code) that indicates whether a three-dimensional point is present in each of child nodes belonging to the current node. The information of the first node is information (the occupancy information of the first node) that indicates whether a three-dimensional point is present in the first node. The information of the second node is information (the occupancy information of the second node) that indicates whether a three-dimensional point is present in the second node.

[0626]

For example, in the encoding, the three-dimensional data encoding device selects a coding table based on whether the three-dimensional point is present in the first node, and entropy encodes the information (e.g., the occupancy code) of the current node using the coding table selected.

[0627]

For example, as illustrated in FIG. 92 and FIG. 93, in the encoding, the three-dimensional data encoding device permits reference to information (e.g., occupancy information) of a child node of the first node, the child node being included in the neighboring nodes.

[0628]

With this, since the three-dimensional data encoding device enables reference to more detailed information of a neighboring node, the three-dimensional data encoding device can improve the coding efficiency.

[0629]

For example, as illustrated in FIG. 79, in the encoding, the three-dimensional data encoding device selects a neighboring node to be referred to from the neighboring nodes according to a spatial position of the current node in the parent node.

[0630]

With this, the three-dimensional data encoding device can refer to an appropriate neighboring node according to the spatial position of the current node in the parent node.

[0631]

For example, the three-dimensional data encoding device includes a processor and memory, and the processor performs the above process using the memory.

[0632]

The three-dimensional data decoding device according to the present embodiment decodes information (e.g., an occupancy code) of a current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, where N is an integer greater than or equal to 2. As illustrated in FIG. 77 and FIG. 78, in the decoding, the three-dimensional data decoding device permits reference to information (e.g., occupancy information) of a first node included in neighboring nodes spatially neighboring the current node, and prohibits reference to information of a second node included in the neighboring nodes, the first node having a same parent node as the current node, the second node having a different parent node from the parent node of the current node. To put it another way, in the decoding, the three-dimensional data decoding device permits reference to information (e.g., an occupancy code) of the parent node, and prohibits reference to information (e.g., an occupancy code) of another node (a parent neighbor node) in the same layer as the parent node.

[0633]

With this, the three-dimensional data decoding device can improve coding efficiency by reference to the information of the first node included in the neighboring nodes spatially neighboring the current node, the first node having the same parent node as the current node. Besides, the three-dimensional data decoding device can reduce a processing amount by not reference to the information of the second node included in the neighboring nodes, the second node having a different parent node from the parent node of the current node. In this manner, the three-dimensional data decoding device can not only improve the coding efficiency but also reduce the processing amount.

[0634]

For example, the three-dimensional data decoding device further

obtains, from a bitstream, prohibition switch information (e.g., limit_refer_flag shown in FIG. 89) indicating whether to prohibit the reference to the information of the second node. In the decoding, the three-dimensional data decoding device selects whether to prohibit or permit the reference to the information of the second node, based on the prohibition switch information.

[0635]

With this, the three-dimensional data decoding device can appropriately perform a decoding process using the prohibition switch information.

[0636]

For example, the information of the current node is information (e.g., an occupancy code) that indicates whether a three-dimensional point is present in each of child nodes belonging to the current node. The information of the first node is information (the occupancy information of the first node) that indicates whether a three-dimensional point is present in the first node. The information of the second node is information (the occupancy information of the second node) that indicates whether a three-dimensional point is present in the second node.

[0637]

For example, in the decoding, the three-dimensional data encoding device selects a coding table based on whether the three-dimensional point is present in the first node, and entropy decodes the information (e.g., the occupancy code) of the current node using the coding table selected.

[0638]

For example, as illustrated in FIG. 92 and FIG. 93, in the decoding, the three-dimensional data decoding device permits reference to information (e.g., occupancy information) of a child node of the first node, the child node being included in the neighboring nodes.

[0639]

With this, since the three-dimensional data decoding device enables reference to more detailed information of a neighboring node, the three-dimensional data decoding device can improve the coding efficiency.

[0640]

For example, as illustrated in FIG. 79, in the decoding, the three-dimensional data decoding device selects a neighboring node to be referred to from the neighboring nodes according to a spatial position of the current node in the parent node.

[0641]

With this, the three-dimensional data decoding device can refer to an appropriate neighboring node according to the spatial position of the current node in the parent node.

[0642]

For example, the three-dimensional data decoding device includes a processor and memory, and the processor performs the above process using the memory.

[0643]

EMBODIMENT 10

A method of reducing the number of coding tables will be described in the present embodiment.

[0644]

When a coding table is provided for each of combinations of positions (eight patterns) of a current node in a parent node and occupancy state patterns (eight patterns) of three neighboring nodes of the current node, $8 \times 8 = 64$ coding tables are necessary. It should be noted that the combination is hereinafter also referred to as a neighbor occupancy pattern. In addition, a node in an

occupancy state is also referred to as an occupied node. Additionally, a neighboring node in an occupancy state is also referred to as a neighboring occupied node.

[0645]

In the present embodiment, a total number of coding tables is reduced by assigning the same coding table to similar neighbor occupancy patterns. Specifically, neighbor occupancy patterns are grouped by performing a conversion process on the neighbor occupancy patterns. More specifically, neighbor occupancy patterns that become the same due to a conversion process are grouped into the same group. In addition, one coding table is assigned to each group.

[0646]

For example, as shown in FIG. 94, translation along an x-axis, y-axis, or z-axis is used as a conversion process. Alternatively, as shown in FIG. 95, rotation along the x-axis, y-axis, or z-axis (about the x-axis, y-axis, or z-axis) is used.

[0647]

Moreover, grouped neighbor occupancy patterns may be classified using the following rule. For example, as shown in FIG. 96, a rule may be that a surface in which occupied nodes and a current node are present is horizontal or vertical to a coordinate plane (an x-y plane, y-z plane, or x-z plane). Alternatively, as shown in FIG. 97, a rule may be that an adjacent surface is defined by directions in which neighboring occupied nodes are present relative to a current node.

[0648]

FIG. 98 is a diagram illustrating an example of translation along each of the x-axis, y-axis, and z-axis. FIG. 99 is a diagram illustrating examples of

rotation along the x-axis. FIG. 100 is a diagram illustrating examples of rotation along the y-axis. FIG. 101 is a diagram illustrating examples of rotation along the z-axis. FIG. 102 is a diagram illustrating examples of horizontality and verticality to a coordinate plane. FIG. 103 is a diagram illustrating examples of an adjacent surface pattern.

[0649]

FIG. 104 is a diagram illustrating an example of dividing 64 neighbor occupancy patterns into 6 groups. In other words, six coding tables are used in this example. Besides, the number of coding tables is reduced by rotation along the z-axis in the example.

[0650]

Specifically, as shown in FIG. 104, a neighbor occupancy pattern that includes a zero neighboring occupied node among three neighboring nodes (i.e., an occupancy number is zero) is classified into group 0. A neighbor occupancy pattern that includes one neighboring occupied node among three neighboring nodes (i.e., an occupancy number is one) and is horizontal to the x-y plane is classified into group 1. A neighbor occupancy pattern that includes one neighboring occupied node among three neighboring nodes (i.e., an occupancy number is one) and is vertical to the x-y plane is classified into group 2. A neighbor occupancy pattern that includes two neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is two) and is vertical to the x-y plane is classified into group 3. A neighbor occupancy pattern that includes two neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is two) and is horizontal to the x-y plane is classified into group 4. A neighbor occupancy pattern that includes three neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is three) is classified into group 5.

[0651]

Moreover, one coding table is used for each of the groups. Furthermore, each group includes neighbor occupancy patterns that become the same due to the rotation along the z-axis. It should be noted that regarding group 4, translation along the z-axis is also considered.

[0652]

For example, in the case of a three-dimensional map having an x-y plane as a ground surface, buildings similar in shape are often present on the x-y plane. In such a case, for example, when building A is rotated in a z-axis direction, building A is likely to overlap building B. In this case, there is a possibility of improving the coding efficiency when building B is encoded, by using, when an occupancy code of building B is encoded, a coding table for an occupancy code updated by encoding building A. Since, regarding coding tables obtained by rotating shapes in the z-axis direction as belonging to the same group, it is possible to update the coding tables without an effect of the rotation in the z-axis direction, it is possible to improve the coding efficiency. Moreover, for example, when building C is translated on the x-y plane, building C is likely to overlap building D. In this case, there is a possibility of improving the coding efficiency when building D is encoded, by using, when an occupancy code of building D is encoded, a coding table for an occupancy code updated by encoding building C. Since, regarding coding tables relating to the translation on the x-y plane as belonging to the same group, it is possible to update the coding tables without an effect of the translation on the x-y plane, it is possible to improve the coding efficiency.

[0653]

FIG. 105 is a diagram illustrating an example of dividing 64 neighbor occupancy patterns into 8 groups. In other words, eight coding tables are used

in this example. Besides, the number of coding tables is reduced by rotation along the z-axis and using adjacent surfaces in the example.

[0654]

Specifically, as shown in FIG. 105, a neighbor occupancy pattern that includes a zero neighboring occupied node among three neighboring nodes (i.e., an occupancy number is zero) is classified into group 0. A neighbor occupancy pattern that includes one neighboring occupied node among three neighboring nodes (i.e., an occupancy number is one) and is horizontal to the x-y plane is classified into group 1. A neighbor occupancy pattern that includes one neighboring occupied node among three neighboring nodes (i.e., an occupancy number is one) and is vertical to the x-y plane is classified into group 2. A neighbor occupancy pattern that includes two neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is two), is vertical to the x-y plane, and includes an adjacent surface in the z direction (i.e., the neighboring occupied nodes of the current node are present in the z direction) is classified into group 3. A neighbor occupancy pattern that includes two neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is two), is vertical to the x-y plane, and includes an adjacent surface in the −z direction (i.e., the neighboring occupied nodes of the current node are present in the −z direction) is classified into group 4.

[0655]

A neighbor occupancy pattern that includes two neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is two) and is horizontal to the x-y plane is classified into group 5. A neighbor occupancy pattern that includes three neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is three) and includes an adjacent surface in the z-direction is classified into group 6. A neighbor occupancy

pattern that includes three neighboring occupied nodes among three neighboring nodes (i.e., an occupancy number is three) and includes an adjacent surface in the −z-direction is classified into group 7.

[0656]

Moreover, one coding table is used for each of the groups. Furthermore, each group includes neighbor occupancy patterns that become the same due to the rotation along the z-axis. It should be noted that regarding group 5, translation along the z-axis is also considered.

[0657]

A mapping rule can be created using each of the examples shown in FIG. 104 and FIG. 105. FIG. 106 is a diagram illustrating an example of a mapping rule (a conversion table) when 3 coding tables are used for 64 neighbor occupancy patterns. In the example shown in FIG. 106, one of indexes (table 0 to table 2) of the 3 coding tables is assigned to each of the 64 neighbor occupancy patterns (pattern 0 to pattern 63). This rule is represented by, for example, a look-up table (LUT).

[0658]

Moreover, a mapping rule may be created by adding a new rule to a given mapping rule or deleting part of a rule. In other words, neighbor occupancy patterns grouped according to a first rule may be further grouped according to a second rule. To put it another way, neighbor occupancy patterns may be assigned to first coding tables using a first conversion table, the first coding tables may be assigned to second coding tables using a second conversion table, and arithmetic encoding or arithmetic decoding may be performed using the second coding tables. For example, the classification shown in FIG. 104 may be performed by further combining some of classified groups after the classification shown in FIG. 105 is performed.

[0659]

FIG. 107 is a diagram illustrating an example of a conversion table for performing such classification. Coding table index 1 shown in FIG. 107 indicates an index of a coding table derived according to a given mapping rule, and coding table index 2 shown in FIG. 107 indicates an index of a coding table representing a new mapping rule.

[0660]

For example, coding table index 1 indicates one of indexes (table 0 to table 7) of eight coding tables that are indexes of coding tables obtained by the classification shown in FIG. 105. In addition, coding table index 2 indicates one of indexes (table 0 to table 5) of six coding tables corresponding to the classification shown in FIG. 104.

[0661]

Specifically, since group 4 and group 5 shown in FIG. 105 correspond to group 4 shown in FIG. 104, as shown in FIG. 107, table 4 and table 5 indicated by coding table index 1 are assigned to table 4 indicated by coding table index 2. Likewise, table 6 and table 7 indicated by coding table index 1 are assigned to table 5 indicated by coding table index 2.

[0662]

The following describes an outline of a mapping process. A mapping rule is used to identify a unique index of a coding table.

[0663]

FIG. 108 is a diagram illustrating an outline of a mapping process for determining an index of a coding table from 64 neighbor occupancy patterns. As shown in FIG. 108, a neighbor occupancy pattern including a position of a current node is inputted to a mapping rule, and a table index (an index of a coding table) is outputted. The number of patterns is reduced by the mapping

rule. For example, the mapping rule shown in FIG. 106 is used as the mapping rule. As shown in FIG. 106, the same table index is assigned to different neighbor occupancy patterns.

[0664]

Next, entropy encoding is performed using the coding table to which the obtained table index is assigned.

[0665]

FIG. 109 is a diagram illustrating an outline of a mapping process when a table index is given. As shown in FIG. 109, table index 1 is inputted to a mapping rule, and table index 2 is outputted. The number of patterns is reduced by the mapping rule. For example, the mapping rule shown in FIG. 107 is used as the mapping rule. As shown in FIG. 107, same table index 2 is assigned to different table indexes 1.

[0666]

Next, entropy encoding is performed using the coding table to which obtained table index 2 is assigned.

[0667]

The following describes configurations of a three-dimensional data encoding device and a three-dimensional data decoding device according to the present embodiment. FIG. 110 is a block diagram of three-dimensional data encoding device 3600 according to the present embodiment. Three-dimensional data encoding device 3600 shown in FIG. 110 includes octree generator 3601, geometry information calculator 3602, index generator 3603, coding table selector 3604, and entropy encoder 3605.

[0668]

Octree generator 3601 generates, for example, an octree from inputted three-dimensional points (a point cloud), and generates an occupancy code of

each node included in the octree. Geometry information calculator 3602 obtains occupancy information indicating whether a neighboring reference node of a current node is occupied. For example, geometry information calculator 3602 calculates occupancy information of a neighboring reference node from an occupancy code of a parent node to which a current node belongs. It should be noted that geometry information calculator 3602 may change a neighboring reference node according to a position of the current node in the parent node. In addition, geometry information calculator 3602 need not refer to occupancy information of each node in a neighboring parent node.

[0669]

Index generator 3603 generates an index of a coding table using neighboring information (e.g., a neighbor occupancy pattern).

[0670]

Coding table selector 3604 selects a coding table to be used for entropy encoding an occupancy code of the current node using the index of the coding table generated by index generator 3603.

[0671]

An occupancy code is encoded as a decimal number or a binary number. For example, when binary encoding is used, the index of the coding table generated by index generator 3603 is used in selecting binary context to be used for entropy encoding by entropy encoder 3605. Moreover, when a decimal number or M-ary encoding is used, M-ary context is selected using an index of a coding table.

[0672]

Entropy encoder 3605 generates a bitstream by entropy encoding the occupancy code using the selected coding table. In addition, entropy encoder 3605 may append information indicating the selected coding table to the

bitstream.

[0673]

FIG. 111 is a block diagram of three-dimensional data decoding device 3610 according to the present embodiment. Three-dimensional data decoding device 3610 shown in FIG. 111 includes octree generator 3611, geometry information calculator 3612, index generator 3613, coding table selector 3614, and entropy decoder 3615.

[0674]

Octree generator 3611 generates an octree of a space (nodes) using, for example, the header information of a bitstream. For example, octree generator 3611 generates a large space (a root node) using the size of a space along the x-axis, y-axis, and z-axis directions appended to the header information, and generates an octree by generating eight small spaces A (nodes A0 to A7) by dividing the space into two along each of the x-axis, y-axis, and z-axis directions. In addition, nodes A0 to A7 are set as a current node in sequence.

[0675]

Geometry information calculator 3612 obtains occupancy information indicating whether a neighboring reference node of a current node is occupied. For example, geometry information calculator 3612 calculates occupancy information of a neighboring reference node from an occupancy code of a parent node to which a current node belongs. It should be noted that geometry information calculator 3612 may change a neighboring reference node according to a position of the current node in the parent node. In addition, geometry information calculator 3612 need not refer to occupancy information of each node in a neighboring parent node.

[0676]

Index generator 3613 generates an index of a coding table using

neighboring information (e.g., a neighbor occupancy pattern).

[0677]

Coding table selector 3614 selects a coding table to be used for entropy decoding an occupancy code of the current node using the index of the coding table generated by index generator 3613.

[0678]

An occupancy code is decoded as a decimal number or a binary number. For example, when binary encoding is used, an index of a coding table mapped in a previous block is used in selecting binary context to be used for entropy decoding the next block. Moreover, when a decimal number or M-ary encoding is used, M-ary context is selected using an index of a coding table.

[0679]

Entropy decoder 3615 generates three-dimensional points (a point cloud) by entropy decoding the occupancy code using the selected coding table. It should be noted that entropy decoder 3615 may obtain information of the selected coding table appended to the bitstream, by performing decoding, and use the coding table indicated by the obtained information.

[0680]

Each bit of an occupancy code (8 bits) included in a bitstream indicates whether a corresponding one of eight small spaces A (node A0 to node A7) includes a point cloud. The three-dimensional data decoding device further generates an octree by dividing small space node A0 into eight small spaces B (node B0 to node B7), and obtains information indicating whether each node of small spaces B includes a point cloud by decoding the occupancy code. As stated above, the three-dimensional data decoding device decodes an occupancy code of each node while generating an octree by dividing a large space into small spaces.

[0681]

The following describes a flow of processing performed by each of the three-dimensional data encoding device and the three-dimensional data decoding device. FIG. 112 is a flowchart of a three-dimensional data encoding process in the three-dimensional data encoding device. First, the three-dimensional data encoding device defines a space (a current node) including part or all of an inputted three-dimensional point cloud (S3601). Next, the three-dimensional data encoding device generates eight small spaces (nodes) by dividing the current node into eight (S3602). Then, the three-dimensional data encoding device generates an occupancy code of the current node according to whether each node includes a point cloud (S3603).

[0682]

After that, the three-dimensional data encoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3604).

[0683]

Next, the three-dimensional data encoding device converts the neighbor occupancy pattern into an index of a coding table (S3605). Then, the three-dimensional data encoding device selects a coding table to be used for entropy encoding, based on the index (S3606).

[0684]

After that, the three-dimensional data encoding device entropy encodes the occupancy code of the current node using the selected coding table (S3607).

[0685]

Finally, the three-dimensional data encoding device repeats a process of dividing each node into eight and encoding an occupancy code of each node until

each node cannot be divided (S3608). In other words, steps S3602 to S3607 are repeated recursively.

[0686]

FIG. 113 is a flowchart of a three-dimensional data decoding process in the three-dimensional data decoding device. First, the three-dimensional data decoding device defines a space (a current node) to be decoded, using the header information of a bitstream (S3611). Next, the three-dimensional data decoding device generates eight small spaces (nodes) by dividing the current node into eight (S3612). Then, the three-dimensional data decoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3613).

[0687]

After that, the three-dimensional data decoding device converts the neighbor occupancy pattern into an index of a coding table (S3614). Next, the three-dimensional data decoding device selects a coding table to be used for entropy decoding, based on the index (S3615). Then, the three-dimensional data decoding device entropy decodes an occupancy code of the current node using the selected coding table (S3616).

[0688]

Finally, the three-dimensional data decoding device repeats a process of dividing each node into eight and decoding an occupancy code of each node until each node cannot be divided (S3617). In other words, steps S3612 to S3616 are repeated recursively.

[0689]

The following describes coding modes. In mode 1, an occupancy code of a parent node is referred to, neighboring nodes of the parent node are searched,

and neighbor occupancy pattern n is obtained. Index $i_1 = f(n)$ of a coding table is obtained using mapping rule f.

[0690]

In mode 2, occupancy code c of a parent node is referred to. Index $i_2 = g(c)$ of a coding table is obtained using mapping rule g.

[0691]

In mode 3, occupancy code c of a parent node is referred to. Index $i_3 = h(c)$ of a coding table is obtained using mapping rule h.

[0692]

In mode 1, two pieces of information from two sources are used as neighbor occupancy information. The first information is occupancy information of a neighboring node in a parent node obtained from an occupancy code of the parent node. The second information is occupancy information of a neighboring node outside the parent node, and is obtained by searching neighboring nodes of the parent node. In other words, this information is occupancy information of, among neighboring nodes of a current node, a neighboring node belonging to a parent node different from a parent node of the current node. Hereinafter, this process of searching neighboring nodes of a parent node is also referred to as search parent neighbor.

[0693]

In mode 2 and mode 3, occupancy information of a neighboring node in a parent node obtained from an occupancy code of the parent node is used, and occupancy information of a neighboring node outside the parent node is not used. Besides, mode 2 and mode 3 differ in mapping rule for obtaining an index of a coding table.

[0694]

Moreover, in order to develop methods for reducing neighbor occupancy

patterns, modes mutually differing in mapping rule may be added. For example, a mapping rule is obtained by combining the rules shown in FIG. 94 to FIG. 103, etc.

[0695]

The following describes examples of a coding mode. In coding mode 1 (CodingMode = 1), 64 neighbor occupancy patterns are obtained by referring to an occupancy code of a parent node and performing search parent neighbor. Here, the 64 neighbor occupancy patterns are a combination ($2^6 = 64$) of pieces of occupancy information of 6 neighboring nodes, and a position of a current node is not considered. In addition, a look-up table that converts the 64 neighbor occupancy patterns into indexes of ten coding tables is used.

[0696]

In coding mode 2 (CodingMode = 2), 64 neighbor occupancy patterns for which a position of a current node is considered are obtained by referring to an occupancy code of a parent node. Here, the 64 neighbor occupancy patterns are a combination ($8 \times 8 = 64$) of positions (8 patterns) of the current node in the parent node and pieces of occupancy information of three neighboring nodes ($2^3 = 8$). In addition, a look-up table that converts the 64 neighbor occupancy patterns into indexes of 6 coding tables is used.

[0697]

In coding mode 3 (CodingMode = 3), 64 neighbor occupancy patterns for which a position of a current node is considered are obtained by referring to an occupancy code of a parent node. In addition, a look-up table that converts the 64 neighbor occupancy patterns into indexes of 8 coding tables is used.

[0698]

Furthermore, as another example of a coding mode (CodingMode), for example, the three-dimensional data encoding device appends, to a bitstream, a

search flag (search_flag) indicating whether to perform search parent neighbor. When search_flag = 1, the three-dimensional data encoding device may calculate a neighbor occupancy pattern by performing search parent neighbor, generate an index of one of N coding tables from a value of the neighbor occupancy pattern, and perform arithmetic encoding on an occupancy code using the coding table having the index.

[0699]

When search_flag = 0, the three-dimensional data encoding device may calculate a neighbor occupancy pattern without performing search parent neighbor, generate an index of one of M coding tables from a value of the neighbor occupancy pattern, and perform arithmetic encoding on an occupancy code using the coding table having the index, M being an integer less than or equal to N. It is possible to control a balance between the coding efficiency and the amount of processing by selecting search parent neighbor and a coding table to be used according to a value of search_flag in the above manner.

[0700]

The following describes a flow of processing performed by each of the three-dimensional data encoding device and the three-dimensional data decoding device when a coding mode is selected.

[0701]

FIG. 114 is a flowchart of a three-dimensional data encoding process in the three-dimensional data encoding device. First, the three-dimensional data encoding device defines a space (a current node) including part or all of an inputted three-dimensional point cloud (S3621). Next, the three-dimensional data encoding device generates eight small spaces (nodes) by dividing the current node into eight (S3622). Then, the three-dimensional data encoding device generates an occupancy code of the current node according to whether

each node includes a point cloud (S3623).

[0702]

After that, the three-dimensional data encoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3624).

[0703]

Next, the three-dimensional data encoding device determines whether a coding mode is a predetermined mode (S3625). For example, when CodingMode = 1, the three-dimensional data encoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data encoding device determines that a coding mode is not a predetermined mode.

[0704]

It should be noted that, for example, a search flag (search_flag) indicating whether to perform search parent neighbor may be used instead of CodingMode. In this case, when, for example, search_flag = 1, the three-dimensional data encoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data encoding device determines that a coding mode is not a predetermined mode.

[0705]

When the coding mode is the predetermined mode (YES in S3625), the three-dimensional data encoding device obtains occupancy information (remaining neighbor occupancy patterns) of remaining neighboring nodes by performing search parent neighbor for searching all encoded child nodes (S3626). Then, the three-dimensional data encoding device combines the neighbor occupancy pattern calculated from the occupancy code of the parent node in step

S3624 and the remaining neighbor occupancy patterns calculated by search parent neighbor in step S3626 (S3627).

[0706]

After that, the three-dimensional data encoding device converts a neighbor occupancy pattern obtained by the combination into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 10 coding tables (S3628).

[0707]

On the other hand, when the coding mode is not the predetermined mode (NO in S3625), the three-dimensional data encoding device converts the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3624 into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 6 coding tables (S3629).

[0708]

It should be noted that although an example in which the number of coding tables is changed between 10 and 6 has been given here, the number of coding tables is not always limited to this. For example, the three-dimensional data encoding device may change the number of coding tables between N and M, M being an integer less than or equal to N.

[0709]

Next, the three-dimensional data encoding device selects a coding table to be used for entropy encoding, based on the index obtained in step S3628 or S3629 (S3630). After that, the three-dimensional data encoding device entropy encodes the occupancy code of the current node using the selected coding table (S3631).

[0710]

It should be noted that the three-dimensional data encoding device may

encode, as header information, information (CodingMode) indicating a coding mode. In addition, the three-dimensional data encoding device may append, to a bitstream, a search flag (search_flag) indicating whether to perform search parent neighbor, instead of CodingMode.

[0711]

Finally, the three-dimensional data encoding device repeats a process of dividing each node into eight and encoding an occupancy code of each node until each node cannot be divided (S3632). In other words, steps S3622 to S3631 are repeated recursively.

[0712]

FIG. 115 is a flowchart of a three-dimensional data decoding process in the three-dimensional data decoding device. First, the three-dimensional data decoding device defines a space (a current node) to be decoded, using the header information of a bitstream (S3641).

[0713]

It should be noted that the three-dimensional data decoding device may decode information (CodingMode) indicating a coding mode included in the header information. In addition, the three-dimensional data decoding device may decode a search flag (search_flag) indicating whether to perform search parent neighbor, instead of CodingMode.

[0714]

Next, the three-dimensional data decoding device generates eight small spaces (nodes) by dividing the current node into eight (S3642). Then, the three-dimensional data decoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3643).

[0715]

After that, the three-dimensional data decoding device determines whether a coding mode is a predetermined mode (S3644). For example, when CodingMode = 1, the three-dimensional data decoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data decoding device determines that a coding mode is not a predetermined mode.

[0716]

It should be noted that, for example, a search flag (search_flag) indicating whether to perform search parent neighbor may be used instead of CodingMode. In this case, when, for example, search_flag = 1, the three-dimensional data decoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data decoding device determines that a coding mode is not a predetermined mode.

[0717]

When the coding mode is the predetermined mode (YES in S3644), the three-dimensional data decoding device obtains occupancy information (remaining neighbor occupancy patterns) of remaining neighboring nodes by performing search parent neighbor for searching all encoded child nodes (S3645). Next, the three-dimensional data decoding device combines the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3643 and the remaining neighbor occupancy patterns calculated by search parent neighbor in step S3645 (S3646).

[0718]

Then, the three-dimensional data decoding device converts a neighbor occupancy pattern obtained by the combination into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 10 coding tables (S3647).

[0719]

On the other hand, when the coding mode is not the predetermined mode (NO in S3644), the three-dimensional data decoding device converts the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3643 into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 6 coding tables (S3648).

[0720]

It should be noted that although an example in which the number of coding tables is changed between 10 and 6 has been given here, the number of coding tables is not always limited to this. For example, the three-dimensional data decoding device may change the number of coding tables between N and M, M being an integer less than or equal to N.

[0721]

After that, the three-dimensional data decoding device selects a coding table to be used for entropy decoding, based on the index obtained in step S3647 or S3648 (S3649). Next, the three-dimensional data decoding device entropy decodes an occupancy code of the current node using the selected coding table (S3650).

[0722]

Finally, the three-dimensional data decoding device repeats a process of dividing each node into eight and decoding an occupancy code of each node until each node cannot be divided (S3651). In other words, steps S3642 to S3650 are repeated recursively.

[0723]

As stated above, the three-dimensional data encoding device according to the present embodiment performs the process shown in FIG. 116. First, the three-dimensional data encoding device encodes a first flag (e.g., search_flag)

indicating whether a node having a parent node different from a parent node of a current node is to be referred to in encoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2 (S3661). In other words, the three-dimensional data encoding device generates a bitstream including the first flag.

[0724]

The three-dimensional data encoding device selects a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performs arithmetic encoding on information of the current node using the coding table selected (S3663), when the first flag indicates that the node is to be referred to (YES in S3662).

[0725]

The three-dimensional data encoding device selects a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performs arithmetic encoding on information of the current node using the coding table selected (S3664), when the first flag indicates that the node is not to be referred to, M being an integer different from N (NO in S3662).

[0726]

With this, since it is possible to reduce the number of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0727]

For example, N is greater than M.

[0728]

For example, when the three-dimensional data encoding device selects a coding table from the M coding tables, the three-dimensional data encoding device selects the coding table from the M coding tables by reference to a correspondence table (e.g., the table shown in FIG. 106) according to the occupancy states of the neighboring nodes, the correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and the M coding tables, L being an integer greater than M.

[0729]

For example, when the three-dimensional data encoding device selects a coding table from the M coding tables, the three-dimensional data encoding device selects the coding table from the M coding tables by reference to a first correspondence table (e.g., the table shown in FIG. 106) and a second correspondence table (e.g., the table shown in FIG. 107), according to the occupancy states of the neighboring nodes, the first correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and I coding tables, the second correspondence table indicating a correspondence relationship between the I coding tables and the M coding tables, L being an integer greater than I, I being an integer greater than M.

[0730]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An

identical coding table (e.g., group 1 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction horizontal to an x-y plane.

[0731]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 2 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction vertical to an x-y plane.

[0732]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 3 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is horizontal to an x-y plane.

[0733]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node

and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 4 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is vertical to an x-y plane.

[0734]

For example, the three-dimensional data encoding device includes a processor and memory, and the processor performs the above process using the memory.

[0735]

Moreover, the three-dimensional data decoding device according to the present embodiment performs the process shown in FIG. 117. The three-dimensional data decoding device decodes a first flag (e.g., search_flag) indicating whether a node having a parent node different from a parent node of a current node is to be referred to in decoding of the current node included in an N-ary tree structure of three-dimensional points included in three-dimensional data, N being an integer greater than or equal to 2 (S3671). In other words, the three-dimensional data decoding device obtains the first flag from a bitstream.

[0736]

The three-dimensional data decoding device selects a coding table from N coding tables according to occupancy states of neighboring nodes of the current node, and performing arithmetic decoding on information of the current node using the coding table selected (S3673), when the first flag indicates that the node is to be referred to (YES in S3672).

[0737]

The three-dimensional data decoding device selects a coding table from M coding tables according to the occupancy states of the neighboring nodes of the current node, and performing arithmetic decoding on information of the current node using the coding table selected (S3674), when the first flag indicates that the node is not to be referred to (NO in S3672), M being an integer different from N.

[0738]

With this, since it is possible to reduce the number of coding tables, it is possible to reduce the amount of processing. Moreover, since it is possible to set a coding table appropriately by changing the number of coding tables according to whether a node having a parent node different from a parent node of a current node is to be referred to, it is possible to reduce the amount of processing while suppressing the reduction of coding efficiency.

[0739]

For example, N is greater than M.

[0740]

For example, when the three-dimensional data decoding device selects a coding table from the M coding tables, the three-dimensional data decoding device selects the coding table from the M coding tables by reference to a correspondence table (e.g., the table shown in FIG. 106) according to the occupancy states of the neighboring nodes, the correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and the M coding tables, L being an integer greater than M.

[0741]

For example, when the three-dimensional data decoding device selects a coding table from the M coding tables, the three-dimensional data decoding

device selects the coding table from the M coding tables by reference to a first correspondence table (e.g., the table shown in FIG. 106) and a second correspondence table (e.g., the table shown in FIG. 107), according to the occupancy states of the neighboring nodes, the first correspondence table indicating a correspondence relationship between L occupancy patterns indicating the occupancy states of the neighboring nodes and I coding tables, the second correspondence table indicating a correspondence relationship between the I coding tables and the M coding tables, L being an integer greater than I, I being an integer greater than M.

[0742]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 1 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction horizontal to an x-y plane.

[0743]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 2 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which one of the three neighboring nodes is occupied and neighbors the current node in a direction vertical to an x-y plane.

[0744]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 3 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is horizontal to an x-y plane.

[0745]

For example, the occupancy states of the neighboring nodes when the first flag indicates that the node is not to be referred to are occupancy patterns represented by combinations of a position of the current node in a parent node and occupancy states of three neighboring nodes in the parent node. An identical coding table (e.g., group 4 shown in FIG. 104) among the M coding tables is assigned to, among the occupancy patterns, occupancy patterns in which two of the three neighboring nodes are occupied and a plane defined by the two of the three neighboring nodes occupied and the current node is vertical to an x-y plane.

[0746]

For example, the three-dimensional data decoding device includes a processor and memory, and the processor performs the above process using the memory.

[0747]

EMBODIMENT 11

Another method of reducing the number of coding tables will be

described in the present embodiment. In the present embodiment, neighbor occupancy patterns are grouped based on an occupancy number that is the number of neighboring occupied nodes. To put it another way, one of four coding tables is used according to the number of neighboring occupied nodes regardless of a position of a current node and positions of the neighboring occupied nodes.

[0748]

FIG. 118 is a diagram illustrating an example of dividing 64 neighbor occupancy patterns into 4 groups. A neighbor occupancy pattern having an occupancy number of zero is classified into group 0. A neighbor occupancy pattern having an occupancy number of one is classified into group 1. A neighbor occupancy pattern having an occupancy number of two is classified into group 2. A neighbor occupancy pattern having an occupancy number of three is classified into group 3.

[0749]

As stated above, the three-dimensional data encoding device may select a coding table according to how many neighboring nodes among neighboring nodes of a current node are occupied. It follows that the same coding table is assigned to similar shapes regardless of translation and rotation. Accordingly, it is possible to improve the coding efficiency while reducing the number of coding tables.

[0750]

For example, when the above mode is used, coding mode 4 is added as a coding mode. In coding mode 4 (CodingMode = 4), 64 neighbor occupancy patterns are obtained by referring to an occupancy code of a parent node and performing search parent neighbor. In addition, a look-up table that converts the 64 neighbor occupancy patterns into indexes of 4 coding tables is used.

[0751]

Furthermore, as another example of a coding mode (CodingMode), for example, the three-dimensional data encoding device appends, to a bitstream, a search flag (search_flag) indicating whether to perform search parent neighbor. When search_flag = 1, the three-dimensional data encoding device may calculate a neighbor occupancy pattern by performing search parent neighbor, generate an index of one of ten coding tables from a value of the neighbor occupancy pattern, and perform arithmetic encoding on an occupancy code using the coding table having the index.

[0752]

When search_flag = 0, the three-dimensional data encoding device may calculate a neighbor occupancy pattern without performing search parent neighbor, generate an index of one of four coding tables from a value of the neighbor occupancy pattern, and perform arithmetic encoding on an occupancy code using the coding table having the index. It is possible to control a balance between the coding efficiency and the amount of processing by selecting search parent neighbor and a coding table to be used according to a value of search_flag in the above manner.

[0753]

FIG. 119 is a flowchart of this process. First, the three-dimensional data encoding device determines whether search_flag is 1 (S3701). It should be noted that the three-dimensional data encoding device may encode, for example, a flag (search_skip_flag) indicating whether to skip search parent neighbor, instead of search_flag. For example, the three-dimensional data encoding device appends search_skip_flag to a bitstream. When search_skip_flag = 1, the three-dimensional data encoding device calculates a neighbor occupancy pattern without performing search parent neighbor,

generates an index of one of four coding tables from a value of the neighbor occupancy pattern, and performs arithmetic encoding on an occupancy code using the coding table having the index. When search_skip_flag = 0, the three-dimensional data encoding device calculates a neighbor occupancy pattern by performing search parent neighbor, generates an index of one of ten coding tables from a value of the neighbor occupancy pattern, and performs arithmetic encoding on an occupancy code using the coding table having the index.

[0754]

When search_flag is 1 (YES in S3701), the three-dimensional data encoding device generates an index of a coding table using conversion table A that converts values of 64 neighbor occupancy patterns into indexes of 10 coding tables (S3702).

[0755]

When search_flag is not 1 (NO in S3701), the three-dimensional data encoding device generates an index of a coding table using conversion table B that converts values of 64 neighbor occupancy patterns into indexes of 4 coding tables (S3703).

[0756]

It should be noted that the three-dimensional data encoding device may use the same probability value initializing method on coding tables assigned to the same neighbor occupancy pattern in conversion table A and conversion table B. For example, the three-dimensional data encoding device may initialize, for table 5 of conversion table A and table 2 of conversion table B assigned to pattern 4, probability values of the coding tables by the same method. Since this eliminates the need for providing an initializing method for each conversion table, it is possible to reduce the amount of processing.

[0757]

The following describes a flow of processing performed by each of the three-dimensional data encoding device and the three-dimensional data decoding device according to the present embodiment.

[0758]

FIG. 120 is a flowchart of a three-dimensional data encoding process in the three-dimensional data encoding device. First, the three-dimensional data encoding device defines a space (a current node) including part or all of an inputted three-dimensional point cloud (S3711). Next, the three-dimensional data decoding device generates eight small spaces (nodes) by dividing the current node into eight (S3712). Then, the three-dimensional data encoding device generates an occupancy code of the current node according to whether each node includes a point cloud (S3713).

[0759]

After that, the three-dimensional data encoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3714).

[0760]

Next, the three-dimensional data encoding device determines whether a coding mode is a predetermined mode (S3715). For example, when CodingMode = 1, the three-dimensional data encoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data encoding device determines that a coding mode is not a predetermined mode.

[0761]

It should be noted that, for example, a search flag (search_flag) indicating whether to perform search parent neighbor may be used instead of

CodingMode. In this case, when, for example, search_flag = 1, the three-dimensional data encoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data encoding device determines that a coding mode is not a predetermined mode.

[0762]

When the coding mode is the predetermined mode (YES in S3715), the three-dimensional data encoding device obtains occupancy information (remaining neighbor occupancy patterns) of remaining neighboring nodes by performing search parent neighbor for searching all encoded child nodes (S3716). Then, the three-dimensional data encoding device combines the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3714 and the remaining neighbor occupancy patterns calculated by search parent neighbor in step S3716 (S3717).

[0763]

After that, the three-dimensional data encoding device converts a neighbor occupancy pattern obtained by the combination into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 10 coding tables (S3718).

[0764]

On the other hand, when the coding mode is not the predetermined mode (NO in S3715), the three-dimensional data encoding device converts the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3714 into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 4 coding tables (S3719).

[0765]

It should be noted that although an example in which the number of coding tables is changed between 10 and 4 has been given here, the number of

coding tables is not always limited to this. For example, the three-dimensional data encoding device may change the number of coding tables between N and M, M being an integer less than or equal to N.

[0766]

Next, the three-dimensional data encoding device selects a coding table to be used for entropy encoding, based on the index obtained in step S3718 or S3719 (S3720). Then, the three-dimensional data encoding device entropy encodes the occupancy code of the current node using the selected coding table (S3721).

[0767]

It should be noted that the three-dimensional data encoding device may encode, as header information, information (CodingMode) indicating a coding mode. In addition, the three-dimensional data encoding device may append, to a bitstream, a search flag (search_flag) indicating whether to perform search parent neighbor, instead of CodingMode.

[0768]

Finally, the three-dimensional data encoding device repeats a process of dividing each node into eight and encoding an occupancy code of each node until each node cannot be divided (S3722). In other words, steps S3712 to S3721 are repeated recursively.

[0769]

FIG. 121 is a flowchart of a three-dimensional data decoding process in the three-dimensional data decoding device. First, the three-dimensional data decoding device defines a space (a current node) to be decoded, using the header information of a bitstream (S3731).

[0770]

It should be noted that the three-dimensional data decoding device may

decode information (CodingMode) indicating a coding mode included in the header information. In addition, the three-dimensional data decoding device may decode a search flag (search_flag) indicating whether to perform search parent neighbor, instead of CodingMode.

[0771]

Next, the three-dimensional data decoding device generates eight small spaces (nodes) by dividing the current node into eight (S3732). Then, the three-dimensional data decoding device calculates (obtains) occupancy information (a neighbor occupancy pattern) of a neighboring reference node of the current node from an occupancy code of a parent node of the current node (S3733).

[0772]

After that, the three-dimensional data decoding device determines whether a coding mode is a predetermined mode (S3734). For example, when CodingMode = 1, the three-dimensional data decoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data decoding device determines that a coding mode is not a predetermined mode.

[0773]

It should be noted that, for example, a search flag (search_flag) indicating whether to perform search parent neighbor may be used instead of CodingMode. In this case, when, for example, search_flag = 1, the three-dimensional data decoding device determines that a coding mode is a predetermined mode; and in the other cases, the three-dimensional data decoding device determines that a coding mode is not a predetermined mode.

[0774]

When the coding mode is the predetermined mode (YES in S3734), the three-dimensional data decoding device obtains occupancy information

(remaining neighbor occupancy patterns) of remaining neighboring nodes by performing search parent neighbor for searching all encoded child nodes (S3735). Next, the three-dimensional data decoding device combines the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3733 and the remaining neighbor occupancy patterns calculated by search parent neighbor in step S3735 (S3736).

[0775]

Then, the three-dimensional data decoding device converts a neighbor occupancy pattern obtained by the combination into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 10 coding tables (S3737).

[0776]

On the other hand, when the coding mode is not the predetermined mode (NO in S3734), the three-dimensional data decoding device converts the neighbor occupancy pattern calculated from the occupancy code of the parent node in step S3733 into an index, using a look-up table that converts 64 neighbor occupancy patterns into indexes of 4 coding tables (S3738).

[0777]

It should be noted that although an example in which the number of coding tables is changed between 10 and 4 has been given here, the number of coding tables is not always limited to this. For example, the three-dimensional data decoding device may change the number of coding tables between N and M, M being an integer less than or equal to N.

[0778]

After that, the three-dimensional data decoding device selects a coding table to be used for entropy decoding, based on the index obtained in step S3737 or S3738 (S3739). Next, the three-dimensional data decoding device entropy

decodes an occupancy code of the current node using the selected coding table (S3740).

[0779]

Finally, the three-dimensional data decoding device repeats a process of dividing each node into eight and decoding an occupancy code of each node until each node cannot be divided (S3741). In other words, steps S3732 to S3740 are repeated recursively.

[0780]

It should be noted that although the operation of the three-dimensional data encoding device has been mainly described above, the three-dimensional data decoding device may perform the same operation. In addition, information such as various types of flags generated by the three-dimensional data encoding device is included in a bitstream. The three-dimensional data decoding device obtains information such as various types of flags included in a bitstream, and performs a process by referring to the information.

[0781]

EMBODIMENT 12

A method of removing a redundant coding table will be described in the present embodiment. FIG. 122 is a diagram for illustrating redundant coding tables. When table indexes likely to be used in table selection methods are not continuous, redundant coding tables are created.

[0782]

In response to this, it is possible to remove redundant tables by creating coding tables dynamically. As a result, it is possible to save hardware resources. For example, it is possible to prevent redundant tables from being created by calculating the number of actually necessary tables at compile time or execution time (run-time).

[0783]

The following describes examples of redundant coding tables. In the first example, redundant tables are created when one selection method is given. FIG. 123 is a diagram for illustrating this example.

[0784]

A statically created coding table includes redundant tables. It is difficult to accurately determine the number of tables in a design stage of algorithm. Accordingly, it is necessary to prepare more tables than actually necessary.

[0785]

On the other hand, when a coding table is created dynamically, the number of tables is inputted to algorithm at compile time or execution time. In consequence, it is possible to remove redundant tables.

[0786]

In the second example, redundant tables are created in binary encoding in which a dependency relationship with neighboring nodes is used. In order to improve the coding efficiency, neighboring information is used in binary encoding. FIG. 124 is a diagram illustrating an example of a current node. A method of using neighboring information may differ between when a current node includes no neighboring node in a parent node and when a current node includes one or more neighboring nodes in a parent node.

[0787]

The following describes case 1. FIG. 125 is a diagram illustrating an operation in case 1. In case 1, an index of a coding table is based on a value of each encoded bit in a parent node.

[0788]

In case 1, the three-dimensional data encoding device changes a coding

table according to a value (ENC) of each bit of an occupancy code of a current node. For example, when bit 0 is 1 and bit 1 is 0, the three-dimensional data encoding device may use index = 1 (= 1 + 0) as a coding table for performing arithmetic encoding on bit 2. In other words, by changing a coding table according to a value of each bit of an encoded occupancy code, for example, the three-dimensional data encoding device can select a coding table having a high probability of 1 when an occurrence frequency of 1 is high in each bit, and select a coding table having a high probability of 0 when an occurrence frequency of 0 is high in each bit. For this reason, it is possible to improve the coding efficiency.

[0789]

Moreover, when NC = 0, that is, neighboring nodes of the current node are not occupied nodes, the three-dimensional data encoding device may use the method of selecting a coding table described in case 1. For this reason, when an association with neighboring nodes is weak, the three-dimensional data encoding device can select a coding table according to occurrence frequencies of 0 and 1 in an occupancy code. Accordingly, case 1 makes it more possible to improve the coding efficiency while reducing the number of coding tables than case 2 to be described.

[0790]

When an occupancy code is 8 bits in the example shown in FIG. 125, a total number of coding tables is $1 + 2 + 3 + \ldots + 8 = 36$.

[0791]

Next, the following describes case 2. FIG. 126 is a diagram illustrating an operation in case 2. In case 2, an index of a coding table is determined based on an index of a neighbor configuration (NC) and an encoded bit (ENC). For example, NCs correspond to the above-mentioned neighbor occupancy patterns.

[0792]

Moreover, a set of different NCs is used. In example 1, the number of neighboring occupied nodes in a parent node is used. In other words, the number of NCs is four (NC = {0, 1, 2, 3}).

[0793]

In example 2, positions of neighboring occupied nodes are used. When 6 neighboring nodes are used, the number of NCs is 64 (NC = {0, 1, 2, ..., 63}).

[0794]

In example 3, some NCs are combined. As a result, the number of NCs is optional. For example, 64 NCs are combined into 10 NCs based on geometry information.

[0795]

In case 2, the three-dimensional data encoding device may also change a coding table according to occupancy information (NC) of a neighboring node of a current node and a value (ENC) of each bit of an occupancy code.

[0796]

FIG. 127 is a diagram illustrating a configuration example of six neighboring nodes. For example, when one of the six neighboring nodes shown in FIG. 127 is occupied, NC = 1 (a case in which a value of NC = 0 to 9 is taken), and bit 0 is 1, the three-dimensional data encoding device selects a coding table for performing arithmetic encoding on bit 1, using information of NC = 1 and bit 0 = 1. In other words, the three-dimensional data encoding device changes a coding table according to occupancy information of a neighboring node and a value of each bit of an encoded occupancy code. For this reason, since the three-dimensional data encoding device can select an appropriate coding table accordingly when, for example, one of neighboring nodes is occupied and an occurrence frequency of 1 is high in each bit, the three-dimensional data encoding device can improve the coding efficiency. Additionally, when NC > 0,

that is, at least one neighboring node of a current node is an occupied node, the three-dimensional data encoding device may use the method of selecting a coding table described in case 2. For this reason, since the three-dimensional data encoding device can select a coding table according to an association with the neighboring nodes and occurrence frequencies of 0 and 1 in an occupancy code, the three-dimensional data encoding device can improve the coding efficiency.

[0797]

The following describes the third example. In the third example, for example, NC = 10 (a case in which a value of NC = 0 to 9 is taken). Redundant tables are created by binary encoding in which neighboring information is used. FIG. 128 is a diagram for illustrating the third example.

[0798]

In the third example, when NC = 0, a coding table is selected based on the number of bits (ENC) indicating an occupancy state and included in encoded bits. When NC > 0, a coding table is selected based on an ENC and an NC (1 to 9).

[0799]

FIG. 128 shows the number of coding tables for each bit when a method of selecting a coding table is set to case 2 regardless of a value of an NC (case 2 for all); the number of coding tables for each bit when case 1 is used as a method of selecting a coding table in the case of NC = 0 and case 2 is used as the same in the case of NC > 0; and the number of tables not to be used (the number of redundant tables) that is a difference between those numbers.

[0800]

When those two cases are used, equations for calculating a total number of tables are complex, and it is not easy to generate indexes. Consequently, it

is difficult to create a fixed number of coding tables. This is because the number of tables increases linearly when NC = 0, but the number of tables increases exponentially when NC > 0. Accordingly, the number of redundant tables (tables not to be used) increases, and it is impossible to use indexes of the redundant tables for the both cases.

[0801]

It should be noted that the three-dimensional data encoding device may remove coding tables created but not to be used. Since this makes it possible to reduce an initialization time for coding tables, it is possible to achieve speeding up and the reduction of memory utilization.

[0802]

The following describes the fourth example. In the fourth example, for example, NC = 4 (a case in which a value of NC = 0 to 3 is taken). Redundant tables are created by binary encoding in which neighboring information is used. FIG. 129 is a diagram for illustrating the fourth example. It should be noted that the fourth example differs from the third example in a total number of NC numbers. As shown in FIG. 129, redundant tables (tables not to be used) are created in the fourth example, as is the case with the third example.

[0803]

The following describes examples of redundant coding tables. FIG. 130 is a diagram illustrating an example of coding tables when coding tables are created statically for each bit in the above third example (NC = 10).

[0804]

As shown in FIG. 130, for NC0, the number of tables increases linearly with an increase in bit number. For NC1 to NC9, the number of tables increases exponentially with an increase in bit number.

[0805]

Here, different coding is used between NC0 and NC > 0. As a result, a total number of tables increases linearly for NC0 and increases exponentially for NC > 0.

[0806]

FIG. 131 is a diagram illustrating an example of coding tables when coding tables are created statically for each bit in the above fourth example (NC = 4). It should be noted that the three-dimensional data encoding device need not create redundant tables by separating a table for NC0 and a table for NC > 0 as shown in FIG. 132.

[0807]

In the fifth example, the three-dimensional data encoding device removes redundant tables by creating tables for NC0 and tables for NC > 0 separately. FIG. 133 is a diagram for illustrating the fifth example. FIG. 134 is a diagram illustrating total numbers of coding tables after redundant tables are removed.

[0808]

Next, the following describes a process of creating a coding table having a dynamic size. FIG. 135 is a diagram for illustrating this process. In order to create a required and accurate number of coding tables, a setting (a setting input) for coding table size is inputted to the three-dimensional data encoding device.

[0809]

FIG. 136 is a diagram illustrating the size of each table when setting input = 10. As shown in FIG. 136, size of table Size is defined as Size = $2^{bit} \times (x - 1) + bit + 1$ in source code. The size of each table is determined after indefinite number x is obtained. Here, $2^{bit} \times (x - 1)$ corresponds to a size when NC > 0 in the second example, and bit + 1 corresponds to a size when NC = 0.

[0810]

By using such a setting, it is possible to create coding tables dynamically at compile time or execution time.

[0811]

FIG. 137 is a flowchart of a process of creating a dynamic coding table. First, the three-dimensional data encoding device determines whether a dynamic table size flag (dynamicTableSize) is 1 (S3801). dynamicTableSize is a flag indicating whether to use a dynamic table size. The three-dimensional data encoding device generates the flag and a bitstream including the flag. Besides, the three-dimensional data decoding device obtains the flag included in the bitstream.

[0812]

When dynamicTableSize is 1 (YES in S3801), the three-dimensional data encoding device loads a setting for dynamic size (S3802). Next, the three-dimensional data encoding device creates a coding table based on the loaded setting (S3803). This coding table includes no redundant tables.

[0813]

When dynamicTableSize is not 1 (NO in S3801), the three-dimensional data encoding device creates a static coding table (S3804). This coding table may include redundant tables.

[0814]

It should be noted that although an example in which the 8-bit occupancy code is used has been given above, the present disclosure is not necessarily limited to this. For example, the present procedure may be applied to a code having another bit number such as ten bits. In addition, a code to be used is not limited to an occupancy code.

[0815]

FIG. 138 is a diagram illustrating the size of each table when setting input = 4. It should be noted that a method of calculating the size of a table is the same as when setting input = 10.

[0816]

The three-dimensional data encoding device may change a process of creating a table based on a search flag (search_flag). For example, the three-dimensional data encoding device appends, to a bitstream, a search flag (search_flag) indicating whether to perform search parent neighbor. When search_flag = 1, the three-dimensional data encoding device may prepare coding tables when NC = {0, 1, ..., 9}; calculate a neighbor occupancy pattern (NC) by performing search parent neighbor; generate an index of one of the coding tables using a value of the neighbor occupancy pattern and a value of each bit of an encoded occupancy code; and perform arithmetic encoding on an occupancy code using the coding table having the index.

[0817]

Moreover, when search_flag = 0, the three-dimensional data encoding device may prepare coding tables when NC = {0, 1, ..., 3}; calculate a neighbor occupancy pattern (NC) without performing search parent neighbor; generate an index of one of the coding tables using a value of the neighbor occupancy pattern and a value of each bit of an encoded occupancy code; and perform arithmetic encoding on an occupancy code using the coding table having the index.

[0818]

It is possible to control a balance between the coding efficiency, the amount of processing, and the amount of memory by (i) controlling the number of coding tables to be created by changing a value of search_flag and (ii) selecting search parent neighbor and a coding table to be used in the above manner.

[0819]

FIG. 139 is a flowchart of this process. First, the three-dimensional data encoding device determines whether search_flag is 1 (S3811). It should be noted that the three-dimensional data encoding device may encode, for example, a flag (search_skip_flag) indicating whether to skip search parent neighbor, instead of search_flag. For example, the three-dimensional data encoding device appends search_skip_flag to a bitstream. When search_skip_flag = 1, the three-dimensional data encoding device prepares coding tables when NC = {0, 1, ..., 3}; calculates a neighbor occupancy pattern (NC) without performing search parent neighbor; generates an index of one of the coding tables using a value of the neighbor occupancy pattern and a value of each bit of an encoded occupancy code; and performs arithmetic encoding on an occupancy code using the coding table having the index. Moreover, when search_skip_flag = 0, the three-dimensional data encoding device prepares coding tables when NC = {0, 1, ..., 9}; calculates an NC by performing search parent neighbor; generates an index of one of the coding tables using a value of the NC and a value of each bit of an encoded occupancy code; and performs arithmetic encoding on an occupancy code using the coding table having the index.

[0820]

When search_flag is 1 (YES in S3811), the three-dimensional data encoding device prepares coding tables when NC = {0, 1, ..., 9} (S3812). At this time, the three-dimensional data encoding device may remove redundant tables.

[0821]

When search_flag is not 1 (NO in S3811), the three-dimensional data encoding device prepares coding tables when NC = {0, 1, ..., 3} (S3813). At this time, the three-dimensional data encoding device may remove redundant tables.

[0822]

It should be noted that although the operation of the three-dimensional data encoding device has been mainly described above, the three-dimensional data decoding device may perform the same operation. In addition, information such as various types of flags generated by the three-dimensional data encoding device is included in a bitstream. The three-dimensional data decoding device obtains information such as various types of flags included in a bitstream, and performs a process by referring to the information.

[0823]

A three-dimensional data encoding device, a three-dimensional data decoding device, and the like according to the embodiments of the present disclosure have been described above, but the present disclosure is not limited to these embodiments.

[0824]

Note that each of the processors included in the three-dimensional data encoding device, the three-dimensional data decoding device, and the like according to the above embodiments is typically implemented as a large-scale integrated (LSI) circuit, which is an integrated circuit (IC). These may take the form of individual chips, or may be partially or entirely packaged into a single chip.

[0825]

Such IC is not limited to an LSI, and thus may be implemented as a dedicated circuit or a general-purpose processor. Alternatively, a field programmable gate array (FPGA) that allows for programming after the manufacture of an LSI, or a reconfigurable processor that allows for reconfiguration of the connection and the setting of circuit cells inside an LSI may be employed.

[0826]

Moreover, in the above embodiments, the structural components may be implemented as dedicated hardware or may be realized by executing a software program suited to such structural components. Alternatively, the structural components may be implemented by a program executor such as a CPU or a processor reading out and executing the software program recorded in a recording medium such as a hard disk or a semiconductor memory.

[0827]

The present disclosure may also be implemented as a three-dimensional data encoding method, a three-dimensional data decoding method, or the like executed by the three-dimensional data encoding device, the three-dimensional data decoding device, and the like.

[0828]

Also, the divisions of the functional blocks shown in the block diagrams are mere examples, and thus a plurality of functional blocks may be implemented as a single functional block, or a single functional block may be divided into a plurality of functional blocks, or one or more functions may be moved to another functional block. Also, the functions of a plurality of functional blocks having similar functions may be processed by single hardware or software in a parallelized or time-divided manner.

[0829]

Also, the processing order of executing the steps shown in the flowcharts is a mere illustration for specifically describing the present disclosure, and thus may be an order other than the shown order. Also, one or more of the steps may be executed simultaneously (in parallel) with another step.

[0830]

A three-dimensional data encoding device, a three-dimensional data

decoding device, and the like according to one or more aspects have been described above based on the embodiments, but the present disclosure is not limited to these embodiments. The one or more aspects may thus include forms achieved by making various modifications to the above embodiments that can be conceived by those skilled in the art, as well forms achieved by combining structural components in different embodiments, without materially departing from the spirit of the present disclosure.

INDUSTRIAL APPLICABILITY

[0831]

The present disclosure is applicable to a three-dimensional data encoding device and a three-dimensional data decoding device.

REFERENCE MARKS IN THE DRAWINGS

[0832]

100, 400 three-dimensional data encoding device

101, 201, 401, 501 obtainer

102, 402 encoding region determiner

103 divider

104, 644 encoder

111 three-dimensional data

112, 211, 413, 414, 511, 634 encoded three-dimensional data

200, 500 three-dimensional data decoding device

202 decoding start GOS determiner

203 decoding SPC determiner

204, 625 decoder

212, 512, 513 decoded three-dimensional data

403 SWLD extractor

404 WLD encoder

405 SWLD encoder

411 input three-dimensional data

412 extracted three-dimensional data

502 header analyzer

503 WLD decoder

504 SWLD decoder

620, 620A three-dimensional data creation device

621, 641 three-dimensional data creator

622 request range determiner

623 searcher

624, 642 receiver

626 merger

631, 651 sensor information

632 first three-dimensional data

633 request range information

635 second three-dimensional data

636 third three-dimensional data

640 three-dimensional data transmission device

643 extractor

645 transmitter

652 fifth three-dimensional data

654 sixth three-dimensional data

700 three-dimensional information processing device

701 three-dimensional map obtainer

702 self-detected data obtainer

703 abnormal case judgment unit

704 coping operation determiner

705 operation controller

711 three-dimensional map

712 self-detected three-dimensional data

810 three-dimensional data creation device

811 data receiver

812, 819 communication unit

813 reception controller

814, 821 format converter

815 sensor

816 three-dimensional data creator

817 three-dimensional data synthesizer

818 three-dimensional data storage

820 transmission controller

822 data transmitter

831, 832, 834, 835, 836, 837 three-dimensional data

833 sensor information

901 server

902, 902A, 902B, 902C client device

1011, 1111 data receiver

1012, 1020, 1112, 1120 communication unit

1013, 1113 reception controller

1014, 1019, 1114, 1119 format converter

1015 sensor

1016, 1116 three-dimensional data creator

1017 three-dimensional image processor

1018, 1118 three-dimensional data storage

1021, 1121 transmission controller

1022, 1122 data transmitter

1031, 1032, 1135 three-dimensional map

1033, 1037, 1132 sensor information

1034, 1035, 1134 three-dimensional data

1117 three-dimensional data merger

1201 three-dimensional map compression/decoding processor

1202 sensor information compression/decoding processor

1211 three-dimensional map decoding processor

1212 sensor information compression processor

1300 three-dimensional data encoding device

1301 divider

1302 subtractor

1303 transformer

1304 quantizer

1305, 1402 inverse quantizer

1306, 1403 inverse transformer

1307, 1404 adder

1308, 1405 reference volume memory

1309, 1406 intra predictor

1310, 1407 reference space memory

1311, 1408 inter predictor

1312, 1409 prediction controller

1313 entropy encoder

1400 three-dimensional data decoding device

1401 entropy decoder

1900 three-dimensional data encoding device

1901, 1911 octree generator

1902, 1912 similarity information calculator

1903, 1913 coding table selector

1904 entropy encoder

1910 three-dimensional data decoding device

1914 entropy decoder

2100 three-dimensional data encoding device

2101, 2111 octree generator

2102, 2112 geometry information calculator

2103, 2113 coding table selector

2104 entropy encoder

2110 three-dimensional data decoding device

2114 entropy decoder

3600 three-dimensional data encoding device

3601, 3611 octree generator

3602, 3612 geometry information calculator

3603, 3613 index generator

3604, 3614 coding table selector

3605 entropy encoder

3610 three-dimensional data decoding device

3615 entropy decoder

Claims

1. A three-dimensional data encoding method, comprising:

generating an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2;

encoding a first parameter;

determining a value indicated by the first parameter;

when the first parameter is determined as indicating a first value, (i) selecting a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic encoding on information of the current node using the selected contexts; and

when the first parameter is determined as indicating a second value different from the first value, (i) selecting a group from M groups according to the occupancy states of neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic encoding on information of the current node using the selected contexts.

2. The three-dimensional data encoding method according to claim 1, wherein N is greater than M.

3. A three-dimensional data decoding method, comprising:

obtaining an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2;

obtaining a first parameter;

determining a value indicated by the first parameter;

when the first parameter is determined as indicating a first value, (i) selecting a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic decoding on information of the current node using the selected contexts;

when the first parameter is determined as indicating a second value different from the first value, (i) selecting a group from M groups according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selecting contexts based on the selected group, and (iii) performing arithmetic decoding on information of the current node using the selected contexts.

4. The three-dimensional data decoding method according to claim 3, wherein N is greater than M.

5. A three-dimensional data encoding device that encodes three-dimensional points each including attribute information, the three-dimensional data encoding device comprising:

a processor; and

memory,

wherein using the memory, the processor:

generates an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2;

encodes a first parameter;

determines a value indicated by the first parameter;

when the first parameter is determined as indicating a first value, (i) selects a group from N groups according to occupancy states of neighboring nodes neighboring a current node and

located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic encoding on information of the current node using the selected contexts; and

when the first parameter is determined as indicating a second value different from the first value, (i) selects a group from M groups according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic encoding on information of the current node using the selected contexts.

6. A three-dimensional data decoding device that decodes three-dimensional points each including attribute information, the three-dimensional data decoding device comprising:

a processor; and

memory,

wherein using the memory, the processor:

obtains an n-ary tree structure of three-dimensional points included in three-dimensional data, n being an integer greater than or equal to 2;

obtains a first parameter;

determines a value indicated by the first parameter;

when the first parameter is determined as indicating a first value, (i) selects a group from N groups according to occupancy states of neighboring nodes neighboring a current node and located inside or outside a parent node of the current node, N being an integer greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic decoding on information of the current node using the selected contexts;

when the first parameter is determined as indicating a second value different from the first value, (i) selects a group from M groups

according to occupancy states of the neighboring nodes neighboring the current node and located inside the parent node of the current node, M being an integer different from N and greater than 1, (ii) selects contexts based on the selected group, and (iii) performs arithmetic decoding on information of the current node using the selected contexts.

7. The three-dimensional data encoding method according to claim 1, wherein

when the first parameter is determined as indicating the first value, a first pattern of the occupancy states of neighboring nodes is selected as a first group from the N groups,

when the first parameter is determined as indicating the second value, the first pattern of the occupancy states of neighboring nodes is selected as a second group from the M groups, and

the first group and the second group indicate that all of the neighboring nodes neighboring the current node do not include a three-dimensional point.

8. The three-dimensional data encoding method according to claim 7, wherein

the number of patterns of the occupancy states of neighboring nodes assigned to the first group of the N groups is one, and

the number of patterns of the occupancy states of neighboring nodes assigned to the second group of the M groups is one.

9. The three-dimensional data encoding method according to claim 1, wherein

when the first parameter is determined as indicating the first value, a first pattern of the occupancy states of neighboring nodes is selected as a first group from the N groups,

when the first parameter is determined as indicating the second value, the first pattern of the occupancy states of neighboring nodes

is selected as a second group from the M groups, and

the first group and the second group indicate that at least one of the neighboring nodes neighboring the current node includes a three-dimensional point.

10. The three-dimensional data encoding method according to claim 9, wherein

Q patterns of the occupancy states of neighboring nodes are assigned to the first group of the N groups,

P patterns of the occupancy states of neighboring nodes are assigned to the second group of the M groups, and

P is greater than Q.

11. The three-dimensional data decoding method according to claim 3, wherein

when the first parameter is determined as indicating the first value, a first pattern of the occupancy states of neighboring nodes is selected as a first group from the N groups,

when the first parameter is determined as indicating the second value, the first pattern of the occupancy states of neighboring nodes is selected as a second group from the M groups, and

the first group and the second group indicate that all of the neighboring nodes neighboring the current node do not include a three-dimensional point.

12. The three-dimensional data decoding method according to claim 11, wherein

the number of patterns of the occupancy states of neighboring nodes assigned to the first group of the N groups is one, and

the number of patterns of the occupancy states of neighboring nodes assigned to the second group of the M groups is one.

13. The three-dimensional data decoding method according to claim 3, wherein

when the first parameter is determined as indicating the first value, a first pattern of the occupancy states of neighboring nodes is selected as a first group from the N groups,

when the first parameter is determined as indicating the second value, the first pattern of the occupancy states of neighboring nodes is selected as a second group from the M groups, and

the first group and the second group indicate that at least one of the neighboring nodes neighboring the current node includes a three-dimensional point.

14. The three-dimensional data decoding method according to claim 13, wherein

Q patterns of the occupancy states of neighboring nodes are assigned to the first group of the N groups,

P patterns of the occupancy states of neighboring nodes are assigned to the second group of the M groups, and

P is greater than Q.