(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
7 March 2013 (07.03.2013)

WIPO | PCT

(10) International Publication Number
WO 2013/029232 A1

(51) International Patent Classification:
H04N 7/26 (2006.01) *G06T 17/00* (2006.01)

(21) International Application Number:
PCT/CN2011/079095

(22) International Filing Date:
30 August 2011 (30.08.2011)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant *(for all designated States except US)*: **TECHNICOLOR (CHINA) TECHNOLOGY CO., LTD.** [CN/CN]; 8th Floor, Building A, Technology Fortune Center, No. 8 Xueqing Road, Haidian District, Beijing 100192 (CN).

(72) Inventors; and
(75) Inventors/Applicants *(for US only)*: **TIAN, Jiang** [CN/CN]; 8th Floor, Building A, Technology Fortune Center, No. 8 Xueqing Road, Haidian District, Beijing 100192 (CN). **LUO, Tao** [CN/CN]; 8th Floor, Building A, Technology Fortune Center, No. 8 Xueqing Road, Haidian District, Beijing 100192 (CN). **CAI, Kangying** [CN/CN]; 8th Floor, Building A, Technology Fortune Center, No. 8 Xueqing Road, Haidian District, Beijing 100192 (CN). **JIANG, Wenfei** [CN/CN]; 8th Floor, Building A, Technology Fortune Center, No. 8 Xueqing Road, Haidian District, Beijing 100192 (CN).

(74) Agent: LIU, SHEN & ASSOCIATES; A0601, Huibin Building, No. 8 Beichen Dong Street, Chaoyang District, Beijing 100101 (CN).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, QA, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:
— *with international search report (Art. 21(3))*

(54) Title: MULTI-RESOLUTION 3D TEXTURED MESH CODING

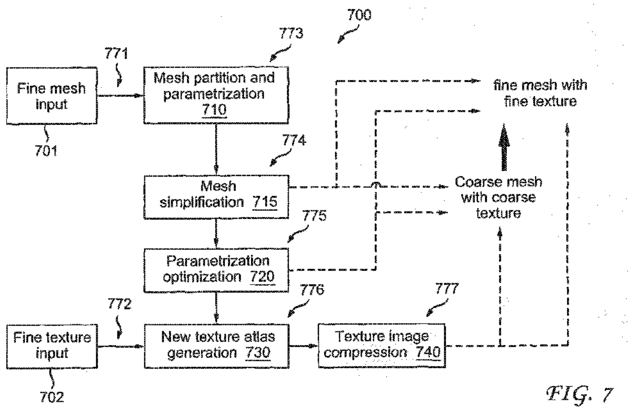
700
771
773
Fine mesh input
701
Mesh partition and parametrization 710
fine mesh with fine texture
774
Mesh simplification 715
Coarse mesh with coarse texture
775
Parametrization optimization 720
772
776
777
Fine texture input
702
New texture atlas generation 730
Texture image compression 740

FIG. 7

(57) Abstract: There is provided a method for encoding a three-dimensional model represented by a mesh of two-dimensional polygons. The method includes simplifying the mesh to obtain a simplified mesh. The method further includes generating a texture image. The texture image represents textures for the mesh and the simplified mesh. The method also includes encoding the texture image to form a base layer and an enhancement layer. The base layer corresponds to a coarse representation of the texture image and the enhancement layer provides a refinement of the base layer.

WO 2013/029232 A1

MULTI-RESOLUTION 3D TEXTURED MESH CODING

TECHNICAL FIELD

Implementations are described that relate to 3D models. Various particular implementations relate to transmitting 3D models with texture in multi-resolution.

BACKGROUND

3D models are used to represent 3D objects. A mesh is a collection of vertices, edges and faces that defines the shape of an object in 3D computer graphics. The faces usually include triangles, quadrilaterals or other simple convex polygons. Typically, connectivity, geometry, and property data are used to represent a 3D mesh. Connectivity data describes the adjacency relationship between vertices. Geometry data specifies vertex locations. Property data specifies several attributes such as the normal vector, material reflectance, and texture coordinates. A 3D model requires a significant amount of data to represent its shape information and texture images. Such models place large strains on computation, storage, transmission, and display resources.

An important problem in 3D modeling is to allow a multi-resolution transmission of the object. That is, the 3D object can be received at different levels of qualities given its network and computational constraints. For example, a user with a low bandwidth may choose to only receive a coarse representation of the object while a user with a high bandwidth may choose to receive a fine representation of the object. Both multi-resolution and progressive coding share a common goal to further improve rendering performance. The difference is that multi-resolution defines several versions of a model at different levels of detail, whereas progressive coding transmit the approximation of a model in a more “continuous way”.

Most existing multi-resolution compression and transmission techniques aim to encode the geometry of a 3D model efficiently. Consequently, these approaches study the progressive coding of the geometry only, without considering the progressive coding of texture images or color materials.

A first prior art approach provides a multi-resolution encoding method for the texture associated with a multi-resolution mesh. Different texture images are generated for different levels of detail to obtain successive refinements. FIGs. 1A-D respectively show texture images 110, 120, 130, 140 having successive refinements of texture, in accordance with the first prior art approach. The texture image in FIG. 1A has a refinement associated with the highest level of details, and the texture image in FIG. 1D has a refinement associated with the lowest level of details. The successive refinements do not correspond to finer or more detailed versions of the same element, but to different texture images associated with different levels of details.

The first prior art approach uses a common mesh unwrapping system agreed upon by both the encoder and the decoder. A new texture atlas is generated from the mesh unwrapping, following a set of fixed and predefined rules known to both the encoder and the decoder. The shape of the new texture atlas determines its “useful versus useless” parts. Due to the common unwrapping system, the shape of the new texture atlas and the references to the texture coordinates need not to be transmitted to the decoder since they can be automatically and losslessly reconstructed in the decoder. Information saving is therefore obtained without the need to transmit the texture coordinates.

FIG. 2 show a texture atlas 200 generated from a mesh unwrapping, in accordance with the first prior art approach. This texture atlas 200 is composed by several images carefully laid out inside a rectangle.

A second prior art approach provides a method to construct a progressive mesh such that all meshes therein share a common texture parametrization. To create a single texture image that can be used to texture all meshes in a progressive mesh sequence, this method considers two quality metrics simultaneously. The second prior art approach minimizes texture stretch (small texture distances mapped onto large surface distances) to balance sampling rates over all locations and directions on the surface. By minimizing the largest texture stretch across all domain points, the second prior art approach creates a balanced parametrization where no domain direction is too stretched and thus undersamples its corresponding mapped 3D direction. The second prior art approach also minimizes texture deviation (“slippage” error based on parametric correspondence) to obtain accurate textured mesh approximations. To measure texture deviation, the second prior art approach uses a heuristic of measuring the incremental texture deviation between two consecutive meshes.

FIG. 3 shows an existing surface signal 310 sampled to obtain a texture image 320, in accordance with the second prior art approach. As seen, the intent thereof is to create a parametrization for representing an existing signal already associated with the mesh surface.

FIG. 4 shows an exemplary mapping 450 of a 2D texture domain 410 to a 3D surface 420, in accordance with the second prior art approach. Singular values are represented as follows: γ, Γ. The mapping computes the distance L from the 2D texture domain to the 3D surface 420 using the following equation: $L^2 = \sqrt{(\gamma^2 + \Gamma^2)/2}$ as described below. That is, to optimize a parametrization's ability to balance frequency content everywhere over the surface in every direction, the second prior art approaches defined a “texture stretch” metric on triangle meshes.

Given a triangle *T* with 2D texture coordinates p_1, p_2, p_3, $p_i = (s_i, t_i)$, and corresponding 3D coordinates q_1, q_2, q_3, the unique affine mapping $S(p) = S(s,t) = q$ is as follows

$$S(p) = (\langle p, p_2, p_3 \rangle q_1 + \langle p, p_3, p_1 \rangle q_2 + \langle p, p_1, p_2 \rangle q_3) / \langle p_1, p_2, p_3 \rangle$$

where $\langle a,b,c \rangle$ denotes area of triangle *abc*. Since the mapping is affine, its partial derivatives are constant over (*s,t*) and given by the following:

$$S_s = \partial S / \partial s = (q_1(t_2 - t_3) + q_2(t_3 - t_1) + q_3(t_1 - t_2)) / (2A)$$
$$S_t = \partial S / \partial t = (q_1(s_3 - s_2) + q_2(s_1 - s_3) + q_3(s_2 - s_1)) / (2A)$$
$$A = \langle p_1, p_2, p_3 \rangle = ((s_2 - s_1)(t_3 - t_1) - (s_3 - s_1)(t_2 - t_1)) / 2$$

The larger and smaller singular values of the Jacobian $[S_s, S_t]$ are given respectively by the following:

$$\Gamma = \sqrt{1/2\left((a+c) + \sqrt{(a-c)^2 + 4b^2}\right)} \quad \text{max singular value}$$
$$\gamma = \sqrt{1/2\left((a+c) - \sqrt{(a-c)^2 + 4b^2}\right)} \quad \text{min singular value}$$

where $a = S_s \cdot S_s$, $b = S_s \cdot S_t$, and $c = S_t \cdot S_t$. The singular values Γ and γ represent the largest and smallest length obtained when mapping unit-length vectors from the texture domain to the surface, i.e. the largest and smallest local “stretch”. We define two stretch norms over triangle *T* as follows:

$$L^2(T) = \sqrt{(\Gamma^2 + \gamma^2)/2}$$

Hence, the method of the second prior art begins by partitioning the mesh into charts using planarity and compactness heuristics. It creates a stretch-minimizing parametrization within each chart, and resizes the charts based on the resulting stretch. Next, the second prior art approach simplifies the mesh while respecting the chart boundaries. The parametrization is re-optimized to reduce both stretch and deviation over the whole progressive mesh sequence. Finally, the charts are packed into a texture atlas.

FIG. 5 shows a partition result 500 for texture mapping progressive meshes obtained by the second prior art approach.

SUMMARY

These and other drawbacks and disadvantages of the prior art are addressed by the present invention, which is directed to multi-resolution 3D textured mesh coding.

According to an aspect of the present principles, there is provided a method for encoding a three-dimensional model represented by a mesh of two-dimensional polygons. The method includes simplifying the mesh to obtain a simplified mesh. The method further includes generating a texture image. The texture image represents textures for the mesh and the simplified mesh. The method also includes encoding the texture image to form a base layer and an enhancement layer. The base layer corresponds to a coarse representation of the texture image and the enhancement layer provides a refinement of the base layer.

According to another aspect of the present principles, there is provided an apparatus for encoding a three-dimensional model represented by a mesh of two-dimensional polygons. The apparatus includes a mesh simplifier for simplifying the mesh to obtain a simplified mesh. The apparatus further includes a texture image generator for generating a texture image. The texture image represents textures for the mesh and the simplified mesh. The apparatus also includes an encoder for encoding the texture image to form a base layer and an enhancement layer. The base layer corresponds to a coarse representation of the texture image and the enhancement layer provides a refinement of the base layer.

According to still another aspect of the present principles, there is provided a method. The method includes decoding a simplified mesh, a mesh, and a coarse representation of a texture map from a bitstream. The coarse representation of the texture map corresponds to both the simplified mesh and the mesh. The method further includes forming a three-dimensional model for the mesh, using the coarse representation of the texture map. The method also includes decoding a refinement of the texture map to form a refined representation of the texture map. The method additionally includes enhancing the three-dimensional model for the mesh, using the refined representation of the texture map.

According to yet another aspect of the present principles, there is provided an apparatus. The apparatus includes a decoder for decoding a simplified mesh, a mesh, and a coarse representation of a texture map from a bitstream. The coarse representation of the texture map corresponds to both the simplified mesh and the mesh. The apparatus further includes a rendering device for forming a three-dimensional model for the mesh using the coarse representation of the texture map. The decoder decodes a refinement of the texture map to form a refined representation of the texture map, and the rendering device enhances the three-dimensional model for the mesh using the refined representation of the texture map.

These and other aspects, features and advantages of the present invention will become apparent from the following detailed description of exemplary embodiments, which is to be read in connection with the accompanying drawings.

BRIEF DESCRIPTION OF THE DRAWINGS

FIGs. 1A-D respectively show texture images 110, 120, 130, 140 having successive refinements of texture, in accordance with the first prior art approach;

FIG. 2 show a texture atlas 200 generated from a mesh unwrapping, in accordance with the first prior art approach;

FIG. 3 shows an existing surface signal 310 sampled to obtain a texture image 320, in accordance with the second prior art approach;

FIG. 4 shows an exemplary mapping of a 2D texture domain 410 to a 3D surface 420, in accordance with the second prior art approach;

FIG. 5 shows a partition result 500 for texture mapping progressive meshes, in accordance with the second prior art approach;

FIG. 6 shows various exemplary 3D representations with different levels of qualities obtained using progressive meshes and progressive coding of the texture image, in accordance with an embodiment of the present principles;

FIG. 7 is a high level block diagram showing both a system and method for multi-resolution 3D textured mesh coding, in accordance with an embodiment of the present principles;

FIG. 8 shows an exemplary texture atlas 800 to which the present principles may be applied, in accordance with an embodiment of the present principles;

FIG. 9 shows an exemplary bitstream format 900, in accordance with an embodiment of the present principles;

FIG. 10 shows an exemplary method 1000 for providing a multi-resolution 3D textured mesh coding to one or more users, in accordance with an embodiment of the present principles;

FIG. 11 shows an exemplary environment 1100 to which the present principles may be applied, in accordance with an embodiment of the present principles;

FIG. 12 shows an example of a boundary problem 1200 to which the present principles may be applied, in accordance with an embodiment of the present principles;

FIG. 13 shows a segmentation 1310 of an image 1300, in accordance with an embodiment of the present principles;

FIG. 14 shows a flowchart of a method 1400 implementing a feature detection step for texture image pattern aware partitioning, in accordance with an embodiment of the present principles;

FIG. 15 further describes step 1430 of FIG. 14, in accordance with an embodiment of the present principles; and

FIG. 16 shows a flowchart of a method 1600 implementing a chart growing step for texture image pattern aware partitioning, in accordance with an embodiment of the present principles.

DETAILED DESCRIPTION

The present principles are directed to multi-resolution 3D textured mesh coding. Advantageously to that end, the present principles provide a method and apparatus for transmitting both shape and texture progressively. According to one or more embodiments of the present principles, a progressive mesh is constructed such that all meshes in the progressive mesh sequence share a common texture parametrization (i.e., a map having common texture coordinates for each of the corresponding progressive levels). Consequently, the corresponding texture coordinates are ready for progressive transmission. For each level of detail (LOD), we just need to transmit the texture coordinates for the related refined vertices, while for unchanged vertices, their texture coordinates remain constant during the whole procedure.

The common texture image is progressively encoded to provide different levels of qualities. After the texture image is encoded, a bitstream corresponding to a low level of quality is transmitted first to provide a coarse representation of details. As more data corresponding to texture refinements are transmitted, a finer representation of the texture image can be reconstructed at the receiving side.

In one embodiment, an encoder based on wavelet transformation can be used. In other embodiments, other scalable encoders, for example, such as a JPEG2000 encoder and an H.264 SVC encoder, can also be used to provide different levels of resolutions and fidelities. Of course, the present principles are not limited to solely the preceding encoders and, thus, other encoders of different types, different generations, and so forth, may also be used in accordance with the present principles, while maintaining the spirit of the present principles.

FIG. 6 shows various exemplary 3D representations with different levels of qualities obtained using progressive meshes and progressive coding of the texture image, in accordance with an embodiment of the present principles. To that end, 3 progressive meshes are provided, namely mesh 610, mesh 620, and mesh 630. Mesh 610 has the fewest number of triangles in representing the object’s shape. As more triangles are progressively used in meshes 620 and 630, the shape of the object gets improved (e.g., unnecessary corners are “softened” and more details are added, and so froth). All three meshes 610, 620, and 630 share an underlying common texture image. The common texture is encoded using different resolutions. In the example of FIG. 6, the common texture is encoded at a lowest resolution (640), a medium resolution (650), and a highest resolution (660). Meshes 670, 680, and 690 are also encoded. Mesh 670 has the lowest resolution, while mesh 690 has the highest resolution.

It is to be appreciated that mesh 620 includes the information from mesh 610, and medium resolution texture 650 includes the content from lowest resolution texture 640. Mesh 610 can be combined with either lowest resolution texture 640 or medium resolution texture 650, and lowest resolution texture 640 can be combined with either 620 or 610. If medium resolution texture 650 is used, lowest resolution texture 640 is not needed to obtain mesh 680. Similarly, if mesh 620 is used, mesh 610 is not needed to obtain mesh 680.

The present description illustrates the principles of the present invention. It will thus be appreciated that those skilled in the art will be able to devise various arrangements that, although not explicitly described or shown herein, embody the principles of the invention and are included within its spirit and scope.

All examples and conditional language recited herein are intended for pedagogical purposes to aid the reader in understanding the principles of the invention and the concepts contributed by the inventor to furthering the art, and are to be construed as being without limitation to such specifically recited examples and conditions.

Moreover, all statements herein reciting principles, aspects, and embodiments of the invention, as well as specific examples thereof, are intended to encompass both structural and functional equivalents thereof. Additionally, it is intended that such equivalents include both currently known equivalents as well as equivalents developed in the future, i.e., any elements developed that perform the same function, regardless of structure.

Thus, for example, it will be appreciated by those skilled in the art that the block diagrams presented herein represent conceptual views of illustrative circuitry embodying the principles of the invention. Similarly, it will be appreciated that any flow charts, flow diagrams, state transition diagrams, pseudocode, and the like represent various processes which may be substantially represented in computer readable media and so executed by a computer or processor, whether or not such computer or processor is explicitly shown.

The functions of the various elements shown in the figures may be provided through the use of dedicated hardware as well as hardware capable of executing software in association with appropriate software. When provided by a processor, the functions may be provided by a single dedicated processor, by a single shared processor, or by a plurality of individual processors, some of which may be shared. Moreover, explicit use of the term “processor” or “controller” should not be construed to refer exclusively to hardware capable of executing software, and may implicitly include, without limitation, digital signal processor (“DSP”) hardware, read-only memory (“ROM”) for storing software, random access memory (“RAM”), and non-volatile storage.

Other hardware, conventional and/or custom, may also be included. Similarly, any switches shown in the figures are conceptual only. Their function may be carried out through the operation of program logic, through dedicated logic, through the interaction of program control and dedicated logic, or even manually, the particular technique being selectable by the implementer as more specifically understood from the context.

In the claims hereof, any element expressed as a means for performing a specified function is intended to encompass any way of performing that function including, for example, a) a combination of circuit elements that performs that function or b) software in any form, including, therefore, firmware, microcode or the like, combined with appropriate circuitry for executing that software to perform the function. The invention as defined by such claims resides in the fact that the functionalities provided by the various recited means are combined and brought together in the manner which the claims call for. It is thus regarded that any means that can provide those functionalities are equivalent to those shown herein.

FIG. 7 is a high level block diagram showing both a system and method (collectively represented by the reference figure numeral 700) for multi-resolution 3D textured mesh coding, in accordance with an embodiment of the present principles. The system 700 generates bitstreams providing different levels of quality for various representations of a 3D object. For example, a bitstream corresponding to a simplified mesh (e.g., such as mesh 610) and a low-resolution texture image (e.g., such as texture 640) provides a low quality and, therefore, a coarse representation (e.g., such as multi-resolution mesh 670) of the 3D object. When more information, for example, additional information about the texture image, is received in the bitstream, a texture image with a higher resolution can be formed. Similarly, when more details on the mesh are received, a finer mesh can be generated. With a texture image at a higher resolution and a finer mesh, the bitstream corresponds to a finer mesh with fine texture. Of course, as is readily appreciated by one of ordinary skill in the art, more levels are possible than those shown in FIG. 7.

When a coarse representation is rendered at a decoder and quality improvement is desired, only additional information corresponding to the quality refinement needs to be transmitted to the decoder. This advantageously leads to information savings.

The system 700 includes a mesh partition and parametrization device 710, a mesh simplifier 715, a parametrization optimizer 720, a new texture atlas generator 730, and a texture image compressor 740. Each of these elements perform various steps, and hence, these elements and their associated functions will be described in further detail hereinafter with respect to various embodiments of the present principles. It is to be appreciated that the preceding elements can be implemented in hardware, software, or a combination thereof. In an embodiment, each element can include a processor and associated memory, or two or more of these elements may share a processor and memory. These and other configurations for these elements are readily contemplated by one of ordinary skill in the art.

The system takes as inputs, for example, a fine mesh 701 and a fine texture 702 representative of a 3D object. In particular, at step 771, mesh partition and parametrization device 710 receives the fine mesh 701 as an input thereof, and at step 772, the new atlas generator 730 receives the fine texture 702 as an input thereof. At step 773, the mesh 701 is partitioned and parametrized by mesh partition and parametrization device 710 to obtain a set of texture charts. At step 774, the mesh 701 is simplified by mesh simplifier 715 to define a progressive mesh. At step 775, parametrization optimization is applied to the progressive mesh by parametrization optimizer 720. At step 776, the set of texture charts are packed into a square texture image to form a new texture atlas by new atlas generator 730. At step 777, the square texture image (i.e., the texture atlas) is compressed by texture image compressor 740. Steps 773 through 777 are described in further detail herein below.

<u>Mesh Partition and Parametrization</u>

A single unfolding of an arbitrary mesh onto a texture image may create regions of high distortion, so generally a mesh must be partitioned into a set of charts, which are regions with disk-like topology. Regarding step 773, the mesh 701 is partitioned into a set of charts, and each chart is parametrized by creating a one-to-one mapping from the surface region to a 2D polygon. Thus, the texture charts are these (parametrized) 2D polygons. Hence, regarding step 773, we partition (segment) the mesh 701 into a set of charts, parametrize the charts individually onto 2D polygons, and pack the 2D polygons into a single texture image. The single texture image can be compressed and transmitted in a progressive way, meaning that the single texture image can have different resolutions. In an embodiment, step 773 can also include resizing the texture charts (2D polygons) according to their stretch so that charts that have more stretch get more texture area.

Existing segmentation techniques do not consider the texture image pattern factor, which will lead to artifacts. In order to minimize artifacts, the segmentation algorithm is designed such that the “meaningful texture image patch” is grouped into one chart while meeting some geometric constraints.

In our case, we need to incorporate the influence from the pattern of the initial texture image 702 into the segmentation operation. Let G be the gradient magnitude from the Sobel operator at some point on the mesh 701 for a particular objective function. The objective function in our case will be a weighted sum of the geometric cost and $\int_{tri} Gds$, the latter being the integral of G over a triangle as follows:

$$\alpha \times geometricCost + \beta \times \sum_{i=0,\ldots,n} \int_{tri} Gds,$$

where α, β are relative weights, *n* is the number of triangles inside a chart, and *geometricCost* is the sum of: (1) the mean-squared distance of the chart to the best-fitting plane through the chart; and (2) the perimeter length of this chart. It is to be appreciated that the partitioning (per step 773) is an iterative process. We keep a list of this objective function for each possible merge operation and, after each operation, we update this list.

To parametrize the charts, we adapt the minimization of a geometric stretch metric of the aforementioned second prior art approach shown and described with respect to FIG. 4. This metric penalizes under-sampling, and results in samples that are uniformly distributed over the surface.

Moreover, we provide another approach to partitioning, namely a texture image pattern aware mesh partitioning approach described in further detail herein below with respect to FIGs. 14-16.

Mesh Simplification

Regarding step 774, the input mesh is simplified to define a progressive mesh. One objective during the simplification is to minimize texture deviation. The texture deviation is measured as the geometric error according to parametric correspondence. In particular, the texture deviation between a simplifying mesh M^i and the original mesh M^n at a point $p^i \in M^i$ is defined as $\left\| p^i - p^n \right\|$, where p^n is the point on M^n with the same parametric location (as p^i) in the texture domain.

Parametrization Optimization

Regarding step 775, we optimize the parametrization over the entire progressive mesh minimizing stretch and deviation at all levels of details (after we have determined the progressive mesh simplification sequence as per step 774). The objective function will be a weighted sum of the texture stretch and deviation on all meshes $M^0 \cdots M^n$ as follows:

$$Energy = \sum_{i=0,\cdots n} weight(i)[\alpha \times Stretch(i) + \beta \times Deviation(i)] ,$$

where α, β are relative weights between stretch and deviation, and $weight(i)$ is the relative weight assigned to each LOD mesh (i.e., mesh *i*) in the progressive mesh sequence.

New Texture Atlas Generation

Regarding step 776, we then pack the texture charts (obtained in step 773) in a square texture image to form a new texture atlas. While we mention a square texture image here for illustrative purposes, it is readily appreciated that other geometric shapes can also be used including, but not limited to, rectangles. Most existing techniques aim at minimizing the resolution of the atlas. However, atlas resolution is not equal to compressed size. FIG. 8 shows an exemplary texture atlas (image) 800 to which the present principles may be applied, in accordance with an embodiment of the present principles. The texture atlas 800 includes various texture charts. As shown in FIG. 8, significant similarity exists among the charts in the texture atlas 800, where chart instances sharing the same pattern tend to have similar textures.

We categorize the charts based on the color histogram and texture similarity. The aim of the packing process includes minimizing the atlas resolution, and putting charts of the same category together as much as possible. Some optimization operations may be necessary in order to make the texture atlas more “compact”. After execution of step 776, we have optimized texture coordinates for each vertex.

Texture Image Compression

Regarding step 777, we perform texture image compression. To that end, in an embodiment, we apply predictive compression to the texture image, and make the texture image ready for progressive transmission. Any image/video encoders that can perform encoding progressively be used to compress the texture image. For example, but clearly not meant to represent an exhaustive list, a wavelet-based image encoder, a JPEG2000 encoder, and an H.264/AVC encoder can be used to implement the present principles.

FIG. 9 shows an exemplary bitstream format 900, in accordance with an embodiment of the present principles. Of course, the format 900 is for illustrative purposes and, thus, other bitstream formats may also be used, while maintaining the spirit of the present principles. For the first level (level 1) that corresponds to the lowest quality, the encoded data corresponding to the vertices' positions, texture coordinates, and texture image and represented as "position level 1" in block 910, "texture coordinates level 1" in block 920, and "image level 1" in block 930, respectively. For the next level (level 2), only the difference between level 2 and level 1 are encoded for vertices' positions and texture coordinates in "position level 2" in block 940 and in "texture coordinates level 2" in block 950, respectively. The texture image corresponding to level 2 is encoded into "image level 2" in block 960. Note that the texture image is progressively encoded. Thus, to reconstruct the texture image at level 2, encoded data for both image levels 1 and 2 are needed. Similarly, to reconstruct the vertices' positions and texture coordinates at level 2, the vertices' positions and texture coordinates at level 1 are also required.

For each additional quality level, additional data for vertices' positions, texture coordinates and texture image are used to represent a refinement over the previous level. When data for all levels are received, the decoder can render the 3D object with a fine mesh and fine texture. In the example shown in FIG. 9, when the data corresponding to the highest level n, "position level n" in block 970, "texture coordinates level n", and "image level n" are received, as well as its previous levels, the highest quality can be rendered at the decoder.

Thus, according to the present principles, vertices' positions, texture coordinates, and texture images all allow for progressive transmission. This flexibility in progressive transmission is important to certain interactive applications.

FIG. 10 shows an exemplary method 1000 for providing a multi-resolution 3D textured mesh coding to one or more users, in accordance with an embodiment of the present principles. The method 1000 pertains to a receiving/decoding side, in compliment to method 700 described above. At step 1010, a bitstream is received.

At step 1020, the corresponding vertices' positions, texture coordinates, and texture image conveyed within the bitstream are decoded. When the bitstream only includes data for the lowest quality level, the decoding is based on the bitstream itself. When the bitstream includes data for a higher quality level, the vertices' positions, texture coordinates, and the texture image will be decoded based on the bitstream for the current quality level and previously decoded data for the lower levels. At step 1030, a 3D object is rendered using the vertices' positions, texture coordinates, and the texture image. At step 1040, it is determined whether or not the user is satisfied with the quality of the 3D object (rendered per step 1030). If so, then the method is terminated. Otherwise, the method proceeds to step 1050. At step 1050, the user is afforded the opportunity to request more data to refine the 3D object and if the user so requests, then the decoder consequently receives additional data that correspond to a higher level of quality and the method returns to step 1020. Accordingly, the additional received data will be decoded at step 1020 and used for refining the 3D object at step 1030.

FIG. 11 shows an exemplary environment 1100 to which the present principles may be applied, in accordance with an embodiment of the present principles. The environment involves a server 1110, one or more networks (hereinafter simply represented by the Internet) 1120, and various user devices 1131 and 1132. The server receives a representation 1105 of a 3D object in the form of a multi-resolution mesh with texture, and implements the present principles as represented, for example, in FIG. 7, to perform geometry streaming and texture streaming over the Internet 1120. The various user devices 1131 and 1132 receive the geometry streams and texture streams, and decode the same, for example as described with respect to FIG. 10 to provide 3D representations 1185 (such as 670, 680, and 690 of FIG. 6) having various qualities associated therewith to the user devices 1131 and 1132. The same may involve the user 1199 requesting a higher quality (per step 1050 of method 1000) via some feedback mechanism, which is then provided to the server. Accordingly, each of the user devices 1131 and 1132 include a respective decoder 1150. Such a decoder may be implemented in hardware, software, or a combination thereof. In an embodiment, the decoder can include a processor and associated memory, or may use the processor and associated memory of the device within which it is found. Moreover, each of the user devices

1131 and 1132 include a rendering device (e.g., a display) 1160. These and other configurations are readily contemplated by one of ordinary skill in the art.

It is to be appreciated that the environment 1100 is provided for exemplary purposes and, thus, the present principles may be applied to many other environments, as readily determined by one of ordinary skill in the art, while maintaining the spirit of the present principles.

Texture Image Pattern Aware Mesh Partitioning

The following provides an alternative approach to partitioning than that described above with respect to step 773 of FIG. 7. That is, the following approach provides an effective way to reduce artifacts for texture mapping of multi-resolution mesh. It is to be appreciated that given the teachings of the present principles provided herein, the following texture image pattern aware partitioning approach can be applied to various scenarios regarding texture mapping, unfolding of meshes, and so forth, as readily appreciated by one of ordinary skill in the art, while maintaining the spirit of the present principles.

In multi-resolution of geometry and texture, geometry is modified through the simplification operator. In the case of texture mapping, a modified geometry imposes a different texture mapping function which may have a boundary problem. FIG. 12 shows an example of a boundary problem 1200 to which the present principles may be applied, in accordance with an embodiment of the present principles. When an edge of a vertex on a boundary 1205 in a chart is simplified, texture maps for the two charts 1221 and 1222 may lead to artifacts. The reason is that the texture image is mapped onto different geometries. Using texture images for the different geometries poses a difficulty with respect to mapping. That is, if two neighboring points are mapped onto the distant boundaries of a texture image, the texture mapping function for these two points may result in blurred effect.

Thus, to minimize artifacts, we will design the segmentation algorithm in such a way as to make the meaningful texture image patch' into one chart. In other words, it is suitable to generate large charts with most of their boundaries in 'color-sharpness' zones. FIG. 13 shows a segmentation 1310 of an image 1300, in accordance with an embodiment of the present principles. This image 1300 is partitioned into different sub regions of homogeneity. The boundary of a "meaningful texture image patch" usually corresponds to a high image gradient zone. Thus, we will take into account the image gradient as a factor when we partition the input model. It is also possible to use other and/or more elaborate criteria, as readily determined by one of ordinary skill in the art, given the teachings of the present principles provided herein. Nonetheless, for illustrative purposes, we note that such other and/or more elaborate criteria capable of being used for segmentation may include, but is not limited to, a model- based approach (i.e., considering one or more model parameters), a Histogram-based approach, and so forth.

Usually in texture mapping the model to be textured is decomposed into charts homomorphic to discs, where each chart is parametrized, and the unfolded charts are packed in a texture space. In the decomposition step, if a texture image pattern has not been considered, texture artifacts may occur on the boundaries.

The present principles take into account the texture image pattern when partitioning the mesh into charts. To minimize artifacts, the segmentation algorithm is designed in such a way as to avoid chart boundaries in low image gradient zones. In other words, it is suitable to generate large charts with most of their boundaries in “color-sharp” zones.

The partition operation decomposes the model into a set of charts, and is intended to meet the following requirements as much as possible and/or practical given the intended application and available resources:

1. Charts boundaries should be positioned in such a way that most of the discontinuities between the charts will be located in zones where they will not cause texture artifacts;
2. Charts must be homomorphic to discs, and it must be possible to parametrize them without introducing too much deformation;
3. Reduce color discontinuity; and
4. Minimize texture distortion.

In an embodiment, the texture image pattern aware partitioning approach can be considered to involve two stages. The first stage pertains to feature detection, which finds boundaries corresponding to high image gradient zones of the model. The second stage pertains to chart growing, which makes the charts meet at these features curves. We note that the terms “borders”, “boundaries”, and “feature curves” are used interchangeably herein with respect to the pattern aware mesh partition approach.

Detect features

The feature detection phase can be outlined as follows:

- Compute a color-sharpness criterion on the edges (where “boundaries” and “edges” are used interchangeably herein). For illustrative purposes, we use image gradient as the color-sharpness criterion. It is also possible to use other and/or more elaborate criteria.
- Choose a threshold so that a certain proportion of the edges is filtered out.
- For each of the remaining edges, grow a feature curve by applying Algorithm 1.

Let us define $sharpness = G$, where *G* is the gradient magnitude from the Sobel operator. It measures the change of color of the texture image. We would like to generate charts with most of their boundaries in “color-sharp” zones, which means the larger a value for *sharpness’*s of a given candidate boundary under consideration, the more chance for that candidate boundary to be an actual chart boundary. The following algorithm (hereinafter algorithm 1) is provided directed to feature detection:

```
obtain_feature_curve(edge start)
        vector<edge> detected_feature
        edge $h' = start$

        do
                use depth-first search to find the string S of edges starting with
                $h'$ and such that:
                    • two consecutive edges of S share a vertex
                    • the length of S is not larger than threshold1
                    • sharpness(S) $\longleftarrow \sum_{e \in S} sharpness(e)$ is maximum. When
                      depth-first search is utilized to find strings of edges
                      starting with h’, you may have a couple of such strings.
                      The string which has the maximum sharpness is selected.
                    • no edge of S goes backward (relative to $h'$)
                    • no edge of S is tagged as a feature neighbor

                $h' \longleftarrow$ second item of S
```

```
            append $h'$ to detected_feature
        while(sharpness(S) > threshold2)

        if (length(detected_feature) > min_feature_length) then
            tag the elements of detected_feature as features
            tag the edges in the neighborhood of detected_feature as feature
            neighbors

        end

    end
```

Algorithm 1 attempts to predict the best paths, and filters out the small features caused by noise. FIG. 14 shows a flowchart of a method 1400 implementing a feature detection step (Algorithm 1) for texture image pattern aware partitioning, in accordance with an embodiment of the present principles. At step 1410, a color-sharpness criterion is computed on the edges. For illustrative purposes, we use image gradient for the color-sharpness criterion. However, it is to be appreciated that other and/or more elaborate criteria can be used, while maintaining the spirit of the present principles. At step 1420, a threshold is chosen such that a certain proportion of the edges is filtered out. At step 1430, for each of the remaining edges, a feature curve is grown from the remaining edges. Step 1430, involves applying Algorithm 1 to each of the remaining edges to grow the feature curves therefrom. The purpose of these feature curves is to serve as the "meeting boundary" of different charts. These charts are obtained through growing from a set of seeds, as described in further detail herein below. FIG. 15 further describes step 1430 of FIG. 14, in accordance with an embodiment of the present principles. At step 1505, edge h' is set equal to start (i.e., a starting point for detecting features from which the feature curves are generated). At step 1510, a depth-first search is used to find the string S of edges starting with edge h' which satisfies each of the following conditions: two consecutive edges of S share a vertex; the length of S is not larger than threshold1; *sharpness(S)* $\longleftarrow \sum_{e \in S} sharpness(e)$ is maximum; no edge of S goes backward (relative to edge h'); and no edge of S is tagged as a feature neighbor. The most advantage as far as predicting the best paths and filtering out the small features caused by noise can be obtained by using all of the preceding five conditions. At step 1515, edge h' is set equal to the 2nd item of string S of edges. At step 1520, edge h' is appended to detected_feature. At step 1525, it is determined whether or not the sharpness(S) is greater than the threshold. If so, the method returns to step 1510. Otherwise, the method proceeds to step 1530. At step 1530, it is determined whether or not the length of detected_feature is greater than the min_feature_length. If so, then the method proceeds to step 1535. Otherwise, the method 1500 is terminated. At step 1535, the element detected_feature is tagged as a (detected) features, and the edges in the neighborhood of the (detected) feature are tagged as feature neighbors.

Expand charts

Once the color-sharpness features have been detected, the charts can be created. In an embodiment, our approach is a greedy algorithm, expanding all the charts simultaneously from a set of seeds as follows:

- A front is propagated from the borders and feature curves detected by the previous algorithm (Algorithm 1), to compute a distance_to_features function at each facet of a geometric shape in the charts (to be merged). Then the seeds are found to be the local maxima of this distance_to_features function.
- Charts are merged if they meet at a small distance from their seed.

The following algorithm (hereinafter algorithm 2) is provided directed to chart growing:

```
priority_queue<edge> Heap sorted by dist(facet(edge))
set<edge> chart_boundaries initialized with all the edges of the surface

for each facet F where dist(F) is a local maximum
    create a new chart with seed F
    add the edges of F to Heap
end

while(Heap is not empty)
        • edge $h \longleftarrow e \in Heap$ such that dist(e) is maximum
        • remove h from Heap
        • facet F ⟵ facet(h)
        • facet $F_{opp} \longleftarrow$ the opposite facet of F relative to h
    if ( chart($F_{opp}$) is undefined ) then
    if(geometry_constraint_satisfied)then
                • add $F_{opp}$ to chart(F)
                • remove E from chart_boundaries
                • remove edges which do not link two other
                  chart boundary edges from
                  chart_boundaries
                • add the edges of $F_{opp}$ belonging to
                  chart_boundaries to Heap
        else
```

```
                    • create a new chart with $F_{opp}$
                    • add the edges of $F_{opp}$ to Heap
            else if ( chart($F_{opp}$) $\neq$ chart(F) and
            geometry_constraint_satisfied and
                max_dist(chart(F)) - dist(F) $< \varepsilon$ and
                max_dist(chart($F_{opp}$)) - dist(F) $< \varepsilon$ then
                merge chart(F) and chart($F_{opp}$)

            end
        end
    end
```

This region growing uses the following:

- *distance_to_feature* is stored in each facet *F*, and denoted by *dist(F);*
- for each chart *C*, *max_dist(C)* denotes the maximum distance to features for all the facets of *C;*
- the set of edges *chart_boundaries* represents the borders of all charts;
- ***geometry_constraint_satisfied*** means that the cost (sum of planarity measure and compactness measure) of the merging operation does not exceed a user-specified threshold.

FIG. 16 shows a flowchart of a method 1600 implementing a chart growing step (Algorithm 2) for texture image pattern aware partitioning, in accordance with an embodiment of the present principles. At step 1610, a front is propagated from the borders and feature curves (detected by method 500, pertaining to Algorithm 1), to compute a distance_to_features function at each facet. At step 1620, the seeds are found to be the local maxima of the distance_to_features function. At step 1630, charts are merged if they meet at a small distance (e.g., below a given threshold distance) from their seed.

These and other features and advantages of the present invention may be readily ascertained by one of ordinary skill in the pertinent art based on the teachings herein. It is to be understood that the teachings of the present invention may be implemented in various forms of hardware, software, firmware, special purpose processors, or combinations thereof.

Most preferably, the teachings of the present invention are implemented as a combination of hardware and software. Moreover, the software may be implemented as an application program tangibly embodied on a program storage unit. The application program may be uploaded to, and executed by, a machine comprising any suitable architecture. Preferably, the machine is implemented on a computer platform having hardware such as one or more central processing units ("CPU"), a random access memory ("RAM"), and input/output ("I/O") interfaces. The computer platform may also include an operating system and microinstruction code. The various processes and functions described herein may be either part of the microinstruction code or part of the application program, or any combination thereof, which may be executed by a CPU. In addition, various other peripheral units may be connected to the computer platform such as an additional data storage unit and a printing unit.

It is to be further understood that, because some of the constituent system components and methods depicted in the accompanying drawings are preferably implemented in software, the actual connections between the system components or the process function blocks may differ depending upon the manner in which the present invention is programmed. Given the teachings herein, one of ordinary skill in the pertinent art will be able to contemplate these and similar implementations or configurations of the present invention.

Although the illustrative embodiments have been described herein with reference to the accompanying drawings, it is to be understood that the present invention is not limited to those precise embodiments, and that various changes and modifications may be effected therein by one of ordinary skill in the pertinent art without departing from the scope or spirit of the present invention. All such changes and modifications are intended to be included within the scope of the present invention as set forth in the appended claims.

CLAIMS

1. A method for encoding a three-dimensional model represented by a mesh of two-dimensional polygons, comprising:

simplifying (774) the mesh to obtain a simplified mesh;

generating (776) a texture image, the texture image representing textures for the mesh and the simplified mesh; and

encoding (777) the texture image to form a base layer and an enhancement layer, the base layer corresponding to a coarse representation of the texture image and the enhancement layer providing a refinement of the base layer.

2. The method of claim 1, further comprising:

partitioning (773) the mesh and the simplified mesh into a plurality of charts;

parametrizing (773) the plurality of charts by minimizing an energy function which depends on gradients of the texture image.

3. The method of claim 1, wherein encoding (777) the texture image is performed by a wavelet-based encoder.

4. The method of claim 1, further comprising:

encoding (740) the mesh and the simplified mesh to an encoded mesh and an encoded simplified mesh respectively;

outputting the encoded simplified mesh, the encoded mesh, the base layer of the texture map, and the enhancement layer of the texture map.

5. The method of claim 4, wherein the encoded mesh is encoded in response to a difference between the mesh and the simplified mesh.

6. The method of claim 4, wherein the outputting of at least one of the encoded mesh and the enhancement layer of the texture map is performed in response to a user request.

7. The method of claim 1, further comprising:
partitioning the mesh into a plurality of charts;
wherein partitioning the mesh into a plurality of charts comprises:
computing (1410) a color-sharpness criterion on edges in the texture image, wherein the color-sharpness criterion relates to a gradient of the texture image;
filtering (1420) out a certain proportion of the edges using a threshold related to the color-sharpness criterion to obtain remaining edges; and
growing (1430) a respective feature curve from each of the remaining edges.

8. An apparatus for encoding a three-dimensional model represented by a mesh of two-dimensional polygons, comprising:
a mesh simplifier (715) for simplifying the mesh to obtain a simplified mesh;
a texture image generator (730) for generating a texture image, the texture image representing textures for the mesh and the simplified mesh;
an encoder (740) for encoding the texture image to form a base layer and an enhancement layer, the base layer corresponding to a coarse representation of the texture image and the enhancement layer providing a refinement of the base layer.

9. The apparatus of claim 8, further comprising a mesh partition and parametrization device for partitioning the mesh and the simplified mesh into a plurality of charts, and parametrizing the plurality of charts by minimizing an energy function which depends on gradients of the texture image.

10. The apparatus of claim 8, wherein said encoder (740) is a wavelet-based encoder.

11. The apparatus of claim 8, wherein said encoder (740) encodes the mesh and the simplified mesh to an encoded mesh and an encoded simplified mesh respectively, and outputs the encoded simplified mesh, the encoded mesh, the base layer of the texture map, and the enhancement layer of the texture map.

12. The apparatus of claim 11, wherein the encoded mesh is encoded in response to a difference between the mesh and the simplified mesh.

13. The apparatus of claim 8, further comprising a mesh partitioning device (710) for partitioning the mesh into a plurality of charts by computing a color-sharpness criterion on edges in the texture image, filtering out a certain proportion of the edges using a threshold related to the color-sharpness criterion to obtain remaining edges, and growing a respective feature curve from each of the remaining edges, wherein the color-sharpness criterion relates to a gradient of the texture image.

14. A method, comprising:

decoding (1020) a simplified mesh, a mesh, and a coarse representation of a texture map from a bitstream, the coarse representation of the texture map corresponding to both the simplified mesh and the mesh;

forming (1030) a three-dimensional model for the mesh, using the coarse representation of the texture map;

decoding (1050) a refinement of the texture map to form a refined representation of the texture map; and

enhancing (1030) the three-dimensional model for the mesh, using the refined representation of the texture map.

15. The method of claim 14, wherein the coarse representation and the refinement of the texture map are decoded using a wavelet-based decoder.

16. The method of claim 14, wherein the step of decoding the mesh comprises:

decoding a difference between the simplified mesh and the mesh; and

combining the difference and the simplified mesh to form the mesh.

17. The method of claim 14, further comprising receiving (1050) a user request to enhance said three-dimensional model.

18. An apparatus, comprising:

a decoder (1150) for decoding a simplified mesh, a mesh, and a coarse representation of a texture map from a bitstream, the coarse representation of the texture map corresponding to both the simplified mesh and the mesh; and

a rendering device (1160) for forming a three-dimensional model for the mesh using the coarse representation of the texture map,

wherein the decoder (1150) decodes a refinement of the texture map to form a refined representation of the texture map, and the rendering device (1160) enhances the three-dimensional model for the mesh using the refined representation of the texture map.

19. The apparatus of claim 18, wherein at least a portion of the decoder (1150) that decodes the coarse representation and the refinement of the texture map comprises a wavelet-based decoder.

20. The apparatus of claim 18, wherein said decoder (1150) decodes the mesh by decoding a difference between the simplified mesh and the mesh, and combining the difference and the simplified mesh to form the mesh.

21. The apparatus of claim 18, further comprising a feedback mechanism for receiving a user request to enhance said three-dimensional model.

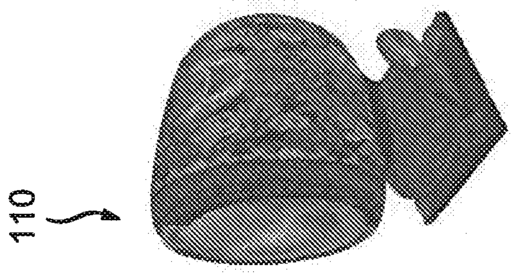

FIG. 1A

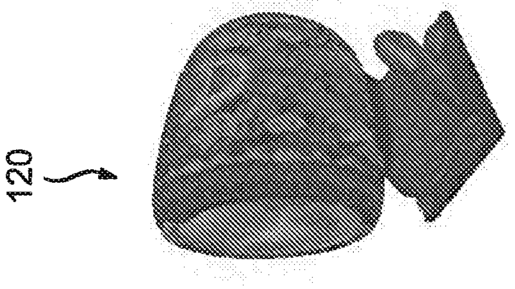

FIG. 1B

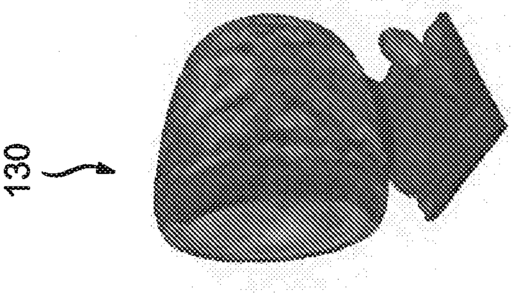

FIG. 1C

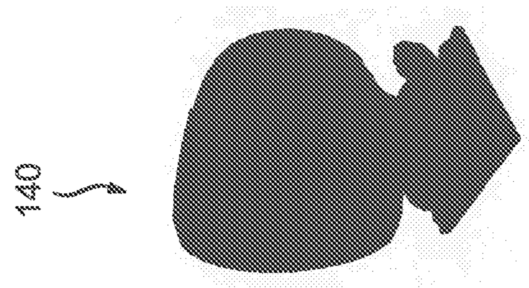

FIG. 1D

FIG. 2

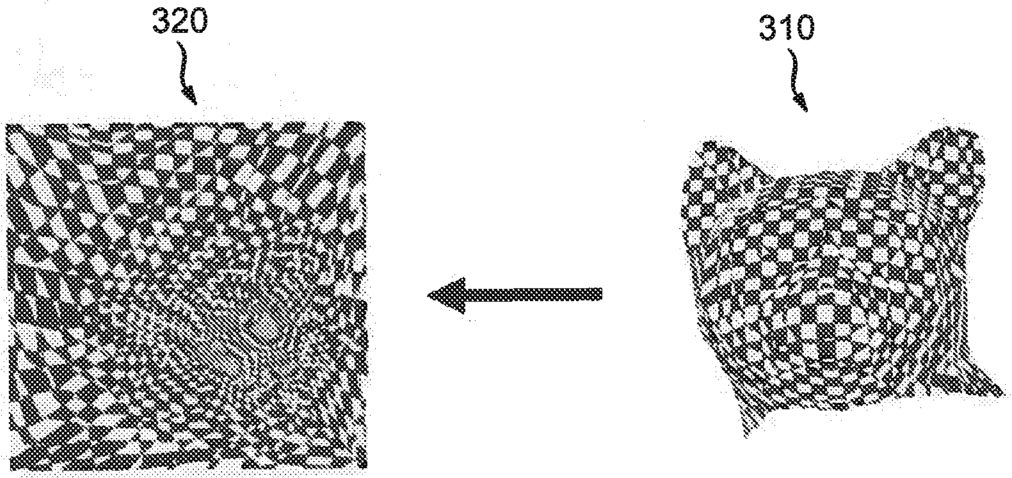

FIG. 3

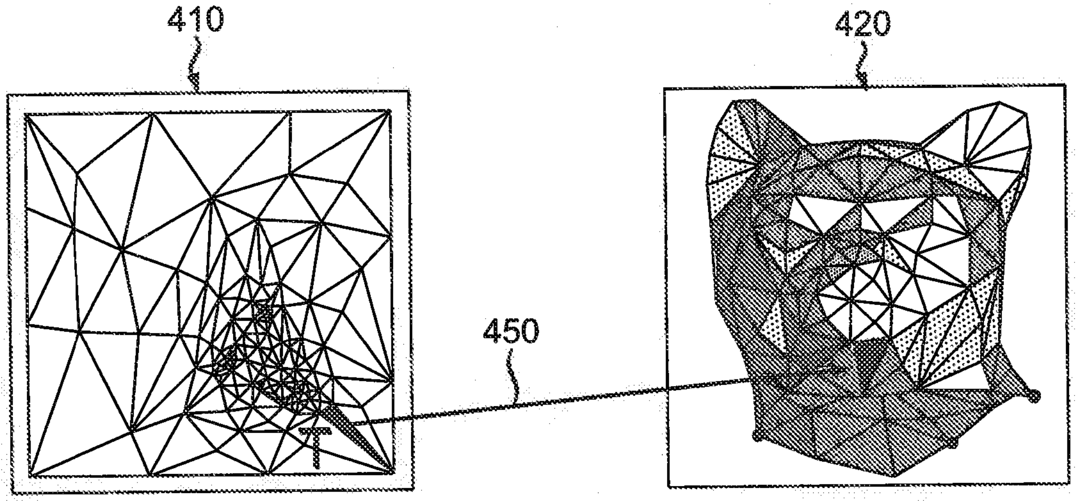

FIG. 4

FIG. 5

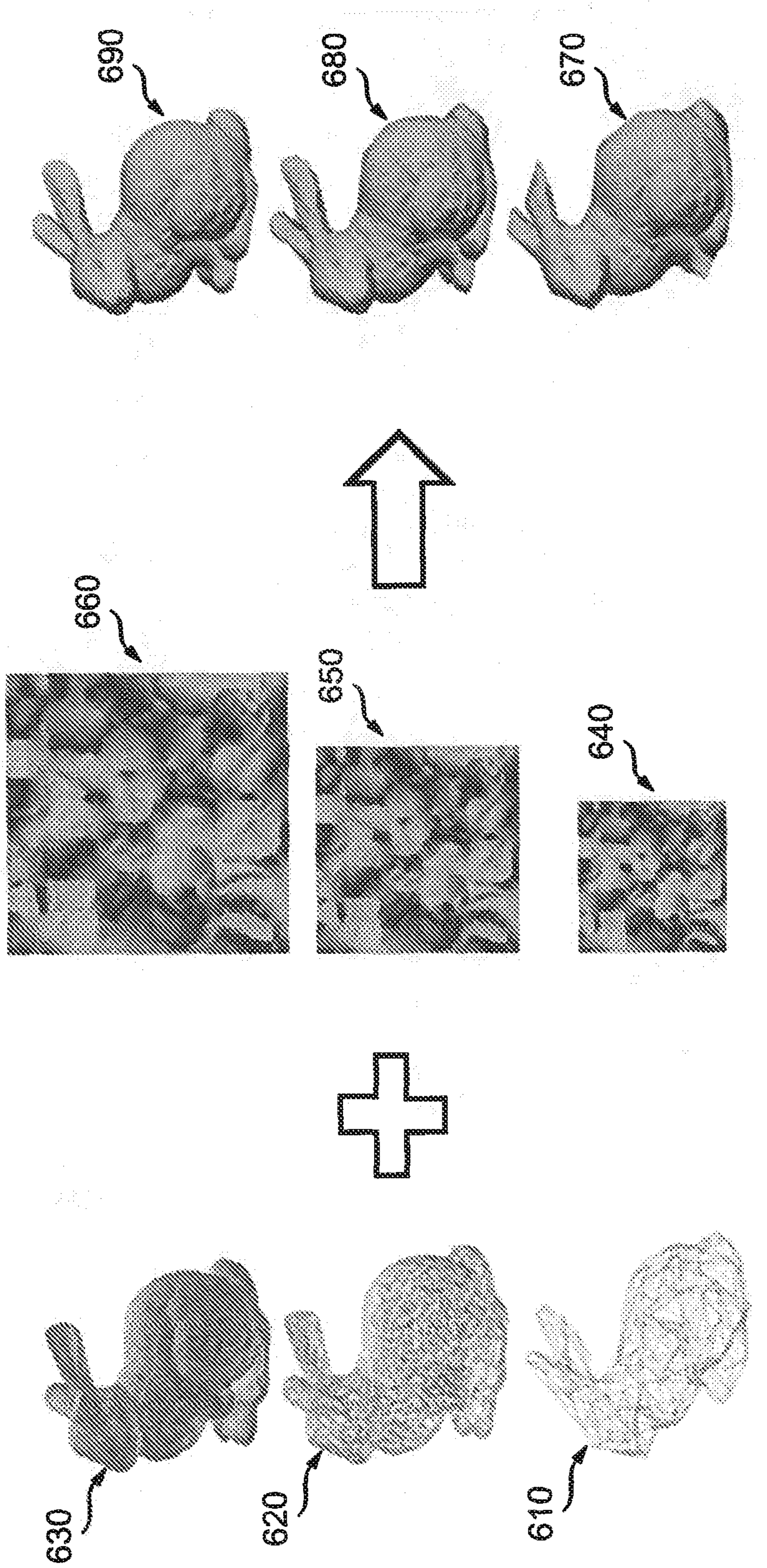

FIG. 6

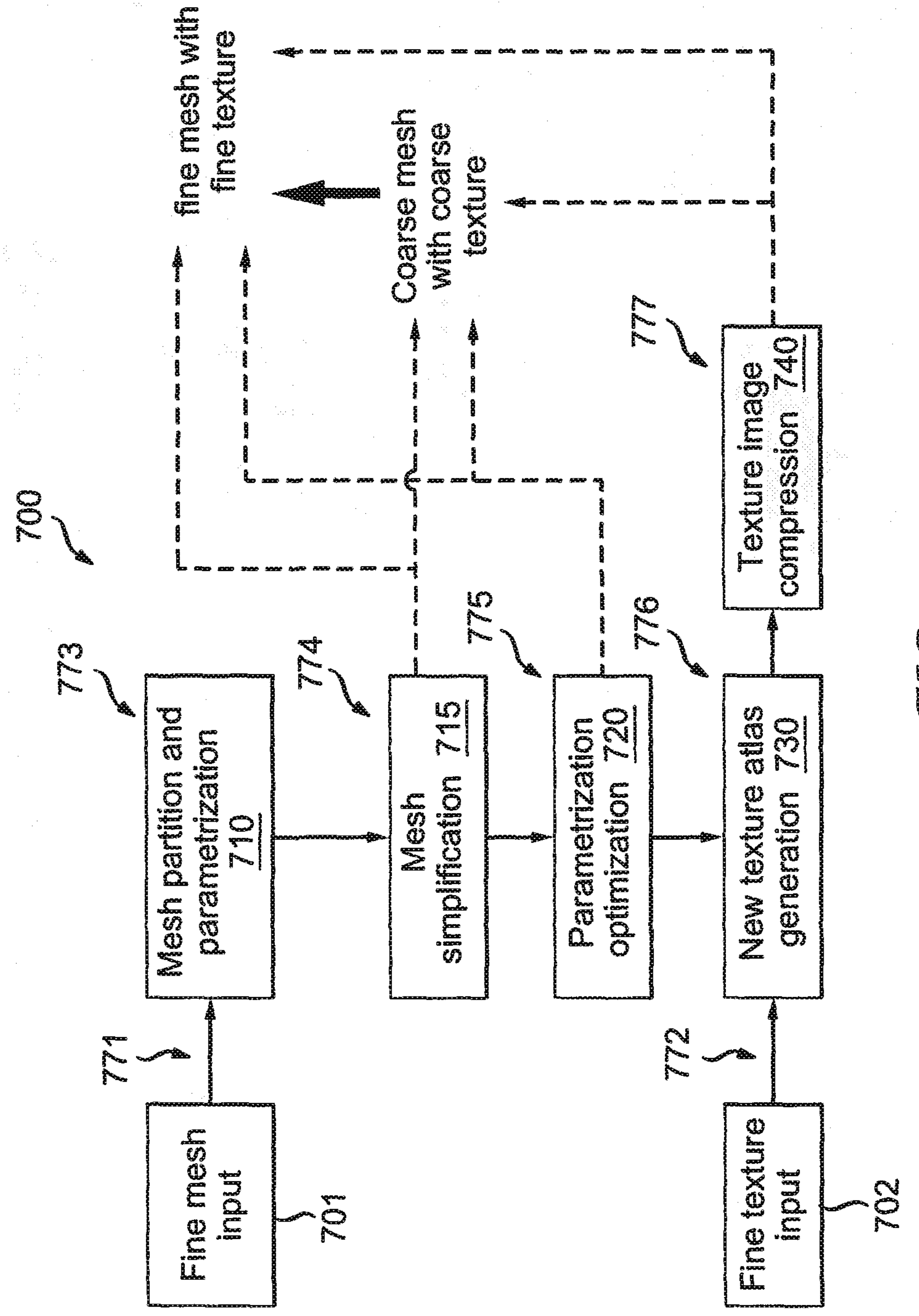
700
Fine mesh input
701
771
Mesh partition and parametrization 710
773
Mesh simplification 715
774
Parametrization optimization 720
775
New texture atlas generation 730
776
Fine texture input
702
772
Texture image compression 740
777
fine mesh with fine texture
Coarse mesh with coarse texture

FIG. 7

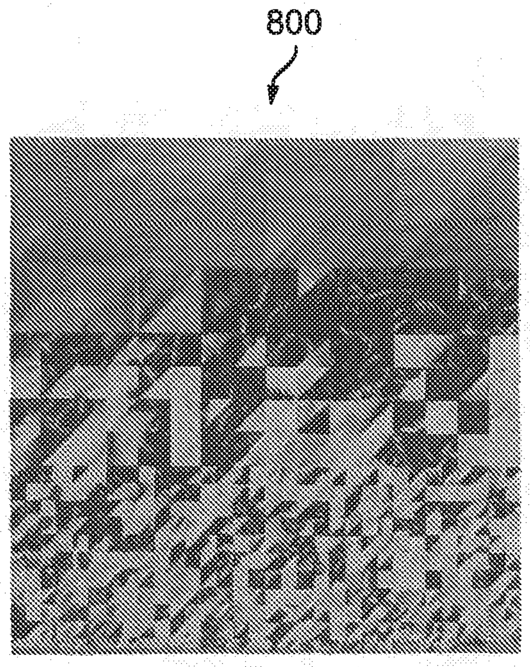

FIG. 8

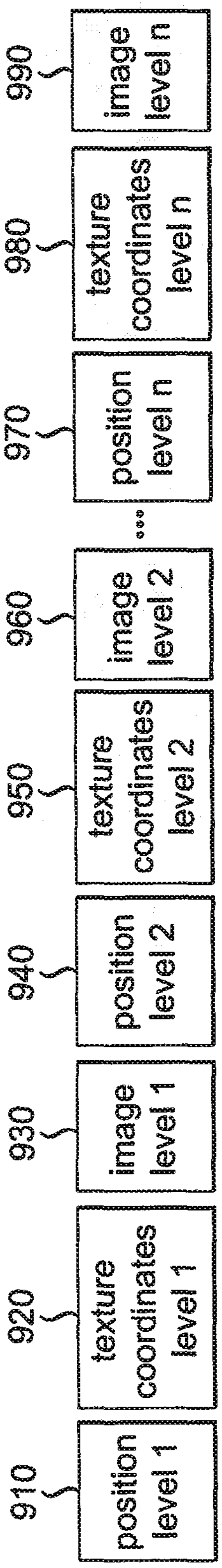
910
position level 1
920
texture coordinates level 1
930
image level 1
940
position level 2
950
texture coordinates level 2
960
image level 2
...
970
position level n
980
texture coordinates level n
990
image level n

FIG. 9

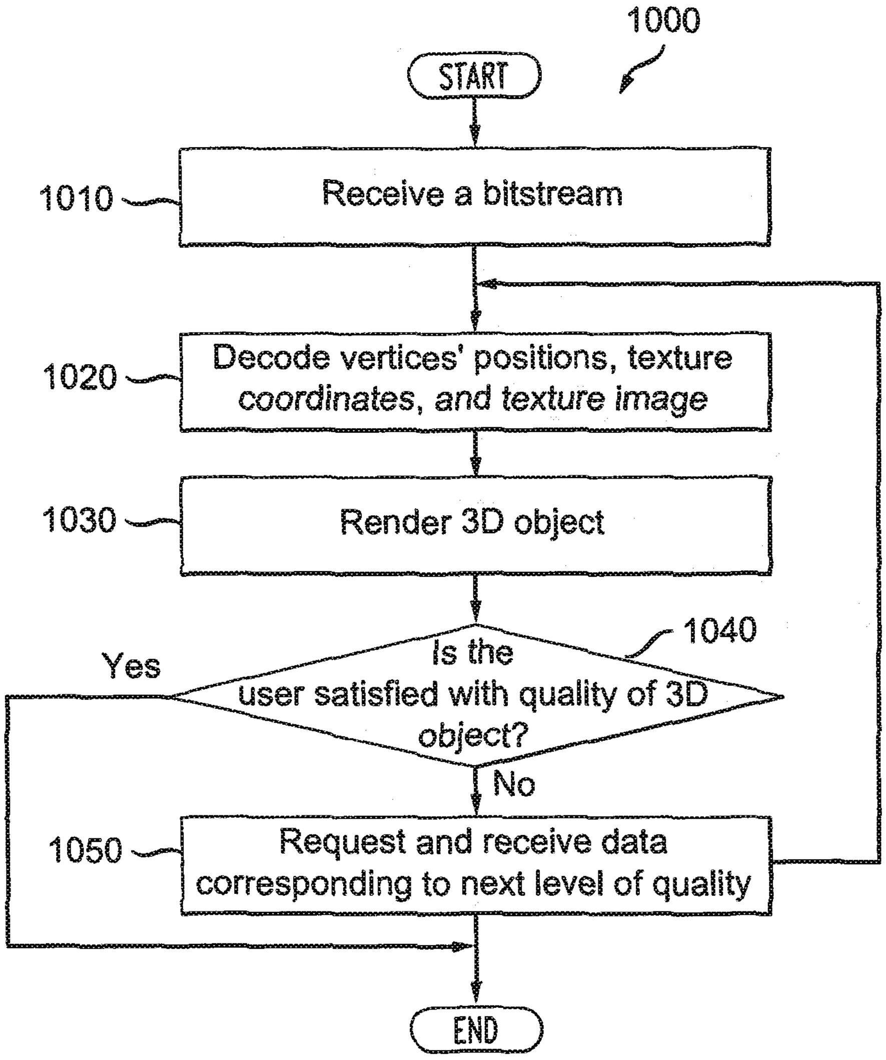
1000
START
1010
Receive a bitstream
1020
Decode vertices' positions, texture coordinates, and texture image
1030
Render 3D object
1040
Is the user satisfied with quality of 3D object?
Yes
No
1050
Request and receive data corresponding to next level of quality
END

FIG. 10

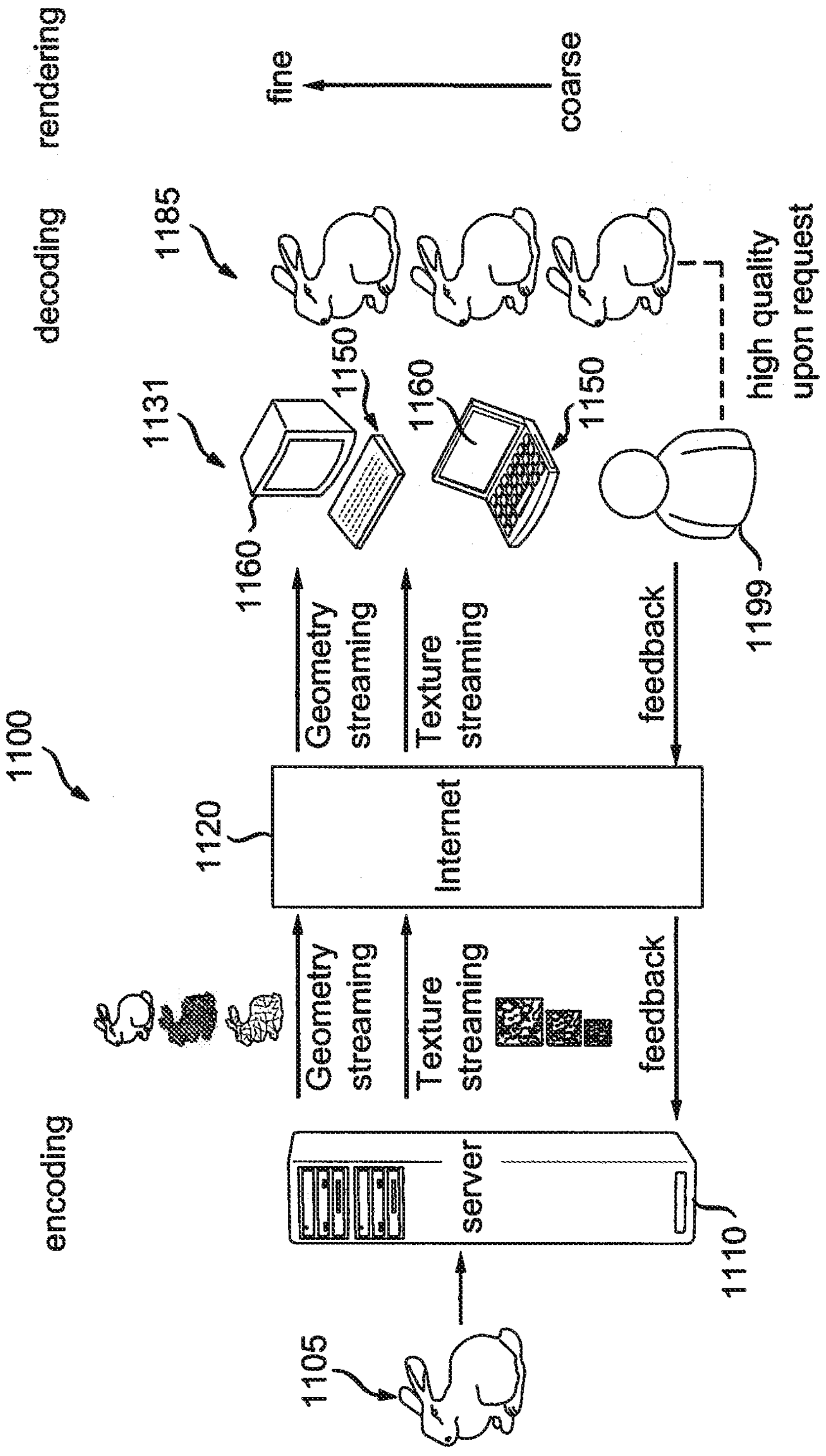
encoding
decoding
rendering
1100
1105
1110
server
Geometry streaming
Texture streaming
feedback
1120
Internet
Geometry streaming
Texture streaming
feedback
1160
1131
1150
1160
1150
1199
1185
high quality upon request
fine
coarse

FIG. 11

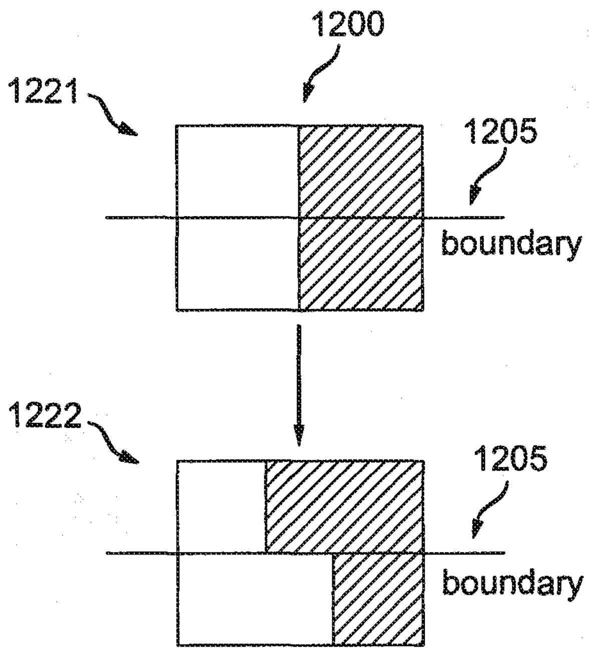

FIG. 12

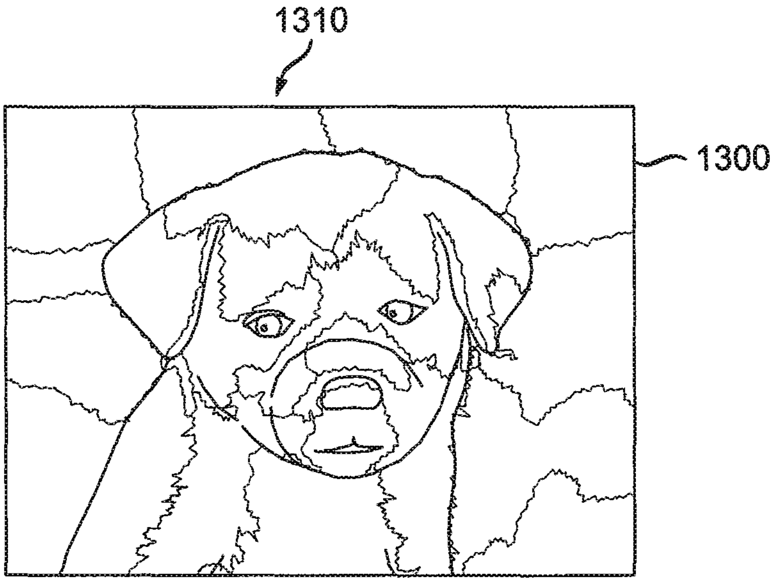

FIG. 13

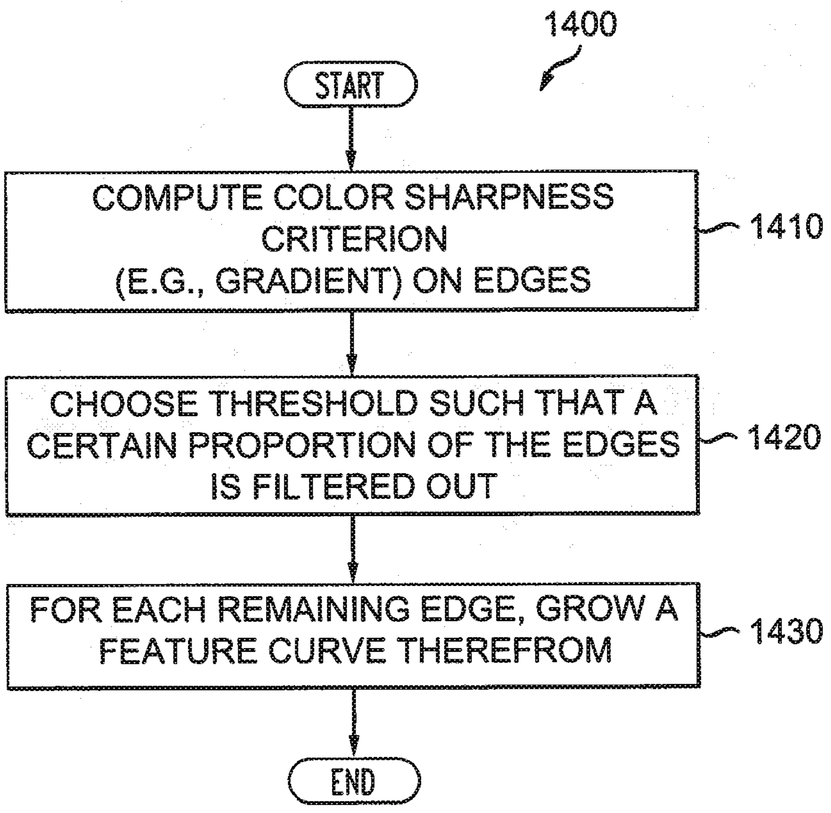
1400
START
COMPUTE COLOR SHARPNESS CRITERION (E.G., GRADIENT) ON EDGES
1410
CHOOSE THRESHOLD SUCH THAT A CERTAIN PROPORTION OF THE EDGES IS FILTERED OUT
1420
FOR EACH REMAINING EDGE, GROW A FEATURE CURVE THEREFROM
1430
END

FIG. 14

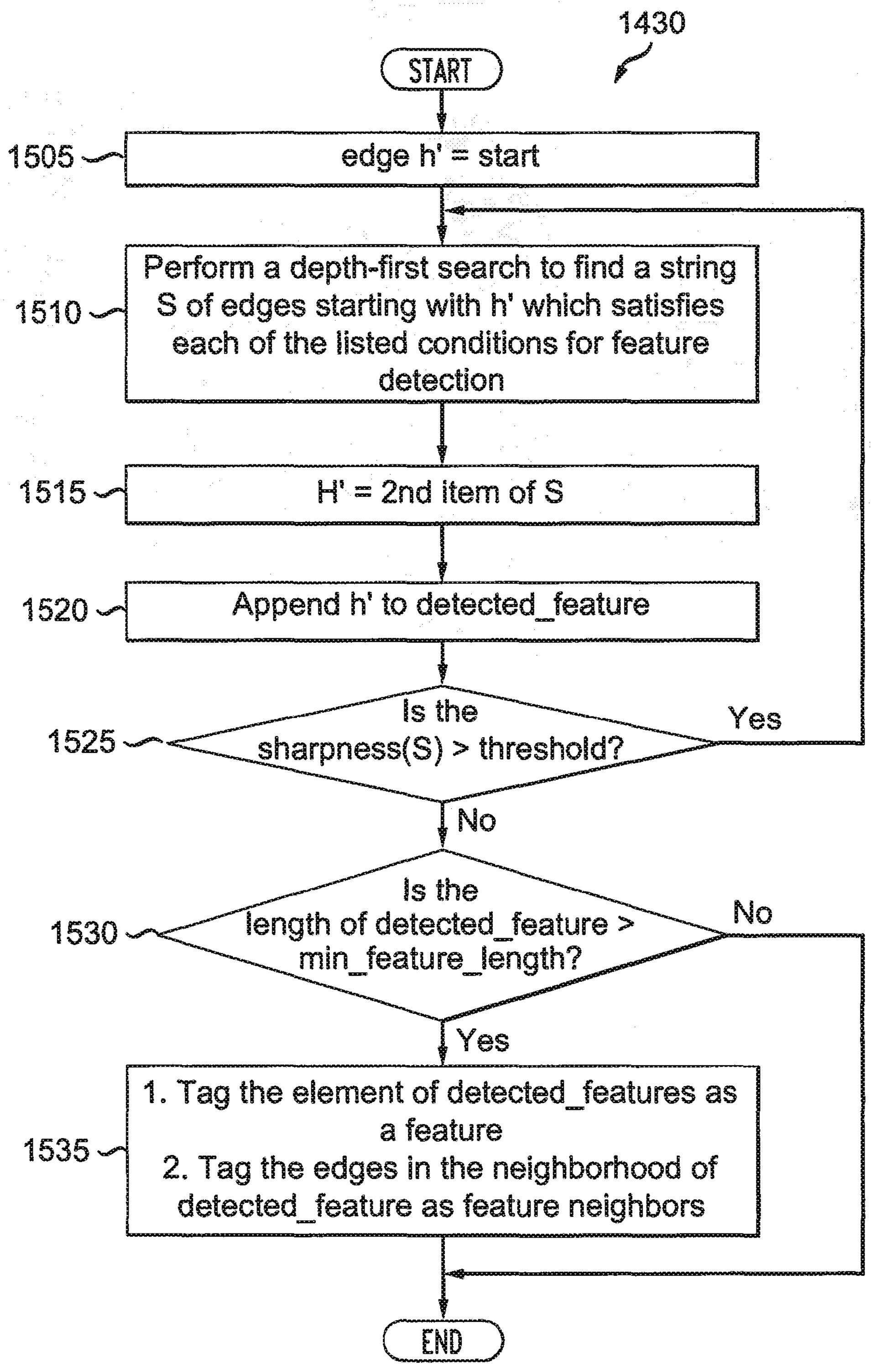
1430
START
1505
edge h' = start
1510
Perform a depth-first search to find a string S of edges starting with h' which satisfies each of the listed conditions for feature detection
1515
H' = 2nd item of S
1520
Append h' to detected_feature
1525
Is the sharpness(S) > threshold?
Yes
No
1530
Is the length of detected_feature > min_feature_length?
No
Yes
1535
1. Tag the element of detected_features as a feature
2. Tag the edges in the neighborhood of detected_feature as feature neighbors
END

FIG. 15

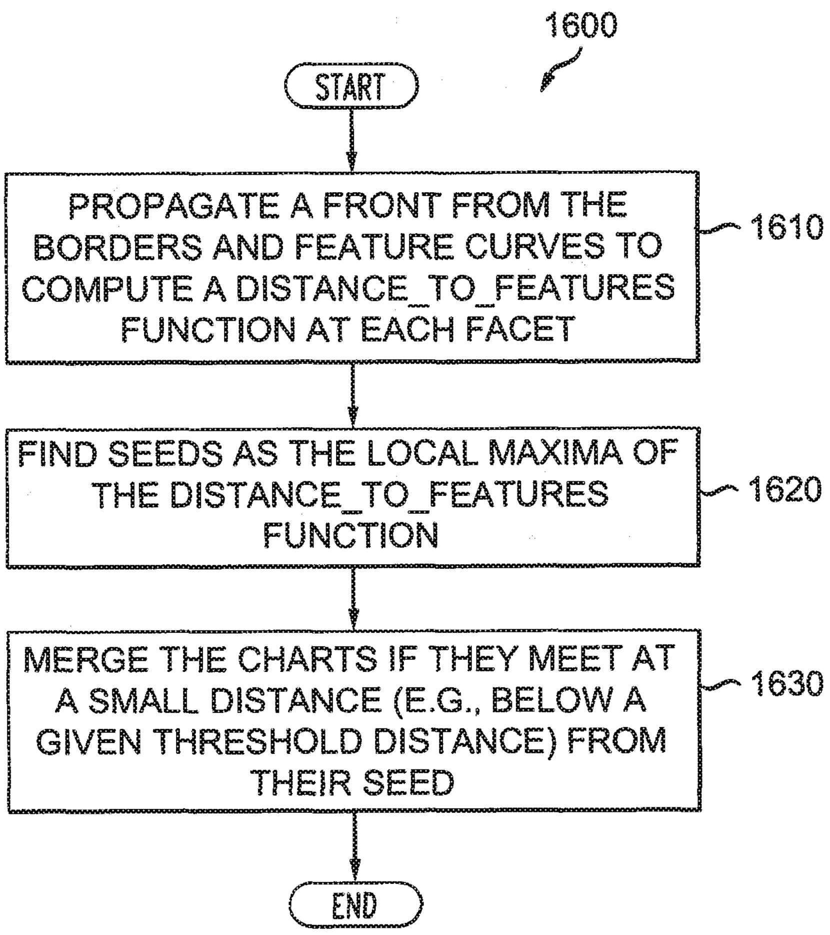
1600
START
PROPAGATE A FRONT FROM THE BORDERS AND FEATURE CURVES TO COMPUTE A DISTANCE_TO_FEATURES FUNCTION AT EACH FACET
1610
FIND SEEDS AS THE LOCAL MAXIMA OF THE DISTANCE_TO_FEATURES FUNCTION
1620
MERGE THE CHARTS IF THEY MEET AT A SMALL DISTANCE (E.G., BELOW A GIVEN THRESHOLD DISTANCE) FROM THEIR SEED
1630
END

FIG. 16

INTERNATIONAL SEARCH REPORT

International application No.
PCT/CN2011/079095

A. CLASSIFICATION OF SUBJECT MATTER

SEE THE EXTRA SHEET

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

IPC:H04N7/-,G06T17-,G06T15-,H04N13/-,G09G5/-

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

CNKI,CNPAT,WPI,EPODOC,CNTXT,VEN,CNABS: simpl+, mesh,3D,three w dimensional, encod+, decod+, coding, layer, basic, base, enhance+,texture,image,refine+,level,stereo

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CN1349716A(KONINK PHILIPS ELECTRONICS N.V.) 15 May 2002(15.05.2002) page 2 lines 15-25 and page 3 lines 1-9 of the description, claims 1-2, the abstract, fig. 1	1,3-6,8,10-12,14-21
Y	CN101364310A(BEIJING LINGTU SOFTWARE CO.,LTD.)11 Feb.2009(11.02.2009) the abstract, claims 1-18	1,3-6,8,10-12,14-21
A	CN101119485A(UNIV. BEIJING AERONAUTICS&ASTRONAUTICS) 06 Feb.2008(06.02.2008) the whole document	1-21
A	US2010277571A1(XU, Bugao et al.)04 Nov.2010(04.11.2010)the whole document	1-21
A	US7274372B1(LAKE, Adam T. et al.) 25 Sep.2007(25.09.2007)the whole document	1-21
A	US6678419B1(MALVAR, Henrique S.) 13 Jan.2004(13.01.2004)the whole document	1-21

☐ Further documents are listed in the continuation of Box C. ☒ See patent family annex.

* Special categories of cited documents:

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier application or patent but published on or after the international filing date

"L" document which may throw doubts on priority claim (S) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

"&"document member of the same patent family

Date of the actual completion of the international search 14 May 2012(14.05.2012)	Date of mailing of the international search report **31 May 2012 (31.05.2012)**
Name and mailing address of the ISA/CN The State Intellectual Property Office, the P.R.China 6 Xitucheng Rd., Jimen Bridge, Haidian District, Beijing, China 100088 Facsimile No. 86-10-62019451	Authorized officer MENG, Jia Telephone No. (86-10)62413419

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.
PCT/CN2011/079095

Patent Documents referred in the Report	Publication Date	Patent Family	Publication Date
CN1349716A	15.05.2002	WO0149036A1	05.07.2001
		EP1161839A1	12.12.2001
		KR20010105361A	28.11.2001
		US2002159518A1	31.10.2002
		US6985526B2	10.01.2006
CN101364310A	11.02.2009	NONE	
CN101119485A	06.02.2008	CN100566414C	02.12.2009
US2010277571A1	04.11.2010	NONE	
US7274372B1	25.09.2007	NONE	
US6678419B1	13.01.2004	WO0059116A1	05.10.2000
		AU3922200A	16.10.2000
		EP1188244A1	20.03.2002
		KR20020008133A	29.01.2002
		US6477280B1	05.11.2002
		JP2002540711U	26.11.2002
		DE60015755E	16.12.2004
		KR100733949B1	02.07.2007
		JP4426118B2	03.03.2010

INTERNATIONAL SEARCH REPORT

International application No.	PCT/CN2011/079095

CLASSIFICATION OF SUBJECT MATTER

H04N7/26(2006.01)i

G06T17/00(2006.01)i