METHOD AND SYSTEM FOR GENERATING A THREE-DIMENSIONAL GRAPHICAL SURFACE FROM THREE-DIMENSIONAL DATA.

This application claims priority from Provisional Application Serial No. 00/123456, filed December 1, 2006.

ABSTRACT

Three-dimensional surfaces representing two-dimensional contours or isosurfaces are extracted from three-dimensional data by comparing each three-dimensional data cube against a pre-established case table that identifies surface boundary locations within each cube, generating geometry, and marching to the next, typically adjacent, cube in the dataset.

FIELD OF THE INVENTION

This invention relates to graphical display on a digital computer system, and in particular to the tessellation and rendering of three-dimensional surfaces – also known as two-dimensional contours or isosurfaces – from three-dimensional data.

DESCRIPTION OF THE RELATED ART

In many applications is it required to generate and visualize three-dimensional data representing any number of real, simulated, or otherwise acquired events. Typically, the data is acquired by means of three-dimensional scanning devices that produce regular, ordered three-dimensional grid datasets, though it need not be so. Within these datasets, each point coordinate (a vertex), typically the position of each grid-defined interior cube (i.e. a voxel), is associated with one or more data values. Each vertex point value signifies a particular level or strength of signal from the scanning device, if said scanning device is used. The key point to the generation of three-dimensional surfaces is the isolation of all similarly valued vertex points, or the interpolation between them to the same extent.

Several means to isolate similar vertex point values within a three-dimensional dataset are known from prior art, including histographic methods that simply count the number of occurrences of each vertex point value. This method returns the relative quantity of each vertex point value present in the dataset, but provides no information about the coordinate of each vertex point value or the relationship between similar vertex point values, or across multiple vertex point values.

Other related work includes the binary identification of each cube as either containing or not containing a desired vertex point value. While it is possible to build each binary cube as geometry, this method produces aliased images with little to no surface detail. This method can be computationally expensive, as each voxel must be treated as geometry, whether visible or not.

Given the inherent difficulties of representing three-dimensional data on two-dimensional media, the need is clear for a simple method to isolate surfaces within that three-dimensional data that share the same data value, analogous to one-dimensional contour lines on a two-dimensional topological map. Also clear is the need to treat three-dimensional surface generation as tessellated geometry, rather than binary voxels, to produce a finer quality surface that fits in the established geometry rendering pipeline. Finally, there is a need for this three-dimensional surface generation to be computationally simple and fast, including operating in real time on current and future digital computers and display devices.

SUMMARY OF THE INVENTION

These and other features of the invention are achieved by comparing three-dimensional data cubes (voxels) against a pre-established lookup case table, to identify the active edges of the expected inclusion of a contour value within the cube, an interpolation means to establish the exact boundary location, and the digital computer process to render the resulting geometry.

These and other features of the invention will be more readily understood upon consideration of the attached drawings and of the following detailed description of those drawings and the presently-preferred and other embodiments of the invention.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features of the invention are set forth with particularity in the appended claims. The invention itself, however, both as to its organization and method of operation, together with further objects and advantages thereof, may best be understood by reference to the following description taken in conjunction with the accompanying drawings in which:

FIG 1. is a flow chart of the three-dimensional surface extraction technique, including optimization for resolving potential case ambiguities.

FIG 2. is a graphical representation of fifteen potential cases in the lookup case table, illustrating where boundary locations may be expected given the provided vertex element values.

FIG 3. is a graphical representation of a three-dimensional dataset, arranged in a regular grid structure, with a single marching cube shaded.

FIG 4. illustrates a single marching cube, suggested vertex point naming conventions, and saddle point S1.

FIG 5. illustrates the process of acquiring boundary location values from any given lookup case, with an example contour value of 1.0.

FIG 6. is a graphical representation of the fifteen potential cases of FIG 2, plus additional lookup subcases to resolve potential ambiguities within the geometry.

DETAILED DESCRIPTION OF THE INVENTION

Referring first to FIG 1, three-dimensional (3D) data 101 may be fixed in any fashion common and acceptable to the practice of data storage. Three-dimensional (3D) data 101 need not be arranged in a regular grid; there are a plurality of current and future methods designed to translate non-regular three-dimensional data into the regular grid described here. After the description of the preferred embodiment below, it will be understood that other means of working with non-regular data are available, including using non-cube and non-regular marching objects, such as tetrahedral – to fulfill the spirit or scope of this invention.

Referring now to FIG 3, the standard means of representing three-dimensional raw data is in regular grid 301, where each junction point of the grid, illustrated as spheres, stores one or many vertex data values and vertex data coordinates that correspond to the relative position within the grid. Typically, this data is produced from a scanning device such as magnetic resonance (MR) or computed tomography (CT) imaging devices, though it need not be so. Other common means of generating three-dimensional data include computer simulations. In the preferred embodiment, the data is arranged in a regular grid structure, represented by one of a plurality of data storage structures.

In FIG 1, three-dimensional (3D) data 101 is import data 102 into the digital computer required for the invention. Import data 102 may comprise of any known art digital storage device, linked or pointed to by the digital computer invention. Preferably, the storage means has a fast access time, and sufficient indexing to allow for fast access of any given data element.

Import data 102 links three-dimensional (3D) data 101 and interprets said three-dimensional (3D) data 101 as structured uniform grid 104. Referring to FIG 3, in accordance with the preferred embodiment of this invention, each cube 302 is constructed of eight rectilinear data points known as vertex points V1-V8, where each vertex points V1-V8 is independently accessible, containing its position relative to three-dimensional (3D) data 101 as a whole, and the data value itself.

It is to be understood that FIG 1 and FIG 3, like all figures referred to, are merely illustrative of the invention and its preferred embodiment, and other structural and naming conventions may be used and still fall within the scope of this invention.

At least one of a plurality of means may be employed to define contour value 103. In the preferred embodiment, at least a default value is available prior to the invention implementation, though this need not be so, and a dynamic or heuristic method may be used to identify contour value 103. Other means of embodiment to create contour value 103 include, but are not limited to, data-centric probabilistic methods or user-directed

acquisition through trial and error. Preferably, contour value 103 falls within the range of values contained within three-dimensional (3D) data 101 vertex points V1-V8 values.

In the preferred embodiment, the invention operates on each cube 302 independently, and at conclusion of computation per cube 302, marches 107 to the next available, typically adjacent, cube. In our description, this process is completed per cube for every cube available within three-dimensional (3D) data 101, though many different implementation are available to optimize the quantity of cubes operated on, including isolating only cubes containing contour value 103 via a tree data structure, branch-on-need (BON) octree data structure, parallel processing, or any other optimization process of known or derived art. For the sake of completeness for the discussion of this particular embodiment, for each 8 point cube 105 signifies the completion of this process through every cube 302 available.

For each point 106 operates on each of the preferred eight vertex points V1-V8, where each vertex points V1-V8 value is compared to contour value 103, and returns a greater-than or less-than signifier, and in the preferred embodiment rare equal-to results being assigned consistently to either greater-than or less-than. In the preferred embodiment, each vertex comparison can return a binary bit, with the eight vertex points that comprise each square consisting of a byte. For example (referring to FIG 4) if vertex points V1, V3, V5, and V7 are found to contain values equal-to or greater-than contour value 103, and vertex points V2, V4, V6, and V8 contain values less-than contour value 103, one possible returned byte might be 01010101. In the preferred embodiment, this byte will act as the value to use with lookup case table FIG 2 and lookup subcase table FIG 6. Other possible comparison function and signifiers, as well as encoding processes are available and practical, and typically return similar results, the only requirement being that the methods are used consistently, and all are assumed to fall within the spirit or scope of this invention.

Two possible cube cases, encoded using the above example as 11111111 and 00000000, contain vertex point values that are either all greater-than or all less-than contour value 103, and this is taken to mean that there is no expected inclusion of geometry 114. The cubes may be quickly marched 107 over according to the employed means to move to the next cube 302. For the anticipated purpose of the invention, these cube cases are not relevant, but may be used for other processing or indexing functions that contribute to the spirit of the invention.

It is observable that the above preferred encoding method has a potential 2^8 = 256 cases to check against in a lookup case table. In the preferred embodiment, many of these cases can be condensed using arguments of symmetry and rotation, and can be depicted as a graphical representation of this condensed lookup case table. In lookup case table FIG 2, each cube 302 is illustratively depicted with vertex point V1-V8 value greater-than contour value 103 as a solid sphere, and less-than vertex point V1-V8 value as a completely transparent sphere. From lookup case table FIG 2, expected inclusions of geometry 114 signifying the transversal of contour value 103 are graphically represented. The method to encode active edge 112 of each lookup case table FIG 2, which signifies where geometry 114 is expected to pass from current cube 302 to the next depends on

implemented graphics geometry renderer 115 employed, relevant to the implemented digital computer, and are within the spirit and scope of the invention. The important information is the identification of active edges 112, which will be further located with precision by one of a plurality of interpolation methods to identify interpolated edge boundary location 113, described below.

Using a plurality of active edges 112 of each cube 302, the exact interpolated edge boundary location 113 may be found using one of a plurality of interpolation methods. In the preferred embodiment, a linear interpolation along active edge 112, that is, between the two vertex point V1-V8 values of active edge 112, will identify interpolated edge boundary location 113. Other interpolation methods are meant to include a plurality of methods, including nearest neighbor, statistical variance, and other methods common to the art.

The interpolation method to identify interpolated edge boundary location 113 is abstracted as an example in FIG 5. Using the vertex point naming convention used for example in FIG 4, vertex point V1-V8 values have been assigned. For example, vertex point V1 has been assigned vertex point V1-V8 value 0.1, vertex point V2 has been assigned vertex point V1-V8 value 0.8, and so on. Additionally, contour value 103 has been assigned value 1.0. In the full implementation of this invention, these values would be determined using a plurality of methods illustrated within the invention description. Using the above described preferred encoding convention, this cube would be encoded using the byte 10110000, which in a full implementation would point 109 to lookup case table FIG 2 case 206. Using said case, active edges 112 are quickly identified as edges (V1, V5), (V2, V6), (V4, V8), (V5, V7), (V7, V8). In the preferred embodiment, linear interpolation is preformed along this edge to locate the exact position of interpolated edge boundary location 113 of contour value 103. Keeping in mind that each vertex point V1-V8 signifies a location on structured uniform grid 104 as well as a vertex point V1-V8 value, each interpolated edge boundary location 113 are fully identified.

Using at least one of a plurality of means, interpolated edge boundary location 113 are structured as geometry 114. In the preferred embodiment, this is taken to be a triangle mesh, with each interpolated boundary location used as a vertex point in said triangle mesh. Other geometry tessellation means that are known to the art can be employed, are fall within the spirit and scope of this invention.

Geometry 114 is passed to implemented graphics geometry renderer 115. A plurality of methods exist to render geometry.

Observation of lookup case table FIG 2 will illustrate that with certain cases, ambiguities exist where the interpolated edge boundary location 113 can be connected as geometry 114 in a plurality of relationships. This would produce undesirable holes in geometry 114. Using lookup subcase table FIG 6 resolves ambiguities. A graphical representation of lookup subcase table FIG 6 illustrates how a plurality of lookup case tables FIG 2 and lookup subcase table FIG 6 are possible, and are taken to fall within the spirit or scope of this invention.

The value of saddle point S1 is identified by one of a plurality of interpolation methods. In the preferred embodiment, trilinear interpolation is used. The value of saddle point S1 is compared against contour value 103, and lookup subcase table FIG 6 case is identified using the process described above for lookup case table FIG 2, or by the implemented means taken to fall within the spirit or scope of this invention.

It is to be understood that the above described embodiments are merely illustrative of numerous and varied other embodiments which may constitute applications of the principles of the invention. Such other embodiments may be readily devised by those skilled in the art without departing from the spirit or scope of this invention and it is our intent they be deemed within the scope of our invention.

CLAIMS

What is claimed is:

1. A method of operating on a digital computer, comprising:

   storing three-dimensional data;

   organizing said three-dimensional data into regular three-dimensional grid;

   processing a plurality of marching cubes, each formed by a plurality of regular division of said regular three-dimensional grid;

   processing a plurality of vertex elements associated with each said marching cube;

   comparing said vertex elements to establish value comparison between said vertex element and defined contour value; and

   comparing said marching cube to establish inclusion of said defined contour value.

2. A digital computer programmed to perform the method of claim 1.

3. A computer-readable medium storing a computer program implementing the method of claim 1.

4. A signal transmitting a computer program implementing the method of claim 1.

5. The method of claim 1 further comprising:

   establishing said defined contour value by a plurality of processes.

6. A digital computer programmed to perform the method of claim 5.

7. A computer-readable medium storing a computer program implementing the method of claim 5.

8. A signal transmitting a computer program implementing the method of claim 1.

9. The method of claim 1 further comprising:

listing a plurality of said marching cubes with said established inclusion of said defined contour value; and

ranking said list and said marching cube with at least one said vertex element of said marching cube being below said defined contour value, and at least one said vertex element of said marching cube being above said defined contour value.

10. A digital computer programmed to perform the method of claim 9.

11. A computer-readable medium storing a computer program implementing the method of claim 9.

12. A signal transmitting a computer program implementing the method of claim 9.

13. The method of claim 1 further comprising:

identifying said marching cube with said established inclusion against pre-established lookup case table, and returning said identification;

comparing said identification of said marching cube with said lookup case table to identify active edges where said established inclusion is expected to pass boundary location to adjacent said marching cubes, or beyond said three-dimensional data, or beyond said three-dimensional grid, or beyond viewing frame;

interpolating using one of a plurality of interpolation means for interpolating across said active edge to locate said boundary location of said established inclusion;

graphically associating each said boundary location as three-dimensional geometry vertex elements; and

tessellating with one or several of a plurality of tessellation means for associating a plurality of said three-dimensional geometry vertex elements together as three-dimensional geometry.

14. A digital computer programmed to perform the method of claim 13.

15. A computer-readable medium storing a computer program implementing the method of claim 13.

16. A signal transmitting a computer program implementing the method of claim 13.

17. The method of claim 1 further comprising:

rendering visually or otherwise rendering said three-dimensional geometry to a plurality of standard outputs.

18. A digital computer programmed to perform the method of claim 17.

19. A computer-readable medium storing a computer program implementing the method of claim 17.

20. A signal transmitting a computer program implementing the method of claim 17.

21. The method of claim 13, further comprising:

identifying subcase identifying said marching cube with said established inclusion against ambiguity-less lookup subcase table, and returning said subcase identification;

comparing said subcase identification of said marching cube with said ambiguity-less lookup subcase table to identify said active edges where said established inclusion is expected to boundary location to adjacent said marching cubes, or beyond said three-dimensional data, or beyond said three-dimensional grid, or beyond said viewing frame; and

interpolating using one of a plurality of interpolation means for locating saddle point to resolve ambiguity of said subcase identification of said marching cube.

22. A digital computer programmed to perform the method of claim 21.

23. A computer-readable medium storing a computer program implementing the method of claim 21.

24. A signal transmitting a computer program implementing the method of claim 21.

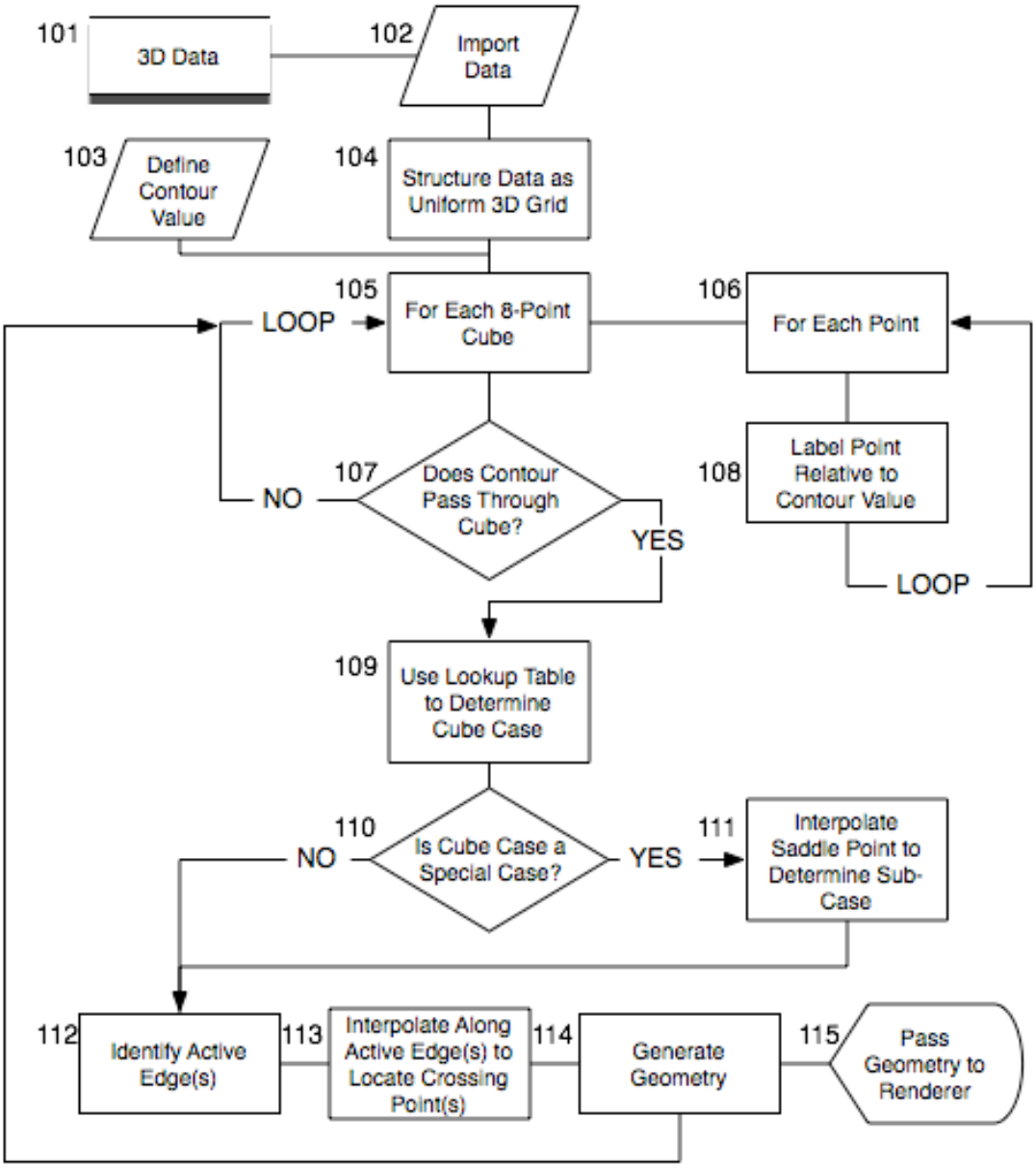
101
3D Data
102
Import Data
103
Define Contour Value
104
Structure Data as Uniform 3D Grid
105
LOOP
For Each 8-Point Cube
106
For Each Point
107
NO
Does Contour Pass Through Cube?
YES
108
Label Point Relative to Contour Value
LOOP
109
Use Lookup Table to Determine Cube Case
110
NO
Is Cube Case a Special Case?
YES
111
Interpolate Saddle Point to Determine Sub-Case
112
Identify Active Edge(s)
113
Interpolate Along Active Edge(s) to Locate Crossing Point(s)
114
Generate Geometry
115
Pass Geometry to Renderer

FIG 1

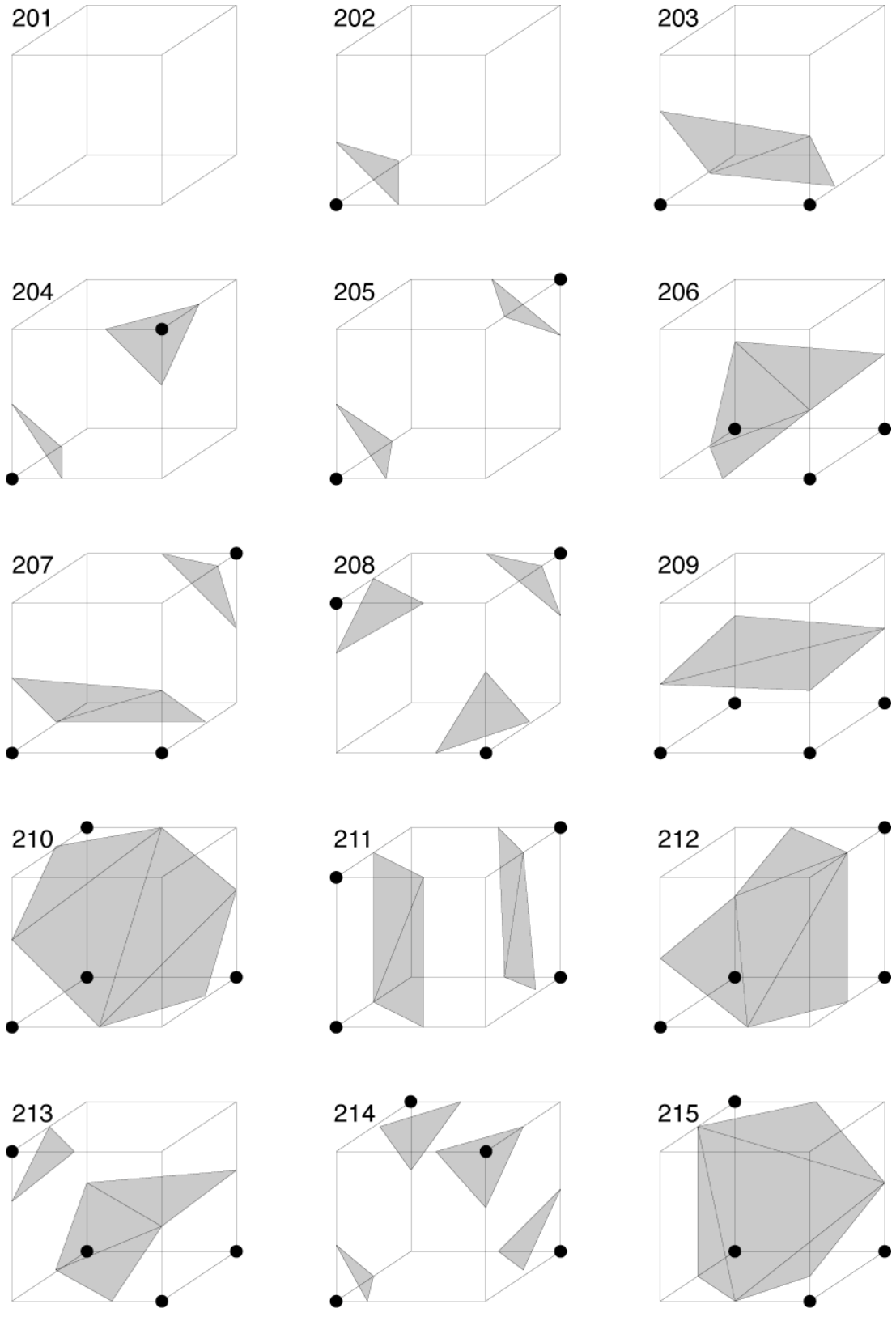

FIG 2

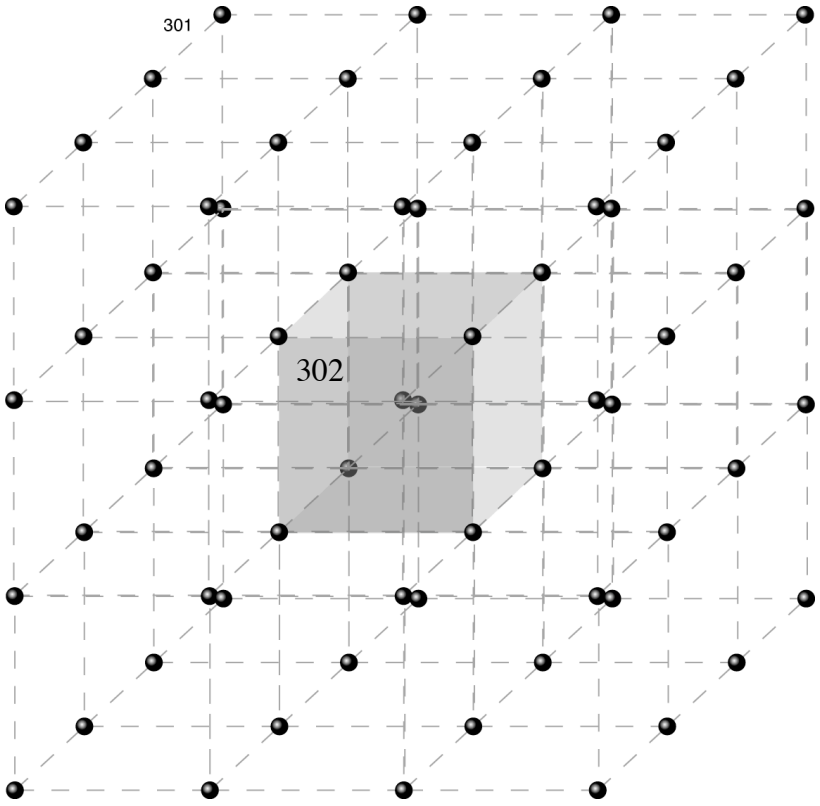

FIG 3

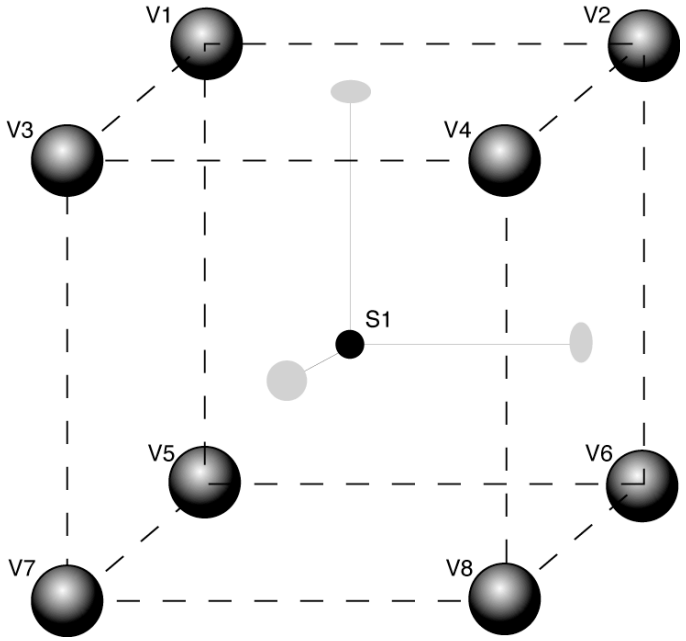

FIG 4

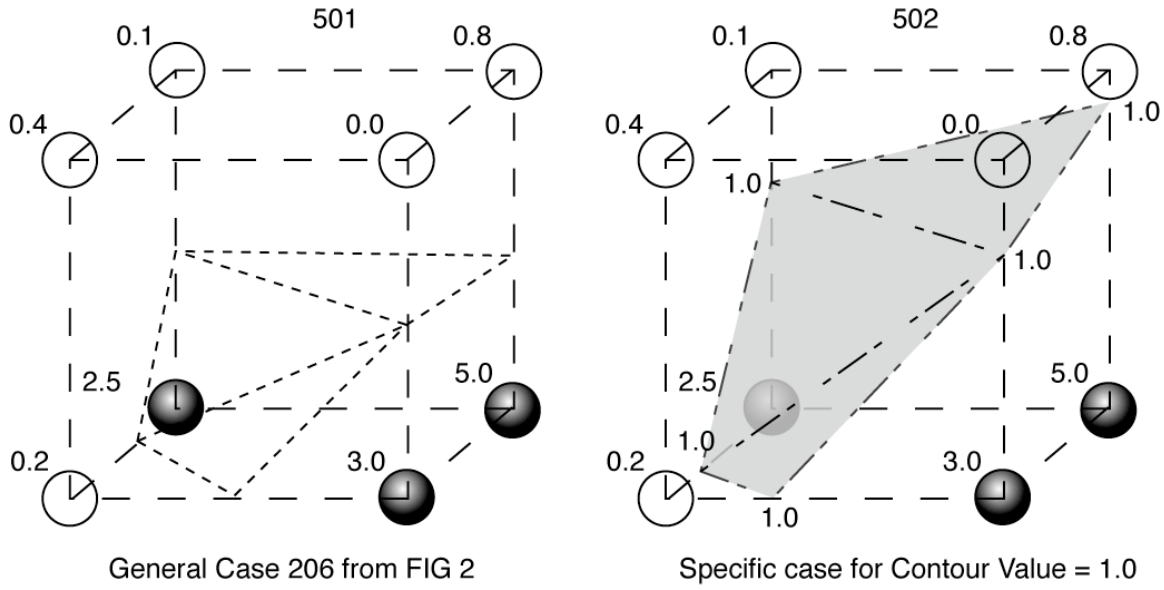

FIG 5

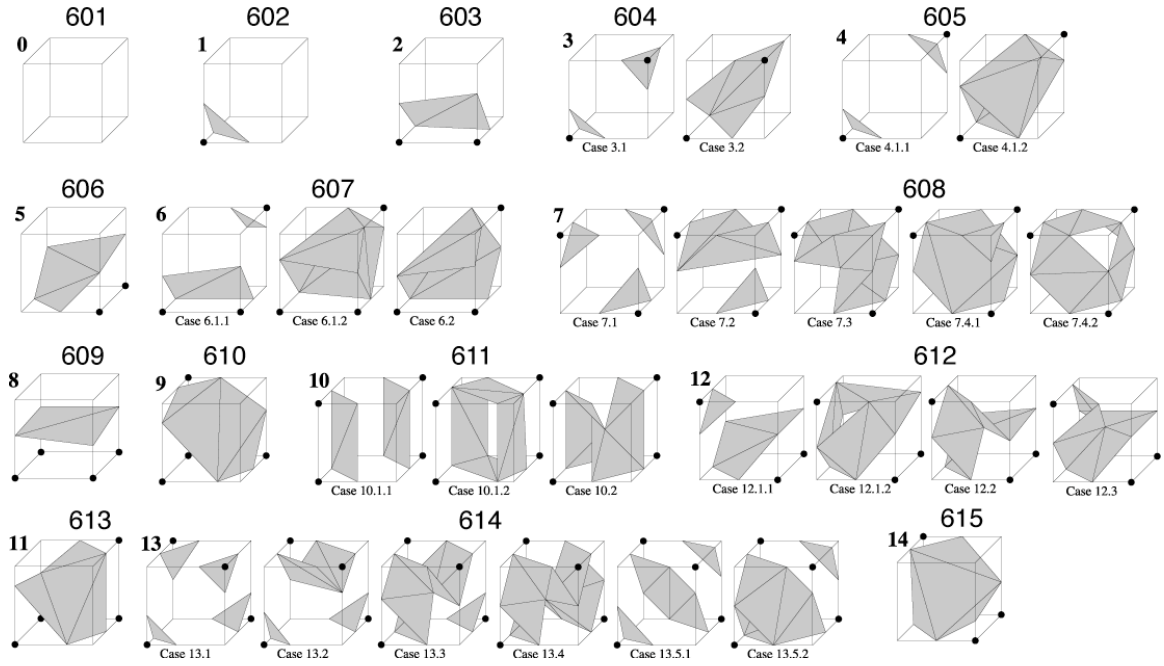

FIG 6

NOTE: FIG 2 and FIG 6 are borrowed from Evgeni V. Chernyaev's paper "Marching Cubes 33: Construction of Topologically Correct Isosurfaces," an improved implementation of the marching cube algorithm described in this patent exercise.