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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
9 March 2017 (09.03.2017)

WIPO | PCT

(10) International Publication Number
WO 2017/040006 A1

(51) International Patent Classification:
G06T 17/10 (2006.01) *G06T 17/20* (2006.01)
G06F 17/50 (2006.01)

(21) International Application Number:
PCT/US2016/046960

(22) International Filing Date:
15 August 2016 (15.08.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
14/842,039 1 September 2015 (01.09.2015) US

(71) Applicant: **SIEMENS PRODUCT LIFECYCLE MANAGEMENT SOFTWARE INC.** [US/US]; 5800 Granite Parkway, Suite 600, Plano, Texas 75024-6612 (US).

(72) Inventors: **MAKEM, Jonathan**; 2 Regency Square, Cambridge Cambridgeshire CB13WL (GB). **MUKHERJEE, Nilanjan**; 7812 Meadowcreek Drive, Cincinnati, Ohio 45244 (US).

(74) Agent: **PARMELEE, Christopher L.**; Siemens Corporation-Intellectual Property Dept., 3501 Quadrangle Blvd. Ste. 230, Orlando, Florida 32817 (US).

(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, BN, 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, IR, IS, JP, KE, KG, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, 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.

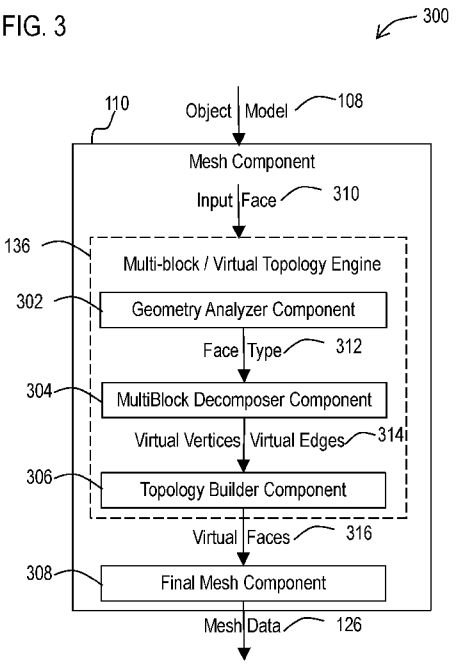
(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, 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, KM, ML, MR, NE, SN, TD, TG).

Published:

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

(54) **Title**: MESH GENERATION SYSTEM AND METHOD

FIG. 3



(57) **Abstract**: A system (100) and method is provided that facilitates generating meshes for object models of structures for use with finite element analysis simulations carried out on the structure. The system may include at least one processor (102) configured to classify a type (312, 508) of an input face (204, 310, 502) of a three dimensional (3D) object model (108) of a structure based at least in part on a number of loops (504, 506) included by the input face. The processor may also select based on the classified type of the input face a multi-block decomposition algorithm (1006, 1008, 1010, 1012, 1014) from among a plurality of multi-block decomposition algorithms that the processor is configured to use. Further the processor may use the selected multi-block decomposition algorithm to determine locations of a plurality of blocks (1506) across the input face. In addition the processor may mesh each block to produce mesh data (126) defining a mesh (202) that divides the input face into a plurality of quadrilateral elements (210).

WO 2017/040006 A1

Mesh Generation System and Method

TECHNICAL FIELD

[0001] The present disclosure is directed, in general, to computer-aided engineering (CAE), computer-aided design (CAD), visualization, and manufacturing systems, product data management (PDM) systems, product lifecycle management (PLM) systems, and similar systems, that are used to create, use, and manage data for products and other items (collectively referred to herein as product systems).

BACKGROUND

[0002] PLM systems may include components that facilitate the design and simulated testing of product structures. Such components may benefit from improvements.

SUMMARY

[0003] Variously disclosed embodiments include systems and methods that may be used to facilitate generating meshes for object models of structures for use with finite element analysis simulations carried out on the structure via computer-aided engineering (CAE) software. In one example, a system may comprise at least one processor. The at least one processor may be configured to classify a type of an input face of a three dimensional (3D) object model of a structure based at least in part on a number of loops included by the input face. The at least one processor may also be configured to select, based on the classified type of the input face, a multi-block decomposition algorithm from among a plurality of multi-block decomposition algorithms that the processor is configured to use. Further, the at least one processor may be configured to use the selected multi-block decomposition algorithm to determine locations of a plurality of blocks (e.g., such as via split lines) across the input face. In addition the at least one processor may be configured to mesh each block to produce data representative of a mesh that divides the input face into a plurality of quadrilateral elements.

[0004] In another example, a method may include various acts carried out through operation of at least one processor. Such a method may include classifying a type of an input face of a three dimensional (3D) object model of a structure, based at least in part on a number of loops included by the input face; selecting based on the classified type of the input face, a multi-block decomposition algorithm from among a plurality of multi-block decomposition algorithms that

the processor is configured to use; using the selected multi-block decomposition algorithm to determine locations of a plurality of blocks (e.g., such as via split lines) across the input face; and meshing each block to produce data representative of a mesh that divides the input face into a plurality of quadrilateral elements.

[0005] A further example may include non-transitory computer readable medium encoded with executable instructions (such as a software component on a storage device) that when executed, causes at least one processor to carry out this described method.

[0006] Another example may include an apparatus including at least one hardware, software, and/or firmware based processor, computer, component, controller, means, module, and/or unit configured for carrying out functionality corresponding to this described method.

[0007] The foregoing has outlined rather broadly the technical features of the present disclosure so that those skilled in the art may better understand the detailed description that follows. Additional features and advantages of the disclosure will be described hereinafter that form the subject of the claims. Those skilled in the art will appreciate that they may readily use the conception and the specific embodiments disclosed as a basis for modifying or designing other structures for carrying out the same purposes of the present disclosure. Those skilled in the art will also realize that such equivalent constructions do not depart from the spirit and scope of the disclosure in its broadest form.

[0008] Before undertaking the Detailed Description below, it may be advantageous to set forth definitions of certain words or phrases that may be used throughout this patent document. For example, the terms "include" and "comprise," as well as derivatives thereof, mean inclusion without limitation. The singular forms "a", "an" and "the" are intended to include the plural forms as well, unless the context clearly indicates otherwise. Further, the term "and/or" as used herein refers to and encompasses any and all possible combinations of one or more of the associated listed items. The term "or" is inclusive, meaning and/or, unless the context clearly indicates otherwise. The phrases "associated with" and "associated therewith," as well as derivatives thereof, may mean to include, be included within, interconnect with, contain, be contained within, connect to or with, couple to or with, be communicable with, cooperate with, interleave, juxtapose, be proximate to, be bound to or with, have, have a property of, or the like.

[0009] Also, although the terms "first", "second", "third" and so forth may be used herein to describe various elements, functions, or acts, these elements, functions, or acts should not be limited by these terms. Rather these numeral adjectives are used to distinguish different elements, functions or acts from each other. For example, a first element, function, or act could be termed a second element, function, or act, and, similarly, a second element, function, or act could be termed a first element, function, or act, without departing from the scope of the present disclosure.

[0010] In addition, phrases such as "processor is configured to" carry out one or more functions or processes, may mean the processor is operatively configured to or operably configured to carry out the functions or processes via software, firmware, and/or wired circuits. For example, a processor that is configured to carry out a function/process may correspond to a processor that is actively executing the software/firmware which is programmed to cause the processor to carry out the function/process and/or may correspond to a processor that has the software/firmware in a memory or storage device that is available to be executed by the processor to carry out the function/process. It should also be noted that a processor that is "configured to" carry out one or more functions or processes, may also correspond to a processor circuit particularly fabricated or "wired" to carry out the functions or processes (e.g., an ASIC or FPGA design). Further the phrase "at least one" before an element (e.g., a processor) that is configured to carry out more than one function may correspond to one or more elements (e.g., processors) that each carry out the functions and may also correspond to two or more of the elements (e.g., processors) that respectively carry out different ones of the one or more different functions.

[0011] The term "adjacent to" may mean: that an element is relatively near to but not in contact with a further element; or that the element is in contact with the further portion, unless the context clearly indicates otherwise.

[0012] Definitions for certain words and phrases are provided throughout this patent document, and those of ordinary skill in the art will understand that such definitions apply in many, if not most, instances to prior as well as future uses of such defined words and phrases. While some terms may include a wide variety of embodiments, the appended claims may expressly limit these terms to specific embodiments.

BRIEF DESCRIPTION OF THE DRAWINGS

[0013] Fig. 1 illustrates a functional block diagram of an example system that facilitates generating meshes for object models.

[0014] Fig. 2 illustrates an example of a well-structured quadrilateral mesh.

[0015] Fig. 3 illustrates an example framework for a mesh component including a multi-block virtual topology engine.

[0016] Fig. 4 illustrates an example flow diagram 400 of an example for a geometry analyzer component.

[0017] Fig. 5 illustrates examples of input surfaces and corresponding single loop and multi loop classifications.

[0018] Fig. 6 illustrates an example of features for classifying a cylindrical type input face.

[0019] Fig. 7 illustrates an example medial axis determined for an input surface.

[0020] Fig. 8 illustrates medial edges and medial vertices of a medial axis.

[0021] Fig. 9 illustrates example topologies of beads and annuli.

[0022] Fig. 10 illustrates a flow diagram for a multi-block decomposer component.

[0023] Fig. 11 illustrates examples of virtual edge lines that define blocks that subdivide an input face.

[0024] Fig. 12 illustrates an example of concavity at a vertex that is removed via a medial axis decomposition algorithm.

[0025] Fig. 13 illustrates an example of Delaunay edges extending from each concavity that may be evaluated via a medial axis decomposition algorithm.

[0026] Fig. 14 illustrates an example placement of split lines defined by the touching points of an inscribed circle at positions on each medial edge where the diameter of the circle is either maximum or minimum.

[0027] Fig. 15 illustrates an example placement of split lines added for a multi-loop annulus type input face.

[0028] Fig. 16 illustrates an example placement of split lines added for a multi-loop annulus type input face.

[0029] Fig. 17 illustrates an example of flattening and aligning a cylindrical input surface with the orthogonal axes of the global Cartesian (X, Y) coordinate system.

[0030] Figs. 18 and 19 illustrate an example of extending orthogonals to partition an input face via an Art Gallery multi-blocking algorithm.

[0031] Fig. 20 illustrates a notch diagram which depicts relationships between reflex vertices and their pairs for an input face.

[0032] Fig. 21 illustrates a final mesh in 3D space on an Art-Gallery decomposed single loop input face with 9 reflex vertices.

[0033] Fig. 22 illustrates an example of a split line added to an input face via the visibility based decomposition algorithm.

[0034] Fig. 23 illustrates an example in which an input face has been split into quad-regions by the visibility based decomposition algorithm.

[0035] Fig. 24 illustrates an example of the mesh generated on the input face in Fig. 23 after blocks are produced by the visibility based decomposition algorithm.

[0036] Fig. 25 illustrates an example where a topology builder component generates a virtual topology based on block location data for an input surface.

[0037] Fig. 26 illustrates a flow diagram of an example methodology that facilitates generating meshes for object models.

[0038] Fig. 27 illustrates a block diagram of a data processing system in which an embodiment can be implemented.

DETAILED DESCRIPTION

[0039] Various technologies that pertain to systems and methods for generating meshes for object models will now be described with reference to the drawings, where like reference numerals represent like elements throughout. The drawings discussed below, and the various embodiments used to describe the principles of the present disclosure in this patent document are by way of illustration only and should not be construed in any way to limit the scope of the disclosure. Those skilled in the art will understand that the principles of the present disclosure may be implemented in any suitably arranged apparatus. It is to be understood that functionality that is described as being carried out by certain system elements may be performed by multiple elements. Similarly, for instance, an element may be configured to perform functionality that is described as being carried out by multiple elements. The numerous innovative teachings of the present application will be described with reference to exemplary non-limiting embodiments.

[0040] Many forms of product systems (such as CAE and CAD software) are operative to manipulate and process various types of three dimensional (3D) object models (such as stored in CAD files) that represent a structure for an object (e.g., parts, assemblies and subassemblies). In particular CAE software may be used to carry out testing and/or simulations on object models. Such simulations for example, may include the application of various loads and the calculation of how the structure behaves responsive to such loads based on a generated mesh for the structure. For example, in an example embodiment the simulation software may carry out stress analysis for a structure using a generated mesh for the object model of the structure. In other examples, simulation software may be capable of using meshes to simulate the physical effect of an automobile collision (or other test) on many connected structures represented by object models.

[0041] To carry out simulations on a structure, an object model of a structure may undergo a meshing process which divides a surface of the structure into a mesh of many connected four sided geometric shapes. Simulation software may use such meshes, as well as user configurable properties of one or more object models (such as the materials the structures are to be made of) and user configurable analysis parameters, to carry out mathematical processes such as finite element analysis in order to derive information regarding the effects (e.g., physical changes, displacements) that a simulation (e.g., a static load, collision) may have on a structure.

[0042] With reference to Fig. 1, an example system 100 that facilitates generating meshes for object models is illustrated. The system 100 may include at least one processor 102 that is configured to execute at least one application software component 104 from a memory 106 in order to carry out the various features described herein. The application software component may be configured (i.e., programmed) to cause the processor to carry out various acts and functions described herein. For example, the application software component may be configured to cause the processor 102 to access 3D object model data 108 from memory and/or some other data store 138 (e.g., file system, PLM database).

[0043] The application software component 104 may correspond to and/or include a mesh component 110 that generates a mesh on an object model. Also the application software component may correspond to and/or include a simulation software component 112 (such as CAE software) that may be comprised of one or more software components that include and/or communicate with the mesh component 110 that carries out generating a mesh on an object model.

[0044] Further as will be described in more detail below, a mesh software component may be comprised of one or more components that facilitate generation of a mesh. However, it should be appreciated that the features described herein as being carried out by separate components may be carried out by fewer components and/or a single component. Also, features carried out by a single described component may be carried out by two or more software components that communicate with each other.

[0045] An example of CAD/CAM/CAE (Computer-aided design/Computer-aided manufacturing/Computer-aided engineering) software that may be adapted to include at least some of the functionality described herein includes the NX suite of applications that is available from Siemens Product Lifecycle Management Software Inc. (Plano, Texas). However, it should also be understood that the described mesh component may correspond integrated into other types of application software components such as architectural software, animation rendering software, and/or any other 3D software that may use meshes of 3D object models.

[0046] The described system may include at least one display device 114 (such as a display screen) and at least one input device 116. For example, the processor may be included as part of

a PC, notebook computer, workstation, server, tablet, mobile phone, or any other type of computing system. The display device, for example, may include an LCD display, monitor, and/or a projector. The input devices, for example, may include a mouse, pointer, touch screen, touch pad, drawing tablet, track ball, buttons, keypad, keyboard, game controller, camera, motion sensing device that captures motion gestures, or any other type of input device capable of providing the inputs described herein. Also for devices such as a tablet, the processor 102 may be integrated into a housing that includes a touch screen that serves as both an input and display device. Further, it should be appreciated that some input devices (such as a game controller) may include a plurality of different types of input devices (analog stick, d-pad, and buttons).

[0047] Also, it should be noted that the processor described herein may be located in a server that is remote from the display and input devices described herein. In such an example, the described display device and input device may be included in a client device that communicates with the server (and/or a virtual machine executing on the server) through a wired or wireless network (which may include the Internet). In some embodiments, such a client device, for example, may execute a remote desktop application or may correspond to a portal device that carries out a remote desktop protocol with the server in order to send inputs from an input device to the server and receive visual information from the server to display through a display device. Examples of such remote desktop protocols include Teradici's PCoIP, Microsoft's RDP, and the RFB protocol. In another example, such a client device may correspond to a computer running a web browser or thin client application. Inputs from the user may be transmitted from the web browser or thin client application to be evaluated on the server, rendered by the server, and an image (or series of images) sent back to the client computer to be displayed by the web browser or thin client application. Also in some examples, the remote processor described herein may correspond to a combination of a virtual processor of a virtual machine executing in a physical processor of the server.

[0048] The described application software component 104 may be configured to be responsive to object loading inputs 118 received through the input device 116 to access the 3D model data 108 and cause the display device 114 to output a graphical user interface (GUI) 120 that displays a visual representation of a 3D object model 122 based on 3D model data 108. Further, the application software component 104 may be configured to be responsive to mesh generation

inputs 124 received through the input device to cause the mesh component 110 to generate mesh data 126 based on the loaded 3D model data 108. Such mesh data may also be stored in the data store 138 in correlated relation with the object model data 108 (e.g., CAD data).

[0049] In addition, the application software component 104 may be configured to be responsive to simulation inputs 128 received through the input device to cause the simulation component 112 to process the mesh data 126 and produce simulation results data 130. Such simulation results data may correspond to quantitative data regarding the effects of the simulation (stress test, collision) on the structure such as physical changes and displacements. It should also be appreciated that the described application software component 104 may be operative to cause the GUI 120 to display a visual representation 132 of the generated mesh as well as a visual representation 134 of the results of the simulation. Further the described application software component 104 maybe be operative to save such simulation data to the data store 138 in correlated relation with object model data 108.

[0050] In surface meshing, for certain simulation applications, a surface mesh comprising well-structured quadrilateral elements may be preferred over a surface mesh comprising triangular elements in order to enhance the accuracy, robustness, and solver convergence rate of the simulations. An example embodiment of the described mesh component 110 may include a multi-block / virtual topology engine 136 (e.g., a software function) that is configured to carry out the generation of a good quality quadrilateral mesh that minimizes and/or eliminates any need to manually refine sub-portions of the mesh (via a finer mesh with denser quadrilateral elements for example).

[0051] Fig. 2 illustrates an example 200 of such a well-structured quadrilateral mesh 202 of an example multi-loop face 204. As defined herein, a quadrilateral mesh of good quality is defined as a mesh where the majority of interior nodes 206 have a valency (i.e., degree - the number of elements incident to the vertex) of four and each of the four included angles 208 of as many quadrilateral elements 210 as possible (such as at least a majority of them) is substantially 90^{o}, (e.g., 80-100^{o}). To this end, the specific problem that is solved by example embodiments is the automatic generation of such a mesh on general surfaces of industrial parts and components.

[0052] Referring back to Fig. 1, an example embodiment of the mesh component 110 may be configured to automatically cause one or more selected faces of the surface for an object model 108 to be processed by the described multi-block virtual topology engine 136. Such an engine 136 may be configured (i.e., programmed) to sub-divide an input face (inputted to the engine 136) into an assortment of maximal convex "blocks" using a niche system of feature-dependent multi-blocking strategies that are selected in order to best achieve the generation of a well-structured quadrilateral mesh.

[0053] An example framework 300 for the described mesh component 110 and multi-block virtual topology engine 136 is illustrated in Fig. 3. In this example, the engine 136 may be comprised of three tiers of components. A first component 302, which is referred to herein as the geometry analyzer component, may be configured to categorize the type of an input face 310 using a series of topology and/or geometry based feature recognition tools.

[0054] This described multi-block virtual topology engine may also include a second component 304, which is referred to herein as a multi-block decomposer component. Such a multi-block decomposer component 304 may be configured to carry out any one of the many different multi-blocking processes (via selectable algorithms) on the input face 310 depending on the face type 312 classified/identified by the geometry analyzer component 302. The output of the multi-block decomposer component (for each of these different multi-blocking processes) may include data representative of block locations across the input face, such as virtual vertices and virtual edges 314 that are usable to determine the block locations.

[0055] In addition, the described multi-block virtual topology engine 300 may include a third component 306, which is referred to herein as a topology builder component. Such a topology building component 306 may be configured to determine data representative of virtual faces 316 for a virtual block topology that subdivides the input face 310 into the blocks based on the data (e.g., virtual vertices and virtual edges 314) determined by the multi-block decomposer component 304.

[0056] Such virtual faces data 316 produced by the described multi-block virtual topology engine 136 may be passed to a final mesh component 308. Such a final mesh component 308

may be configured to mesh the virtual faces in order to generate structured conformal mesh data 126 across the entire input face 310.

[0057] Fig. 4 illustrates a flow diagram 400 of an example workflow (comprised of several steps) that the described geometry analyzer component 302 may carry out to classify surface face types. In this example, in a first step 402, the geometry analyzer component may be configured (e.g., programmed) to categorize an input face 404 based on the number of loops comprised by the input face. A loop for example may correspond to a continuous surface portion that extends around the input face itself or bounds an opening or aperture through the input face. Thus, an input face with no apertures has a single loop, whereas an input face with one or more apertures therethrough has multiple loops.

[0058] For example, the geometry analyzer component at the first step 402 may determine whether the input face 404 includes a single loop or multiple loops. In this example, all multi-loop faces may be classified as *M0* type faces and all single loop faces may be classified as *S0* type faces. Fig. 5 illustrates examples 500 of various input surfaces 502 and corresponding single loop 504 and multi-loop 506 classifications for these various surfaces that may be determined by the described geometry analyzer component.

[0059] Depending on the face type determined in the first step 402, different determinations may be made by the geometry analyzer component to further face classification. For example, the geometry analyzer component may carry out a second step 406 in which the geometry analyzer component performs cylinder detection on the input face 404. If an *M0 or S0* type face (from the first step) is identified as a cylinder (in the second step), it may be reclassified as an *M1* or *S1* type face respectively by the geometry analyzer component.

[0060] As illustrated in the example 600 of Fig. 6, the geometry analyzer component may be configured to detect a cylindrical surface 602 by solving a best fit axis 604 to the surface 602. If the fit-error to the least square approximation is below a set threshold, then the surface is deemed cylindrical. In example embodiments, the geometry analyzer component may be configured to flag a face as a cylinder based on a determination that all points on a dense sample mesh of the *M*-type face meets the above criteria within predetermined tolerances.

[0061] In addition, to further subsequent processes described herein, the geometry analyzer component may (in addition to categorizing face types) may store in memory (and/or with the CAD data for the surface being analyzed) a corresponding virtual seam edge 606 which is parallel with the cylinder axis 604 of the face for the surface being classified.

[0062] Referring back to Fig. 4, the geometry analyzer component may be configured to carry out a third step 408 in which the geometry transforms all faces from 3D space to flattened UV space using the weighted half-edge method (WHEM) flattening algorithm (or other flattening algorithm) to carry out 2D parameterization of a tessellated 3D face. The WHEM flattening algorithm applied to both linear and nonlinear problems uses a compromise between conformal mapping and triangle altitude preservation techniques to generate 2D domains with highly reduced transformational distortion. The further steps and processes described herein related to classification, multi-block decomposing, and meshing may be carried out in this 2D domain and the resulting meshed surface may then be transformed back to 3D space.

[0063] For example, the geometry analyzer component may be configured to carry out further face type identification in this 2D domain based on the type of classification carried out in the first and second steps. After flattening in the third step 408, all *S0* and *M0* (i.e., non-cylindrical) faces may be processed using medial axis recognition to detect bead and annulus features of the faces.

[0064] For example, in a fourth step 410, the geometry analyzer component may carry out bead detection for all S0 (i.e., single-loop non-cylindrical) faces. Also for example, in a fifth step 412, the geometry 302 may carry out annulus detection for all *M0* (multi-loop non-cylindrical) faces. In this example, any *S0* type face which is identified as a bead may be reclassified as an *S1* type face. Also any *M0* type face which is identified as annulus may be reclassified as an *M2* type face.

[0065] Fig. 5 illustrates example face identification classifications 508 for the example input surfaces 502 that may be determined by the described geometry analyzer component. However, it should be appreciated that alternative embodiments of the described geometry analyzer component may identify and classify fewer, alternative, and/or additional types of faces.

[0066] With reference to the example surface 700 shown in Fig. 7, in example embodiments the geometry analyzer component may be configured to derive a medial axis 702 of a face 704 in the 2D domain via a Delaunay triangulation of a set of points on the object boundary 706. The Delaunay triangulation of a set of points is defined as a triangulation that has the property that the circumcircle 708 of the three points forming the vertices of each Delaunay triangle 710 contains no other points. The medial axis is traced out by the circumcircle center 712 when the distance between consecutive points on the boundary tends to zero and the circumscribed circle 708 of each Delaunay triangle 710 becomes inscribed within the object, as shown in Fig. 7.

[0067] With reference to the example 800 illustrated in Fig. 8, there are two entities which make up the medial axis in 2D, namely, medial edges 802 and medial vertices 804. The points on the medial axis which belong to a medial edge are equidistant from two entities on the boundary. Conversely, those which belong to a medial vertex are equidistant from three entities or more on the boundary. Medial edges are connected by medial vertices, as shown in Fig. 8. Medial edges can be classified topologically based on the type of Delaunay triangle from which their connected medial vertices are defined. The three main types of Delaunay triangle may be classified as Junction (*J*) 806, Normal (*N*) 808, or Corner (*C*) 810, having *0*, *1* and *2* edges on the object boundary 812 respectively, as shown in Fig. 8. Consequently, a medial edge flanked with medial vertices derived from a J and C type Delaunay triangle would be classified as a *J-C* medial edge etc.

[0068] Referring to the examples 900 shown in Fig. 9, the two distinct features which the medial axis may be used by the described geometry analyzer component to detect are medial topologies of beads (*S* type) 902 and annuli (*M* type) 904. All beads have a *C-C* type medial edge. Annuli are special cases where the medial edge has no medial vertices and all axis points are derived from *N* type Delaunay triangles. This edge may be referred to as an *N-N* medial edge.

[0069] Referring back to Fig. 3, as discussed previously, the multi-block decomposer component selects and carries out a particular multi-block subdivision process on a face (in the 2D domain) based on the particular classifications determined by the geometry analyzer component 302. Such selectable multi-block subdivision processes may include: one or more medial axis decomposition algorithms, a Cartesian slabbing decomposition algorithm, and an Art Gallery

decomposition algorithm, based on the particular classification of a face type. However, it should also be appreciated that further alternative embodiments may use fewer, alternative, and/ or additional decomposition algorithms based on different classifications of faces carried out by alternative embodiments of the described geometry analyzer component. For example, as will be described in more detail below one of the medial axis decomposition algorithms may be combined with a visibility based decomposition algorithm in order to process relatively complex face curvatures.

[0070] Fig. 10 illustrates a flow diagram 1000 for the functions carried out by the described multi-block decomposer component. In this example, in a first step 1002, the multi-block decomposer component may be configured (e.g., programmed) to select one of the available decomposition algorithms based on the face type classification of an input face 1004 and cause a second step to be carried out in which the selected decomposition algorithm is used to process the input face.

[0071] For example, if the face type is an S0 type (i.e., single loop non-bead type), the multi-block decomposer component may cause a second step 1006 to be carried out in which an Art Gallery decomposition algorithm is used to process the input face 1004. Also, if the face type is an S1 or M1 type (i.e., single or multi-loop cylinder type), the multi-block decomposer component may cause an alternative second step 1008 to be carried out in which a Cartesian slab decomposition algorithm is used to process the input face 1004. In addition, if the face type is an M2 type (i.e., multi-loop annulus type), the multi-block decomposer component may cause an alternative second step 1010 to be carried out in which a medial axis based annulus decomposition algorithm is used to process the input face 1004. Further, if the face type is an S2 type (i.e., single loop bead type), the multi-block decomposer component may cause an alternative second step 1012 to be carried out in which a medial axis based annulus decomposition algorithm is used to process the input face 1004. In addition, if the face type is an M0 type (i.e., multi-loop non-cylinder non-annulus type), the multi-block decomposer component may cause an alternative second step 1014 to be carried out in which a medial axis based decomposition algorithm is used in combination with a visibility based decomposition algorithm to process the input face 1004.

[0072] The output of each of these decomposition algorithms may be stored in memory or other data store and may correspond to data representative of block locations defined by lines that split the input face into a number of blocks (i.e., virtual faces) and which lines represent virtual edge candidates on the input face. The data that represents such split lines may include the virtual vertices that define the positions of the virtual edges corresponding to such split lines. Fig. 11 illustrates examples 1100 of these determined virtual edge lines 1102 that subdivide each of the different classified faces described previously with respect to Fig. 5 into a several blocks. However, it should be appreciated that alternative embodiments of the described multi-block decomposer component may place the virtual edge lines in different locations depending on the configuration and parameters associated with the decomposition algorithms described herein. In addition, the following provides a more detailed discussion for how this data may be generated by each of these algorithms.

[0073] The described medial axis decomposition algorithms may employ a hybrid technique that uses both circumcenter and triangle height to compute medial axis points. Topology traversal algorithms establish the medial edges. Superfluous medial edges or branches are avoided by analyzing degenerate, flat triangles on the boundary. All associated medial axis attributes of thickness, edge length, touching points are stored with the face. In addition, the touching point information of the inscribed circle is used to define the location of split lines for bead and annulus features. For more complex multi-loop faces, the Delaunay mesh from which the medial axis is derived may be employed to remove concavities and the thickness information from the diameter of the inscribed circle defines the split locations along the medial edge.

[0074] For example, the combined medial axis based and visibility based decomposition algorithm 1014 (shown in Fig. 10) may be applied to input faces classified as M0 type faces (multi-loop non-cylinder and non-annuli type). In the processing of an input face, an initial step for this algorithm may be to remove any concavities from the face. Concavities may be identified by the vertex included angle, θ as illustrated in the example 1200 of Fig. 12. When θ about equals 270° or more this will identify a concavity at a vertex which will be removed when carrying out the algorithm by the described multi-block decomposer component.

[0075] In example embodiments, the Delaunay tessellation of the domain may be carried out by the algorithm to identify concavities at vertices based on this principle. Moreover, triangulation may also be employed to decide upon the optimum splitting of edges to remove the concavity. For example, as illustrated in the example 1300 of Fig. 13, the Delaunay edges extending from each concavity may be determined and evaluated, and the most orthogonal edge 1302 may be selected.

[0076] As illustrated in the example shown in Fig. 14, after all concavities have been removed, the next step of this described algorithm may be to place split lines 1402 defined by the touching points of the inscribed circle at positions on each medial edge where the diameter of the circle is either maximum or minimum. In an example embodiment, the most complicated sub-region which may be generated is where all three medial edges surrounding a medial vertex are split to leave a six sided sub-region 1404.

[0077] This described medial axis algorithm may be combined with a visibility based decomposition algorithm to provide additional split lines for complex faces. A description of the visibility based decomposition algorithm is described in more detail below with respect to Figs. 22-24. The split lines generated by the concavity removal, medial axis decomposition and visibility decomposition may then be stored by the multi-block decomposer component as virtual edge candidates on the input face.

[0078] Also for example, the annulus medial axis decomposition algorithm 1010 may be applied to input faces classified as M2 type faces (multi-loop annulus type). As illustrated in Fig. 15 for annulus, split lines 1502 defined by the touching points of the inscribed circle 1504 are placed at equidistant intervals on the input face by the multi-block decomposer component to produce a plurality of arched shaped blocks 1506. The number of split lines may be a user configurable parameter of the described application software component in order to configure the granularity of the blocking.

[0079] Fig. 15 illustrates an example where the number of split lines 1502 on the input face is 4. Such split lines may produce four four-sided blocks 1504. The determined splits lines may be stored by the multi-block decomposer component as virtual edge candidates on the input face.

[0080] In addition, the bead medial axis decomposition algorithm 1012 may be applied to input faces classified as S2 type faces (single-loop bead type). As illustrated in the example 1600 shown in Fig. 16 for a bead, split lines 1602 defined by the touching points of the inscribed circle 1604 may be placed at an offset from the start and end of the medial edge. This defines three blocks including a rectangular portion 1606 in the middle of the bead, flanked by two curved caps 1608, 1610. Such determined split lines may be stored as virtual edge candidates for the input face.

[0081] In example embodiments, the Cartesian slab decomposition algorithm may be targeted for use with cylinders and periodic surfaces for which an axis exists. Since the slabs are all parallel to the axis and each other, this algorithm ensures a regular and structured decomposition of thin slabs that can be easily meshed giving a structured look and feel to meshes even if they are not map-meshed. Furthermore, the computation of a best fit axis using a least squares approximation enables not only the recognition of cylindrical features but provides reference data for Cartesian grid alignment used by the Cartesian slab decomposition algorithm.

[0082] For example, the Cartesian slabbing decomposition algorithm may be applied to input faces classified as S1 and M1 type faces (single-loop and multi-loop cylinder type). As illustrated in the example 1700 shown in Fig. 17, during the flattening of a 3D cylindrical face 602 (carried out by the geometry analyzer component) to 2D space, the 3D to UV transformation is also applied to the virtual seam edge 606 (determined by the geometry analyzer component). The resulting flattened face 1702 and virtual seam edge 1704 will typically occupy a non-axially aligned orientation in 2D UV space.

[0083] An initial step carried out via the Cartesian slabbing algorithm may be to apply a transformation to the flattened face 1702 in 2D UV space to align it with the orthogonal axes of the global Cartesian (X, Y) coordinate system. As the virtual seam edge 1704 is aligned with the cylinder axis, the transformation may be computed to align it with the Y axis 1706 of the Cartesian grid. This transform is then applied to the flattened face 1702, thus producing an aligned flattened face 1708 (in which the corresponding cylindrical axis 604 is parallel with the Y axis).

[0084] The next stage carried out via the Cartesian slabbing algorithm may be to define the slab shaped blocks which subdivide the face and are parallel to the corresponding cylindrical axis of input face. The number of slab blocks may be a user configurable parameter of the described application software component in order to configure the granularity of the blocking. The slab width may be determined based on the configurable number of slab elements. Split lines may then be generated by computing the intersection between a line extending vertically at each interval defined by the determined slab width and a boundary of the aligned flattened face 1708.

[0085] The Art Gallery multi-blocking algorithm, for example, may be employed on faces with single loops. It depends on a coarse boundary discretization and avoids the use of a Delaunay mesh. For example, the art gallery decomposition algorithm may be applied to input faces classified as S0 type faces (single-loop non-cylinder non-bead type). The Art Gallery Theorem (aka Watchman Theorem), provides ground for optimizing the surveillance of art galleries. It states that for a generic, single-loop polygonal floor with n vertices, n/3 floor guards g(n) or surveillance cameras are sometimes necessary and always sufficient. Several lemmas may be used to provide a cheap and smart way to subdivide a non-convex orthogonal polygon into convex orthogonal regions. Orthogonal partitioning of a concave polygon from its reflex vertices can result in convex partitioning of the face. A useable lemma, for any generic polygon with r reflex vertices, can be therefore written as:

$$\frac{r}{2} + 1 \leq A_n \leq r + 1$$

where A_n is the number of resulting convex sub-areas. Fig. 18 shows an example 1800 of how r (4) shortest orthogonals dropped (i.e., extended) from r (4) reflex vertices can partition a polygon into r+1 (5) convex sub-areas. Fig. 19 shows an example 1900 where the minimal condition where r (8) reflex vertices occur opposite to each other and r/2 (4) shortest orthogonals are dropped (extended) to decompose a concave polygon with r (8) vertices into r/2+1(5) convex sub-areas.

[0086] Improvising upon and extending this lemma to generic polygons, the described Art Gallery decomposition algorithm may be configured to generate structured meshes by first determining boundary concavities. The input face may be first boundary discretized at a size

slightly smaller than a user configurable element size which may correspond to an intended feature size. This means if there are features on the face boundary much smaller than the user configurable element size, the algorithm may be configured to ignore them.

[0087] In addition, at each vertex i of the input face (in 3D UV space), the included angle φ_i is computed between the node at the vertex and its trailing and leading nodal neighbors. Based on this angle, a vertex is marked as concave or reflex following the rule

$$\varphi_i > 270^{o} - \theta_t$$

where φ_t is the variable angular tolerance (typically 60^{o}).

[0088] For illustration purposes a notch diagram 2000 is shown in Fig. 20, which depicts relationships between reflex vertices (VR) 2002 and their pair vertices (VP) 2004 for an input face 2006. In order to develop the notch diagram, the face boundary is first discretized at element size and all reflex vertices are first projected orthogonally to their closest neighboring edge.

[0089] The art gallery decomposition algorithm may be configured to next establish a relationship between each reflex vertex (VR) and its pair vertex (VP). Some of the reflex vertices can be virtual (VR_v) and typically most of the pair vertices are virtual (VP_v). When all reflex vertexes (VR) are linked with their pair vertex (VP) the notch diagram is complete as shown in Fig. 20. In mathematical notations, the notch diagram (ND) can be expressed as a collection of links between n reflex vertices and their orthogonally projected pair vertices given by:

$$ND = \bigcup_{i=1}^{n} \left(V_{ri} \mapsto V_{pi} \right)$$

[0090] The described art gallery decomposition algorithm may be configured to next use a split line criteria to determine the best split line from each reflex vertex out of the many connection possibilities on the boundary. Once the split lines are put in place, the face has been decomposed into an optimum number of convex sub-regions. An example 2100 of a final result (mesh) in 3D space based on an art-gallery decomposed single loop input face with 9 reflex vertices is shown in Fig. 21.

[0091] In a further example embodiment the multi-block decomposer component may also use a visibility based decomposition algorithm on complex input faces. The visibility based decomposition algorithm may be applied to both single and multi-loop faces to append/introduce additional split lines that help breaking down several convex regions into smaller quad-shaped zones for better defined structured mesh generation. In particular, as discussed previously, an example embodiment of the multi-block decomposer component may select and use a combined medial axis based decomposition and visibility based decomposition algorithm for input faces that are classified as multi-loop non-annuli and non-cylindrical (e.g., M0 type input faces)

[0092] The described example visibility based decomposition algorithm does not need a Delaunay mesh and depends on a boundary discretization equal to the global size. The visibility based decomposition algorithm determines a "visibility" of a boundary point with respect to other points on the face boundary as illustrated in the example 2200 of Fig. 22. The concave or reflex vertex j 2202 has two lines of extension jl 2204 and jr 2206 that divide the face contour into 3 zones – D1 2208, D2 2210, and D3 2212. The face area to the left of line jl is D1, that to the right of line jr is D3 and the zone is between is D2. Typically, all points in zone D2 are visible to the point j. Each visible point is thus evaluated by means of a split-line functional expressed for like jk 2214 as:

$$\Phi_{jk} = f_{sym} (W_H H + W_L L)$$

[0093] The minimum value of the functional of all valid visible split line candidates, gives the best split-line. The normalized length parameter or length cost function L is given by $L = l_{jk}/l_d$; where l_{jk} is the length of the split line jk (as shown in Fig. 22) and l_d is the characteristic length, which is the diagonal of the rectangular box bounding the mesh area $B \equiv [(xmin,ymin) , (xmax, ymax)]$. The constant angle and length weight factors are developed heuristically and may correspond to $W_H = 1.0$ and $W_L = 5.0$ for example.

[0094] Also, f_{sym} is a split-line symmetry weight factor which is defined as: $f_{sym} = 0.8$ for symmetric split lines; and $f_{sym} = 1.0$ for non-symmetric lines. Further H, or the split-line angle summation term is expressed as:

$$H = \sum^{4} W_{\alpha i} h_i$$

i=1

[0095] Here the angle weight factors $W_{\alpha i}$ may typically be 1.0 for most angles but may be computed giving more weight to split lines closest to 90 deg. For example, they may be altered to 0.7 and 1.3 as the split lines get close to 90 deg as follows: When $h_1 < 0.033$, $h_2 < 0.033$, $h_3 < 1.0$, and $h_4 < 1.0$, then $W_{\alpha 1} = W_{\alpha 2} = 1.3$ and $W_{\alpha 3} = W_{\alpha 4} = 0.7$; and when $h_1 < 1.0$; $h_2 < 1.0$; $h_3 < 0.033$; $h_4 < 0.033$, then $W_{\alpha 1} = W_{\alpha 2} = 0.7$ and $W_{\alpha 3} = W_{\alpha 4} = 1.3$.

[0096] Here the angle cost functions h_i may be computed in a manner that the better the angle, the lower the value. For quads, this value approaches 0 at 90 deg. Split line angles and very high at angles below 60 deg. They may be expressed as:

$$h_i = 0.5\ [\ \{\cos^2\varphi_i\ .\ f_\alpha\ .\ (1 - \cos^2\varphi_i) - 1\} + 1] \quad | \quad 170^o \geq \varphi_i \geq 30^o$$

$$h_i = 1 \quad | \quad \varphi_i < 30^o\ ;\ \varphi_i > 170^o$$

[0097] where the split angles φ_i 2216, around the split line 2214 are shown in Fig. 22 and the angle cosine factor f_α can be written as $f_\alpha = 4/\cos^2\varphi_{min}$.

[0098] Symmetry of the split-line is measured by proximity of the alternate or corresponding angles. For example, symmetric conditions may correspond to:

$$\varphi_1 \approx \varphi_3 \quad \varphi_2 \approx \varphi_4$$
$$\text{or}$$
$$\varphi_1 \approx \varphi_4 \quad \varphi_2 \approx \varphi_3$$

[0099] This example formulation of the split line functional emphasizes orthogonal splitting, symmetry of decomposition and point proximity (shortness of the split lines). As illustrated in Fig. 22, in this example, the best split line 2214 (jk in this case) for the boundary discretized face has the least split line functional value of all possible visible candidates.

[00100] The described visibility based decomposition algorithm may be configured such that every time a splitting of the face contour happens, the face contour-loop is split into a sub-loop on the right and one to the left of the split line. Recursive splitting may continue until the resultant sub-contours are "quad-shaped". The example, multi-block decomposer component may use a quad shape check algorithm to test out the sub-contours and determine when the sub-

contours are sufficiently "quad-shaped" in order to end the recursive slitting on this portion of the input face.

[00101] Fig. 23 illustrates an example 2300 in which an input face 2302 has been split into quad-regions by the described visibility based decomposition algorithm. In this example a face with two inner loops 2304 is decomposed with the visibility based multi-blocking algorithm into 11 virtual quad-shaped faces. Fig. 24 illustrates an example 2400 of this input face 2302 in 2D space after the 11 blocks produced by the visibility based decomposition algorithm have been meshed.

[00102] Referring back to Fig. 3, as discussed previously, the multi-block/virtual topology engine 136 of the mesh component 110 may also include the topology builder component 306, which processes the data outputted by the multi-block decomposer component 304. Such data is usable to determine the virtual edge candidates (i.e., split lines) that define blocks that subdivide the input face. The data may include vertex relationships with respect to the 2D representations of the input face, which are usable by the topology builder component to determine the virtual edge candidates and corresponding faces.

[00103] For example, the topology builder component may be configured to determine a notch diagram (ND) for the outputs from any of the multi-blocking decomposition algorithms described herein. An example of such a notch diagram 2500 for this purpose is illustrated in Fig. 25. The topology builder component may determine the virtual edges (ve) 2502 between linked vertex pairs (e) 2504, 2506. Given the ND, the virtual edges (ve), virtual vertices (e) and the original face geometry, the topology builder component may construct the virtual faces (V_f) 2508. All of unused, temporarily generated nodes on the face boundary may be deleted at this point by the topology builder component.

[00104] Thus the determined ND in Fig. 25 may correspond to an example of the resulting virtual topology after virtual face decomposition in the 2D parameter space for an input face. It should be appreciated that the described topology builder component may be extremely light-weight, contain minimal data (e.g., mostly pointers/references to nodes/vertices/ edges), and may run in the parameter space of the meshing component.

[00105] Referring back to Fig. 3, as discussed previously, the mesh component 110 may also include a final mesh component 308 that is configured to generate a final mesh on each of the virtual faces generated by the topology builder component 306. When the final mesh is generated on the virtual faces, the nodes/elements of all virtual faces are associated with the original parent face/edges of the input face and the virtual topology is discarded.

[00106] In example embodiments, transfinite meshing is carried out by the final mesh component 308 on the virtual faces first. Such transfinite meshing may be successful on a majority of the virtual faces because they are 4-sided in this example. The remaining non-map-meshable virtual faces may be meshed by a subdivision/paving meshing algorithm. The resulting final mesh may be more structured as a result of the described multi-block decomposition process than what is capable of being produced by a subdivision/ paving algorithm on an undecomposed face.

[00107] In example embodiments, engineers may use simulation results carried out using the mesh data described herein to modify object models (via CAD software) in order to optimize structures to meet desired performance criteria (e.g., safer automobiles). The final mesh and/or CAD data and/or product data corresponding to the modified structure may be stored in the data store 138 (such as a CAD file and/or a PLM database). The described application software component and/or other software application may then carry out various functions based on the modified structure stored in the CAD data and/or product data.

[00108] Such functions may include generating (based on the CAD data and/or product data) engineering drawings and/or a Bill of Material (BOM) that specifies particular components (and quantities thereof) that may be used to build the structure. Such engineering drawings and/or BOM may be printed via a printer on paper, generated in an electronic form such as a Microsoft Excel file or Acrobat PDF file, displayed via a display device, communicated in an e-mail, stored in a data store, or otherwise generated in a form capable of being used by individuals and/or machines to build a product corresponding to the designed structure. Further, it should be appreciated that a machine such as a 3D printer may use data representative of the object model (before or after being modified by CAD software in view of the simulation results) to produce a physical structure (e.g., a part) from the object data.

[00109] With reference now to Fig. 26, various example methodologies are illustrated and described. While the methodologies are described as being a series of acts that are performed in a sequence, it is to be understood that the methodologies may not be limited by the order of the sequence. For instance, some acts may occur in a different order than what is described herein. In addition, an act may occur concurrently with another act. Furthermore, in some instances, not all acts may be required to implement a methodology described herein.

[00110] It is important to note that while the disclosure includes a description in the context of a fully functional system and/or a series of acts, those skilled in the art will appreciate that at least portions of the mechanism of the present disclosure and/or described acts are capable of being distributed in the form of computer-executable instructions contained within non-transitory machine-usable, computer-usable, or computer-readable medium in any of a variety of forms, and that the present disclosure applies equally regardless of the particular type of instruction or signal bearing medium or storage medium utilized to actually carry out the distribution. Examples of non-transitory machine usable/readable or computer usable/readable mediums include: ROMs, EPROMs, magnetic tape, floppy disks, hard disk drives, SSDs, flash memory, CDs, DVDs, and Blu-ray disks. The computer-executable instructions may include a routine, a sub-routine, programs, applications, modules, libraries, a thread of execution, and/or the like. Still further, results of acts of the methodologies may be stored in a computer-readable medium, displayed on a display device, and/or the like.

[00111] Referring now to Fig. 26, a methodology 2600 that facilitates generating meshes for object models is illustrated. The method may start at 2602 and the methodology may include several acts carried out through operation of at least one processor. These acts may include an act 2604 of classifying a type of an input face of a three dimensional (3D) object model of a structure based at least in part on a number of loops included by the input face. In addition the methodology may include an act 2606 of selecting based on the classified type of the input face, a multi-block decomposition algorithm from among a plurality of multi-block decomposition algorithms that the processor is configured to use. Further the methodology may include an act 2608 of using the selected multi-block decomposition algorithm to determine locations of a plurality of blocks across the input face. In addition, the methodology may include an act 2610 of

meshing each block to produce mesh data defining a mesh that divides the input face into a plurality of quadrilateral elements. At 2612 the methodology may end.

[00112] It should be appreciated that the methodology 2600 may include other acts and features discussed previously with respect to the system 100. For example, the methodology may include generating a flattened two dimensional (2D) representation of the input face. Classifying the type of the input face may be based on the 2D representation of the input face. Also using the selected multi-block decomposition algorithm to determine locations of the plurality of blocks across the input face may be carried out based on the 2D representation of the input face.

[00113] In addition, the act 2604 of classifying the type of the input face for a plurality of input faces may include determining whether: at least one of the input faces includes a single loop or multiple loops; whether at least one of the input faces includes a cylindrical surface; whether at least one of the input faces includes a medial axis bead; and whether at least one of the input faces includes a medial axis annulus.

[00114] Further, the acts 2606, 2608 of selecting and using the multi-block decomposition algorithm for a plurality of input faces may include selecting and using a Cartesian slab decomposition algorithm to subdivide at least one of the input faces having a cylindrical surface into several parallel slab shaped blocks that are parallel to the axis of the cylindrical surface.

[00115] Also, the acts 2606, 2608 of selecting and using the multi-block decomposition algorithm for a plurality of input faces may include selecting and using at least one medial axis decomposition algorithm including selecting and using at least one of: a medial axis based bead decomposition algorithm; a medial axis bead type decomposition algorithm ; and/or a combined medial axis based and visibility based decomposition algorithm. The medial axis based bead decomposition algorithm may be used to subdivide at least one of the input faces having a medial axis bead type input face classification into three blocks which include a rectangular block in the middle of the bead, flanked by two curved block caps. The medial axis bead type decomposition algorithm may be used to subdivide at least one of the input faces having a medial axis annulus type input face classification into the plurality of blocks which include a plurality of arched shaped blocks. Also, the combined medial axis based and visibility based decomposition

algorithm may be used to subdivide at least one of the input faces having multi-loop type input face classification based on a determination that the at least one of the input faces is not a cylindrical type input face or a medial axial annuli type input face.

[00116] Further, the acts 2606, 2608 may include selecting and using an Art Gallery decomposition algorithm to subdivide at least one of the input faces having a single loop type input face classification based on a determination that the at least one of the input faces is not a cylindrical type input face or a medial axial bead type input face.

[00117] In this described methodology, the produced mesh may include interior nodes, where the majority of interior nodes have a valency of four, and where the majority of quadrilateral elements of the mesh have four included angles that are between 80-100^{o}.

[00118] Also the described methodology may include an act of determining physical effects on the structure based at least in part on the mesh data and data representative of loads applied to the structure provided through operation of at least one input device. In addition, the methodology may include an act generating at least one graphical user interface (GUI) through a display device, which at least one GUI displays a visual representation of the object model, the mesh for the object model, and the determined physical effects on the structure. Further, the methodology may include an act of responsive to further inputs through the at least one input device, modifying the object model and storing data representative of the modified object model in at least one data store.

[00119] As discussed previously, modifications to object models for structures based on simulation results (produced from the meshes described herein) may be persisted as CAD data and/or product data to a CAD file and/or a PLM data store. Acts associated with generating engineering drawings and/or a BOM may then be carried out based on the CAD data or product data. Further, the methodology may include individuals manually building the structure based on the engineering drawings and/or BOM. Further such acts may include a machine (such as a 3D printer) building a structure based on the CAD data.

[00120] As discussed previously, acts associated with these methodologies (other than any described manual acts such as an act of manually building a structure) may be carried out by one

or more processors. Such processor(s) may be included in one or more data processing systems, for example, that execute software components operative to cause these acts to be carried out by the one or more processors. In an example embodiment, such software components may comprise computer-executable instructions corresponding to a routine, a sub-routine, programs, applications, modules, libraries, a thread of execution, and/or the like. Further, it should be appreciated that software components may be written in and/or produced by software environments/languages/frameworks such as Java, JavaScript, Python, C, C#, C++ or any other software tool capable of producing components and graphical user interfaces configured to carry out the acts and features described herein.

[00121] Fig. 27 illustrates a block diagram of a data processing system 2700 (also referred to as a computer system) in which an embodiment can be implemented, for example, as a portion of a CAE CAD, PLM, and/or other system operatively configured by software or otherwise to perform the processes as described herein. The data processing system depicted includes at least one processor 2702 (e.g., a CPU) that may be connected to one or more bridges/controllers/buses 2704 (e.g., a north bridge, a south bridge). One of the buses 2704, for example, may include one or more I/O buses such as a PCI Express bus. Also connected to various buses in the depicted example may include a main memory 2706 (RAM) and a graphics controller 2708. The graphics controller 2708 may be connected to one or more display devices 2710. It should also be noted that in some embodiments one or more controllers (e.g., graphics, south bridge) may be integrated with the CPU (on the same chip or die). Examples of CPU architectures include IA-32, x86-64, and ARM processor architectures.

[00122] Other peripherals connected to one or more buses may include communication controllers 2712 (Ethernet controllers, WiFi controllers, cellular controllers) operative to connect to a local area network (LAN), Wide Area Network (WAN), a cellular network, and/or other wired or wireless networks 2714 or communication equipment.

[00123] Further components connected to various busses may include one or more I/O controllers 2716 such as USB controllers, Bluetooth controllers, and/or dedicated audio controllers (connected to speakers and/or microphones). It should also be appreciated that various peripherals may be connected to the USB controller (via various USB ports) including

input devices 2718 (e.g., keyboard, mouse, touch screen, trackball, gamepad, camera, microphone, scanners, motion sensing devices), output devices 2720 (e.g., printers, speakers) or any other type of device that is operative to provide inputs or receive outputs from the data processing system. Further it should be appreciated that many devices referred to as input devices or output devices may both provide inputs and receive outputs of communications with the data processing system. Further it should be appreciated that other peripheral hardware 2722 connected to the I/O controllers 2716 may include any type of device, machine, or component that is configured to communicate with a data processing system.

[00124] Additional components connected to various busses may include one or more storage controllers 2724 (e.g., SATA). A storage controller may be connected to a storage device 2726 such as one or more storage drives and/or any associated removable media, which can be any suitable non-transitory machine usable or machine readable storage medium. Examples, include nonvolatile devices, volatile devices, read only devices, writable devices, ROMs, EPROMs, magnetic tape storage, floppy disk drives, hard disk drives, solid-state drives (SSDs), flash memory, optical disk drives (CDs, DVDs, Blu-ray), and other known optical, electrical, or magnetic storage devices drives and/or computer media. Also in some examples, a storage device such as an SSD may be connected directly to an I/O bus 2704 such as a PCI Express bus.

[00125] A data processing system in accordance with an embodiment of the present disclosure may include an operating system 2728, software/firmware 2730, and data stores 2732 (that may be stored on a storage device 2726 and/or the memory 2706). Such an operating system may employ a command line interface (CLI) shell and/or a graphical user interface (GUI) shell. The GUI shell permits multiple display windows to be presented in the graphical user interface simultaneously, with each display window providing an interface to a different application or to a different instance of the same application. A cursor or pointer in the graphical user interface may be manipulated by a user through a pointing device such as a mouse or touch screen. The position of the cursor/pointer may be changed and/or an event, such as clicking a mouse button or touching a touch screen, may be generated to actuate a desired response. Examples of operating systems that may be used in a data processing system may include Microsoft Windows, Linux, UNIX, iOS, and Android operating systems. Also, examples of data

stores include data files, data tables, relational database (e.g., Oracle, Microsoft SQL Server), database servers, or any other structure and/or device that is capable of storing data which is retrievable by a processor.

[00126] The communication controllers 2712 may be connected to the network 2714 (not a part of data processing system 2700), which can be any public or private data processing system network or combination of networks, as known to those of skill in the art, including the Internet. Data processing system 2700 can communicate over the network 2714 with one or more other data processing systems such as a server 2734 (also not part of the data processing system 2700). However, an alternative data processing system may correspond to a plurality of data processing systems implemented as part of a distributed system in which processors associated with several data processing systems may be in communication by way of one or more network connections and may collectively perform tasks described as being performed by a single data processing system. Thus, it is to be understood that when referring to a data processing system, such a system may be implemented across several data processing systems organized in a distributed system in communication with each other via a network.

[00127] Further, the term "controller" means any device, system or part thereof that controls at least one operation, whether such a device is implemented in hardware, firmware, software or some combination of at least two of the same. It should be noted that the functionality associated with any particular controller may be centralized or distributed, whether locally or remotely.

[00128] In addition, it should be appreciated that data processing systems may be implemented as virtual machines in a virtual machine architecture or cloud environment. For example, the processor 2702 and associated components may correspond to a virtual machine executing in a virtual machine environment of one or more servers. Examples of virtual machine architectures include VMware ESCi, Microsoft Hyper-V, Xen, and KVM.

[00129] Those of ordinary skill in the art will appreciate that the hardware depicted for the data processing system may vary for particular implementations. For example, the data processing system 2700 in this example may correspond to a computer, workstation, and/or a server. However, it should be appreciated that alternative embodiments of a data processing

system may be configured with corresponding or alternative components such as in the form of a mobile phone, tablet, controller board or any other system that is operative to process data and carry out functionality and features described herein associated with the operation of a data processing system, computer, processor, and/or a controller discussed herein. The depicted example is provided for the purpose of explanation only and is not meant to imply architectural limitations with respect to the present disclosure.

[00130] As used herein, the terms "component" and "system" are intended to encompass hardware, software, or a combination of hardware and software. Thus, for example, a system or component may be a process, a process executing on a processor, or a processor. Additionally, a component or system may be localized on a single device or distributed across several devices.

[00131] Also, as used herein a processor corresponds to any electronic device that is configured via hardware circuits, software, and/or firmware to process data. For example, processors described herein may correspond to one or more (or a combination) of a microprocessor, CPU, FPGA, ASIC, or any other integrated circuit (IC) or other type of circuit that is capable of processing data in a data processing system, which may have the form of a controller board, computer, server, mobile phone, and/or any other type of electronic device.

[00132] Those skilled in the art will recognize that, for simplicity and clarity, the full structure and operation of all data processing systems suitable for use with the present disclosure is not being depicted or described herein. Instead, only so much of a data processing system as is unique to the present disclosure or necessary for an understanding of the present disclosure is depicted and described. The remainder of the construction and operation of data processing system 2700 may conform to any of the various current implementations and practices known in the art.

[00133] Although an exemplary embodiment of the present disclosure has been described in detail, those skilled in the art will understand that various changes, substitutions, variations, and improvements disclosed herein may be made without departing from the spirit and scope of the disclosure in its broadest form.

[00134] None of the description in the present application should be read as implying that any particular element, step, act, or function is an essential element which must be included in the claim scope: the scope of patented subject matter is defined only by the allowed claims. Moreover, none of these claims are intended to invoke a means plus function claim construction unless the exact words "means for" are followed by a participle.

CLAIMS

What is claimed is:

1. A system (100) for generating meshes comprising:

at least one processor (102) configured to: classify a type (312, 508) of an input face (204, 310, 502) of a three dimensional (3D) object model (108) of a structure based at least in part on a number of loops (504, 506) included by the input face; select based on the classified type of the input face a multi-block decomposition algorithm (1006, 1008, 1010, 1012, 1014) from among a plurality of multi-block decomposition algorithms that the processor is configured to use; use the selected multi-block decomposition algorithm to determine locations of a plurality of blocks (1506) across the input face; and mesh each block to produce mesh data (126) defining a mesh (202) that divides the input face into a plurality of quadrilateral elements (210).

2. The system according to claim 1, wherein the at least one processor is configured to generate a flattened two dimensional (2D) representation (1702) of the input face (602), wherein the at least one processor is configured to determine the classified type of the input face from among a plurality of input face types that the at least one processor is configured to determine based on the 2D representation of the input face, wherein the at least one processor is configured to use the 2D representation of the input face to determine the locations of the plurality of blocks across the input face based on the classified type of the input face, wherein the plurality of input face types that the at least one processor is configured to determine, are configured to distinguish whether: the input face includes a single loop or multiple loops; whether the input face includes a cylindrical surface; whether the input face includes a medial axis bead; and whether the input face includes a medial axis annulus.

3. The system according to claim 2, wherein the plurality of multi-block decomposition algorithms that the at least one processor is configured to select and use to process the 2D representation of the input face include a Cartesian slab decomposition algorithm that is used to subdivide the input face having a cylindrical surface into a several parallel slab shaped blocks

that are parallel to the axis of the cylindrical surface, wherein the plurality of available multi-block decomposition algorithms that the at least one processor is configured to select and use to process the 2D representation of the input face include at least one medial axis decomposition algorithm including at least one of:

a medial axis based bead decomposition algorithm that is configured to subdivide at least one of the input faces having a medial axis bead type input face classification into three blocks which include a rectangular block in the middle of the bead, flanked by two curved block caps;

a medial axis bead type decomposition algorithm that is configured to subdivide at least one of the input faces having a medial axis annulus type input face classification into the plurality of blocks which include a plurality of arched shaped blocks; and/or

a combined medial axis based and visibility based decomposition algorithm that is configured to subdivide at least one of the input faces having multi-loop type input face classification based on a determination that the at least one of the input faces is not a cylindrical type input face or a medial axial annuli type input face.

4. The system according to claim 3, wherein when the combined medial axis based and visibility based decomposition algorithm is selected, the at least one processor is configured to carry out a visibility based decomposition portion of the algorithm in which split lines (1402, 1502, 1602, 2214) are determined that subdivide the input face into blocks based on orthogonally, symmetry, and shortness of candidate split lines.

5. The system according to claim 4, wherein the plurality of available multi-block decomposition algorithms that the at least one processor is configured to select and use to process the 2D representation of the input face include at least one art gallery decomposition algorithm which is used to subdivide the input face having a single loop type input face classification when the at least one processor determines that the input face is not a cylindrical type input face or a medial axial bead type input face.

6. The system according to any one of claim 1 to 5, wherein the mesh includes interior nodes (206), wherein the majority of interior nodes have a valency of four, and wherein the majority of quadrilateral elements have four included angles (208) that are between 80-100°, wherein the at least one processor is configured to determine physical effects (130) on the structure based at least in part on the mesh data and data representative of loads applied to the structure provided through operation of at least one input device (116), wherein the at least one processor is operative to generate at least one graphical user interface (GUI) (120) through a display device (114), which at least one GUI displays a visual representation (122) of the object model, the mesh (132) for the object model, and the determined physical effects (134) on the structure, wherein the at least one processor is configured to enable the at least one GUI to be used to modify the object model and store data representative of the modified object model in at least one data store.

7. The system according to claim 6, further comprising a computer system including a memory (106), an application software component (104), the at least one processor, the at least one input device, and the display device, wherein the application software component is comprised of instructions that when included in the memory and executed by the at least one processor, cause the at least one processor responsive to inputs through the at least one input device to cause an object model to be loaded in the memory, meshed, and have finite element analysis performed thereon.

8. A method for generating meshes comprising:

through operation of at least one processor (102):

classifying a type (312, 508) of an input face (204, 310, 502) of a three dimensional (3D) object model (108) of a structure based at least in part on a number of loops (504, 506) included by the input face;

selecting based on the classified type of the input face, a multi-block decomposition algorithm (1006, 1008, 1010, 1012, 1014) from among a plurality of multi-block decomposition algorithms that the processor is configured to use;

using the selected multi-block decomposition algorithm to determine locations of a plurality of blocks (1506) across the input face; and

meshing each block to produce mesh data (126) defining a mesh (202) that divides the input face into a plurality of quadrilateral elements (210).

9. The method according to claim 8, further comprising through operation of the at least one processor:

generating a flattened two dimensional (2D) representation of the input face, wherein classifying the type of the input face is based on the 2D representation of the input face, wherein using the selected multi-block decomposition algorithm to determine locations of the plurality of blocks across the input face is carried out based on the 2D representation of the input face; and

wherein classifying the type of the input face for a plurality of input faces includes determining whether: at least one of the input faces includes a single loop or multiple loops; whether at least one of the input faces includes a cylindrical surface; whether at least one of the input faces includes a medial axis bead; and whether at least one of the input faces includes a medial axis annulus.

10. The method according to claim 9, further comprising carrying out selecting and using the multi-block decomposition algorithm for a plurality of input faces including selecting and using a Cartesian slab decomposition algorithm to subdivide at least one of the input faces having a cylindrical surface into a several parallel slab shaped blocks that are parallel to the axis of the cylindrical surface.

11. The method according to claim 9, further comprising selecting and using at least one medial axis decomposition algorithm including selecting and using at least one of:

a medial axis based bead decomposition algorithm to subdivide at least one of the input faces having a medial axis bead type input face classification into three blocks which include a rectangular block in the middle of the bead, flanked by two curved block caps;

a medial axis bead type decomposition algorithm to subdivide at least one of the input faces having a medial axis annulus type input face classification into the plurality of blocks which include a plurality of arched shaped blocks; and/or

a combined medial axis based and visibility based decomposition algorithm to subdivide at least one of the input faces having multi-loop type input face classification based on a determination that the at least one of the input faces is not a cylindrical type input face or a medial axial annuli type input face.

12. The method according to claim 11, wherein when the combined medial axis based and visibility based decomposition algorithm is selected, further comprising carrying out a visibility based decomposition portion of the algorithm , in which split lines are determined that subdivide the input face into blocks based on orthogonally, symmetry, and shortness of candidate split lines.

13. The method according to claim 9, further comprising selecting and using an Art Gallery decomposition algorithm to subdivide at least one of the input faces having a single loop type input face classification based on a determination that the at least one of the input faces is not a cylindrical type input face or a medial axial bead type input face.

14. The method according to any one of claims 8 to 13, wherein the mesh includes interior nodes (206), wherein the majority of interior nodes have a valency of four, and wherein the majority of quadrilateral elements have four included angles (208) that are between 80-100^{o}, further comprising through operation of the at least one processor:

determining physical effects on the structure based at least in part on the mesh data and data representative of loads applied to the structure provided through operation of at least one input device;

generating at least one graphical user interface (GUI) through a display device, which at least one GUI displays a visual representation of the object model, the mesh for the object model, and the determined physical effects on the structure;

responsive to further inputs through the at least one input device, modifying the object model and storing data representative of the modified object model in at least one data store.

15. A non-transitory computer readable medium (2726) encoded with executable instructions (2730) that when executed, cause at least one processor (102, 2702) to carry out the method according to any one of claims 8 to 14.

FIG. 1

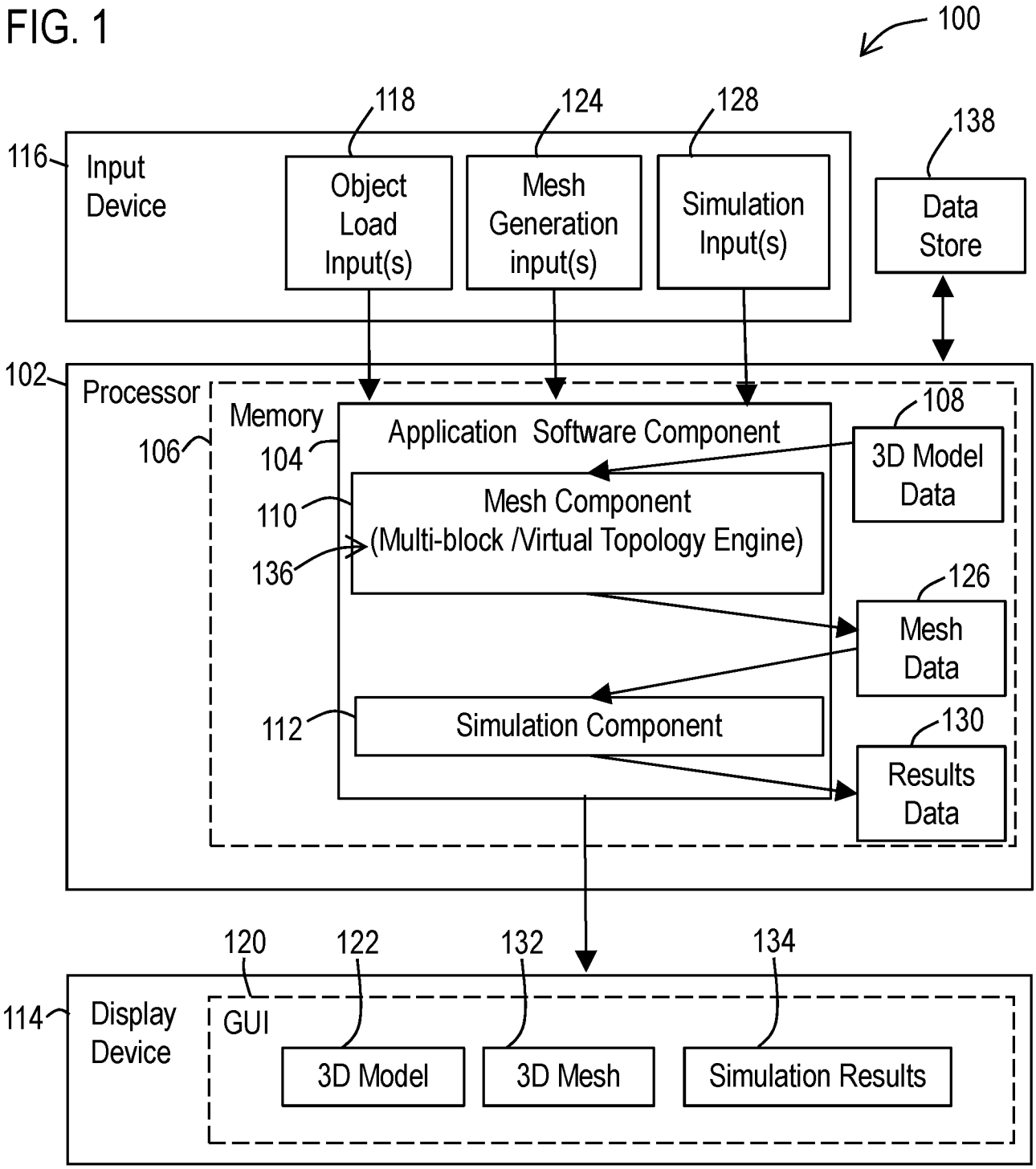

100
118
124
128
138
116
Input Device
Object Load Input(s)
Mesh Generation input(s)
Simulation Input(s)
Data Store
102
Processor
Memory
106
104
Application Software Component
108
3D Model Data
110
Mesh Component
(Multi-block /Virtual Topology Engine)
136
126
Mesh Data
112
Simulation Component
130
Results Data
120
122
132
134
114
Display Device
GUI
3D Model
3D Mesh
Simulation Results

FIG. 2

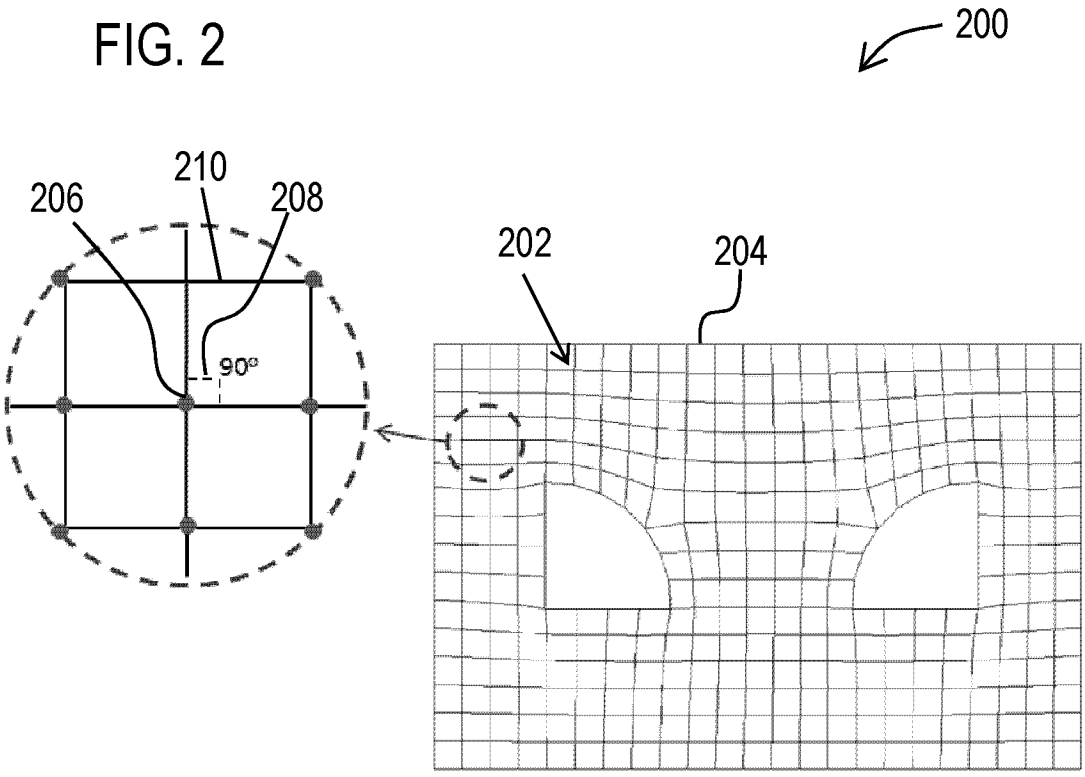


FIG. 3

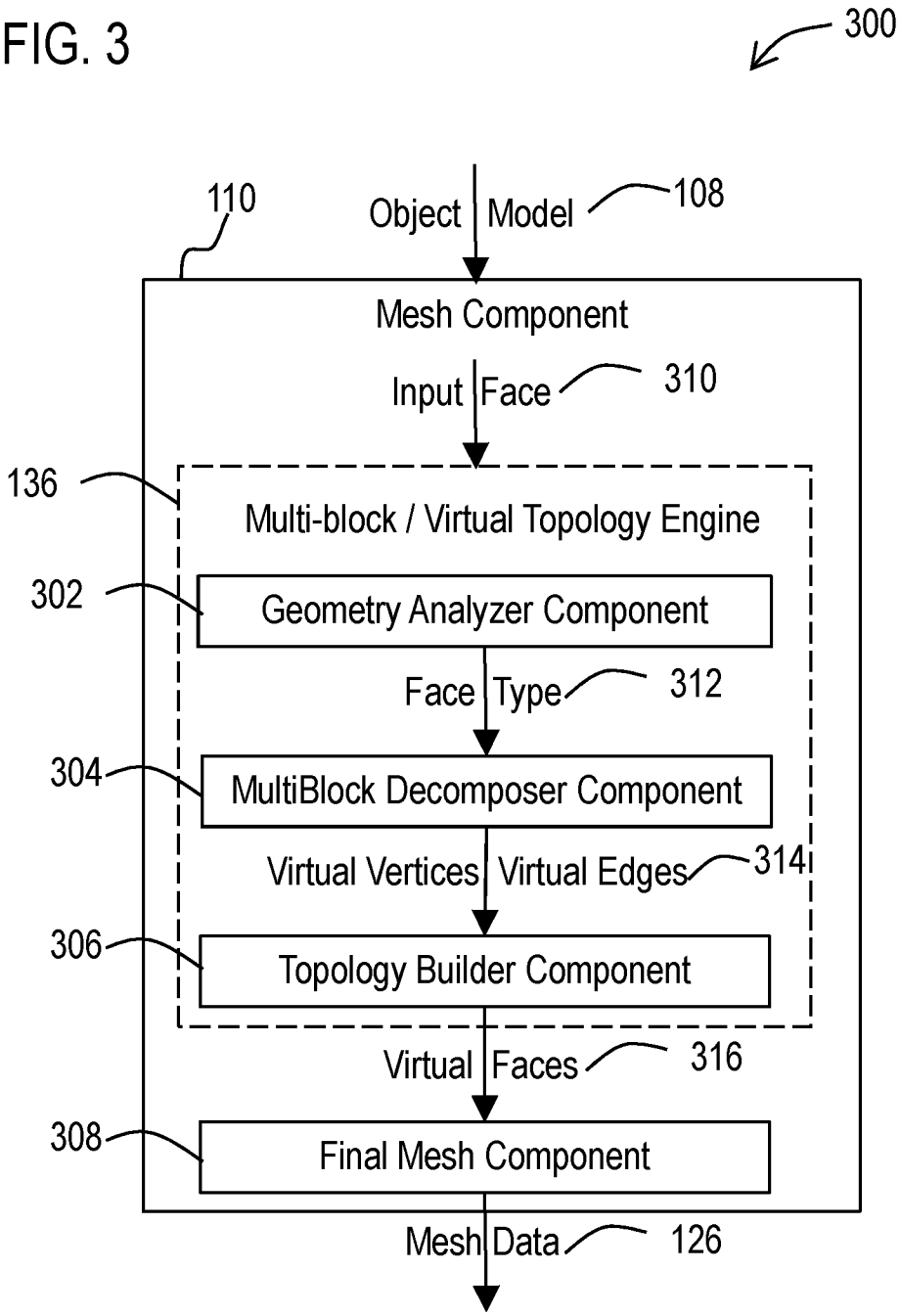

300
110
Object Model
108
Mesh Component
Input Face
310
136
Multi-block / Virtual Topology Engine
302
Geometry Analyzer Component
Face Type
312
304
MultiBlock Decomposer Component
Virtual Vertices
Virtual Edges
314
306
Topology Builder Component
Virtual Faces
316
308
Final Mesh Component
Mesh Data
126

FIG. 4

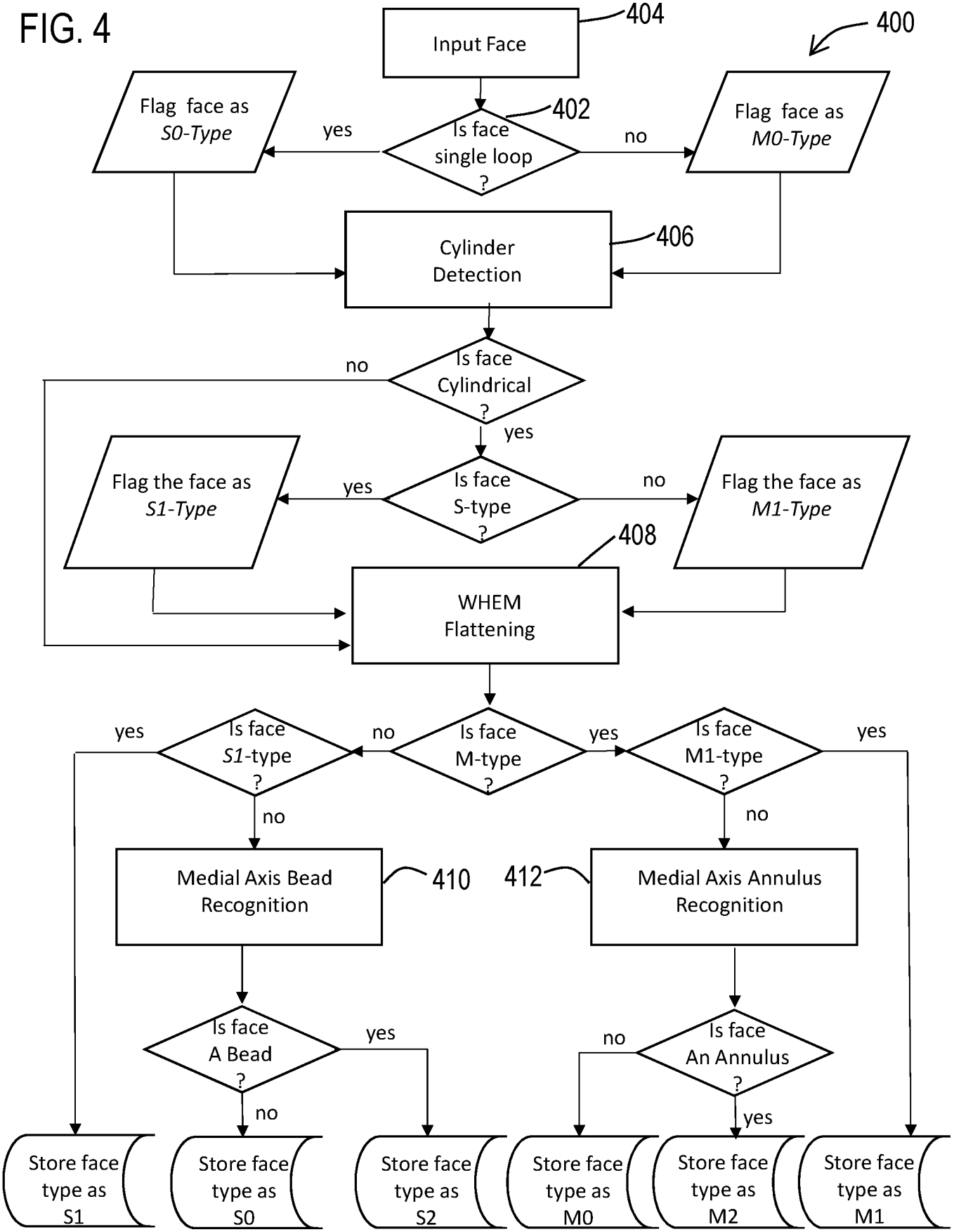

400
Input Face
404
Is face single loop ?
402
yes
Flag face as S0-Type
no
Flag face as M0-Type
Cylinder Detection
406
Is face Cylindrical ?
no
yes
Is face S-type ?
yes
Flag the face as S1-Type
no
Flag the face as M1-Type
WHEM Flattening
408
Is face S1-type ?
yes
Is face M-type ?
no
Is face M1-type ?
yes
no
no
Medial Axis Bead Recognition
410
412
Medial Axis Annulus Recognition
Is face A Bead ?
yes
no
Is face An Annulus ?
no
yes
Store face type as S1
Store face type as S0
Store face type as S2
Store face type as M0
Store face type as M2
Store face type as M1

FIG. 5

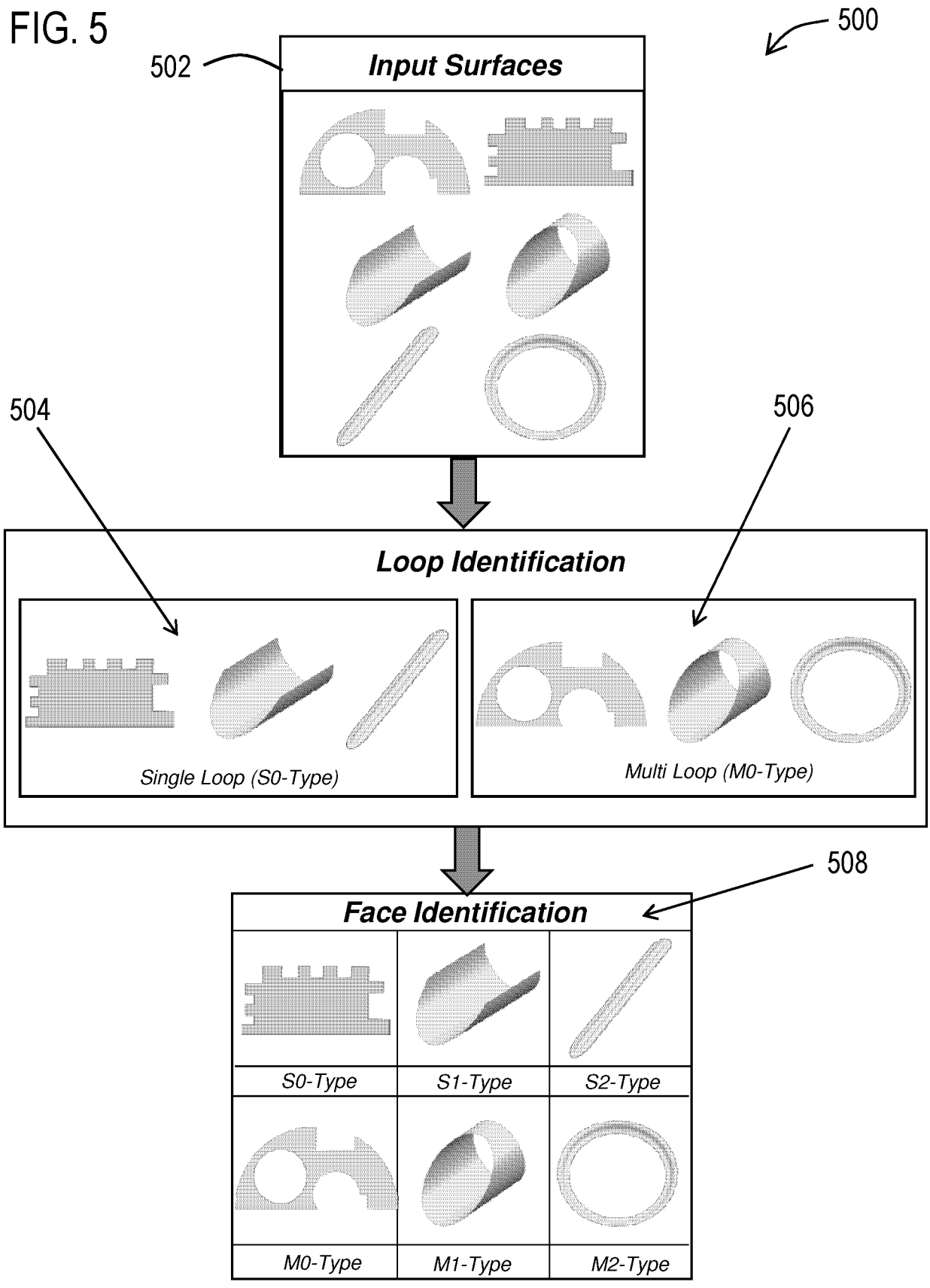

500
502
Input Surfaces
504
506
Loop Identification
Single Loop (S0-Type)
Multi Loop (M0-Type)
508
Face Identification
S0-Type
S1-Type
S2-Type
M0-Type
M1-Type
M2-Type

FIG. 6

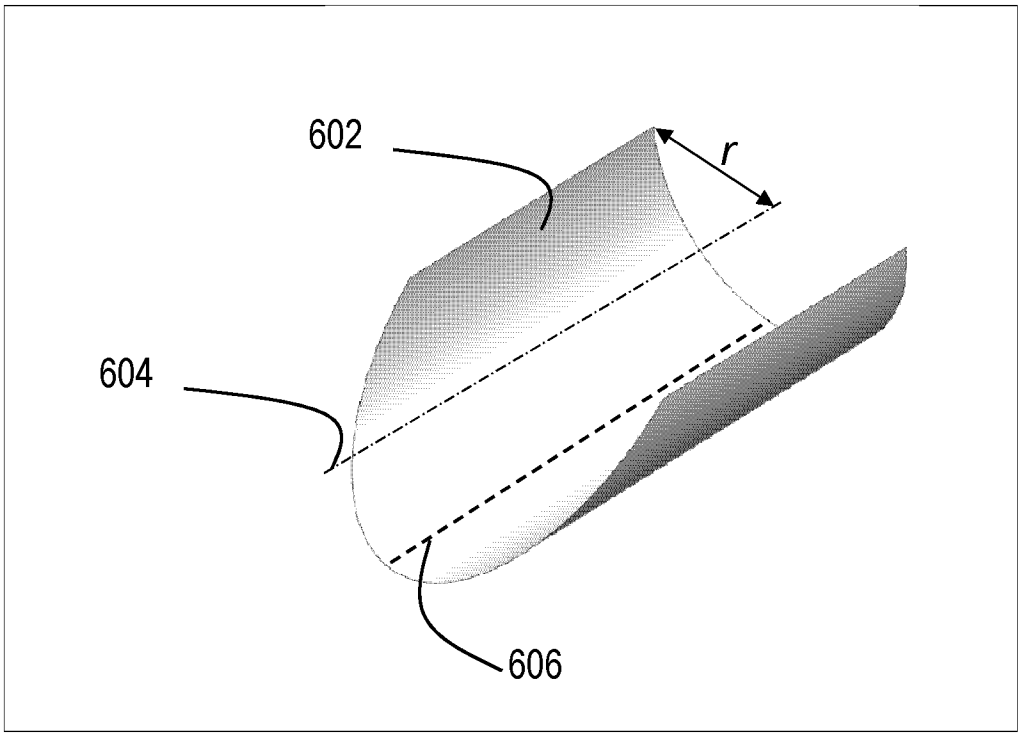


FIG. 7

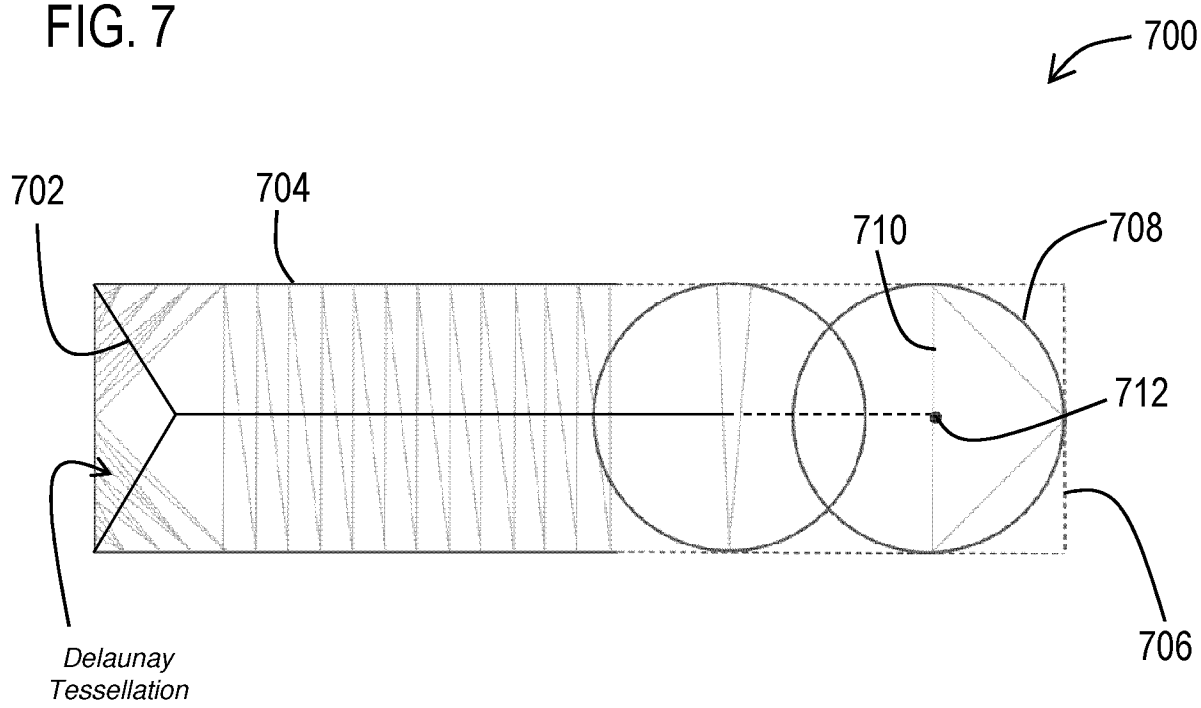

700
702
704
710
708
712
706
Delaunay
Tessellation

FIG. 8

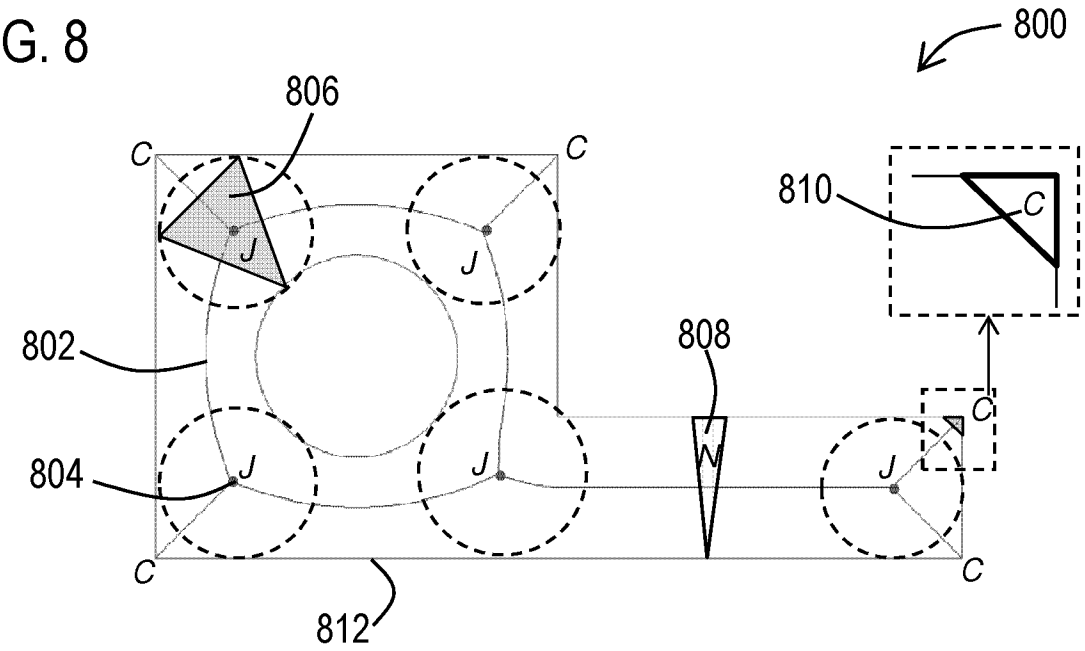

800
806
C
C
J
J
810
C
802
808
C
804
J
J
J
C
C
812

FIG. 9

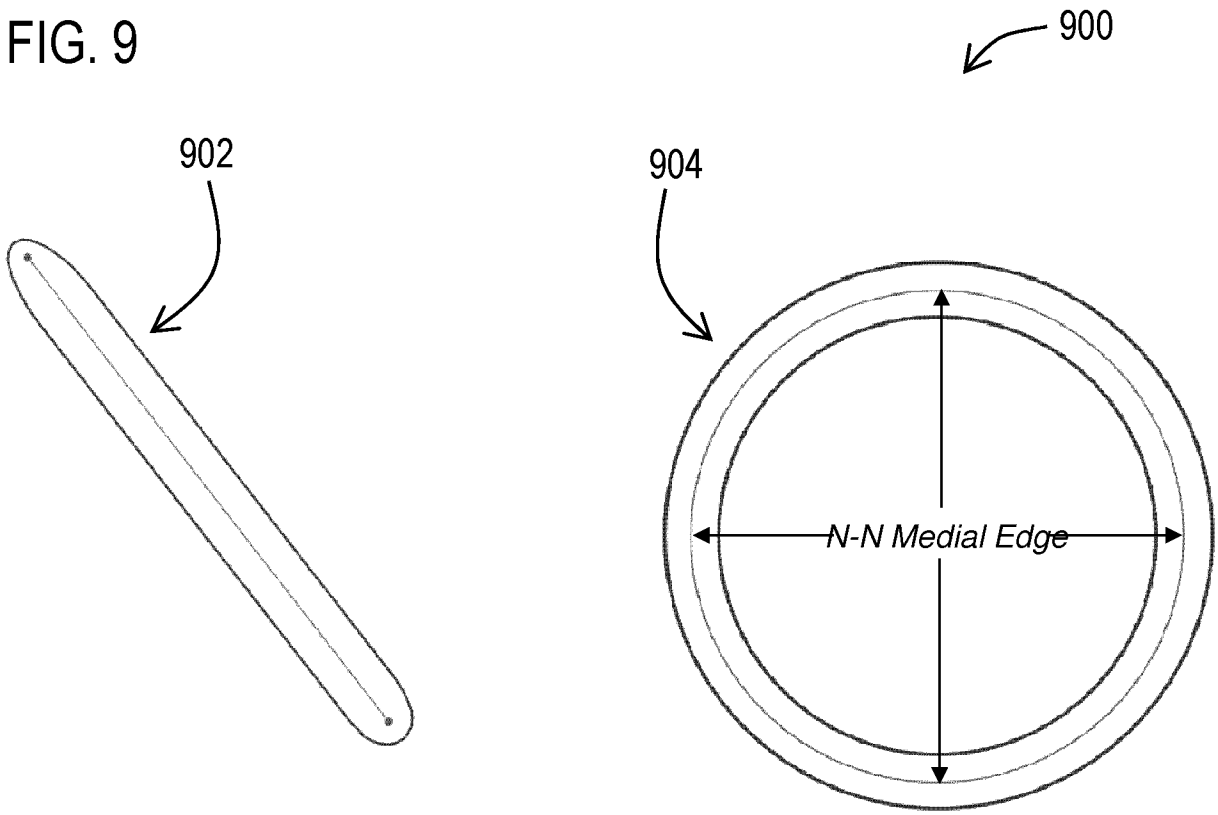


FIG. 10

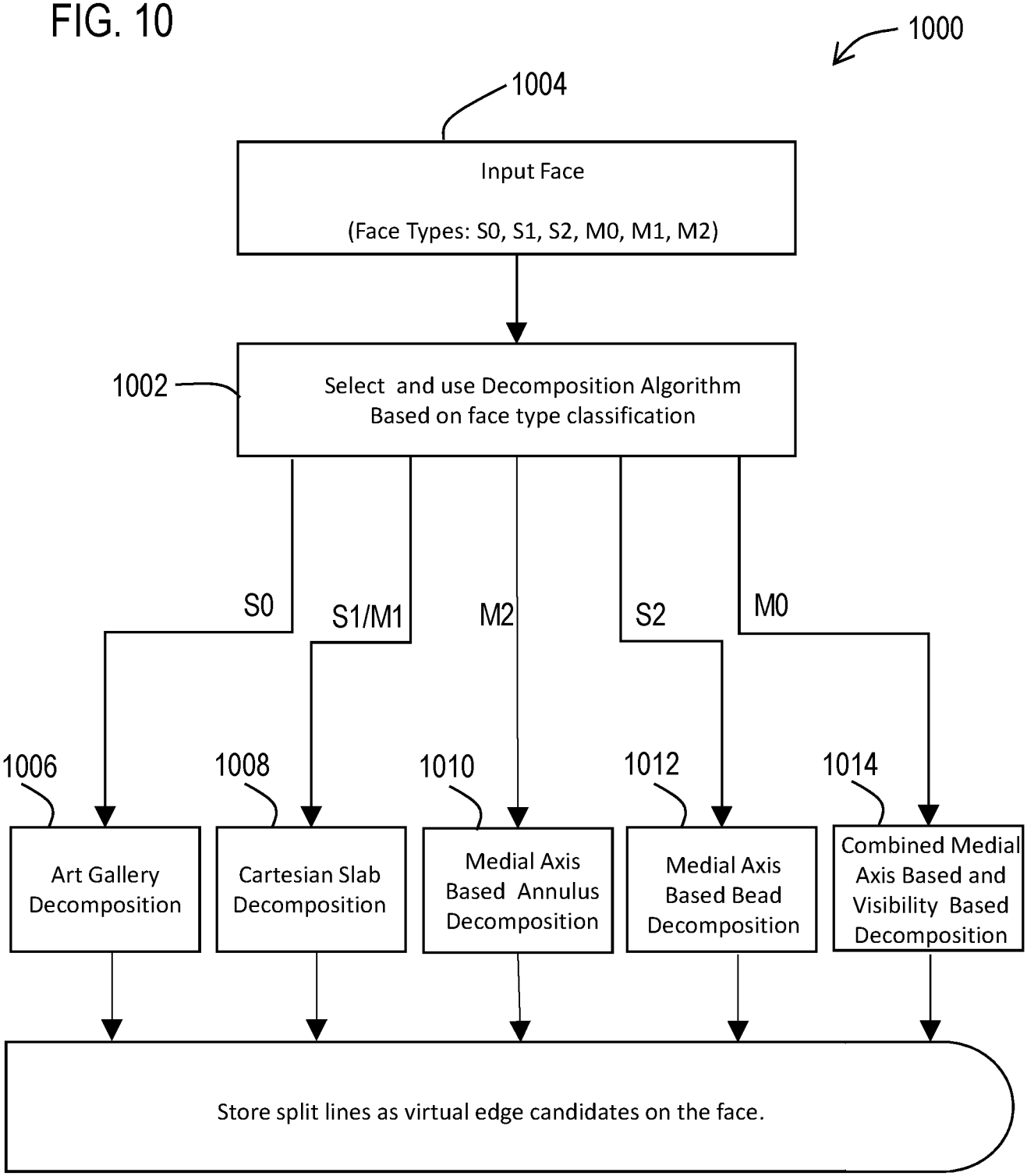

1000
1004
Input Face
(Face Types: S0, S1, S2, M0, M1, M2)
1002
Select and use Decomposition Algorithm
Based on face type classification
S0
S1/M1
M2
S2
M0
1006
1008
1010
1012
1014
Art Gallery Decomposition
Cartesian Slab Decomposition
Medial Axis Based Annulus Decomposition
Medial Axis Based Bead Decomposition
Combined Medial Axis Based and Visibility Based Decomposition
Store split lines as virtual edge candidates on the face.

FIG. 11

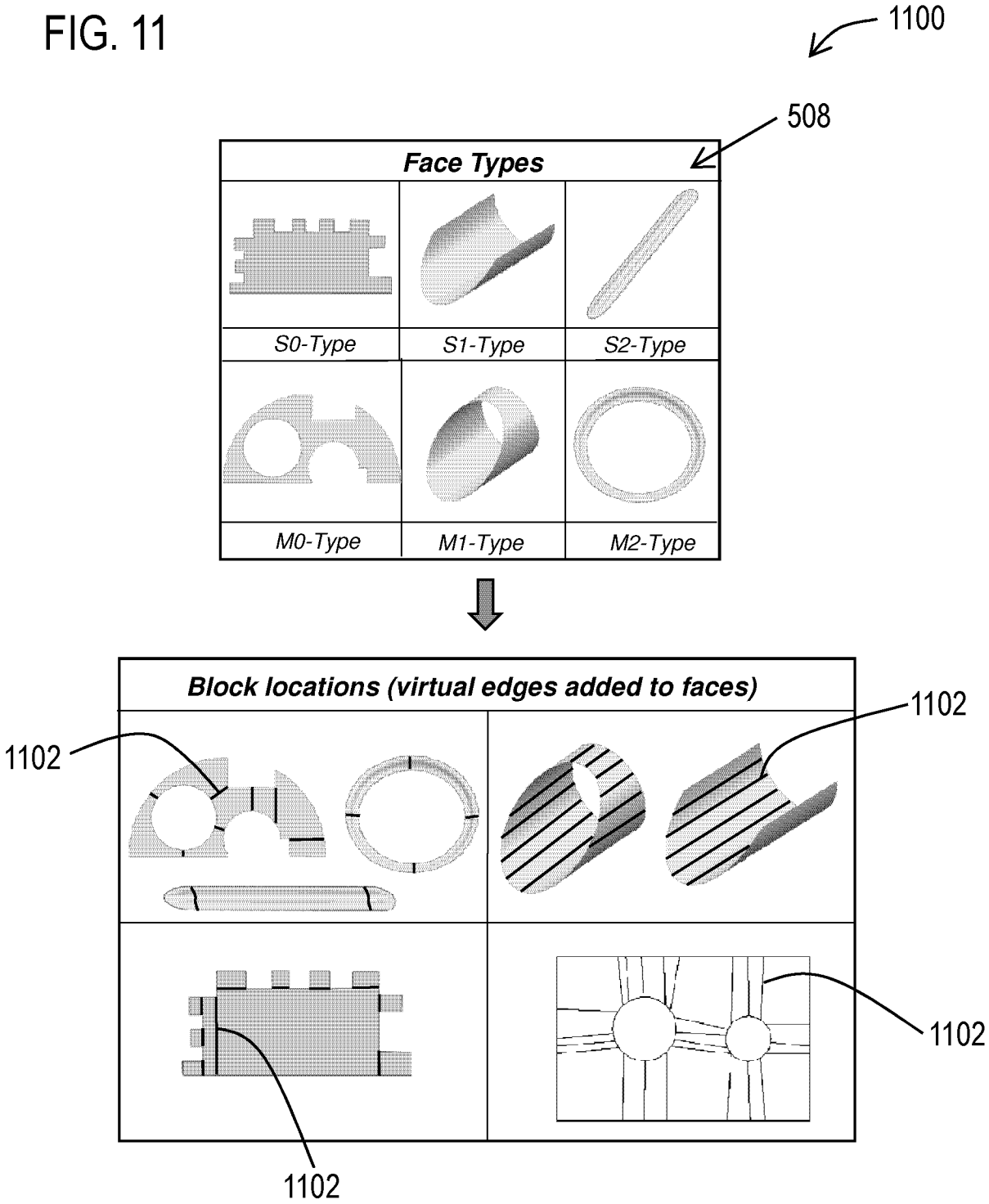

1100
508
Face Types
S0-Type
S1-Type
S2-Type
M0-Type
M1-Type
M2-Type
Block locations (virtual edges added to faces)
1102
1102
1102
1102

FIG. 12

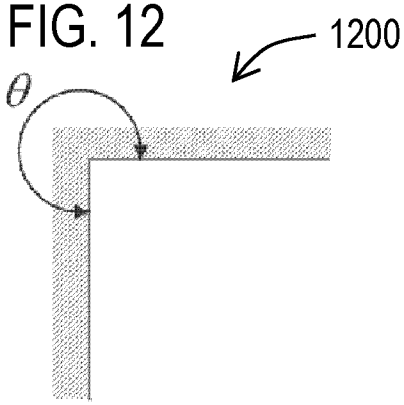

1200
θ

FIG. 13

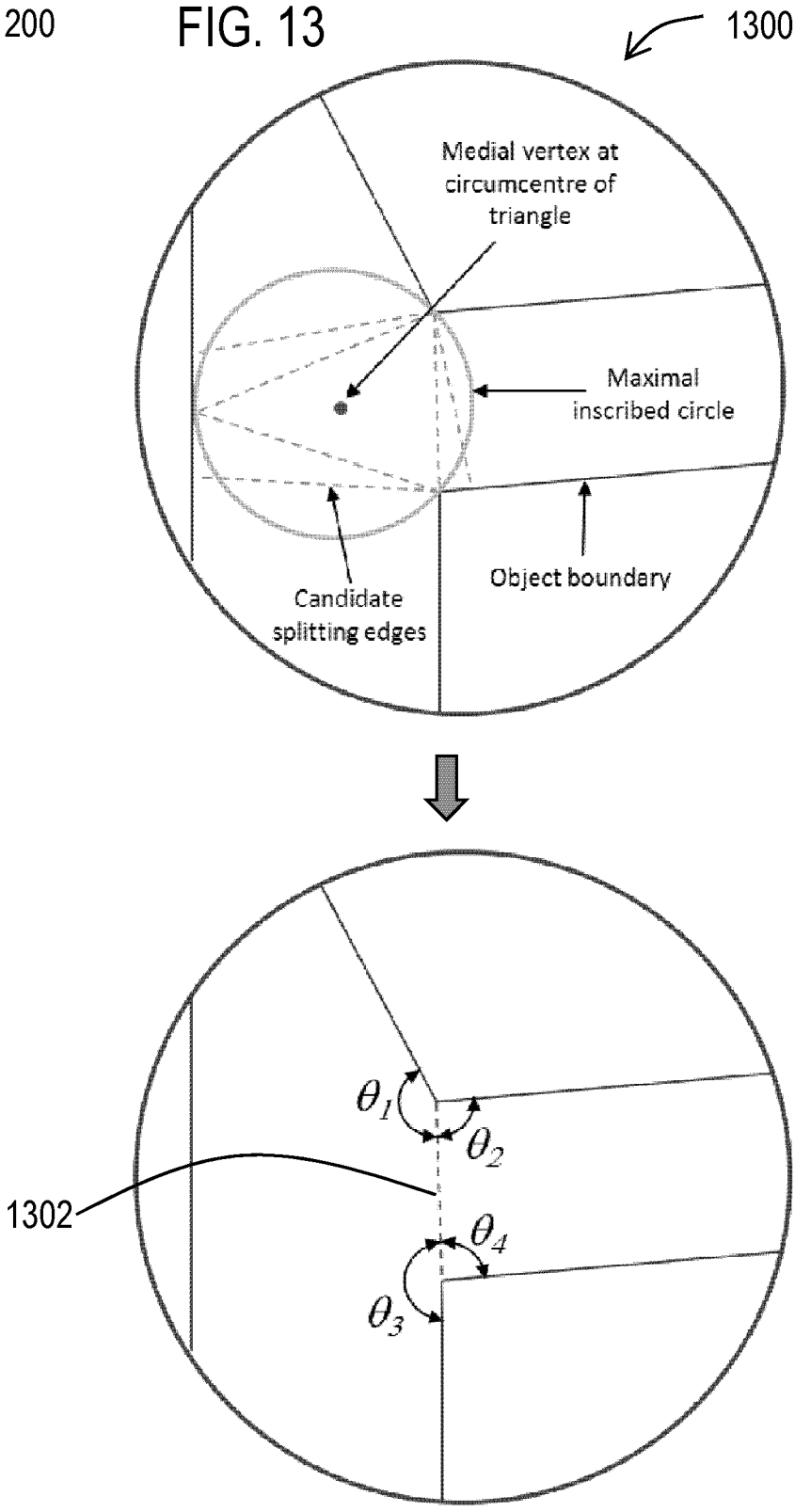

1300
Medial vertex at circumcentre of triangle
Maximal inscribed circle
Object boundary
Candidate splitting edges
θ1
θ2
θ3
θ4
1302

FIG. 14

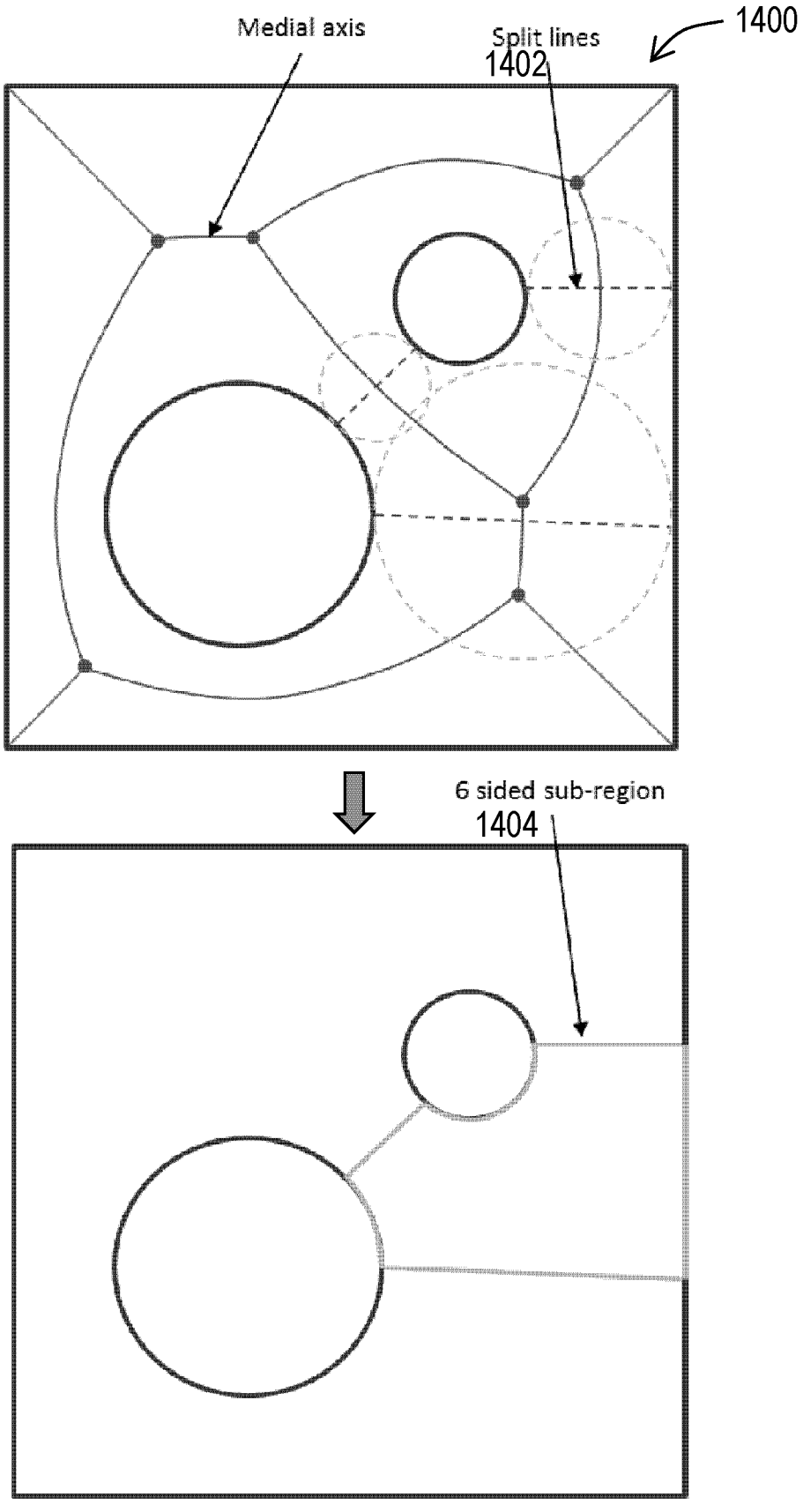


FIG. 15

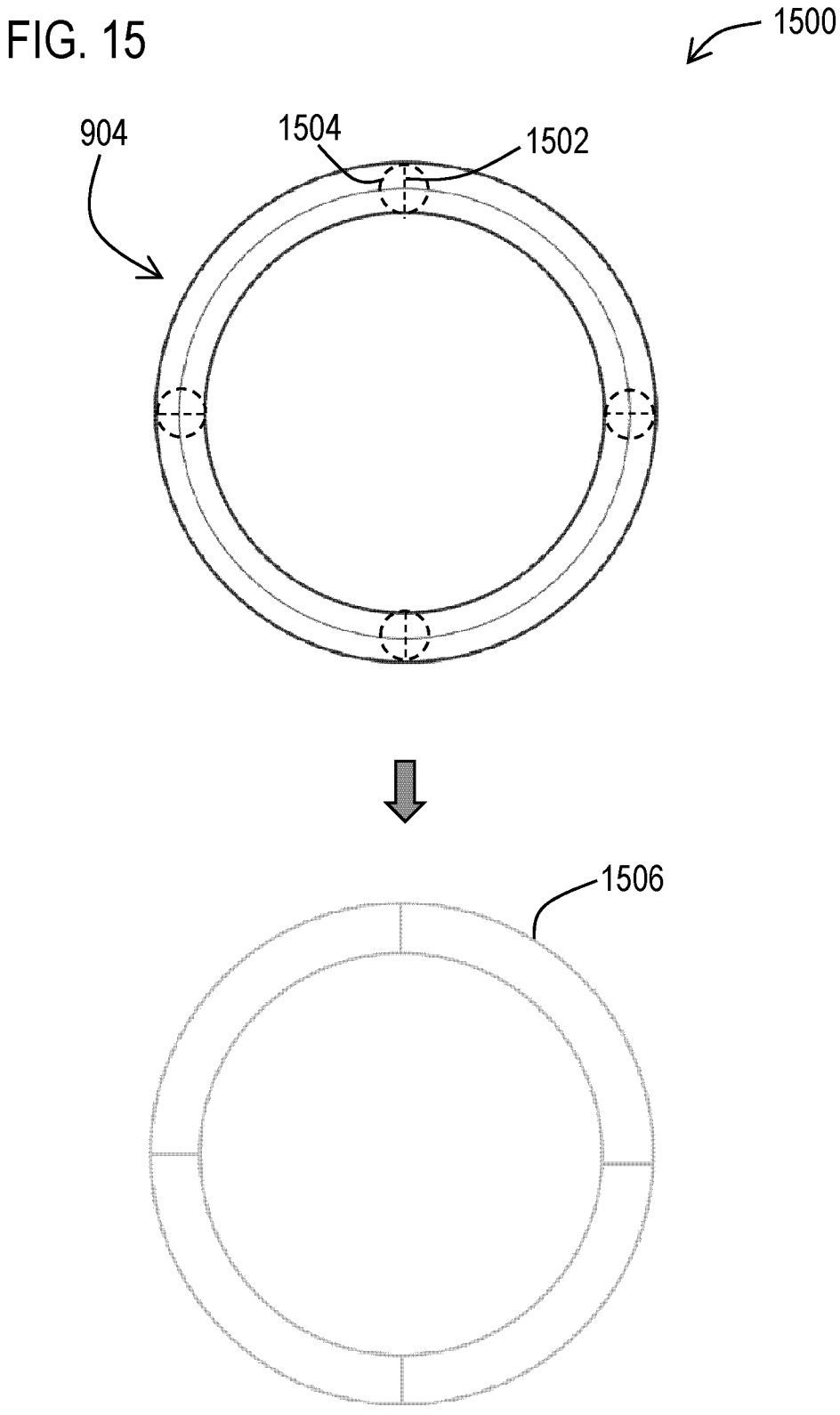


FIG. 16

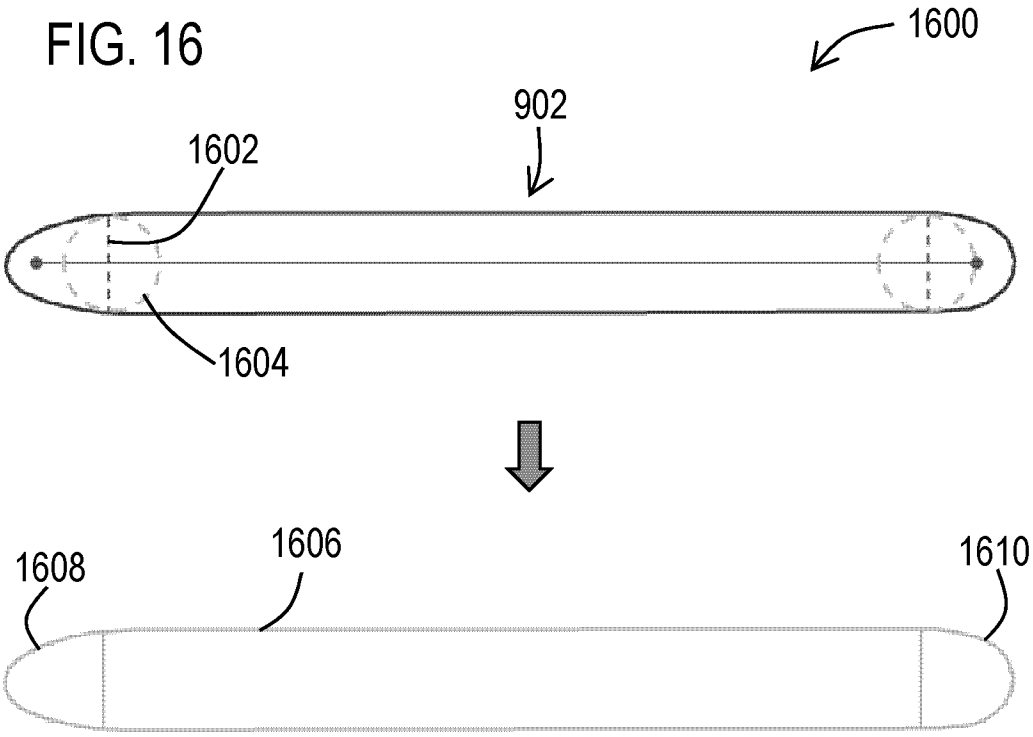


FIG. 17

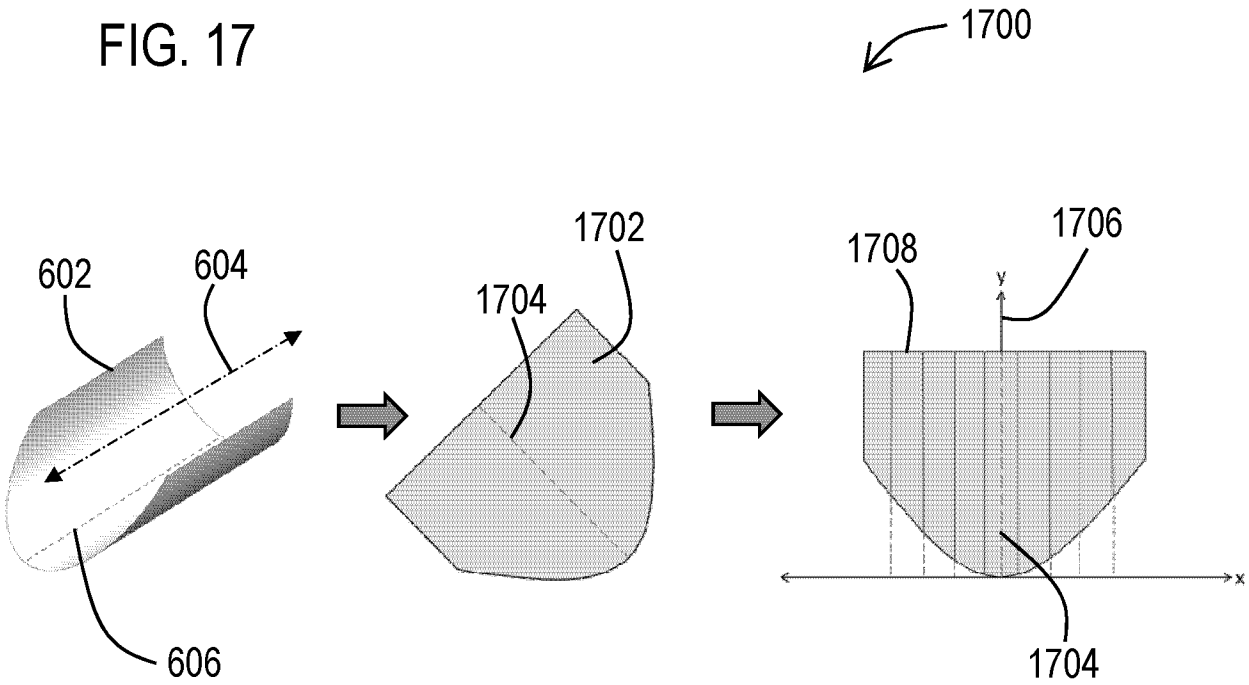


FIG. 18

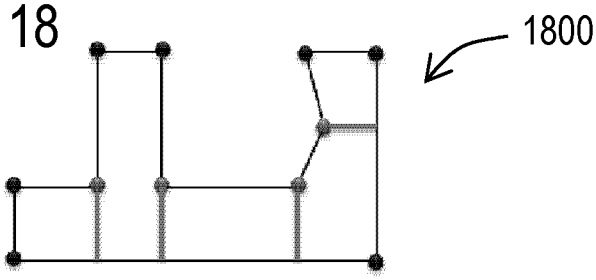


FIG. 19

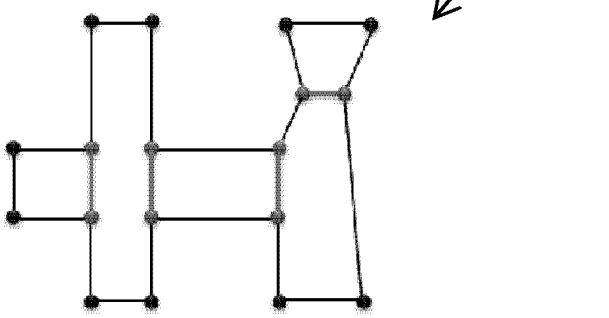


FIG. 20

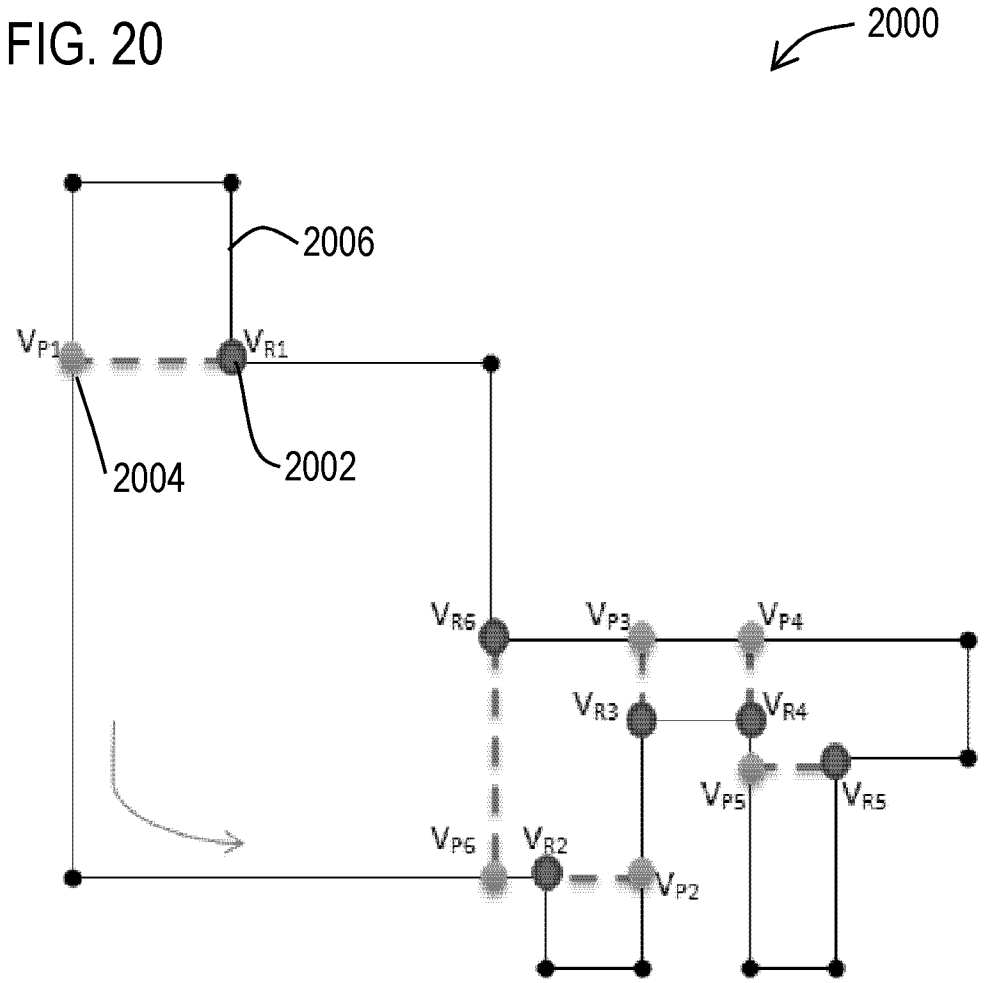

2000
2006
V_{P1}
V_{R1}
2004
2002
V_{R6}
V_{P3}
V_{P4}
V_{R3}
V_{R4}
V_{P5}
V_{R5}
V_{P6}
V_{R2}
V_{P2}

FIG. 21

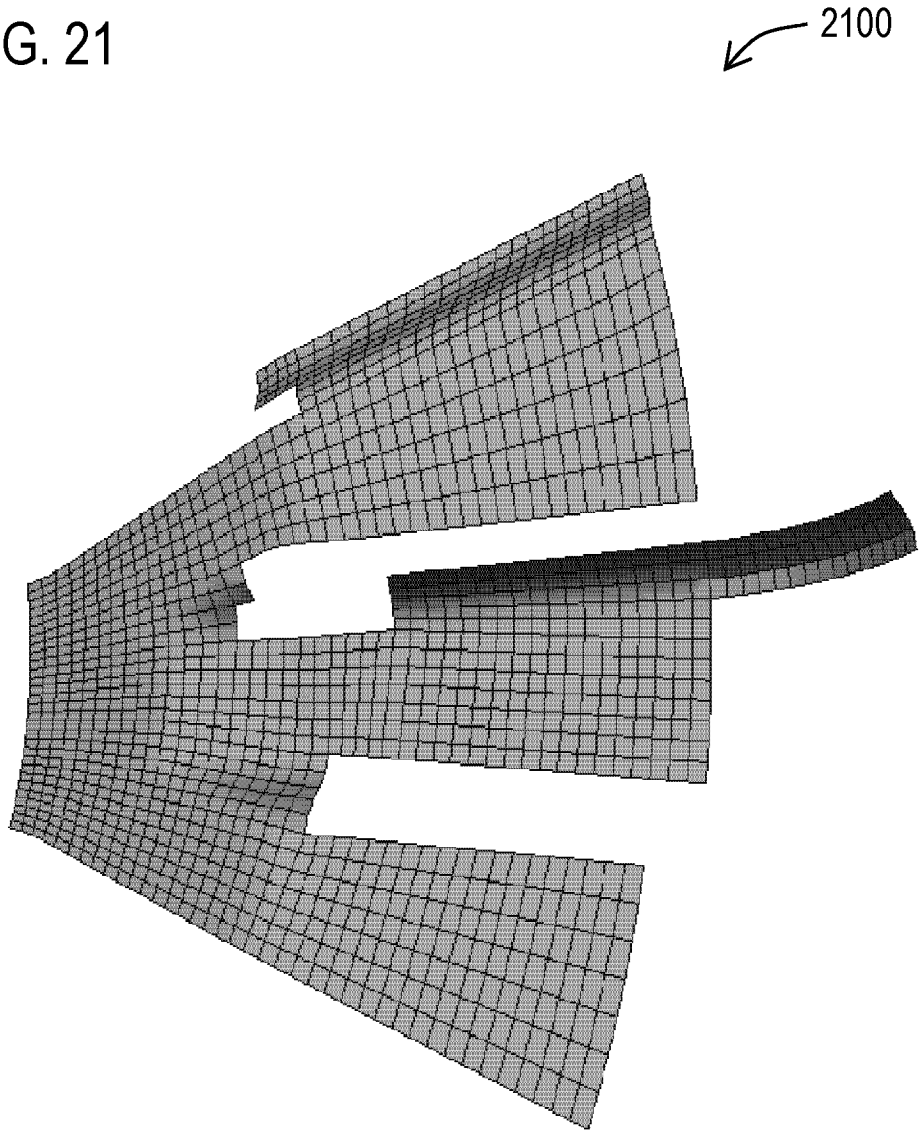


FIG. 22

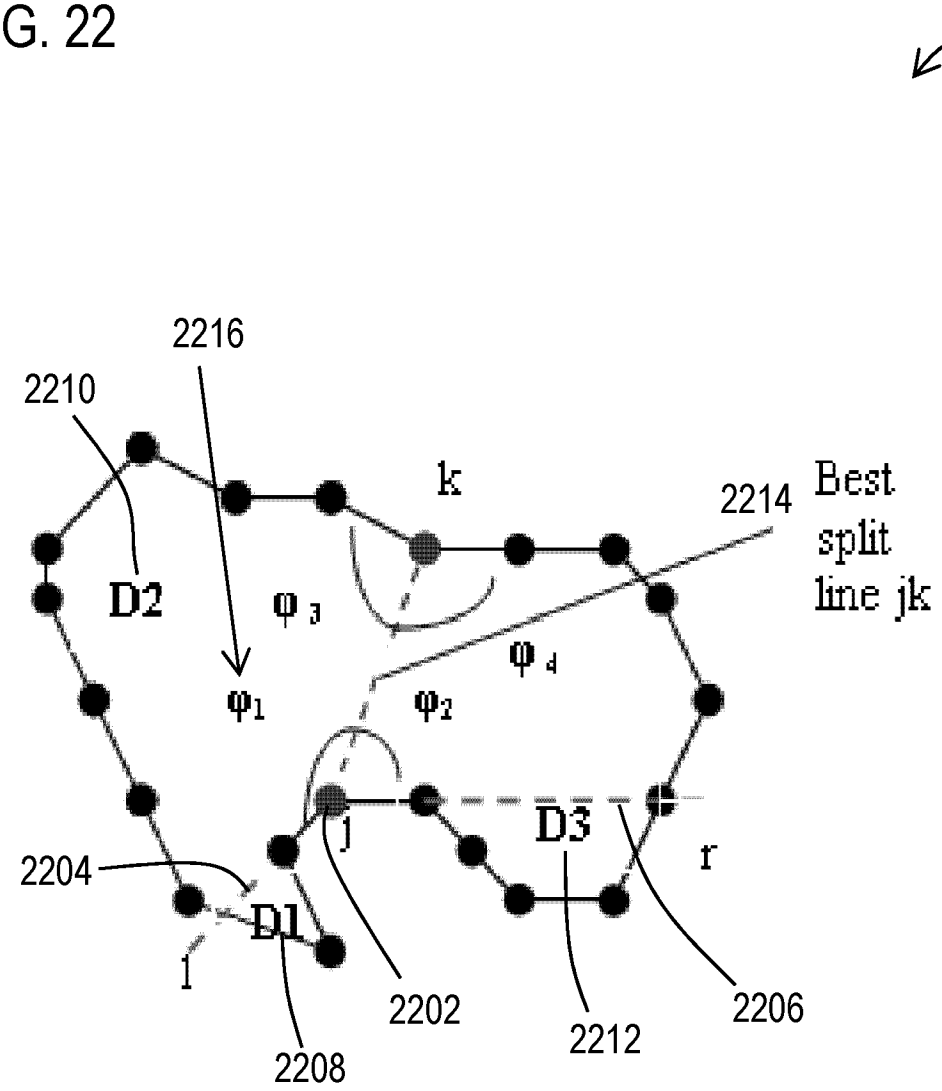

2200
2216
2210
k
2214
Best
split
line jk
D2
φ 3
φ 4
φ1
φ2
j
D3
r
2204
D1
l
2202
2212
2206
2208

FIG. 23

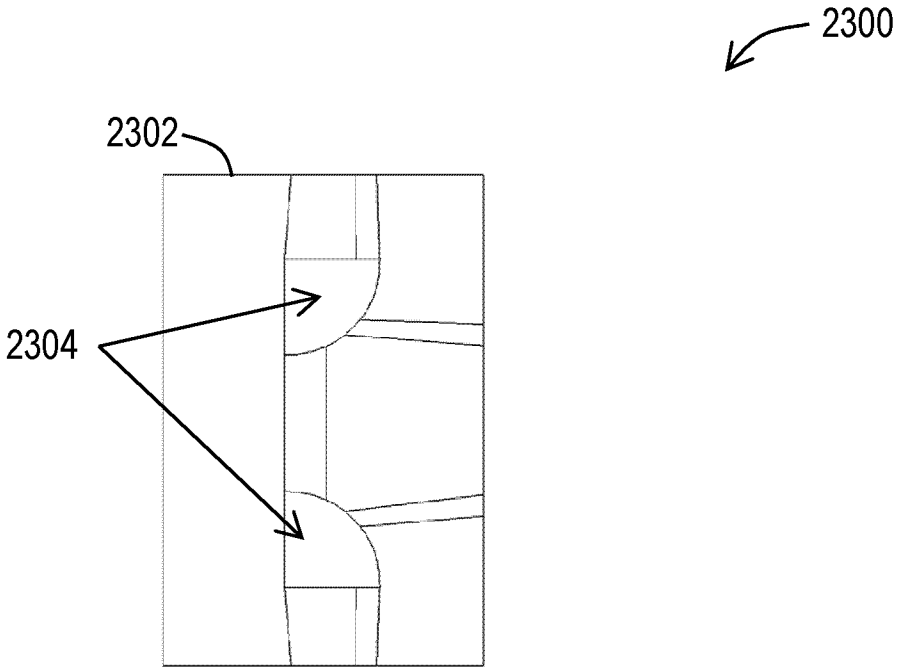


FIG. 24

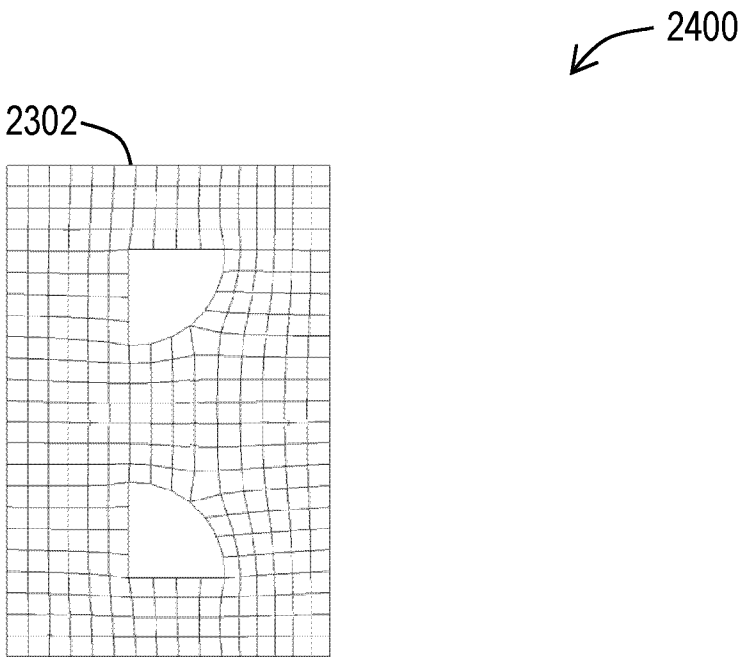


FIG. 25

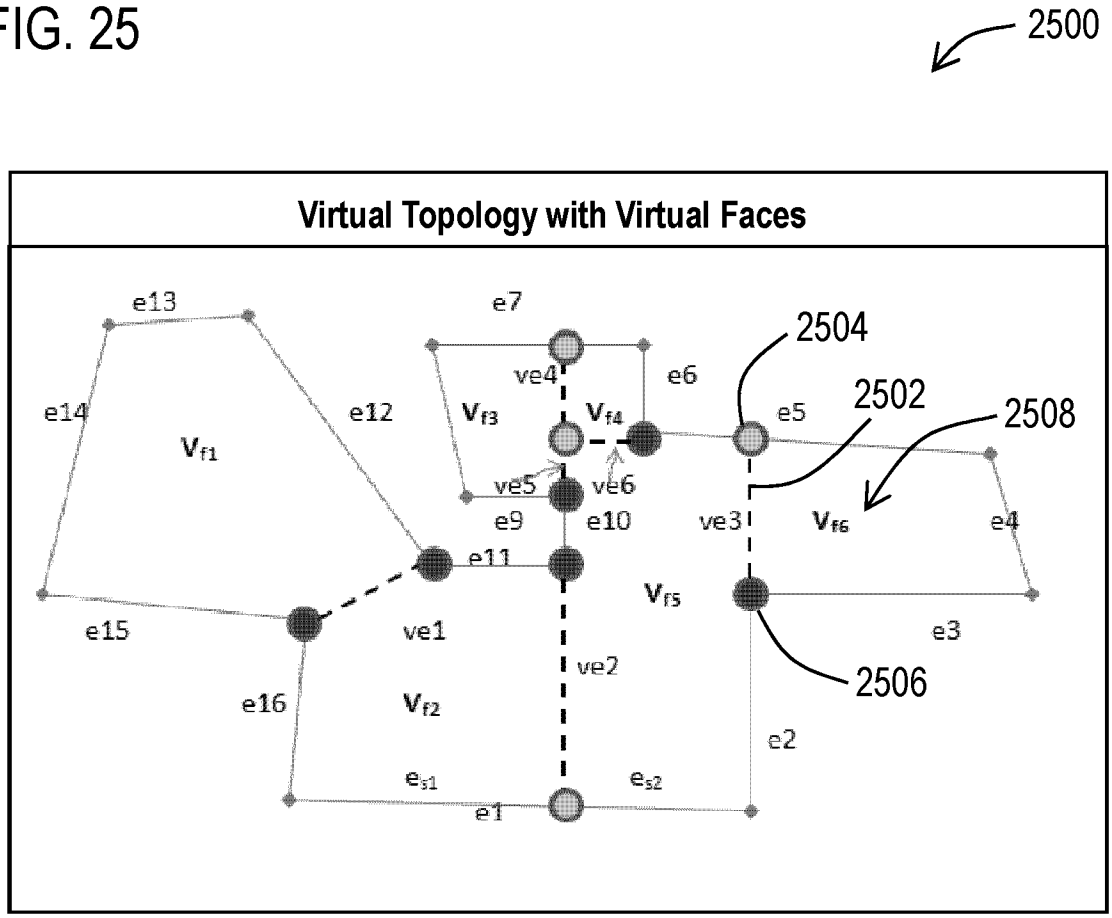

2500
Virtual Topology with Virtual Faces
e13
e7
2504
ve4
e6
e14
e12
Vf3
Vf4
e5
2502
2508
Vf1
ve5
ve6
e9
e10
ve3
Vf6
e4
e11
Vf5
e15
ve1
e3
ve2
2506
e16
Vf2
e2
es1
es2
e1

FIG. 26

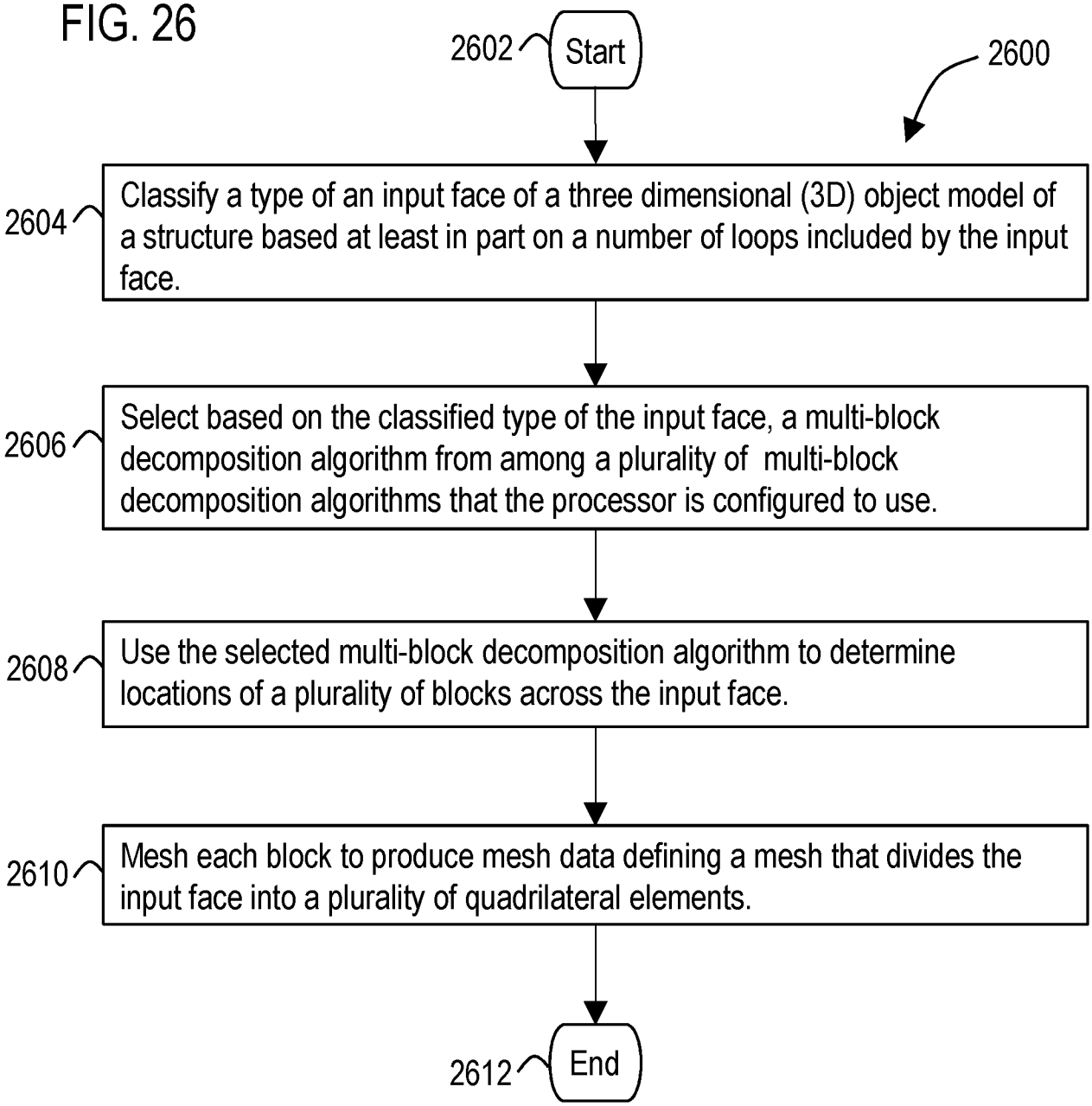
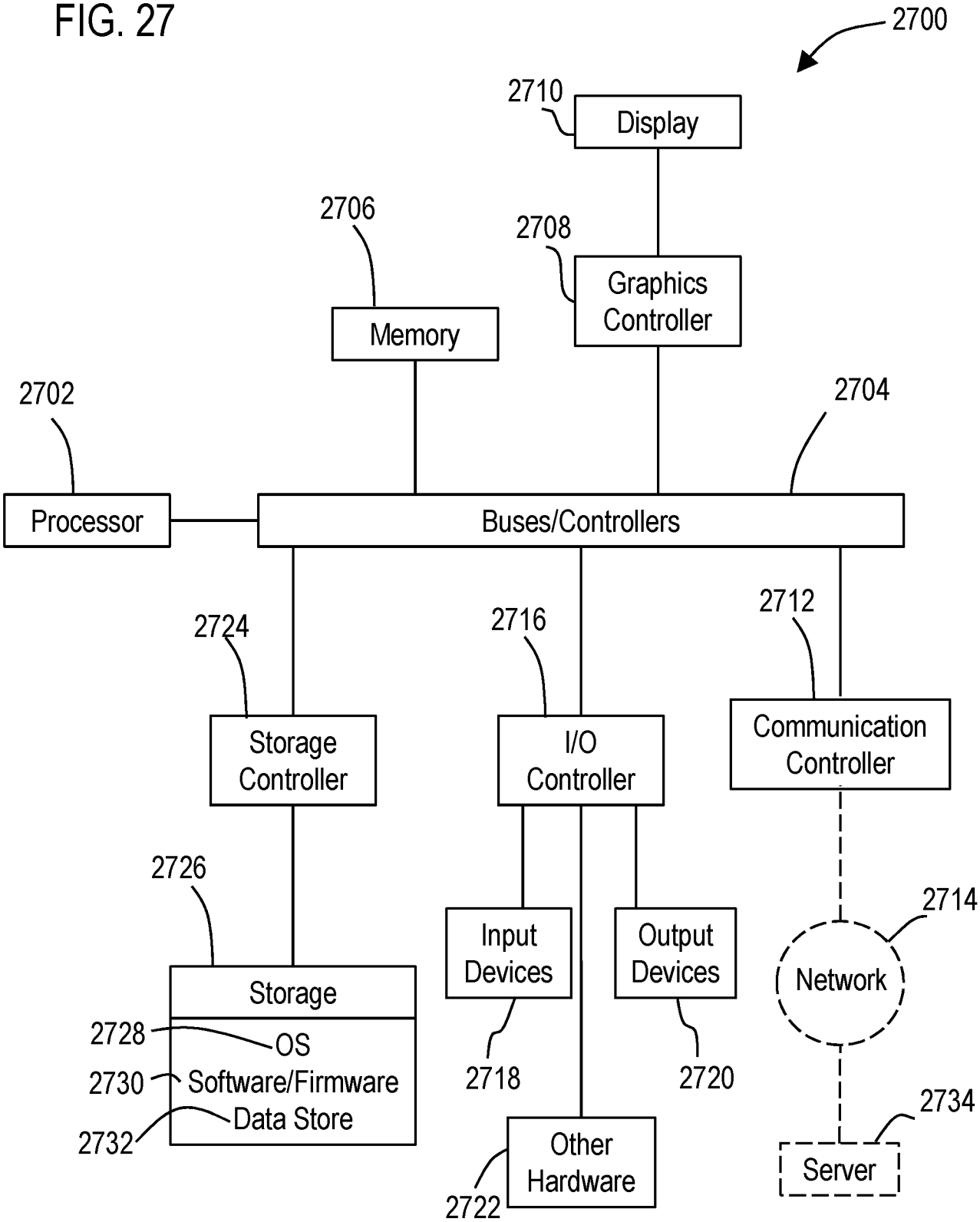

2602
Start
2600
2604
Classify a type of an input face of a three dimensional (3D) object model of a structure based at least in part on a number of loops included by the input face.
2606
Select based on the classified type of the input face, a multi-block decomposition algorithm from among a plurality of multi-block decomposition algorithms that the processor is configured to use.
2608
Use the selected multi-block decomposition algorithm to determine locations of a plurality of blocks across the input face.
2610
Mesh each block to produce mesh data defining a mesh that divides the input face into a plurality of quadrilateral elements.
2612
End

FIG. 27


2700
2710
Display
2706
2708
Graphics Controller
Memory
2702
2704
Processor
Buses/Controllers
2724
2716
2712
Storage Controller
I/O Controller
Communication Controller
2726
2714
Input Devices
Output Devices
Network
Storage
2728
OS
2730
Software/Firmware
2732
Data Store
2718
2720
2734
Other Hardware
Server
2722

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/046960

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06T17/10 G06F17/50 G06T17/20
ADD.

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)
G06T G06F

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)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	HUANG S H ET AL: "Automatic generation of quadrilateral multi-block topology for fea/cfd applications", INDUSTRIAL TECHNOLOGY, 2002. IEEE ICIT '02. 2002 IEEE INTERNATIONAL CO NFERENCE ON DEC. 11-14, 2002, PISCATAWAY, NJ, USA,IEEE, PISCATAWAY, NJ, USA, vol. 2, 11 December 2002 (2002-12-11), pages 1300-1305, XP010637248, DOI: 10.1109/ICIT.2002.1189365 ISBN: 978-0-7803-7657-1 abstract; figures 2,10 ----- -/--	1,2,8,9, 15

[X] 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	Date of mailing of the international search report
12 October 2016	19/10/2016

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	dos Santos, Luís

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2016/046960

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	Robert Schneiders: "Algortihms for Quadrilateral and Hexahedral Mesh Generation", , January 2000 (2000-01), pages 1-56, XP002762807, Retrieved from the Internet: URL:http://www.robertschneiders.de/papers/vki.pdf [retrieved on 2016-10-11] abstract figure 6 -----	1-15