Innovation, Sciences et Développement économique Canada
Office de la Propriété Intellectuelle du Canada

Innovation, Science and Economic Development Canada
Canadian Intellectual Property Office

CA 2984819 C 2023/10/10

(11)(21) 2 984 819

(12) BREVET CANADIEN
CANADIAN PATENT

(13) C

(86) Date de dépôt PCT/PCT Filing Date: 2016/05/11
(87) Date publication PCT/PCT Publication Date: 2016/11/17
(45) Date de délivrance/Issue Date: 2023/10/10
(85) Entrée phase nationale/National Entry: 2017/11/02
(86) N° demande PCT/PCT Application No.: AU 2016/050357
(87) N° publication PCT/PCT Publication No.: 2016/179659
(30) Priorité/Priority: 2015/05/11 (AU2015901712)

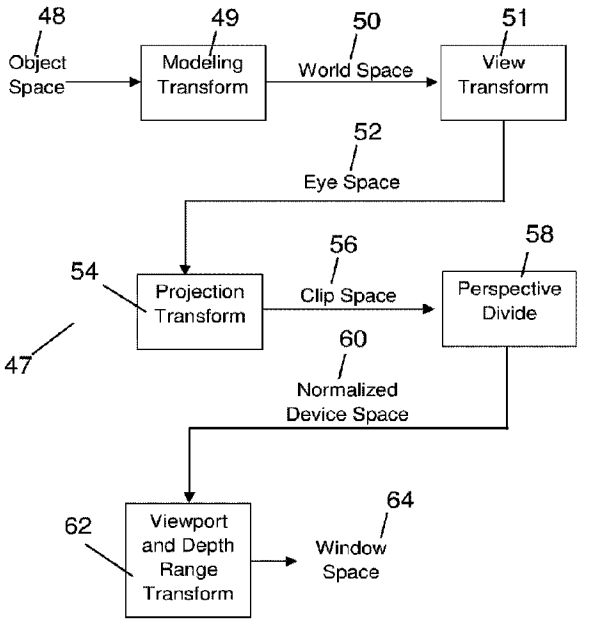
(51) Cl.Int./Int.Cl. *G06T 17/00* (2006.01),
G06T 15/30 (2011.01)

(72) Inventeur/Inventor:
DOUGLAS, SHANE, AU

(73) Propriétaire/Owner:
POINTERRA TECHNOLOGIES PTY LTD, AU

(74) Agent: RIDOUT & MAYBEE LLP

(54) Titre : PROCEDE ET SYSTEME DESTINES AU RENDU INFOGRAPHIQUE
(54) Title: METHOD AND SYSTEM FOR COMPUTER GRAPHICS RENDERING



(57) Abrégé/Abstract:
A method and system of computer graphics rendering implemented upon a processor in communication with a memory device storing a data comprises: loading a data structure representing at least a part of a scene in world space for display into the processor; a transform module transforming the data structure from world space to clip space; a dividing module subdividing the data structure in clip space so as to form child data structures; and a testing module testing which of the plurality of child data structures are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained. The plurality of remaining data structures are subsequently subdivided and tested until the plurality of remaining data structures do not overlap with the viewing frustum. A display module processing the plurality of remaining data structures for generation of an image on a display device.

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

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
17 November 2016 (17.11.2016)

WIPO | PCT

(10) International Publication Number
WO 2016/179659 A1

(51) International Patent Classification:
G06T 17/00 (2006.01) *G06T 15/30* (2011.01)

(21) International Application Number:
PCT/AU2016/050357

(22) International Filing Date:
11 May 2016 (11.05.2016)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2015901712 11 May 2015 (11.05.2015) AU

(71) Applicant: **POINTERRA TECHNOLOGIES PTY LTD** [AU/AU]; c/-, Westar Capital Limited, L45, 108 St Georges Tce, Perth, Western Australia 6000 (AU).

(72) Inventor: **DOUGLAS, Shane**; L45, 108 St Georges Tce, Perth, Western Australia 6000 (AU).

(74) Agent: **IP SENTINELS**; PO Box 1006, Bentley DC, Western Australia 6893 (AU).

(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))*

WO 2016/179659 A1

(54) Title: METHOD AND SYSTEM FOR COMPUTER GRAPHICS RENDERING

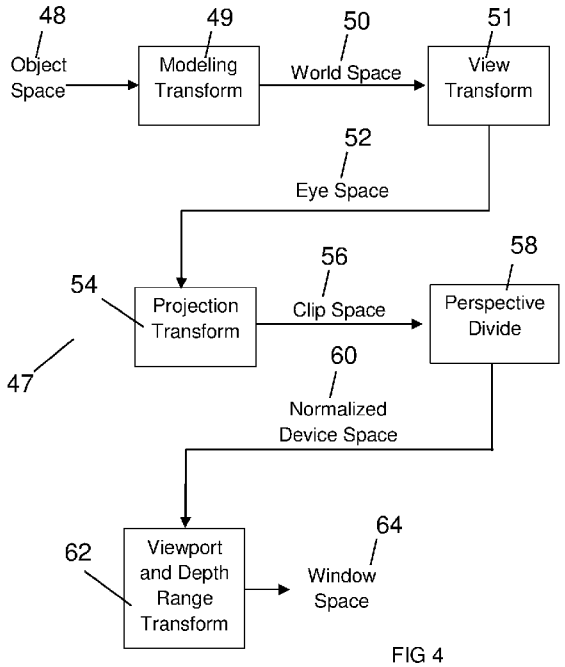


FIG 4

(57) Abstract: A method and system of computer graphics rendering implemented upon a processor in communication with a memory device storing a data comprises: loading a data structure representing at least a part of a scene in world space for display into the processor; a transform module transforming the data structure from world space to clip space; a dividing module subdividing the data structure in clip space so as to form child data structures; and a testing module testing which of the plurality of child data structures are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained. The plurality of remaining data structures are subsequently subdivided and tested until the plurality of remaining data structures do not overlap with the viewing frustum. A display module processing the plurality of remaining data structures for generation of an image on a display device.

Method and System for Computer Graphics Rendering

Field of the Invention

[0001] The present invention relates to a method for computer graphics rendering of data structures and a system for rendering computer graphics for display.

Background

[0002] Computer graphics rendering is the process of converting 2D or 3D models into a generated image for a display on a display screen. For the generation 3D computer graphics, an octree may be used for partitioning a three dimensional space. This partitioning is achieved by recursively subdividing the space into eight octants, to a point where further subdivision is redundant for a given resolution.

[0003] Conventionally, the calculation and processing of the octree is conducted by general-purpose computing on graphics processing units (GPGPU). In this situation, the octree data is loaded onto a GPGPU rendering pipeline for processing. As the processing of the octree is conducted by subdivision in world space with subsequent frustum culling in world space, this process becomes impractical for a large octree due to memory limitations of the graphics processing unit(s). In order to circumvent this issue, the part of the octree that can be seen in a viewing frustum is loaded on demand. Unfortunately with dynamic viewing, this requires constant updates of the loaded portion of the octree, which becomes impractical.

[0004] An alternative solution to the issue would be to use a plurality of central processing units to create an image of the octree and display it on the electronic display screen by bit block transfer. Normally the process of determining which nodes of the octree are within the viewing frustum and transforming their contents to create the image are computationally too expensive to accomplish this at any reasonable frame rate.

[0005] It would be advantageous if a computer graphics rendering method for displaying 3D computer graphics based on data structure (such as octree) calculations was provided that overcame the problems discussed above or which is at least a useful alternative to those methods that have been known.

Summary of Invention

[0006] According an aspect of the invention, there is provided a method of computer graphics rendering implemented upon a processor, the method comprising:

loading a data structure representing at least a part of a scene in world space for display into the processor;
transforming the data structure from world space into clip space;
subdividing the data structure in clip space so as to form child data structures, each child data structure comprising at least one node;
testing which of the plurality of child data structures are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained;
wherein the plurality of remaining data nodes are subsequently subdivided and tested until the plurality of remaining data nodes do not overlap with the viewing frustum, and
processing the plurality of remaining data nodes for generation of an image on a display device.

[0007] According to an embodiment of the invention, the world space data comprises a parent octree.

[0008] In an embodiment transforming the data structure from world space to clip space comprises transforming the parent octree node from world space to clip space.

[0009] In an embodiment subdividing the data structure in clip space so as to form child data structures comprises subdividing the parent octree node in clip space so as to form child octree nodes.

[0010] In an embodiment the plurality of remaining nodes are further repeatedly subdivided into a plurality of new child nodes and tested before the plurality of remaining nodes are processed for generation of an image on the electronic display device.

[0011] In an embodiment the plurality of remaining nodes are recursively subdivided and tested to be within the viewing frustum until a stop condition is met.

[0012] In an embodiment the stop condition is met when the size of a plurality of remaining nodes is too small to effectively display on the display screen.

[0013] In an embodiment the image is generated on the electronic display device by bit block transfer.

[0014] According to another aspect of the invention, there is provided a system of computer graphics rendering, the system comprising:
a processing unit in communication with a memory device storing a data, wherein the processing unit is arranged to control a display device, wherein the processing unit is arranged to create or load a data structure representing at least a part of a scene in world space for display;
a transform module for transforming the data structure from world space to clip space;
a dividing module for subdividing the data structure in clip space so as to form child data structures;
a testing module for testing which of the plurality of child data structures are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained;
wherein the plurality of remaining data structures are subsequently subdivided and tested until the plurality of remaining data structures do not overlap with the viewing frustum, and
a display module for processing the plurality of remaining data structures for generation of an image on a display device.

[0015] According to an embodiment of the invention, the division module is configured such that the plurality of remaining nodes are further subdivided into a plurality of new child nodes and the method repeated from the beginning before the plurality of remaining nodes are subsequently processed to generate an image on the electronic display device.

[0016] In an embodiment the division module is configured such that the plurality of remaining nodes are recursively subjected to subdivision and testing to be within the viewing frustum until the size of a plurality of remaining nodes are too small to effectively display on the electronic display screen.

[0017] According to a preferred embodiment of the invention, the system further comprises a bit block transfer module for transfer of the data structure representing the image to a GPU for generation of the image on the display device.

[0018] According to another aspect of the present invention there is provided a computer program comprising instructions for controlling a processor to perform the method described above.

[0019] Throughout the specification and claims, unless the context requires otherwise, the word “comprise” or variations such as “comprises” or “comprising”, will be understood to

imply the inclusion of a stated integer or group of integers but not the exclusion of any other integer or group of integers.

Description of Drawings

[0020] In order to provide a better understanding of the present invention embodiments will now be described by way of example only, with reference to the drawings, in which:-

[0021] Fig 1 illustrates a computer system for implementing a method according to an embodiment of the present invention;

[0022] Fig 2 illustrates a wireframe cube comprising a simple scene;

[0023] Fig 3 illustrates a subdivided octree of the cube of Fig 2;

[0024] Fig 4 illustrates a flowchart of a typical rendering pipeline;

[0025] Fig 4A is an alternative representation of the method of Figure 4;

[0026] Fig 5 illustrates a flowchart of the method according to an embodiment of the present invention;

[0027] Fig 6 illustrates a set of functional modules of a processor and data transfer between the modules; and

[0027a] Fig 7 is a flow chart of a method of computer graphics rendering according to an embodiment of the present invention.

Description of Embodiments

[0028] Fig 1 shows a block diagram of a typical computer system 10 for carrying out a method of computer graphics rendering according to an embodiment that will be described. The computer system 10 may be for example, a personal computer, a gaming console, a tablet, a smart phone or a dedicated computing system for example a medical imaging device.

[0029] In an embodiment the computer system 10 comprises a central processing unit based system for carrying out the method. The system 10 comprises a mother board 12 which is capable of powering and interfacing to at least one central processing unit 14. The central processing unit 14 may be configured to have a plurality of discrete processors or processors with multiple processing cores. Examples of the central processing unit are the Core series of processors manufactured by Intel Corp or the FX series of processors manufactured by Advanced Micro Devices Inc.

[0030] A storage drive 16 is interfaced to the motherboard 12. The storage drive 16 may comprise one or more typical storage drives, such as by way of example, hard disk drives or solid state drives. An operating system 18 is stored in the storage drive 16 in order to provide instructions to the computer system 10. The mother board 12 is also in

Date Reçue/Date Received 2023-02-28

communication with a random access memory 20 and a read only memory 22. The random access memory 20 is typically used as temporary storage and working space for the operating system. The read only memory 22 is typically used to store instructions for a Basic Input Output System which is accessed by the central processing unit 12 to prepare the computer system 10 to load the operating system 18. An optical device 36 may be interfaced to the mother board 12 to allow access to an optical storage device, such as a CD, DVD, or Blu-ray.

[0031] A graphics processor unit 24 is interfaced to the motherboard 12. The graphics processor unit 24 may also be arranged so as to be integrated to the mother board 12 or the central processing unit 14.

[0032] The mother board 12 may include a network interface module 32, such as by way of example, a local area network adaptor or a wireless network adaptor. The network interface module 32 allows the computer system 10 to connect to a computer network 34, such as by way of example, the internet. The computer program described below, or data for the computer program may be obtained from the computer network 34.

[0033] The computer system 10 is interfaced by a user by means of peripherals, such as by way of example, a keyboard 28, a mouse 21 and a display screen 26. The display screen 26 may be of any suitable type, such as by way of example, a liquid crystal display, a light emitting diode display or a cathode ray tube display. The graphics processor unit 24 controls the images displayed on the display screen 26. The computer system 10 is configured by modules, which may comprise sub-modules, which interact with and exchange data to perform the present invention. The modules may be implemented in hardware, that is as electronic circuits, or in the form of functional arrangements 100 implemented by instructions that control the CPU(s) and/or GPU(s). The control instructions are commonly referred to as computer programs. The computer programs are embodied in a non-transitory computer readable medium 38, such as a permanent optical, magnetic or other electronic storage medium.

[0034] A computer graphics software 40 is installed upon the computer system 10. The software 40 may be regarded as the data generator 102 in Figure 6. The computer graphics software 40 may be a part of an application that requires display of, for example, one or more objects on the display screen 26. The computer graphics software 40 contains or creates data structures for storing data in computer memory to define images and instructions so that the computer graphics software may alter the images in response

to inputs via the peripherals. For example the software application may be a computer game that displays objects that form a scene for the user to interact with according to the game, in a manner where a user input produces a dynamic response to the displayed objects and thus the scene.

[0035] The data structures are typically in the form of positional coordinates that require processing, such as in a render pipeline in order to transform the data from 3D coordinates into 2D graphics for displaying upon the electronic display screen 26. The processing is typically computationally large due to the calculations required to transform the 3D coordinates into 2D graphics that mimic perspective and object size according to distance from a selected viewpoint. In order to simplify processes and reduce the computational requirements of processing the data structures, a preferred data structure of an octree may be used. The octree is a simple equal distance binary recursive subdivision of space starting from an initial cube that contains all the data. An alternative data structure to an octree is a binary space positioning tree.

[0036] Fig 2 shows a wireframe cube 42 for rendering in order to be displayed on the electronic display screen 26. The data of the wireframe cube 42 is stored as an octree data structure. In this situation, the cube 42 would be considered as a node.

[0037] Fig 3 shows the cube 42 becoming a parent node by the subdivision into eight octants, wherein each octant is a child node 44. Additionally, each child node 44 may itself become a parent node by subdivision into eight octants, wherein each octant is a further child node 46. In this situation, the octree data structure may be recursive in nature as each node may be subdivided until a set parameter is met. Typically the process of determining the subdivision of octree coordinates is by averaging those of the parent node.

[0038] In order to determine what parts of an octree are to be included in a 2D graphic for displaying upon the display screen 26, the individual nodes 42, child nodes 44 and further child nodes 46 are subjected to testing by a testing module. The testing, also known as frustum culling, comprises having a viewing frustum, which defines the portion of the octree to be displayed upon the electronic display screen 26, project upon the octree. The testing module then determines whether the individual node 42 falls within the viewing frustum. If the individual node 42 falls outside of the viewing frustum, the individual node 42 is not displayed upon the electronic display screen 26 and is discarded. If the individual node 42 falls within the viewing frustum, the associated child nodes 44 will

undergo the test. This process is recursively repeated until a level is reached where the size of the subsequent children nodes reach a size parameter, such as a size that is too small to effect to the electronic display screen 26, or a size that is at or below a desired resolution.

[0039] Fig 4 shows a typical rendering pipeline flowchart 47. The rendering pipeline flowchart 47 displays the data coordinates and the transformations required to progress from the original data structure coordinates to a final display coordinate for displaying to the display screen 26. The process involves having a set of coordinates in object space 48 undergoing a modelling transformation 49 to have the coordinates converted into a set of coordinates in world space 50. The model coordinates are represented by 4 dimensional homogeneous coordinates (*x, y, z, w*) of vertex positions, with w usually being 1 for object coordinates within the model view and projection matrices being linear transformations. The set of coordinates in world space 50 undergo a view transformation 51 to have the coordinates converted into a set of coordinates in eye space 52. The set of coordinates in eye space 52 undergo a projection transformation 54, which may be either perspective or orthogonal, to have the coordinates converted into a set of coordinates in clip space 56. The set of coordinates in clip space 56 undergo a perspective division 58 by its *w* to have the coordinates converted into a set of coordinates in normalised device space 60. There may be implementation specific variations of the transformation to clip space according to, for example, whether OpenGL or Direct3D is used. The transformation from clip space to normalised device coordinates (NDC) in normalised device space 60 is non-linear. The set of coordinates in normalised device space 60 undergo a viewport and depth range transformation 62 to have the coordinates converted into a set of coordinates in window space 64 which may then be displayed on the display screen 26.

[0040] It is generally known that the portion in which data structures, such as the octree mentioned above, are subjected to the testing is when the data structure is found as a set of coordinates in world space 50. After determining which nodes are to be displayed on the electronic display screen 26, the nodes undergo the multitude of transformations until they are transformed into the set of coordinates in window space 64 in order to be displayed on the electronic display screen 26. It is known that the testing is computationally intensive and the whole process is typically shunted to the rendering pipeline located on the graphics processing unit 24. Furthermore, it is possible to have the central processing unit 14 assist the graphics processing unit 24 by running a parallel rendering pipeline.

[0041] In the present invention the method of computer graphics rendering differs from known techniques, in that the testing is conducted when the coordinates in world space 50 have been transformed into coordinates in clip space 56. Fig 7 shows a flowchart wherein the individual node in world space 42A undergoes a view and projection transformation 66 conducted by a transform module 106 to result with the individual node in clip space 42B. The individual node 42 undergoes a subdivision process 68 of averaging the corners of a plurality of child nodes in clip space 70 by a dividing module 110 prior to being tested by the testing module 116. The testing involves a first step 72 wherein the viewing frustum is projected upon the plurality of child nodes in clip space 70 and child nodes not overlapping with the viewing frustum are discarded, producing a view frustum with remaining child nodes 74. The plurality of child nodes that overlap with the viewing frustum undergo a further subdivision and test 76 to produce a plurality of further child nodes. If the plurality of further child nodes 78 still retains nodes that are outside of (i.e. not fully within) the viewing frustum, these further child nodes undergo further subdivision and processing until there are no longer any overlap of nodes with the viewing frustum, producing a plurality of culled nodes and retained nodes 78. The plurality of retained nodes 78 undergo subsequent transformations in a display module 122 for display on the display screen 26. Because the NDC represents the view frustum as a 3 dimensional cube from (-1, -1, -1) to (1, 1, 1) and is obtained by dividing each clip coordinate by its w it then becomes apparent that frustum culling in clip space is simply done by constraining each coordinate to the range $-w$ to $+w$.

[0042] The plurality of retained nodes 78 may be subjected to a recursive loop 82 of further subdivision and testing, until a required depth or resolution is reached. This produces images more quickly for display on the display screen 26 in comparison to known techniques.

[0043] Additionally, the plurality of retained nodes 78 may be displayed on the display screen 26 by bit block transfer.

[0044] The combined transformation from object to clip coordinates is linear once the top node of the octree is transformed to clip space. The clip space coordinates of the child nodes of the octree can be obtained using the same averaging procedure used in object space. Now having the octree coordinates in clip space these can be used and the simple comparisons for view frustum culling in clip space can be performed to efficiently traverse the octree and obtain the data required to produce an image to "blit" to the screen.

Date Reçue/Date Received 2023-02-28

Additionally, clip space in the *x* and *y* directions map directly to the horizontal and vertical window axes in a proportional manner. Hence it is possible to map clip space *x, y* to window space pixels and hence determine when the dimensions of the node are smaller than a pixel and too small to effect the display so as to determine the stop condition of the recursive subdivision of nodes.

[0045] The invention advantageously provides a method of computer graphics rendering in which the testing is reduced in complexity so that it is computationally relaxed compared to the typical method. As such, the method of computer graphics rendering allows the central processing unit 14 to conduct the transformation of positional coordinates via the rendering pipeline in a manner that does not require the graphics processing unit 24 to be actively employed. Furthermore, due to improved efficiency the method of computer graphics rendering allows for the octree structures to be displayed on the electronic display screen 26 at real time frame rates.

[0046] Modifications may be made to the present invention within the context of that described and shown in the drawings. Such modifications are intended to form part of the invention described in this specification.

Claims

1. A method of computer graphics rendering implemented upon a processor, the method comprising:

loading a data structure representing at least a part of a scene in world space for display into the processor;

transforming the data structure from world space into clip space;

recursively:

(i) subdividing the transformed data structure in clip space so as to form a plurality of child data structures, each child data structure comprising at least one node;

(ii) testing which of the plurality of child data structures in each recursion are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained;

wherein the plurality of remaining data nodes become parent nodes of the transformed data structure for further subdivision in another recursion until the plurality of remaining data nodes do not overlap with the viewing frustum,

and

processing the plurality of remaining data nodes for generation of an image on a display device.

2. The method as claimed in claim 1, wherein the world space data comprises a parent octree node.

3. The method as claimed in claim 2, wherein transforming the data structure from world space to clip space comprises transforming the parent octree node from world space to clip space.

4. The method as claimed in claim 3, wherein subdividing the data structure in clip space so as to form child data structures comprises subdividing the parent octree node in clip space so as to form child octree nodes.

5. The method as claimed in claim 1, wherein the recursion ends when the size of a plurality of remaining nodes is too small to effectively display on the display screen.

Date Reçue/Date Received 2022-09-23

6. The method as claimed in claim 1, wherein the image is generated on the electronic display device by bit block transfer.

7. The method as claimed in claim 1, wherein the loaded data that is transformed includes data structures that reference at least one object that is outside of the viewing frustum.

8. The method as claimed in claim 1, wherein the transforming step is not preceded by a step that culls objects that are outside of the viewing frustum.

9. A system of computer graphics rendering, the system comprising:
a processing unit in communication with a memory device storing a data, wherein the processing unit is arranged to control a display device, wherein the processing unit is arranged to create or load a data structure representing at least a part of a scene in world space for display;
a transform module for transforming the data structure from world space to clip space;
a dividing and testing module for recursively:
(i) subdividing the data structure in clip space so as to form child data structures;
(ii) testing which of the plurality of child data structures are within a viewing frustum, so that child data structures outside of the viewing frustum are discarded and a plurality of remaining nodes are retained;
wherein the plurality of remaining data nodes become parent nodes of the transformed data structure for further subdivision in another recursion until the plurality of remaining data structures do not overlap with the viewing frustum, and
a display module for processing the plurality of remaining data structures for generation of an image on a display device.

10. The system as claimed in claim 9, wherein the processing unit is configured to create or load the data structure in world space so that the data structure comprises a parent octree; the transform module is configured to transform nodes of the parent octree from world space to clip space; and the dividing and testing module is configured to subdivide the transformed nodes of the parent octree in clip space so as to form eight child octree nodes.

Date Reçue/Date Received 2022-09-23

11. The system as claimed in claim 9, wherein the dividing and testing module is configured such that the recursion is stopped when a stop condition is net, wherein the stop condition occurs when the size of a plurality of remaining nodes are too small to effectively display on the electronic display screen.

12. The system as claimed in claim 9, wherein the system further comprises a bit block transfer module for transfer of the data structure representing the image to a GPU for generation of the image on the display device.

13. A non-transitory computer-readable medium storing computer program instructions for controlling a processor to perform the method as claimed in any one of claims 1 to 8.

Date Reçue/Date Received 2022-09-23

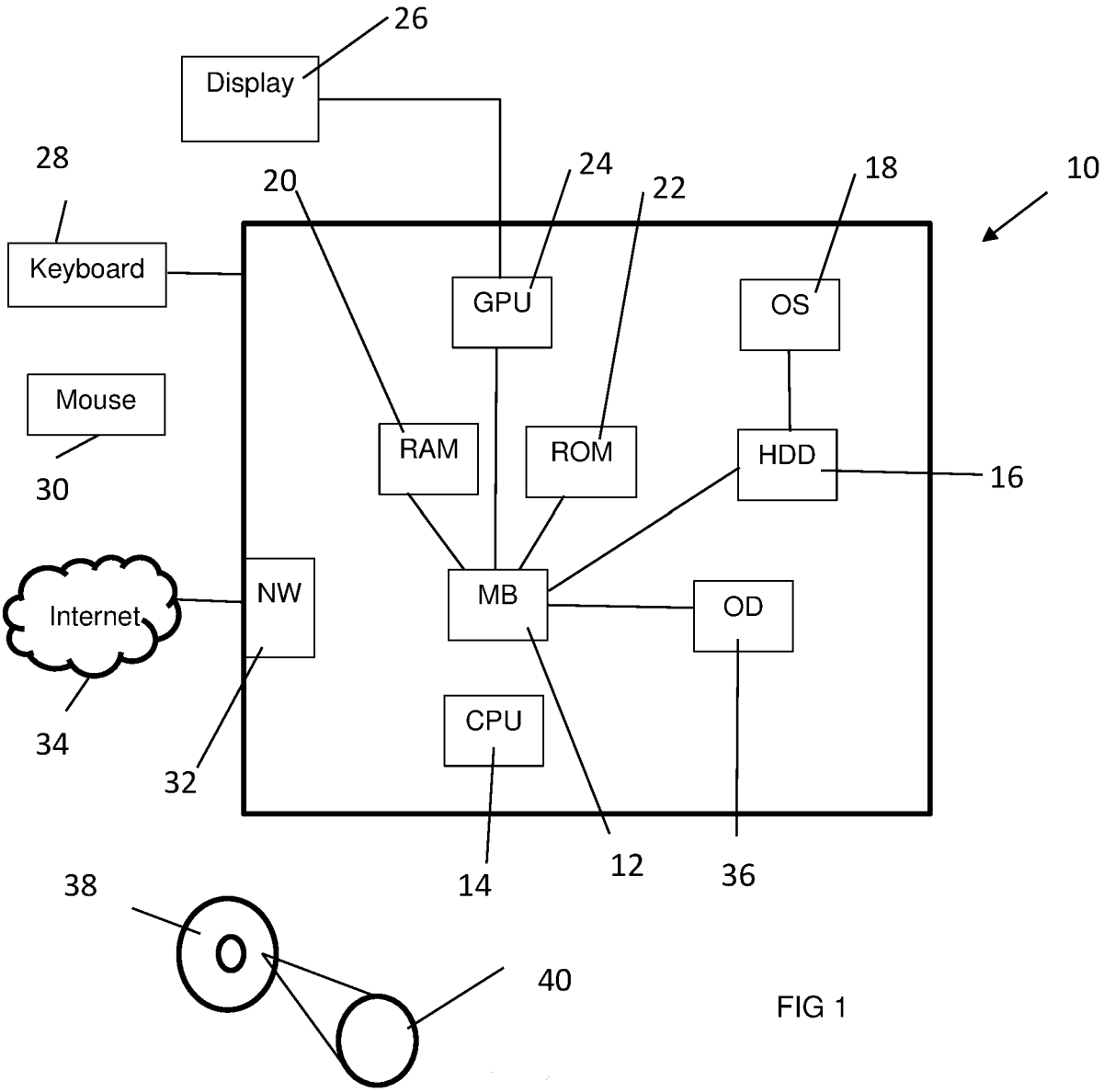

26
Display
28
20
24
22
18
10
Keyboard
GPU
OS
Mouse
RAM
ROM
HDD
16
30
Internet
NW
MB
OD
CPU
34
32
14
12
36
38
40

FIG 1

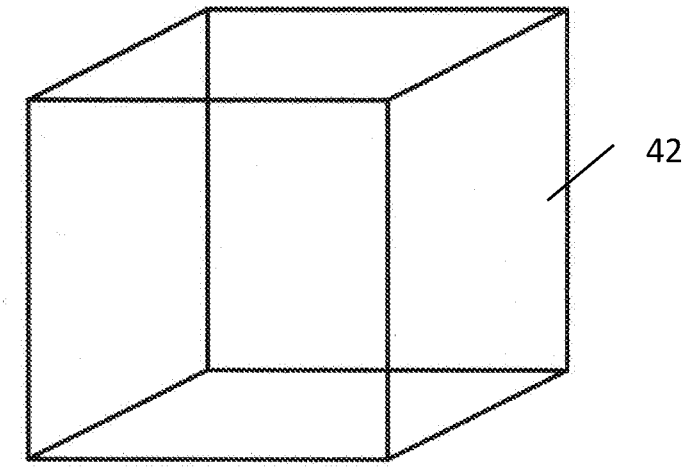

42

FIG 2

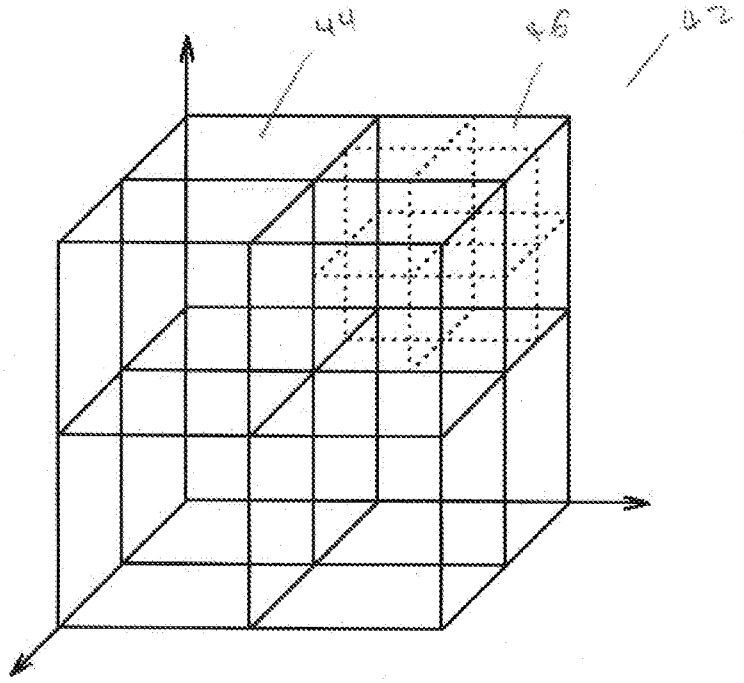

44
46
42

FIG 3

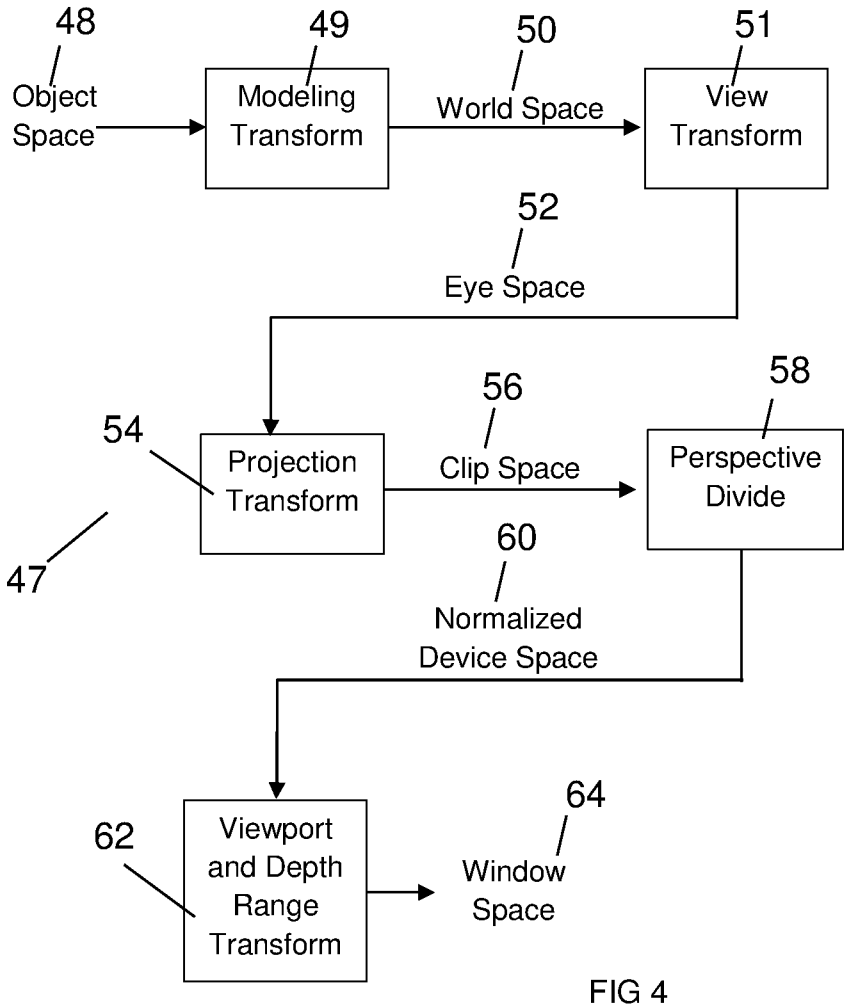

48
Object Space
49
Modeling Transform
50
World Space
51
View Transform
52
Eye Space
54
Projection Transform
56
Clip Space
58
Perspective Divide
47
60
Normalized Device Space
62
Viewport and Depth Range Transform
64
Window Space

FIG 4

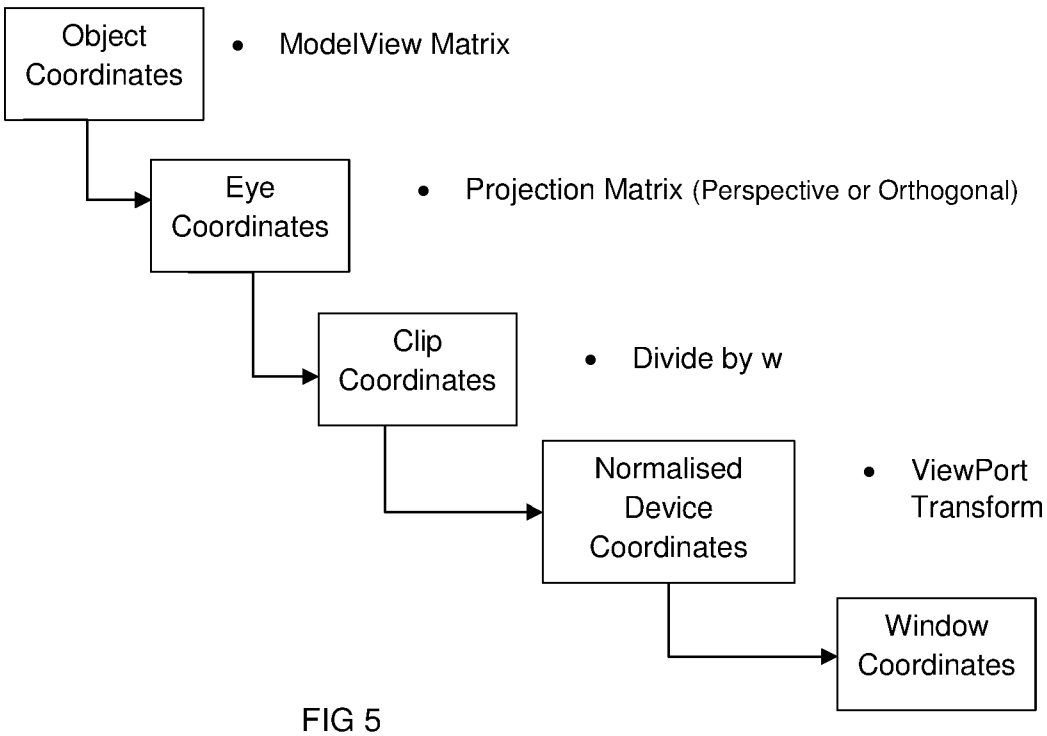

Object Coordinates
ModelView Matrix
Eye Coordinates
Projection Matrix (Perspective or Orthogonal)
Clip Coordinates
Divide by w
Normalised Device Coordinates
ViewPort Transform
Window Coordinates

FIG 5

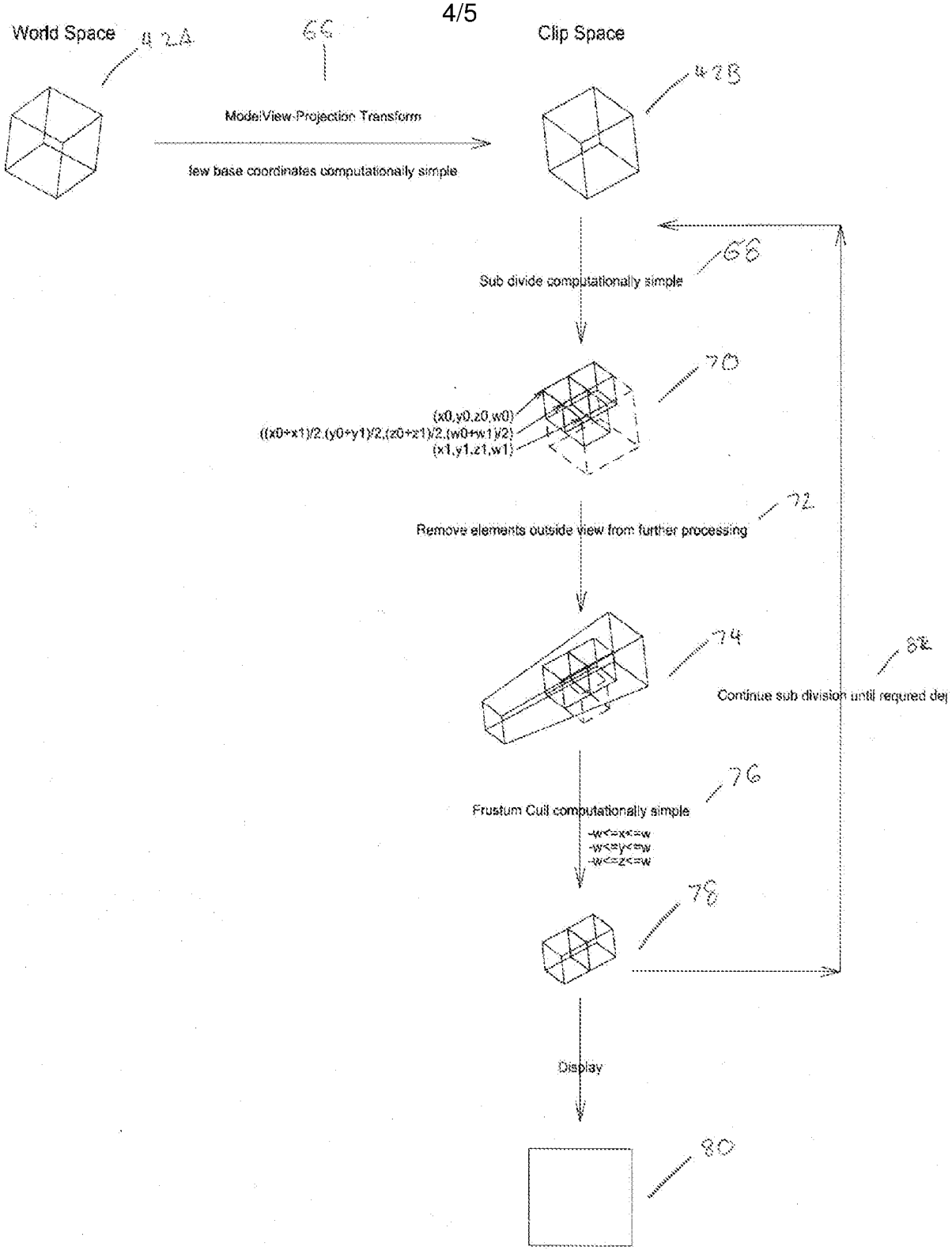

World Space
42A
66
Clip Space
42B
Model/View-Projection Transform
few base coordinates computationally simple
66
Sub divide computationally simple
70
(x0,y0,z0,w0)
((x0+x1)/2,(y0+y1)/2,(z0+z1)/2,(w0+w1)/2)
(x1,y1,z1,w1)
72
Remove elements outside view from further processing
74
Continue sub division until required dep
76
Frustum Cull computationally simple
-w<=x<=w
-w<=y<=w
-w<=z<=w
78
Display
80

FIG 6

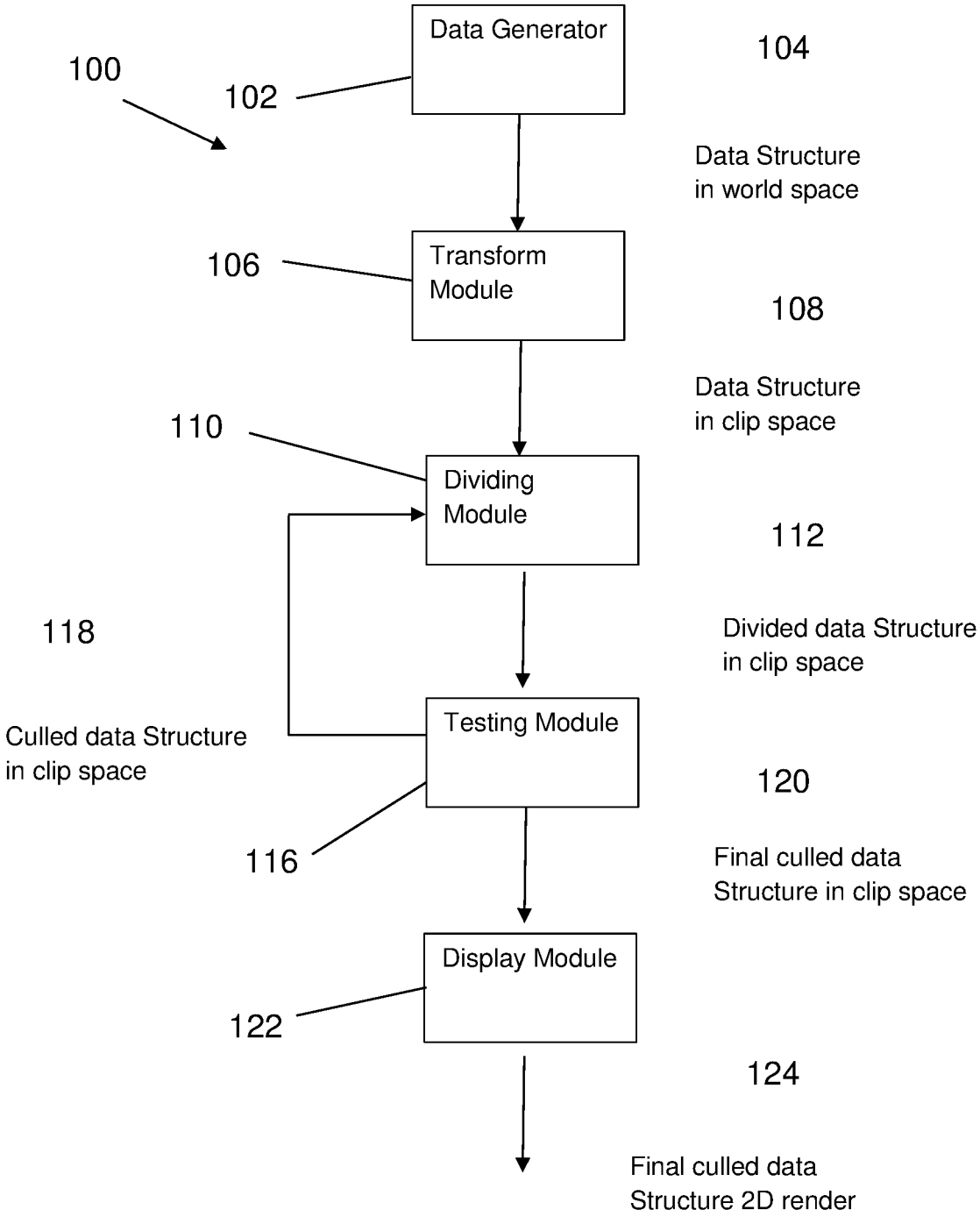
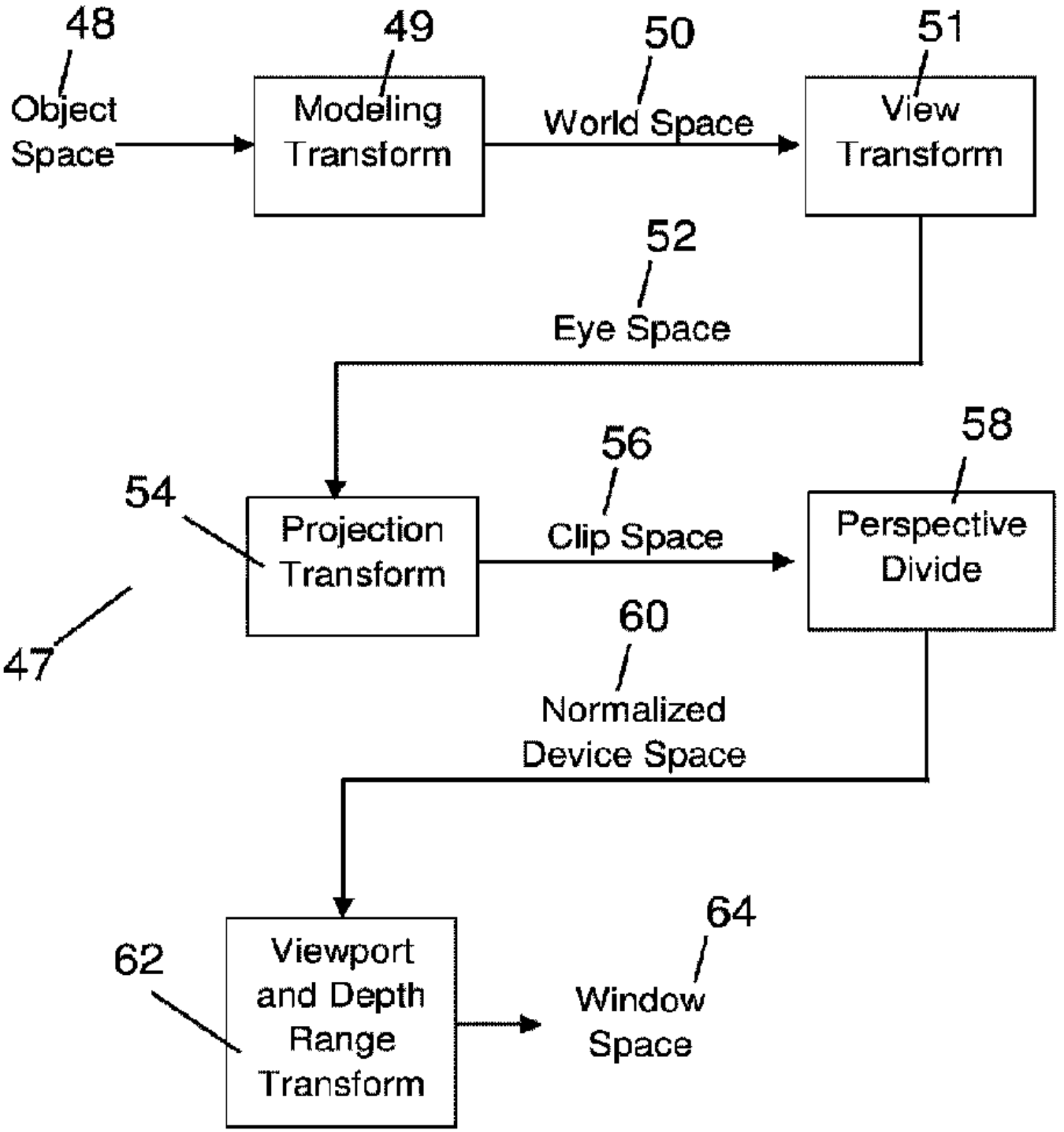

Data Generator
100
102
104
Data Structure
in world space
Transform
Module
106
108
Data Structure
in clip space
110
Dividing
Module
112
118
Divided data Structure
in clip space
Culled data Structure
in clip space
Testing Module
120
116
Final culled data
Structure in clip space
Display Module
122
124
Final culled data
Structure 2D render

FIG 7


48
Object Space
49
Modeling Transform
50
World Space
51
View Transform
52
Eye Space
54
Projection Transform
56
Clip Space
58
Perspective Divide
47
60
Normalized Device Space
62
Viewport and Depth Range Transform
64
Window Space