



(51) International Patent Classification:
G06T 15/00 (2011.01)

(21) International Application Number:
PCT/CN2018/117371

(22) International Filing Date:
26 November 2018 (26.11.2018)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **BEIJING DIDI INFINITY TECHNOLOGY AND DEVELOPMENT CO., LTD.** [CN/CN]; Building 34, No. 8 Dongbeiwang West Road, Haidian District, Beijing 100193 (CN).

(72) Inventor: **CHEN, Zungang**; Room 201, 2/F, Building 3, No. 29 Dongbeiwang South Road, Haidian District, Beijing 100193 (CN).

(74) Agent: **METIS IP (CHENGDU) LLC**; Room 808, 8th Floor, Building B7, Block D, Tianfu Jingrong Center, No. 99 West Section of Hupan Road, Tianfu New District, Chengdu, Sichuan 610213 (CN).

(81) Designated States (*unless otherwise indicated, for every kind of national protection available*): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KR, KW, 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,

(54) Title: SYSTEMS AND METHODS FOR POINT CLOUD RENDERING USING VIDEO MEMORY POOL

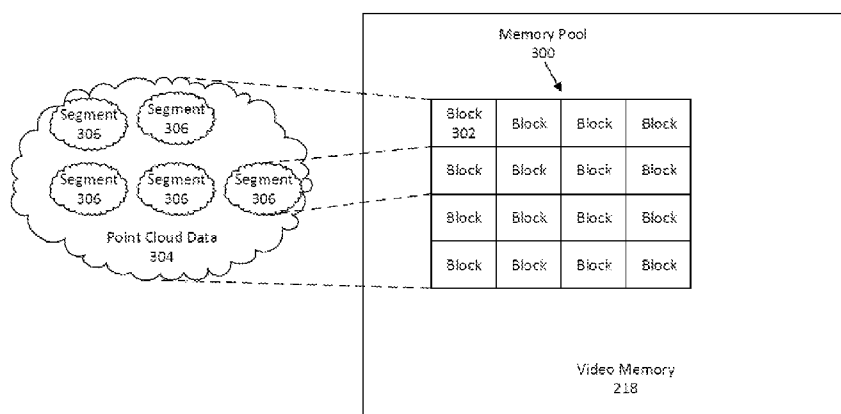


FIG. 3

(57) Abstract: Embodiments of the disclosure provide systems and methods for rendering a point cloud. The system includes a main memory configured to store point cloud data, a video memory, and at least one processor coupled to the main memory and the video memory. The at least one memory is configured to allocate a memory pool in the video memory for the point cloud data and generate a task based on a first segment of the point cloud data. The at least one memory is further configured to execute the task to compile the first segment of the point cloud data into a first block of the memory pool. The at least one processor is further configured to render the compiled first segment of the point cloud data in the video memory.



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))*

SYSTEMS AND METHODS FOR POINT CLOUD RENDERING USING VIDEO MEMORY POOL

TECHNICAL FIELD

5 **[0001]** The present disclosure relates to systems and methods for point cloud rendering, and more particularly to, systems and methods for rendering point cloud using a video memory pool.

BACKGROUND

10 **[0002]** Autonomous driving technology relies heavily on an accurate map. For example, accuracy of the navigation map is critical to functions of autonomous driving vehicles, such as positioning, ambience recognition, decision making and control. High-definition maps may be obtained by aggregating data acquired by various sensors and detectors on vehicles as they drive around. For example, a typical data acquisition system for high-definition maps is
15 usually a vehicle equipped with multiple integrated sensors such as a LiDAR, a Global Positioning System (GPS) receiver, an Inertial Measurement Unit (IMU) sensor, and one or more cameras, to capture features of the road on which the vehicle is driving and the surrounding objects. Data captured may include, for example, center line or border line coordinates of a lane, coordinates and images of an object, such as a building, another vehicle,
20 a landmark, a pedestrian, or a traffic sign.

[0003] LiDAR is known for quickly obtaining three-dimensional (3-D) information of surrounding objects and intensity information of the reflected signals from the surrounding objects. LiDAR can acquire data known as “point cloud data.” The size of the point cloud data acquired by LiDAR increases as the scan frequency increases from several laser pulses
25 per second to over 150 thousand laser pulses per second. Visualization of point cloud data is important for implementing LiDAR in various applications, such as to construct high-definition maps and 3-D buildings and city modeling.

[0004] However, visualization of point cloud data, especially with large data size, is limited by hardware capabilities of computing devices, such as processor and memory speeds.
30 For example, existing point cloud data software solutions use a main memory to load point cloud data files using file operations, and such an operation costs a significant amount of time to compile the point cloud data. Also, the compilation of point cloud data by existing point cloud data software solutions relies on rendering threads for each piece of newly-loaded or

updated data, thereby consuming a major part of the computing resource, particularly when the scene changes.

[0005] Embodiments of the disclosure address the above problems by improved systems and methods for point cloud rendering.

5

SUMMARY

[0006] Embodiments of the disclosure also provide a system for rendering a point cloud. The system includes a main memory configured to store point cloud data, a video memory, and at least one processor coupled to the main memory and the video memory. The at least one memory is configured to allocate a memory pool in the video memory for the point cloud data and generate a task based on a first segment of the point cloud data. The at least one memory is further configured to execute the task to compile the first segment of the point cloud data into a first block of the memory pool. The at least one processor is further configured to render the compiled first segment of the point cloud data in the video memory.

10

[0007] Embodiments of the disclosure provide a method for rendering a point cloud. The method includes allocating, by at least one processor, a memory pool in a video memory for point cloud data stored in a main memory, and generating, by the at least one processor, a task based on a first segment of the point cloud data. The method also includes executing, by the at least one processor, the task to compile the first segment of the point cloud data into a first block of the memory pool. The method further includes rendering, by the at least one processor, the compiled first segment of the point cloud data in the video memory.

15

20

[0008] Embodiments of the disclosure further provide a non-transitory computer-readable medium having instructions stored thereon that, when executed by one or more processors, causes the one or more processors to perform operations. The operations include allocating a memory pool in a video memory for point cloud data stored in a main memory, and generating a task based on a first segment of the point cloud data. The operations also include executing the task to compile the first segment of the point cloud data into a first block of the memory pool. The operations further include rendering the compiled first segment of the point cloud data in the video memory.

25

[0009] It is to be understood that both the foregoing general description and the following detailed description are exemplary and explanatory only and are not restrictive of the invention, as claimed.

30

BRIEF DESCRIPTION OF THE DRAWINGS

[0010] FIG. 1 illustrates a schematic diagram of an exemplary vehicle having sensors, according to embodiments of the disclosure.

[0011] FIG. 2 illustrates a block diagram of an exemplary computing device for point cloud rendering, according to embodiments of the disclosure.

[0012] FIG. 3 illustrates an exemplary video memory pool having multiple blocks and associated point cloud data segments, according to embodiments of the disclosure.

[0013] FIG. 4 illustrates an exemplary process of point cloud rendering, according to embodiments of the disclosure.

[0014] FIG. 5 illustrates a flowchart of an exemplary method for rendering a point cloud, according to embodiments of the disclosure.

[0015] FIG. 6 illustrates a flowchart of an exemplary method for generating and executing a task for compiling a point cloud data segment, according to embodiments of the disclosure.

DETAILED DESCRIPTION

[0016] Reference will now be made in detail to the exemplary embodiments, examples of which are illustrated in the accompanying drawings. Wherever possible, the same reference numbers will be used throughout the drawings to refer to the same or like parts.

[0017] FIG. 1 illustrates a schematic diagram of an exemplary vehicle 100 having a plurality of sensors 140 and 150, according to embodiments of the disclosure. Consistent with some embodiments, vehicle 100 may be a survey vehicle configured for acquiring data for constructing a high-definition map or 3-D buildings and city modeling. It is contemplated that vehicle 100 may be an electric vehicle, a fuel cell vehicle, a hybrid vehicle, or a conventional internal combustion engine vehicle. Vehicle 100 may have a body 110 and at least one wheel 120. Body 110 may be any body style, such as a sports vehicle, a coupe, a sedan, a pick-up truck, a station wagon, a sports utility vehicle (SUV), a minivan, or a conversion van. In some embodiments, vehicle 100 may include a pair of front wheels and a pair of rear wheels, as illustrated in FIG. 1. However, it is contemplated that vehicle 100 may have less wheels or equivalent structures that enable vehicle 100 to move around. Vehicle 100 may be configured to be all wheel drive (AWD), front wheel drive (FWR), or rear wheel drive (RWD). In some embodiments, vehicle 100 may be configured to be operated by an operator occupying the vehicle, remotely controlled, and/or autonomous.

[0018] As illustrated in FIG. 1, vehicle 100 may be equipped with sensor 140 mounted to body 110 via a mounting structure 130. Mounting structure 130 may be an electro-

mechanical device installed or otherwise attached to body 110 of vehicle 100. In some embodiments, mounting structure 130 may use screws, adhesives, or another mounting mechanism. Vehicle 100 may be additionally equipped with sensor 150 inside or outside body 110 using any suitable mounting mechanisms. It is contemplated that the manners in which each sensor 140 or 150 can be equipped on vehicle 100 are not limited by the example shown in FIG. 1, and may be modified depending on the types of sensors 140 and 150 and/or vehicle 100 to achieve desirable sensing performance.

[0019] Consistent with some embodiments, sensors 140 and 150 may be configured to capture data as vehicle 100 moves along a trajectory. For example, sensor 140 may be a LiDAR scanner configured to scan the surrounding and acquire point clouds. LiDAR measures distance to a target by illuminating the target with pulsed laser light and measuring the reflected pulses with a sensor. Differences in laser return times and wavelengths can then be used to make digital 3-D representations of the target. The light used for LiDAR scan may be ultraviolet, visible, or near infrared. Because a narrow laser beam can map physical features with very high resolution, a LiDAR scanner is particularly suitable for high-definition map surveys. In some embodiments, a LiDAR scanner may capture point cloud. As vehicle 100 moves along the trajectory, sensor 140 may continuously capture data. Each set of scene data captured at a certain time range is known as a data frame.

[0020] As illustrated in FIG. 1, vehicle 100 may be additionally equipped with sensor 150, which may include sensors used in a navigation unit, such as a GPS receiver and one or more IMU sensors. A GPS is a global navigation satellite system that provides geolocation and time information to a GPS receiver. An IMU is an electronic device that measures and provides a vehicle's specific force, angular rate, and sometimes the magnetic field surrounding the vehicle, using various inertial sensors, such as accelerometers and gyroscopes, sometimes also magnetometers. By combining the GPS receiver and the IMU sensor, sensor 150 can provide real-time pose information of vehicle 100 as it travels, including the positions and orientations (e.g., Euler angles) of vehicle 100 at each time stamp. In some embodiments, pose information may be used for calibration and/or pretreatment of the point cloud data captured by sensor 140.

[0021] Consistent with the present disclosure, vehicle 100 may include a local computing device 160 inside body 110 of vehicle 100 or communicate with a remote computing device, such as a server, (not illustrated in FIG. 1) for rendering point cloud in an efficiency and effective manner using video memory pool. In some embodiments, local computing device 160 (or remote computing device) can pre-request a video memory pool for point cloud data

and reuse the video memory pool to reduce the time for allocate the video memory for the point cloud data. For example, upon power on, local computing device 160 (or remote computing device) may allocate a video memory pool and divide it into multiple blocks (cells) based on the size of each segment of the point cloud data. In some embodiments, local
5 computing device 160 (or remote computing device) can use task threads to optimize compilation of point cloud data, thereby reducing the workload of rendering threads. By introducing such a pool-based video memory management, it is possible to compile point cloud data using task threads. For example, a main thread can package the information associated with a point cloud data segment into a task and add the task into a task queue such
10 that a task thread can compile the point cloud data segment without any intervention from a rendering thread, thereby increasing point cloud data loading and rendering speed.

[0022] As used herein, a “thread” is the smallest sequence of programmed instructions that can be managed independently, i.e., the smallest unit of execution within a process. A “task thread” is a thread that can execute a task, for example, compiling a point cloud segment. A
15 “main thread” is a thread that can initialize a point cloud rendering process and generate tasks to be executed by one or more task threads. A “rendering thread” is a thread that can render a compiled point cloud segment. Consistent with this disclosure, “rendering” is an automatic process of generating an image from 2-D or 3-D model(s) (e.g., represented by point cloud data or a segment thereof). “Compiling” is an automatic process of loading a point cloud
20 data segment into video memory.

[0023] For example, FIG. 2 illustrates a block diagram of an exemplary computing device 200 for rendering point cloud, according to embodiments of the disclosure. Consistent with the present disclosure, computing device 200 may use various types of data for rendering point cloud of a scene. The various types of data may be captured by sensors 140 and 150
25 equipped on vehicle 100 with respect to the scene, as vehicle 100 moves along a trajectory. The data may include point cloud data captured by sensor 140 (e.g., a LiDAR scanner) and pose information of vehicle 100 acquired by sensor 150 (e.g., a GPS receiver and/or one or more IMU sensors). In some embodiments, the point cloud data may be obtained by transforming the native data from the LiDAR scanner in a local coordinate system into a
30 global coordinate system (e.g., the longitude/latitude coordinates) based on the real-time pose information (e.g., positions and orientations) from the GPS receiver and IMU sensors.

[0024] In some embodiments, as shown in FIG. 2, computing device 200 may include a communication interface 202, a main processor 204, a main memory 206, a storage 208, and a graphic processing unit (GPU) 210. In some embodiments, computing device 200 may

have different modules in a single device, such as an integrated circuit (IC) chip (implemented as an application-specific integrated circuit (ASIC) or a field-programmable gate array (FPGA), or separate devices with dedicated functions. In some embodiments, one or more components of computing device 200 may be located inside vehicle 100 (e.g., local computing device 160 in FIG. 1) or may be alternatively in a mobile device, in the cloud, or another remote location. Components of computing device 200 may be in an integrated device, or distributed at different locations but communicate with each other through a network (not shown). For example, main processor 204 may be a processor on-board vehicle 100, a processor inside a mobile device, or a cloud processor, or any combinations thereof.

[0025] Communication interface 202 may send data to and receive data from components such as sensors 140 and 150 via communication cables, a Wireless Local Area Network (WLAN), a Wide Area Network (WAN), wireless networks such as radio waves, a nationwide cellular network, and/or a local wireless network (e.g., BluetoothTM or WiFi), or other communication methods. In some embodiments, communication interface 202 can be an integrated services digital network (ISDN) card, cable modem, satellite modem, or a modem to provide a data communication connection. As another example, communication interface 202 can be a local area network (LAN) card to provide a data communication connection to a compatible LAN. Wireless links can also be implemented by communication interface 202. In such an implementation, communication interface 202 can send and receive electrical, electromagnetic or optical signals that carry digital data streams representing various types of information via a network. Consistent with some embodiments, communication interface 202 may receive data captured by sensors 140 and 150, including the point cloud data and vehicle pose information, and provide the received data to storage 208 for storage or to main processor 204 for processing.

[0026] Main processor 204 may include any appropriate type of central processing unit (CPU) having one or multiple cores, a general processor (e.g., APU, accelerated processing unit; GPGPU, general-purpose computing on GPU), or any other suitable processor. Main processor 204 may be configured as a separate processor module dedicated to rendering point cloud. Alternatively, main processor 204 may be configured as a shared processor module for performing other functions unrelated to point cloud rendering.

[0027] Main memory 206 and storage 208 may include any appropriate type of mass storage provided to store any type of information that main processor 204 may need to operate. Main memory 206 and storage 208 may be a volatile or non-volatile, magnetic, semiconductor, tape, optical, removable, non-removable, or other type of storage device or

tangible (i.e., non-transitory) computer-readable medium including, but not limited to, a ROM, a flash memory, a dynamic RAM, and a static RAM. Main memory 206 and/or storage 208 may be configured to store one or more computer programs that may be executed by main processor 204 to perform point cloud rendering functions disclosed herein. For example, main memory 206 and/or storage 208 may be configured to store program(s) that may be executed by main processor 204.

[0028] Main memory 206 and/or storage 208 may be further configured to store information and data used by main processor 204. For instance, main memory 206 and/or storage 208 may be configured to store point cloud data to be rendered. The various types of data may be stored permanently, removed periodically, or disregarded immediately after each frame of data is processed.

[0029] GPU 210 may include a graphic processor 217 and a video memory 218 coupled to graphic processor 217. Although graphic processor 217 and main processor 204 are shown as discrete processors in FIG. 2, it is contemplated that graphic processor 217 and main processor 204 may be integrated as a single processor, such as an Application Processor (AP) used in a mobile device or a CPU with integrated graphics. Video memory 218 (also known as a “frame buffer”) is a dedicated memory or a portion of a shared or unified memory that temporarily stores data relating to graphics, e.g., a bitmap of image data. Although main memory 206 and video memory 218 are shown as discrete memories in FIG. 2, it is contemplated that main memory 206 and video memory 218 may be portions of a unified memory shared by main processor 204 and graphic processor 217.

[0030] As shown in FIG. 2, multiple threads may be loaded from main memory 206 and executed by main processor 204, such as a main thread 212, a task thread 214, a rendering thread 216, and the like. It is contemplated that the number of each type of threads 212-216 that can be executed by main processor 204 in parallel or in sequence is not limited to one, and can be any suitable number.

[0031] Main thread 212 may be configured to initialize and manage the point cloud rendering process. Main thread 212 may preprocess the point cloud data to be rendered and set up the environment for point cloud rendering. In some embodiments, main thread 212 may divide the point cloud data into multiple point cloud data segments. In one example, the point cloud data may be divided in accordance with an octree data structure. An octree is a tree data structure in which each internal node has exactly eight children. Octrees can be used to partition a 3-D space by recursively subdividing it into eight octants. Each node (octant) represents a point cloud data segment of the same size. Computing device 200 may

process the point cloud data based on the octree data structure. To set up the working environment, main thread 212 may pre-allocate a memory pool in video memory 218 for the point cloud data. The size of memory pool may be determined based on various factors, such as the hardware configuration and specifications of computing device 200 (e.g., the speed of main processor 204, the size of main memory 206, the size of video memory 218, etc.). In some embodiments, main thread 212 may further divide the memory pool into multiple blocks based on the size of each point cloud data segment.

[0032] For example, FIG. 3 illustrates an exemplary video memory pool 300 having multiple blocks 302 and associated point cloud data segments 306, according to embodiments of the disclosure. Memory pool 300 may be pre-allocated in video memory 218 for point cloud rendering. Memory pool 300 includes multiple blocks 302 of the same size and is the use of pools for memory management that allows dynamic memory allocation. Blocks 302 of memory pool 300 can be allocated, accessed, and released at run time by threads running on main processor 204. For example, as shown in FIG. 3, point cloud data 304 may be divided into multiple point cloud data segments 306, and each point cloud data segment 306 may be dynamically allocated (associated) with a corresponding memory pool block 302 in real time by main thread 212. Each point cloud data segment 306 may be compiled into and access associated memory pool block 302. Once each point cloud data segment 306 has been rendered, associated memory pool block 302 may be dynamically released and thus, can be rescued.

[0033] Referring back to FIG. 2, main thread 212 may be configured to generate a task based on a point cloud data segment. A task is a set of instructions that can be executed by task thread 214 of main processor 204. For example, as shown in FIG. 4, to generate a new task 403, main thread 212 may identify a new point cloud data segment 306 based on a scheduling policy from a scheduler 402. The scheduling policy may aim at one or more objectives, such as maximizing throughput, minimizing wait time, minimizing latency or response time, maximizing fitness, with compromise or preference. Main thread 212 then may package the index associated with new point cloud data segment 306 (and/or any other suitable information associated with new point cloud data segment 306) into new task 403. Main thread 212 may further add new task 403 into a task queue 404. Tasks 403 in task queue 404 may be kept in order, i.e., added into the rear terminal position (enqueue) and removed from the front terminal position (dequeue). In some embodiments, when task queue 404 is empty, main thread 212 may predict next point cloud data segment 306 to be rendered and pre-load it into a memory buffer 408 (e.g., in main memory 206). The predication may

be made by main thread 212 based on a model learned from the historical data and/or the real-time information (e.g., the prior one or more point cloud data segments that have been rendered).

[0034] Referring back to FIG. 2, task thread 214 may be configured to execute the task to compile the point cloud data segment into a memory pool block associated with (allocated) to the point cloud data. As shown in FIG. 3, when task queue 404 is not empty (i.e., having at least one task 403), task thread 214 may retrieve task 403 (e.g., dequeue the first one from the front terminal position) from task queue 404. Task thread 214 may then retrieve corresponding point cloud data segment 306 from memory buffer 408 (e.g., in main memory 206) based on the index associated with point cloud data segment 306 that has been packaged in task 403. Task thread 214 may further associate point cloud data segment 306 with memory pool block 302. For example, task thread 214 may dynamically request an empty memory pool block from the memory pool to be associated with point cloud data segment 306 and compile point cloud data segment 306 into the associated empty memory pool block.

[0035] Consistent with the disclosures of the present application, the compilation of point cloud data segment does not necessarily occur in memory buffer 408. In some embodiments, for point cloud data segments 306 that have not been pre-loaded into memory buffer 408, they still can be compiled by task thread 214 in parallel with the task of compiling point cloud data segment 306 in memory buffer 408 to increase the compilation bandwidth and speed. In some embodiments, another point cloud data segment may be compiled by task thread 214 into its associated memory pool block in parallel with the execution of task 403. That is, multiple point cloud data segments (regardless of whether it has been pre-loaded in memory buffer 408) may be compiled in parallel.

[0036] Referring back to FIG. 2, rendering thread 216 may be configured to render the compiled point cloud data segment(s) in video memory 218. Rendering thread 216 may generate an image of point cloud from the 3-D model represented by the point cloud data. Rendering thread 216 of main processor 204 may work in conjunction with a graphic processor 217 in GPU 210 to render compiled point cloud data segment(s) 410 in video memory 218, as shown in FIG. 4. For example, task thread 214 may add compiled point cloud data segment(s) 410 in a rendering list, and rendering thread 216 may start to render compiled point cloud data segment(s) 410 in the rendering list. Consistent with the disclosures of the present application, rendering thread 216 may continue to render any available compiled point cloud data segment(s) 410 in the rendering list without waiting for the completion of task 403 in task queue 404. In some embodiments, for any point cloud data

segment 306 that has been pre-loaded into memory buffer 408, rendering thread 216 may render it based on its parent node in the octree data structure. For any compiled point cloud data segment 410 in the rendering list that has been rendered, rendering thread 216 may dynamically release its associated memory pool block (e.g., in accordance with the scheduling policy), so that the memory pool block can be reused.

[0037] Referring back to FIG. 2, the rendered point cloud may be displayed on a display 220 that is located inside vehicle 100 or remote from vehicle 100 (e.g., connected with computing device 200 via communication interface 202). Display 220 may include a display such as a Liquid Crystal Display (LCD), a Light Emitting Diode (LED) display, a plasma display, or any other type of display. By implementing the point cloud rendering systems and methods disclosed herein, the point cloud data loading and rendering speed can be significantly increased. As a result, the display of point cloud by display 220 can become smoother, such as during the scene change.

[0038] FIG. 5 illustrates a flowchart of an exemplary method 500 for rendering point cloud, according to embodiments of the disclosure. For example, method 500 may be implemented by a point cloud rendering system of vehicle 100 that includes, among other things, computing device 200. However, method 500 is not limited to that exemplary embodiment. Method 500 may include steps S502-S516 as described below. It is to be appreciated that some of the steps may be optional to perform the disclosure provided herein. Further, some of the steps may be performed simultaneously, or in a different order than shown in FIG. 5.

[0039] In step S502, point cloud data may be divided into segments. For example, before rendering point cloud data 304, main processor 204 may divide it into multiple point cloud data segments 306. In one example, each of point cloud data segments 306 has the same data size. The data size and/or number of point cloud data segments 306 may be determined based on the total size of point cloud data 304 and hardware configuration and specification of computing device 200. In some embodiments, point cloud data may be divided into segments in accordance with an octree data structure, and each divided point cloud data segments may correspond to one node of the octree tree with the same data size.

[0040] In step S504, a memory pool in a video memory may be allocated for the point cloud data. For example, main processor 204 may pre-allocate memory pool 300 in video memory 218 for point cloud data 304. In some embodiments, the size of memory pool 300 may be determined based on the hardware configuration and specifications of computing device 200. Memory pool 300 may then be reserved and dedicated for compiling and rendering point cloud data 304 during the current point cloud rendering process. In step S506,

the memory pool may be divided into blocks. For example, main processor 204 may initialize memory pool 300, including dividing memory pool 300 into multiple memory pool blocks 302 based on the size of each point cloud data segment 306. In some embodiments, each memory pool block 302 may have the same size determined based on the size of each point cloud data segment 306. The allocation, access, and release of each memory pool block 302 may be handled dynamically during the process of point cloud rendering.

[0041] In step S508, a task may be generated based on a first point cloud data segment. For example, main processor 204 may generate a new task 403 based on corresponding point cloud data segment 306 pre-loaded in memory buffer 408 of main memory 206. For example, FIG. 6 illustrates a flowchart of an exemplary method 600 for generating and executing a task for compiling a point cloud data segment, according to embodiments of the disclosure. In some embodiments, step S508 may include steps S602-S606. In step S602, the first point cloud data segment may be identified based on a scheduling policy. For example, main thread 212 may identify new point cloud data segment 306 from remaining point cloud data segments 306 based on the next point cloud data segment to be rendered based on the scheduling policy from scheduler 402. The scheduling policy may be designed to accomplish at least one goal in identifying new point cloud data segment 306, such as maximizing throughput, minimizing wait time, minimizing latency or response time, maximizing fitness, etc. In step S604, an index associated with the first point cloud data segment may be packaged into the task. For example, main thread 212 may package the index of new point cloud data segment 306 into new task 403. It is contemplated that any other suitable information associated with new point cloud data segment 306 may be packaged into next task 403 as well. In step S606, the task may be added into a task queue. For example, main thread 212 may add new task 403 into task queue 404. In some embodiments, when task queue 404 is empty, main thread 212 may predict next point cloud data segment 306 to be rendered and pre-load it into memory buffer 408.

[0042] Referring back to FIG. 5, in step S510, the task may be executed to compile the first point cloud data segment into a first memory pool block. For example, main processor 204 may execute the task to compile the first point cloud data segment into the first memory pool block associated with the first point cloud data segment. In some embodiments, step S510 may be implemented as steps S608-S612 in FIG. 6. In step S608, the task may be retrieved from the task queue. For example, task thread 214 may retrieve new task 403 from task queue 404, for example, by dequeuing the first one from the front terminal position of task queue 404. In step S610, the first point cloud data segment may be retrieved based on

the index in the task. For example, task thread 214 may retrieve corresponding point cloud data segment 306 from memory buffer 408 based on the index associated with point cloud data segment 306 that has been packaged in new task 403. In step S612, the first memory pool block may be associated with the first point cloud data segment. For example, task thread 214 may allocate a corresponding memory pool block to corresponding point cloud data segment 306 for compilation.

[0043] Referring back to FIG. 5, in step S512, a second point cloud data segment may be compiled into a second memory pool block in parallel with executing the task based on the first point cloud data segment. For example, task thread 214 may compile another point cloud data segment 306 that has not been pre-loaded into memory buffer 408 in parallel with step S510.

[0044] In step S514, the compiled point cloud data segments (e.g., the first and second point cloud data segments) may be rendered in the video memory. For example, rendering thread 216 may render compiled point cloud data segments 410 in corresponding memory pool blocks in video memory 218. Rendering thread 216 may continue to render any available compiled point cloud data segment(s) 410 in the rendering list without waiting for the completion of task 403 in task queue 404. For any compiled point cloud data segment 410 in the rendering list that has been rendered, rendering thread 216 may dynamically release its associated memory pool block in accordance with the scheduling policy, so that the memory pool block can be reused. Steps S508-S514 may be repeated until all the point cloud data segments divided from the point cloud data have been rendered. In step S516, once all the point cloud data segments have been rendered, the rendered point cloud data may be displayed. For example, display 220 may visualize the point cloud by displaying the rendered point cloud data.

[0045] Another aspect of the disclosure is directed to a non-transitory computer-readable medium storing instructions which, when executed, cause one or more processors to perform the methods, as discussed above. The computer-readable medium may include volatile or non-volatile, magnetic, semiconductor, tape, optical, removable, non-removable, or other types of computer-readable medium or computer-readable storage devices. For example, the computer-readable medium may be the storage device or the memory module having the computer instructions stored thereon, as disclosed. In some embodiments, the computer-readable medium may be a disc or a flash drive having the computer instructions stored thereon.

[0046] It will be apparent to those skilled in the art that various modifications and variations can be made to the disclosed system and related methods. Other embodiments will be apparent to those skilled in the art from consideration of the specification and practice of the disclosed system and related methods.

- 5 **[0047]** It is intended that the specification and examples be considered as exemplary only, with a true scope being indicated by the following claims and their equivalents.

WHAT IS CLAIMED IS:

1. A system for rendering a point cloud, comprising:
a main memory configured to store point cloud data;
5 a video memory;
at least one processor coupled to the main memory and the video memory and configured to:
allocate a memory pool in the video memory for the point cloud data;
generate a task based on a first segment of the point cloud data; and
10 execute the task to compile the first segment of the point cloud data into a first block of the memory pool; and
render the compiled first segment of the point cloud data in the video memory.
2. The system of claim 1, wherein the at least one processor is further configured to
15 compile a second segment of the point cloud data into a second block of the memory pool in parallel with the execution of the task.
3. The system of claim 1, wherein the at least one processor is further configured to:
divide the point cloud data into a plurality of segments each having a same size; and
20 divide the memory pool into a plurality of blocks based on the size of each segment of the point cloud data.
4. The system of claim 1, wherein to generate the task, the at least one processor is further configured to:
25 identify the first segment of the point cloud data based on a scheduling policy;
package an index associated with the first segment of the point cloud data into the task;
and
add the task into a task queue.
- 30 5. The system of claim 4, wherein to execute the task, the at least one processor is further configured to:
retrieve the task from the task queue;
retrieve the first segment of the point cloud data from the main memory based on the index; and

associate the first block of the memory pool with the first segment of the point cloud data.

6. The system of claim 1, wherein the first segment of the point cloud data has been pre-loaded into a buffer of the main memory.
7. The system of claim 2, wherein the second segment of the point cloud data has not been pre-loaded into a buffer of the main memory.
8. The system of claim 1, wherein the main processor is further configured to release the first block of the memory pool after the compiled first segment of the point cloud data is rendered.
9. A method for rendering a point cloud, comprising:
allocating, by at least one processor, a memory pool in a video memory for point cloud data stored in a main memory;
generating, by the at least one processor, a task based on a first segment of the point cloud data;
executing, by the at least one processor, the task to compile the first segment of the point cloud data into a first block of the memory pool; and
rendering, by the at least one processor, the compiled first segment of the point cloud data in the video memory.
10. The method of claim 9, further comprising compiling, by the at least one processor, a second segment of the point cloud data into a second block of the memory pool in parallel with executing the task.
11. The method of claim 9, further comprising:
dividing, by the at least one processor, the point cloud data into a plurality of segments each having a same size; and
dividing, by the at least one processor, the memory pool into a plurality of blocks based on the size of each segment of the point cloud data.
12. The method of claim 9, wherein generating the task comprises:

identifying the first segment of the point cloud data based on a scheduling policy;
packaging an index associated with the first segment of the point cloud data into the task;
and
adding the task into a task queue.

5

13. The method of claim 12, wherein executing the task comprises:
retrieving the task from the task queue;
retrieving the first segment of the point cloud data from the main memory based on the
index; and

10

associating the first block of the memory pool with the first segment of the point cloud
data.

14. The method of claim 9, wherein the first segment of the point cloud data has been pre-
loaded into a buffer of the main memory.

15

15. The method of claim 10, wherein the second segment of the point cloud data has not
been pre-loaded into a buffer of the main memory.

20

16. The method of claim 9, further comprising releasing, by the at least one processor, the
first block of the memory pool after rendering the compiled first segment of the point cloud
data.

25

17. A non-transitory computer-readable medium having instructions stored thereon that,
when executed by one or more processors, causes the one or more processors to perform
operations comprising:

allocating a memory pool in a video memory for point cloud data stored in a main
memory;

generating a task based on a first segment of the point cloud data;

executing the task to compile the first segment of the point cloud data into a first block
of the memory pool; and

30

rendering the compiled first segment of the point cloud data in the video memory.

18. The computer-readable medium of claim 17, wherein the operations further comprise compiling a second segment of the point cloud data into a second block of the memory pool in parallel with executing the task.
- 5 19. The computer-readable medium of claim 17, wherein the operations further comprise:
dividing the point cloud data into a plurality of segments each having a same size; and
dividing the memory pool into a plurality of blocks based on the size of each segment of the point cloud data.
- 10 20. The computer-readable medium of claim 17, wherein the operations further comprise releasing the first block of the memory pool after rendering the compiled first segment of the point cloud data.

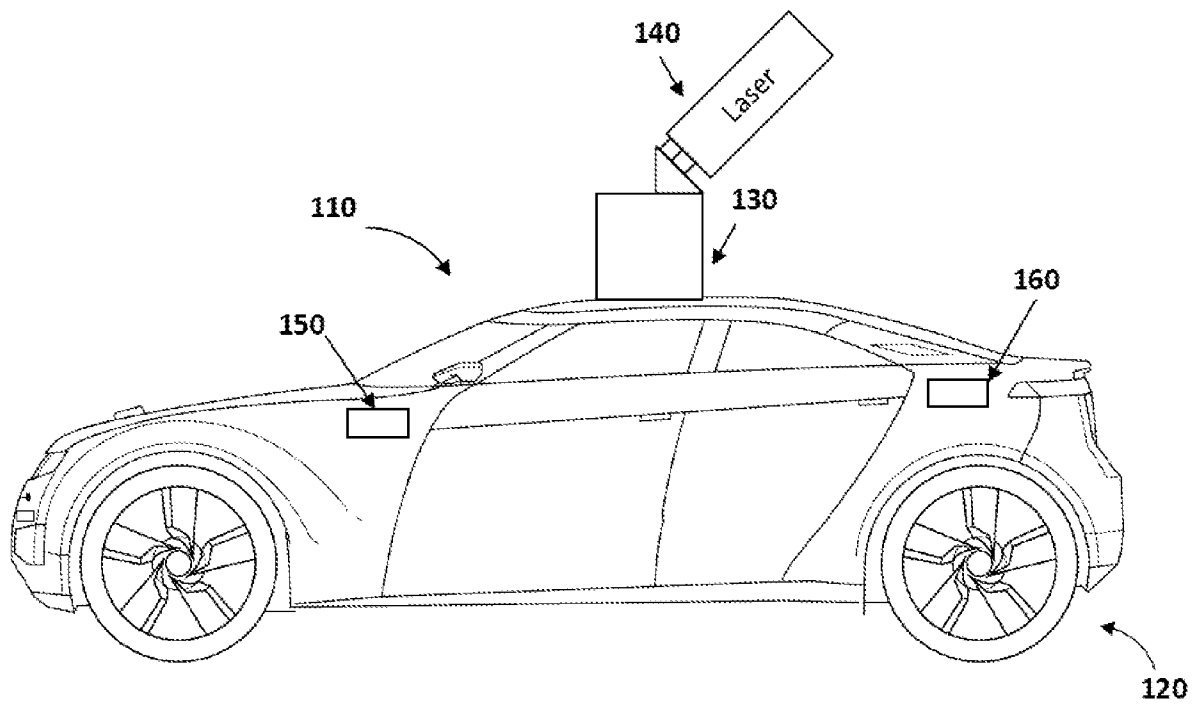


FIG. 1

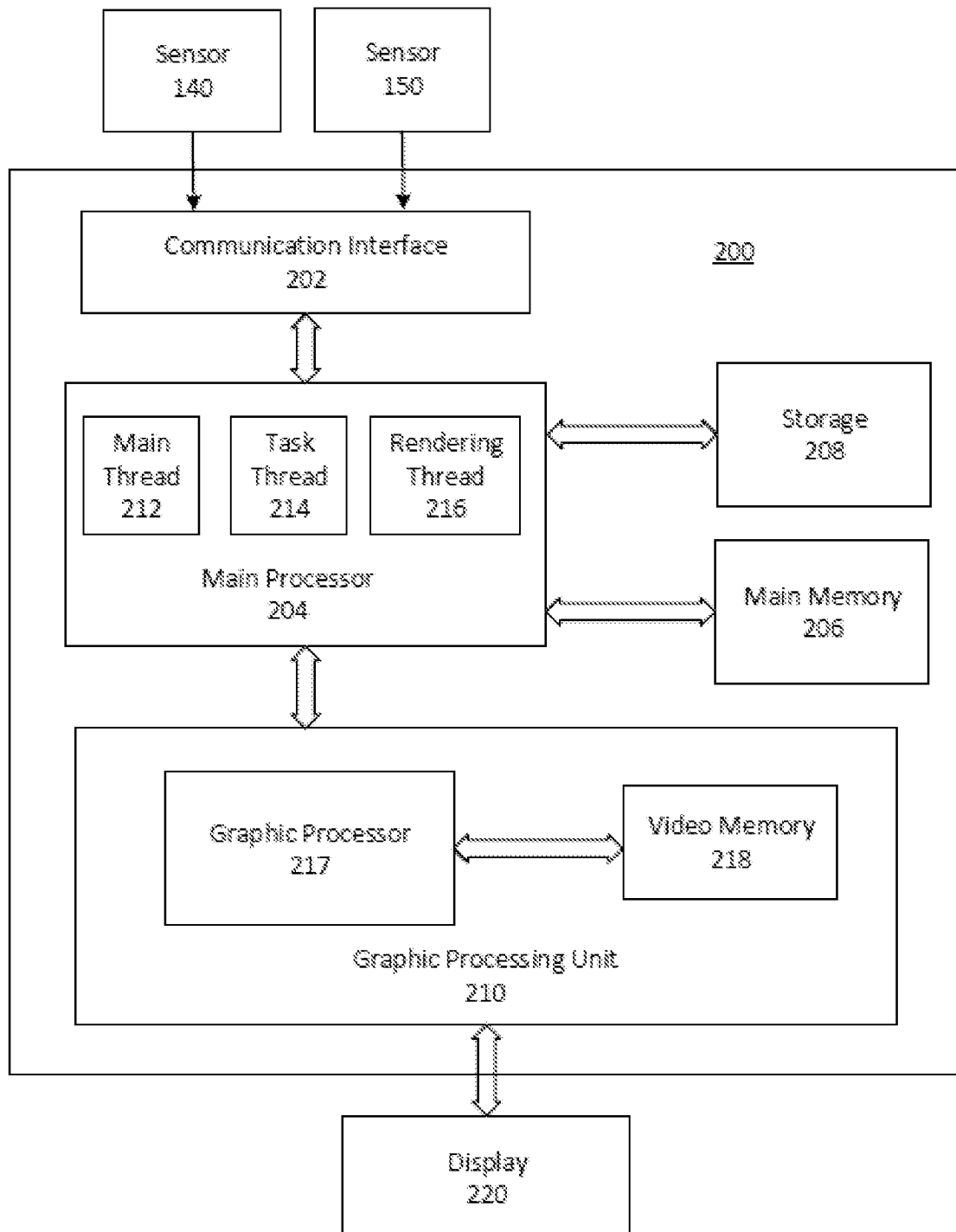


FIG. 2

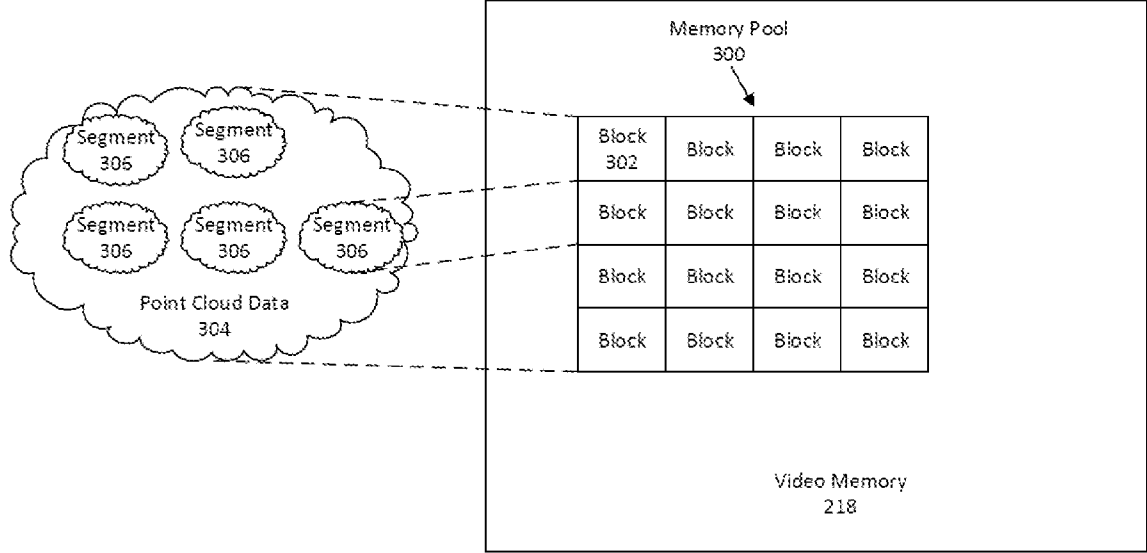


FIG. 3

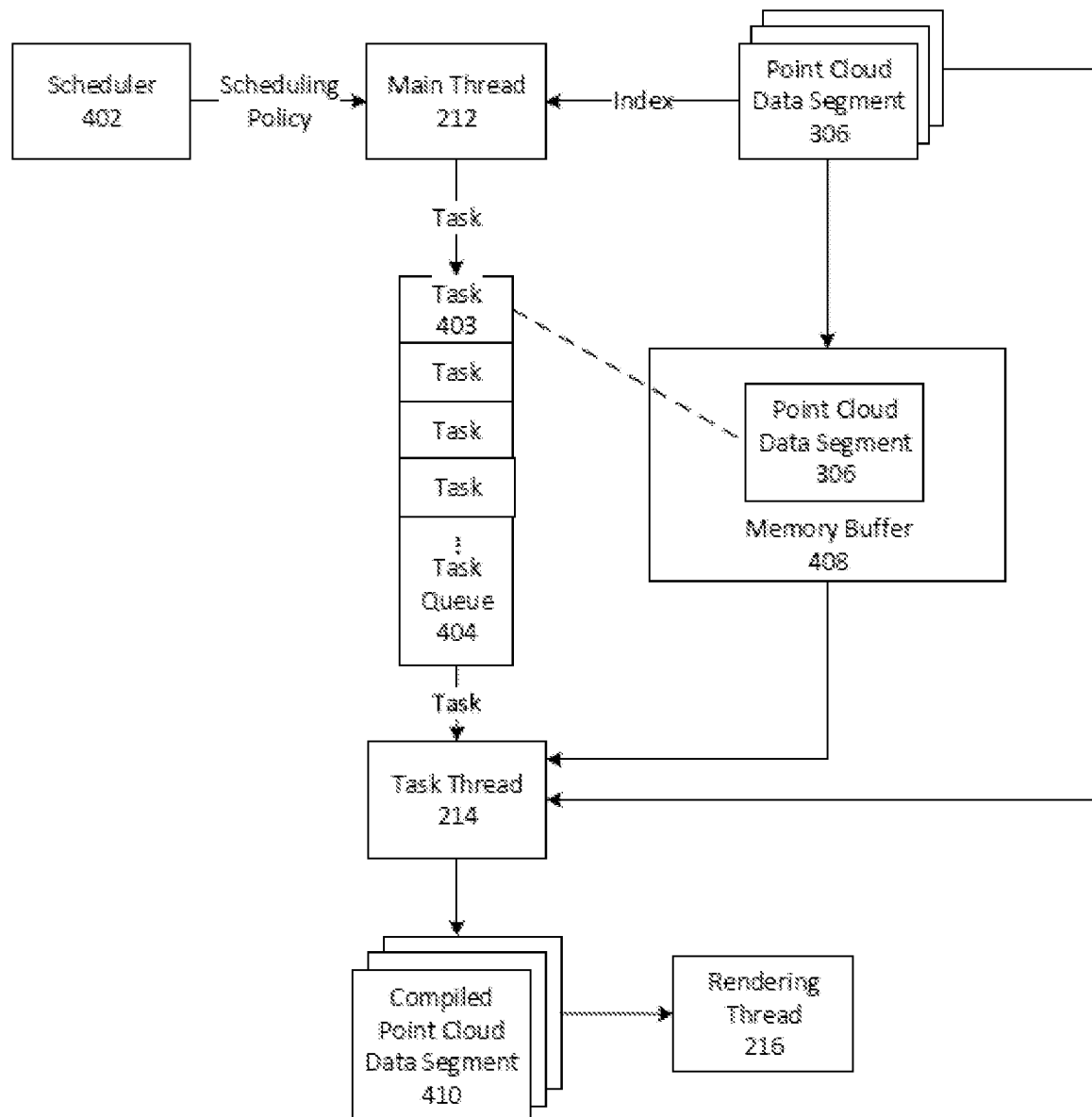


FIG. 4

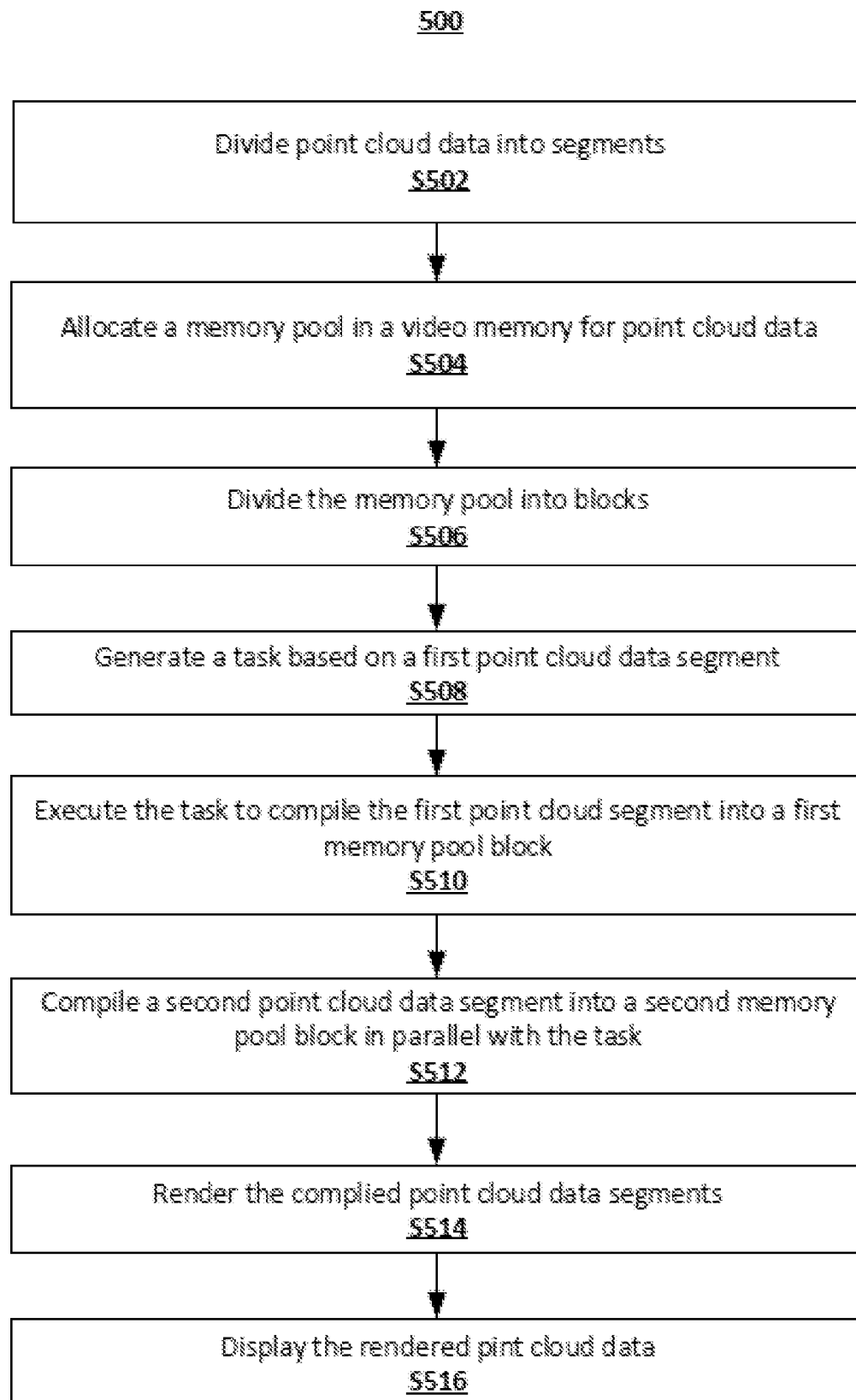


FIG. 5

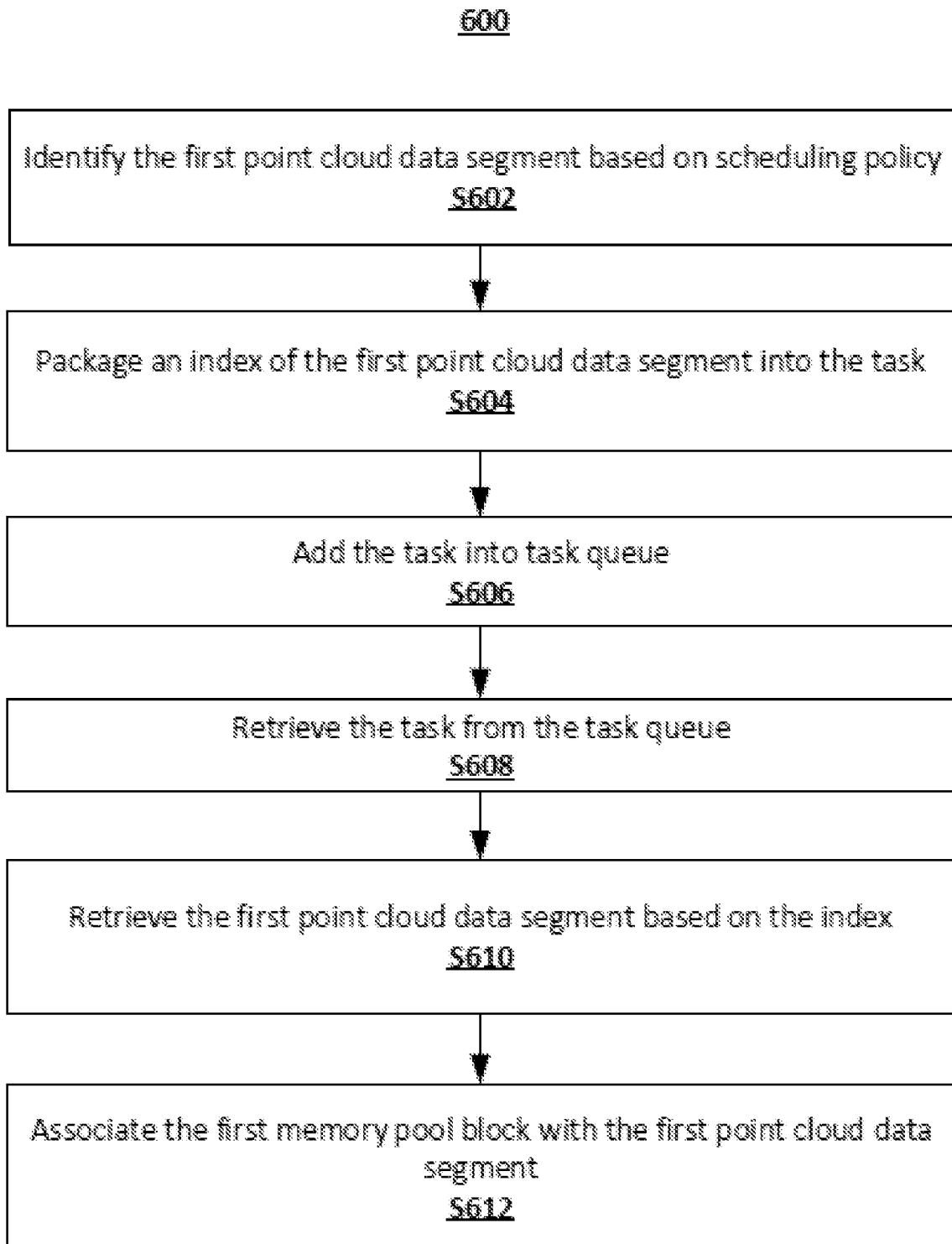


FIG. 6

INTERNATIONAL SEARCH REPORT

International application No.

PCT/CN2018/117371

A. CLASSIFICATION OF SUBJECT MATTER

G06T 15/00(2011.01)i

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)

G06F; G06T

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)

DWPI;VEN;CNABS;CNTXT: point, cloud, data, render, load, display, memory, buffer, video, frame, allocate, divide, partition, segment, block, multiple, thread, parallel

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	CN 104750854 A (WUHAN HI-CLOUD TECHNOLOGY CO., LTD.) 01 July 2015 (2015-07-01) see the description paragraphs [0002]-[0069], figs.1-4	1-20
Y	WO 2015019197 A2 (MOLONEY DAVID ET AL.) 12 February 2015 (2015-02-12) see the description paragraphs [0073]-[0200], figs.1-11	1-20
A	CN 104391906 A (WUHAN HI-CLOUD TECHNOLOGY CO., LTD.) 04 March 2015 (2015-03-04) the whole document	1-20
A	CN 106600570 A (UNIV SOUTHWEST SCI & TECHNOLOGY ET AL.) 26 April 2017 (2017-04-26) the whole document	1-20
A	US 2009060345 A1 (LEICA GEOSYSTEMS AG) 05 March 2009 (2009-03-05) the whole document	1-20



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

07 August 2019

Date of mailing of the international search report

14 August 2019

Name and mailing address of the ISA/CN

National Intellectual Property Administration, PRC
6, Xitucheng Rd., Jimen Bridge, Haidian District, Beijing
100088
China

Authorized officer

SHANG, Qin

Facsimile No. (86-10)62019451

Telephone No. 86-(010)-62089385

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/CN2018/117371

Patent document cited in search report			Publication date (day/month/year)	Patent family member(s)			Publication date (day/month/year)
CN	104750854	A	01 July 2015	CN	104750854	B	02 March 2018
WO	2015019197	A2	19 November 2015	US	2018349147	A1	06 December 2018
				US	9146747	B2	29 September 2015
				JP	2016536692	A	24 November 2016
				KR	20160056881	A	20 May 2016
				EP	3031047	A2	15 June 2016
				WO	2015019197	A2	12 February 2015
				US	2015046678	A1	12 February 2015
				US	2015046677	A1	12 February 2015
				US	9934043	B2	03 April 2018
				CN	105765623	A	13 July 2016
CN	104391906	A	04 March 2015	CN	104391906	B	20 June 2017
CN	106600570	A	26 April 2017	None			
US	2009060345	A1	05 March 2009	US	8204313	B2	19 June 2012
				US	8391610	B2	05 March 2013
				US	2012179690	A1	12 July 2012