

VIRTUAL SCENE-BASED RENDERING METHOD AND APPARATUS, ELECTRONIC DEVICE, COMPUTER-READABLE STORAGE MEDIUM AND COMPUTER PROGRAM PRODUCT

RELATED APPLICATION

[0001] This application claims priority to Chinese Patent Application No. 202111419447.9, filed on November 26, 2021, which is incorporated by reference in its entirety.

FIELD OF THE TECHNOLOGY

[0002] The present disclosure relates to rendering techniques in the field of computer application, and in particular to a virtual scene-based rendering method and apparatus, an electronic device, a computer-readable storage medium, and a computer program product.

BACKGROUND OF THE DISCLOSURE

[0003] In a virtual scene, there is often a need to render liquid surfaces, such as ocean surfaces and water surfaces, and virtual objects interacting with the liquid surfaces. In general, to achieve rendering of the liquid surface, and the virtual object interacting with the liquid surface, fluctuation data of the liquid surface is typically resolved by a graphics processing unit (GPU) to render a high-confidence liquid surface, and the fluctuation data in the GPU is read back to render the virtual object interacting with the liquid surface. However, in the above-mentioned rendering process, there are performance fluctuations during reading back the GPU data, which affects the stability of rendering performance.

SUMMARY

[0004] Embodiments of the present disclosure provide a virtual scene-based rendering method and apparatus, an electronic device, a computer-readable storage medium, and a computer program product, which can improve the stability of rendering performance.

[0005] Technical solutions in the embodiments of the present disclosure are implemented as follows:

[0006] Embodiments of the present disclosure provide a virtual scene-based rendering method, including:

[0007] resolving, by a graphics processing unit (GPU), vertex fluctuation data corresponding to each vertex to be rendered in a model to be rendered to obtain a vertex fluctuation data set corresponding to the model to be rendered;

[0008] dividing, by a central processing unit (CPU), the model to be rendered into a plurality of grids to be rendered, and resolving grid fluctuation data corresponding to each of the grids to be rendered to obtain a grid fluctuation data set corresponding to the plurality of grids to be rendered;

[0009] rendering, by the GPU, a liquid surface to be rendered in a virtual scene based on the vertex fluctuation data set; and

[0010] rendering, by the CPU, a virtual object interacting with the liquid surface to be rendered based on the grid fluctuation data set.

[0011] Embodiments of the present disclosure provide a virtual scene-based rendering apparatus, including:

[0012] a first resolving module, configured to call a graphics processing unit (GPU) to perform the following processing: resolving vertex fluctuation data corresponding to each vertex to be rendered in a model to be rendered to obtain a vertex fluctuation data set corresponding to the model to be rendered;

[0013] a second resolving module, configured to call a central processing unit (CPU) to perform the following processing: dividing the model to be rendered into a plurality of grids to be rendered, and resolving grid fluctuation data corresponding to each of the grids to be rendered to obtain a grid fluctuation data set corresponding to the plurality of grids to be rendered; and

[0014] a data rendering module, configured to call the GPU to perform the following processing: rendering a liquid surface to be rendered in a virtual scene based on the vertex fluctuation data set, and call the CPU to perform the following processing: rendering a virtual object for interacting with the liquid surface to be rendered based on the grid fluctuation data set.

[0015] Embodiments of the present disclosure provide an electronic device for rendering based on a liquid surface, including:

[0016] a memory, configured to store a computer-executable instruction; and

[0017] a processor including a graphics processing unit (GPU) and a central processing unit (CPU), the processor configured to implement, when executing the computer-executable instruction

stored in the memory, the virtual scene-based rendering method provided by embodiments of the present disclosure.

[0018] Embodiments of the present disclosure provide a computer-readable storage medium storing a computer-executable instruction, the computer-executable instruction, when executed by a processor, implementing the virtual scene-based rendering method provided by embodiments of the present disclosure.

[0019] Embodiments of the present disclosure provide a computer program product including a computer program or a computer-executable instruction, the computer program or the computer executable instruction, when executed by a processor, implementing the virtual scene-based rendering method provided by embodiments of the present disclosure.

[0020] Embodiments of the present disclosure have at least the following beneficial effects: since the rendering of the liquid surface to be rendered implemented by the GPU is implemented based on data resolved by the GPU (referred to as the vertex fluctuation data set), and the rendering of the virtual object interacting with the liquid surface to be rendered implemented by the CPU is implemented based on data resolved by the CPU (referred to as the grid fluctuation data set); that is to say, each of the CPU and the GPU can implement the content to be rendered according to the data resolved by itself, so that a resource consumption of the interaction between the CPU and the GPU can be reduced, and thus the stability of rendering performance can be improved.

BRIEF DESCRIPTION OF THE DRAWINGS

[0021] FIG. 1 is a waveform diagram of an exemplary Gerstner wave.

[0022] FIG. 2 is a rendering diagram of an exemplary Gerstner wave.

[0023] FIG. 3 is an architectural diagram of a virtual scene-based rendering system provided by an embodiment of the present disclosure.

[0024] FIG. 4 is a structural diagram of the composition of a terminal in FIG. 3 provided by an embodiment of the present disclosure.

[0025] FIG. 5 is a flow diagram 1 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0026] FIG. 6a is a flow diagram of acquiring a grid fluctuation data set provided by an embodiment of the present disclosure.

[0027] FIG. 6b is a flow diagram 2 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0028] FIG. 6c is a flow diagram 3 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0029] FIG. 6d is a flow diagram 4 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0030] FIG. 6e is a flow diagram 5 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0031] FIG. 6f is a flow diagram 6 of a virtual scene-based rendering method provided by an embodiment of the present disclosure.

[0032] FIG. 6g is a flow diagram of rendering a virtual object provided by an embodiment of the present disclosure.

[0033] FIG. 7 is a flowchart of an exemplary rendering of a liquid surface provided by an embodiment of the present disclosure.

[0034] FIG. 8 is a functional implementation diagram of an exemplary water wave simulation module provided by an embodiment of the present disclosure.

[0035] FIG. 9 is a flow diagram of an exemplary implementation of a sub-module running in a CPU provided by an embodiment of the present disclosure.

[0036] FIG. 10 is a flow diagram of an exemplary implementation of a sub-module running in a GPU provided by an embodiment of the present disclosure.

[0037] FIG. 11 is a diagram of an exemplary shading simulation module implementing water shading provided by an embodiment of the present disclosure.

[0038] FIG. 12 is a flow diagram of an exemplary implementation of a shading rendering module provided by an embodiment of the present disclosure.

[0039] FIG. 13 is a diagram of an exemplary simulation of water fluctuations provided by an embodiment of the present disclosure.

[0040] FIG. 14 is a diagram of another exemplary simulation of water fluctuations provided by an embodiment of the present disclosure.

[0041] FIG. 15 is a diagram of yet another exemplary simulation of water fluctuations provided by an embodiment of the present disclosure.

[0042] FIG. 16 is a diagram of an exemplary interaction of water fluctuations with a virtual object provided by an embodiment of the present disclosure.

[0043] FIG. 17 is a diagram of an exemplary water surface lighting rendering effect provided by an embodiment of the present disclosure.

[0044] FIG. 18 is a diagram of another exemplary water surface lighting rendering effect provided by an embodiment of the present disclosure.

DESCRIPTION OF EMBODIMENTS

[0045] To make the objectives, technical solutions, and advantages of the present disclosure clearer, the following describes the present disclosure in further detail with reference to the accompanying drawings. The described embodiments are not to be considered as a limitation to the present disclosure. All other embodiments obtained by a person skilled in the art without creative efforts shall fall within the protection scope of the present disclosure.

[0046] In the following description, the term "some embodiments" is involved, which describes subsets of all possible embodiments, but it may be understood that "some embodiments" may be the same subset or different subsets of all the possible embodiments, and may be combined with each other without conflict.

[0047] In the following description, the terms "first" and "second" merely distinguish between similar objects and do not denote a particular ordering of the objects, and it may be understood that the terms "first" and "second" may be interchanged either in a particular order or in a sequential order, as permitted, to enable embodiments of the present disclosure described herein to be implemented otherwise than as specifically illustrated or described herein.

[0048] Unless otherwise defined, meanings of all technical and scientific terms used in this specification are the same as those usually understood by a person skilled in the art to which the present disclosure belongs. The terms used in embodiments of the present disclosure are for the purpose of describing embodiments of the present disclosure only and are not intended to limit the present disclosure.

[0049] Before embodiments of the present disclosure are further described in detail, a description is made on nouns and terms in embodiments of the present disclosure, and the nouns and terms in the embodiments of the present disclosure are applicable to the following explanations.

[0050] 1) Fluid dynamics, which is a branch of fluid mechanics, is used for studying a motion law of a fluid as a continuous medium under an action of force and its interaction with a boundary, including an interaction of the fluid and other motion forms. The difference between fluid dynamics and fluid statics is that fluid dynamics studies the fluid in motion. The difference between fluid dynamics and fluid kinematics is that fluid dynamics is used for studying the force

acting on the fluid. In addition, fluid dynamics includes two major parts, hydrodynamics and aerodynamics, and embodiments of the present disclosure involve hydrodynamics, describing ripples generated by a liquid surface under the action of force involved in liquid dynamics.

[0051] 2) A ripple refers to a fluid mechanics effect of environmental forces, such as wind, on a water surface, including Gerstner wave (also referred to as Trochoidal wave). The essence of the Gerstner wave is an exact solution of Euler equations for periodic gravity waves on a liquid surface. A gravity wave refers to a wave in fluid mechanics within a liquid medium or between two medium interfaces (such as between the atmosphere and the ocean). When a clump of liquid leaves the liquid surface (two medium interface types) or reaches a region in the liquid with different liquid densities (liquid medium internal type), through gravity, the clump of liquid swings between equilibrium states in the form of fluctuation, forming the gravity wave. The Euler equation, i.e., differential equation of motion, is a fundamental equation in inviscid fluid dynamics and is the differential equation of motion obtained by applying Newton's second law to inviscid fluid micelles. In fluid dynamics, the Euler equation is a group of equations governing a motion of an inviscid fluid. In embodiments of the present disclosure, various fluctuation data involved are used for rendering ripples of the liquid surface.

[0052] 3) CPU hyper-threading refers to the use of hardware instructions of special characters to simulate a plurality of logical cores into a physical chip, so that a single processor can use thread-level parallel computing, thereby compatible with multi-threaded parallel computing, and then compatible with multi-threaded operating systems and software to improve operating performance. In embodiments of the present disclosure, the resolving of fluctuation data performed by the CPU may be implemented using CPU hyper-threading.

[0053] 4) A physics engine is configured to compute motion, rotation, and collision reflections by assigning true physical properties to rigid objects. In embodiments of the present disclosure, motion information of a virtual object interacting with a liquid surface may be determined by the physics engine.

[0054] 5) A shader, an editable program for implementing image rendering in place of a fixed rendering pipeline, is editable so that various image rendering effects can be implemented without being limited by the fixed rendering pipeline of a graphics card. In embodiments of the present disclosure, the shader includes a compute shader and a fragment shader (also referred to as pixel shader). The compute shader is used for processing some computing work unrelated to rendering using resources of the shader, and in embodiments of the present disclosure, the compute shader is used for performing parallel processing on a large amount (greater than a first number

threshold) of vertex data. The fragment shader is used for performing lighting computing for diffuse and/or specular reflections.

[0055] 6) A rendering pipeline, such as a geometric pipeline, is configured to define computing methods and flows on the GPU.

[0056] 7) A rasterization engine is configured to implement rasterization. Rasterization is a process of converting the vertex data into a fragment, and each element in the fragment may correspond to one pixel in a frame buffer so that rasterization is a process of converting a geometric primitive into a two-dimensional image.

[0057] 8) A virtual scene refers to a scene outputted using a device and being different from the real world, and a visual perception of the virtual scene can be formed by the naked eye or the assistance of the device; for example, a two-dimensional image outputted by a display screen, a three-dimensional image outputted by stereoscopic display technologies such as stereoscopic projection, virtual reality, and augmented reality technology. In addition, various simulated real-world perceptions, such as auditory perception, tactile perception, olfactory perception, and motion perception, may also be formed by various possible hardware.

[0058] The virtual scene displayed (or provided) by an application when running on a terminal device may be a real-world simulation environment, a semi-simulation semi-fictional virtual environment, and a purely fictional virtual environment. The virtual scene may be any one of a two-dimensional virtual scene, a 2.5-dimensional virtual scene, or a three-dimensional virtual scene, and dimensions of the virtual scene are not limited in embodiments of the present disclosure. For example, the virtual scene may include a sky, a land, a sea, etc. the land may include environmental elements such as a desert, a city, and a building, and the virtual object may move or perform other operations (such as an attack operation) within the virtual scene under the control of a user or artificial intelligence (AI).

[0059] 9) Virtual objects are images of various people and things that can interact in the virtual scene, or a movable object in the virtual scene. This movable object may be a virtual character, a virtual animal, an animation character, etc. such as a character, an animal, a plant, an oil bucket, a wall, a stone, and a building, displayed in the virtual scene. This virtual object may be a virtual image in this virtual scene for representing the user. The virtual scene may include a plurality of virtual objects, and each virtual object has its own shape and volume in the virtual scene and occupies some space in the virtual scene. In embodiments of the present disclosure, the virtual object involved is the virtual object interacting with the liquid surface to be rendered.

[0060] In general, in order to render ripples of the liquid surface, this is usually implemented by rendering the Gerstner wave. Referring to FIG. 1, FIG. 1 is a waveform diagram of an exemplary Gerstner wave. As shown in FIG. 1, a Gerstner wave 1-2 under an action of a wind direction 1-1 is rendered. The Gerstner wave is widely used because it takes up computing resources that can be scaled in the rendering process and has good rendering performance and real-time performance. However, since the Gerstner wave is rendered by scheduling a large amount (greater than a second number threshold) of threads, the Gerstner wave is typically rendered by the GPU. Referring to FIG. 2, FIG. 2 is a rendering diagram of an exemplary Gerstner wave. As shown in FIG. 2, a water surface 2-2 is rendered by rendering the Gerstner wave based on parameters set in a region 2-1.

[0061] However, when ripples such as the Gerstner wave, a fast Fourier transform (FFT) wave, and a wave simulated by a wavicle are rendered by the GPU to realize a rendering of the liquid surface, if a motion effect of the virtual object interacting with the liquid surface also needs to be rendered, the fluctuation data on the GPU chip needs to be transmitted to a memory through a GPU bus (GPU BUS) for the CPU to process, because a functional module (such as a buoyancy settlement module, the physics engine, etc.) for rendering the motion effect of the virtual object interacting with the liquid surface is implemented by the CPU. Such is a read-back process, and the read-back of data per frame will result in performance fluctuations on the order of milliseconds (e.g., 5 milliseconds). Therefore, the rendering of the liquid surface and the virtual object interacting with the liquid surface is implemented by rendering ripples such as Gerstner waves by the GPU, influencing the stability of the rendering.

[0062] In addition, when the rendering of the liquid surface is implemented by rendering ripples such as Gerstner waves by the GPU, the CPU is often in an idle state to wait for the processing of the GPU, thus causing a waste of CPU resources.

[0063] Based on this, embodiments of the present disclosure provide a virtual scene-based rendering method and apparatus, an electronic device, a computer-readable storage medium, and a computer program product, which can improve rendering stability and resource utilization. Exemplary application of an electronic device for rendering based on the liquid surface (hereinafter simply referred to as a rendering device) provided by embodiments of the present disclosure is described below. The rendering device provided by embodiments of the present disclosure may be implemented as various types of terminals, such as a smart phone, a smart watch, a notebook computer, a tablet computer, a desktop computer, a smart television, a set-top box, a smart car device, a portable music player, a personal digital assistant, a dedicated messaging device, a

portable game device, and a smart speaker, and may also be implemented as a server. In the following, exemplary application when the rendering device is implemented as the terminal will be described.

[0064] Referring to FIG. 3, FIG. 3 is an architectural diagram of a virtual scene-based rendering system provided by an embodiment of the present disclosure. As shown in FIG. 3, in order to support one virtual scene-based rendering application, in the virtual scene-based rendering system 100, a terminal 400 (referred to as the rendering device, and a terminal 400-1 and a terminal 400-2 are exemplarily shown) is connected to a server 200 through a network 300. The network 300 may be a wide area network or a local area network, or a combination of both, and the server 200 is configured to provide computing services to the terminal 400. In addition, this virtual scene-based rendering system 100 also includes a database 500 for providing data support to the server 200. Also, a case where the database 500 is independent of the server 200 is shown in FIG. 3. Furthermore, the database 500 may be integrated in the server 200, and embodiments of the present disclosure do not limit this.

[0065] The terminal 400 is configured to resolve, by the GPU, vertex fluctuation data corresponding to each vertex to be rendered in a model to be rendered to obtain a vertex fluctuation data set corresponding to the model to be rendered; to divide, by the CPU, the model to be rendered into a plurality of grids to be rendered, and to resolve grid fluctuation data corresponding to each of the grids to be rendered to obtain a grid fluctuation data set corresponding to the plurality of grids to be rendered. In the virtual scene, a liquid surface to be rendered is rendered by the GPU based on the vertex fluctuation data set, and the virtual object interacting with the liquid surface to be rendered is rendered by the CPU based on the grid fluctuation data set (referring to the water surface and a tank floating on the water surface shown in the terminal 400-1, referring to the sea surface and a ship floating on the sea surface shown in the terminal 400-2).

[0066] In some embodiments, the server 200 may be an independent physical server, may be a server cluster or distributed system composed of a plurality of physical servers, and may also be a cloud server providing basic cloud computing services such as a cloud service, a cloud database, cloud computing, a cloud function, cloud storage, a network service, a cloud communication, a middleware service, a domain name service, a security service, a content delivery network (CDN), and a large data and artificial intelligence platform. The terminal 400 may be, but is not limited to, the smart phone, the smart watch, the notebook computer, the tablet computer, the desktop computer, the smart television, the set-top box, the smart car device, the portable music player, the personal digital assistant, the dedicated messaging device, the portable game device, and

the smart speaker, etc. The terminal and the server may be directly or indirectly connected by wired or wireless communication, and are not limited in embodiments of the present disclosure.

[0067] When the server 200 is a cloud game server, the server 200 may render the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, and transmit audio and video information characterizing a rendered liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered to the terminal 400 for playing.

[0068] Referring to FIG. 4, FIG. 4 is a structural diagram of the composition of a terminal in FIG. 3 provided by an embodiment of the present disclosure, and the terminal 400 shown in FIG. 4 includes: at least one processor 410, a memory 450, at least one network interface 420, and a user interface 430. Various components in the terminal 400 are coupled together by a bus system 440. It may be understood that, the bus system 440 is configured to implement connection and communication between the components. In addition to a data bus, the bus system 440 also includes a power bus, a control bus, and a state signal bus. However, for ease of clear description, all types of buses in FIG. 4 are marked as the bus system 440.

[0069] The processor 410 may be an integrated circuit chip, such as a general purpose processor, a digital signal processor (DSP), or other programmable logic devices, discrete gate or transistor logic devices, discrete hardware components, etc. having a signal processing capability, where the general purpose processor may be a microprocessor or any conventional processor, etc.

[0070] The user interface 430 includes one or more output apparatus 431 that enable the presentation of media content, including one or more speakers and/or one or more visual display screens. The user interface 430 also includes one or more input apparatus 432, including user interface elements that facilitate user input, such as a keyboard, a mouse, a microphone, a touch display screen, a camera, other input buttons, and controls.

[0071] The memory 450 may be removable, non-removable, or a combination thereof. An exemplary hardware device includes a solid-state memory, a hard disk drive, an optical disc drive, etc. The memory 450 optionally includes one or more storage devices physically located away from the processor 410.

[0072] The memory 450 includes either a volatile memory or a nonvolatile memory, and may also include both the volatile memory and the nonvolatile memory. The nonvolatile memory may be a read only memory (ROM), and the volatile memory may be a random access memory (RAM). The memory 450 described in embodiments of the present disclosure is intended to include any suitable type of memory.

[0073] In some embodiments, the memory 450 can store data to support various operations, and examples of these data include programs, modules, and data structures, or subsets or supersets thereof, as exemplified below.

[0074] An operating system 451 includes a system program for processing various basic system services and performing hardware-related tasks, such as a framework layer, a core library layer, a driver layer, etc. for implementing various basic businesses and processing hardware-based tasks.

[0075] A network communication module 452 is configured to reach other electronic devices via one or more (wired or wireless) network interfaces 420, and an exemplary network interface 420 includes: Bluetooth, wireless fidelity (Wi-Fi), and a universal serial bus (USB), etc.

[0076] A presentation module 453 is configured to enable the presentation of information (e.g., a user interface configured to operate a peripheral device and display content and information) via one or more output apparatus 431 (e.g., the display screen, the speaker, etc.) associated with the user interface 430.

[0077] An input processing module 454 is configured to detect one or more user inputs or interactions from one of the one or more input apparatus 432 and translate detected inputs or interactions.

[0078] In some embodiments, a virtual scene-based rendering apparatus provided by embodiments of the present disclosure may be implemented in software, and FIG. 4 shows a rendering apparatus 455 stored in a memory 450, which may be software in the form of a program and a plug-in, etc. including the following software modules: a first resolving module 4551, a second resolving module 4552, a data rendering module 4553, a shading rendering module 4554, and a parameter acquisition module 4555. These modules are logical, and therefore may be arbitrarily combined or further split according to the functions implemented. Functions of the various modules will be described below. The modules are in the form of software, programs and other software run by the CPU and the GPU.

[0079] In some embodiments, the rendering apparatus provided by embodiments of the present disclosure may be implemented in hardware, and as an example, the rendering apparatus provided by embodiments of the present disclosure may be a processor in the form of a hardware decoding processor, which is programmed to perform the virtual scene-based rendering method provided by embodiments of the present disclosure, for example, the processor in the form of the hardware decoding processor may adopt one or more application specific integrated circuit (ASIC),

DSP, programmable logic device (PLD), complex programmable logic device (CPLD), field-programmable gate array (FPGA), or other electronic elements.

[0080] In some embodiments, the terminal or the server may implement the virtual scene-based rendering method provided by embodiments of the present disclosure by running a computer program. For example, the computer program may be a native program or the software module in the operating system; may be a native application (APP), i.e., a program which needs to be installed in the operating system to run, such as a video playing APP or a game application APP (including a local game application APP and a cloud game application APP); may be a mini program, i.e., a program which only needs to be downloaded into a browser environment to run; may also be a mini program which may be embedded into any APP. In general, the above-mentioned computer program may be any form of application, module, or plug-in.

[0081] Hereinafter, the virtual scene-based rendering method provided by embodiments of the present disclosure will be described in conjunction with exemplary application and implementation of the rendering apparatus provided by embodiments of the present disclosure. This virtual scene-based rendering method is performed by the rendering device, and the rendering device includes the CPU and the GPU. In addition, this rendering device may be one device including the CPU and the GPU simultaneously, may be a combination of a device including the CPU and a device including the GPU, or may also be a combination of the two, etc. and embodiments of the present disclosure do not limit this. For example, the CPU and the GPU are provided in a mobile phone or a desktop computer simultaneously; for another example, the desktop computer is externally connected to one or more GPUs through an expansion interface (an ExpressCard interface) or a board card (a mini-PCIe interface), and a built-in and an externally connected GPU are simultaneously used; for yet another example, the CPU is provided in the desktop computer, the GPU is not provided, and one or more GPUs are externally connected by the ExpressCard interface or the mini-PCIe interface; so on.

[0082] Referring to FIG. 5, FIG. 5 is a flow diagram 1 of a virtual scene-based rendering method provided by an embodiment of the present disclosure, which will be described in conjunction with the steps shown in FIG. 5.

[0083] S501. The GPU resolves the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered.

[0084] In embodiments of the present disclosure, the rendering device, when rendering the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered,

is implemented by parallel processing of the CPU and the GPU. In the parallel processing of the CPU and the GPU, the GPU is configured to resolve fluctuation data of a fine granularity corresponding to the model to be rendered, which is referred to herein as the vertex fluctuation data set, and this vertex fluctuation data set is used for rendering ripples of the liquid surface to be rendered.

[0085] Here, when rendering the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, the rendering device calls the GPU so that the GPU resolves fluctuation data of the model to be rendered in units of the vertex to be rendered. When resolving the fluctuation data corresponding to each vertex to be rendered (referred to as the vertex fluctuation data), the rendering device combines the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered, and the vertex fluctuation data set corresponding to the model to be rendered is obtained.

[0086] The liquid surface to be rendered is a surface of a liquid to be rendered, such as a surface of the water (a surface of the sea, a surface of a lake), a surface of other solutions. The model to be rendered is a model for rendering the liquid surface to be rendered and may be a three-dimensional structure model or a two-dimensional structure model, etc. and embodiments of the present disclosure do not limit this. The virtual object interacting with the liquid surface to be rendered may be a virtual object carried by the liquid surface to be rendered, such as the ship, a floating person or object. In addition, the model to be rendered is composed of a mesh, and the mesh is composed of a polygon, and vertices of the polygon are vertices to be rendered so that the model to be rendered includes a plurality of vertices to be rendered. Furthermore, the vertex fluctuation data is fluctuation data of the vertex to be rendered. The vertex fluctuation data set is fluctuation data of the fine granularity (referred to as a granularity of the vertex to be rendered) of the liquid surface to be rendered, and the vertex fluctuation data set is a set composed of a plurality of vertex fluctuation data corresponding to a plurality of vertices to be rendered one by one.

[0087] S502. The CPU divides the model to be rendered into a plurality of grids to be rendered and resolves the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered.

[0088] In embodiments of the present disclosure, during the parallel rendering of the CPU and the GPU of the rendering device, the CPU is configured to resolve fluctuation data of a coarse granularity (referred to as a granularity of the grid to be rendered) corresponding to the model to be rendered, which is referred to herein as the grid fluctuation data set. This grid fluctuation data set is used for determining motion information of the virtual object interacting with the liquid surface to

be rendered, and then rendering the virtual object interacting with the liquid surface to be rendered based on the motion information.

[0089] Here, when rendering the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, the rendering device calls the GPU so that the GPU resolves the fluctuation data of the model to be rendered in units of the vertex to be rendered, and also calls the CPU so that the CPU divides the model to be rendered into a plurality of grids to be rendered, and then resolves the fluctuation data of the model to be rendered in units of the grid to be rendered. When resolving the fluctuation data corresponding to each grid to be rendered (referred to as the grid fluctuation data), the rendering device combines the grid fluctuation data corresponding to each grid to be rendered in the model to be rendered, and the grid fluctuation data set corresponding to the model to be rendered is obtained.

[0090] The grid to be rendered is obtained by dividing a region of the model to be rendered and is a region of a specified size, and the granularity of the grid to be rendered is greater than that of the vertex to be rendered. For example, the grid to be rendered includes at least two vertices to be rendered. In addition, the grid fluctuation data are fluctuation data of the grid to be rendered. The grid fluctuation data set is fluctuation data of the coarse granularity (the granularity of the grid to be rendered) of the model to be rendered, and the grid fluctuation data set is a set composed of a plurality of grid fluctuation data corresponding to a plurality of grids to be rendered one by one. Accordingly, a resolving granularity of the GPU is finer than that of the CPU.

[0091] In some embodiments, S501 and S502 are performed in order. For example, S501 is performed first, and then S502 is performed after S501. In other embodiments, S501 and S502 are not performed in order of execution. For example, S501 and S502 are performed in parallel. That is to say, an execution of S501 is performed during an execution of S502, i.e., in the process of the CPU acquiring the grid fluctuation data set. The execution of S502 is performed during the execution of S501, i.e., in the process of the GPU acquiring the vertex fluctuation data set. Accordingly, in S502, the rendering device divides, by the CPU, the model to be rendered into a plurality of grids to be rendered and resolves the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered, including: in the process of the GPU acquiring the vertex fluctuation data set, the rendering device divides, by the CPU, the model to be rendered into the plurality of grids to be rendered and resolves the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered.

[0092] S503. The GPU renders the liquid surface to be rendered in the virtual scene based on the vertex fluctuation data set, and the CPU renders the virtual object interacting with the liquid surface to be rendered based on the grid fluctuation data set.

[0093] In embodiments of the present disclosure, after the GPU of the rendering device obtains the vertex fluctuation data set for rendering the ripples of the liquid surface to be rendered, the GPU renders the ripples of the liquid surface to be rendered based on the vertex fluctuation data set, and thus rendering the liquid surface to be rendered in the virtual scene is realized. And after the CPU of the rendering device obtains the grid fluctuation data set for rendering a posture of the virtual object interacting with the liquid surface to be rendered, the CPU renders the posture of the virtual object interacting with the liquid surface to be rendered based on the grid fluctuation data set, and thus rendering the virtual object interacting with the liquid surface to be rendered in the virtual scene is realized.

[0094] The rendering of the liquid surface to be rendered is performed by the GPU and the rendering of the virtual object interacting with the liquid surface to be rendered is performed by the CPU.

[0095] In the process of resolving the fluctuation data of the liquid surface to be rendered (referred to as the vertex fluctuation data set) by the GPU, simultaneously resolving of the coarse granularity is performed on the fluctuation data of the liquid surface to be rendered by the CPU. Accordingly, while the GPU realizes accurate rendering of the liquid surface to be rendered, the CPU can render the virtual object interacting with the liquid surface to be rendered based on fluctuation data of the coarse granularity (referred to as the grid fluctuation data set) resolved by the CPU. Thus, since the rendering of the liquid surface to be rendered implemented by the GPU is implemented based on data resolved by the GPU, and the rendering of the virtual object interacting with the liquid surface to be rendered implemented by the CPU is implemented based on data resolved by the CPU; that is to say, each of the CPU and the GPU can implement the content to be rendered according to the data resolved by itself, so that a resource consumption of the interaction between the CPU and the GPU can be reduced, and thus the stability of rendering performance can be improved. In addition, while the GPU resolves the fluctuation data for rendering the ripples of the liquid surface to be rendered, the CPU resolves the fluctuation data for rendering the posture of the virtual object interacting with the liquid surface to be rendered, thereby enabling parallel rendering of the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, improving resource utilization.

[0096] Referring to FIG. 6a, FIG. 6a is a flow diagram of acquiring a grid fluctuation data set provided by an embodiment of the present disclosure. As shown in FIG. 6a, in embodiments of the present disclosure, in S502, the CPU in the rendering device resolves the grid fluctuation data corresponding to each grid to be rendered to obtain the grid fluctuation data set corresponding to a plurality of grids to be rendered, including S5021 to S5023, and each step is respectively described below.

[0097] S5021. Create a first fluctuation data resolving task for each of the grids to be rendered to obtain a plurality of first fluctuation data resolving tasks corresponding to the plurality of grids to be rendered.

[0098] In embodiments of the present disclosure, the CPU may resolve each of the grid fluctuation data corresponding to each of the grids to be rendered either discretely or in parallel. Here, when the CPU is multi-core, hyper-threading may be realized, and at this time, the CPU may obtain the grid fluctuation data set through the parallel processing.

[0099] The CPU creates one task for resolving corresponding grid fluctuation data for each of the grids to be rendered, that is to say, the first fluctuation data resolving task is also created. Accordingly, for the plurality of grids to be rendered, the plurality of first fluctuation data resolving tasks are also created. The first fluctuation data resolving task is used for resolving the grid fluctuation data corresponding to each of the grids to be rendered, and the plurality of grids to be rendered correspond to the plurality of first fluctuation data resolving tasks one by one.

[00100] S5022. Divide the plurality of the first fluctuation data resolving tasks into at least one fluctuation data resolving thread.

[00101] In embodiments of the present disclosure, the CPU allocates a plurality of constructed first fluctuation data resolving tasks to at least one thread (referred to as at least one fluctuation data resolving thread). Here, one thread corresponds to one task queue, the fluctuation data resolving thread is one thread, and this fluctuation data resolving thread corresponds to a task queue composed of at least one first fluctuation data resolving task in the plurality of first fluctuation data resolving tasks. A number of the first fluctuation data resolving task in the plurality of first fluctuation data resolving tasks is greater than or equal to a number of the fluctuation data resolving thread in the at least one fluctuation data resolving thread.

[00102] S5023. Perform the at least one fluctuation data resolving thread to obtain the grid fluctuation data set.

[00103] In embodiments of the present disclosure, when the at least one fluctuation data resolving thread is the plurality of fluctuation data resolving threads, the CPU may perform the plurality of fluctuation data resolving threads in parallel, or may perform the plurality of fluctuation data resolving threads in series, or may be a combination of parallel and serial, etc., and embodiments of the present disclosure do not limit this.

[00104] A process of the CPU performing the plurality of fluctuation data resolving threads is a process of performing each first fluctuation data resolving task to resolve the grid fluctuation data corresponding to each of the grids to be rendered. Accordingly, when the CPU completes the execution of the plurality of fluctuation data resolving threads, the grid fluctuation data corresponding to each grid to be rendered in the liquid surface to be rendered is also obtained, and thus the grid fluctuation data set composed of the grid fluctuation data corresponding to each grid to be rendered in the model to be rendered is also obtained.

[00105] Through using the multi-core and hyper-threading processing of the CPU, the process of resolving the fluctuation data of the coarse granularity of the model to be rendered in the CPU is realized, thereby improving the stability and resource utilization of rendering the liquid surface.

[00106] Referring to FIG. 6b, FIG. 6b is a flow diagram 2 of a virtual scene-based rendering method provided by an embodiment of the present disclosure. As shown in FIG. 6b, in embodiments of the present disclosure, in S501, the GPU of the rendering device resolves the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered, including S5011 and S5012, and each step is respectively described below.

[00107] S5011. Traverse a set of vertices to be rendered corresponding to the model to be rendered.

[00108] The plurality of vertices to be rendered included in the model to be rendered is the set of vertices to be rendered. Here, the GPU traverses the plurality of vertices to be rendered in the set of vertices to be rendered to resolve the vertex fluctuation data of the vertices to be rendered one by one.

[00109] S5012. Resolve the vertex fluctuation data corresponding to each traversed vertex to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered.

[00110] The GPU resolves the fluctuation data for each traversed vertex to be rendered, and the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered

is resolved, and thus the vertex fluctuation data set is obtained by combining the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered.

[00111] In embodiments of the present disclosure, S5011 and S5012 may be implemented by the geometric pipeline (i.e., at least one of the fragment shaders and a vertex shader) in the GPU.

[00112] Referring to FIG. 6c, FIG. 6c is a flow diagram 3 of a virtual scene-based rendering method provided by an embodiment of the present disclosure. As shown in FIG. 6c, in embodiments of the present disclosure, in S501, the GPU of the rendering device resolves the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered, including S5013 and S5015, and each step is respectively described below.

[00113] S5013. Divide the set of vertices to be rendered corresponding to the liquid surface to be rendered based on a division unit to obtain a plurality of meshes to be rendered.

[00114] In embodiments of the present disclosure, the GPU may divide the model to be rendered into the plurality of meshes to be rendered, where the division unit includes a specified number of vertices to be rendered, and thus each mesh to be rendered includes a specified number of vertices to be rendered. The mesh to be rendered differs from the grid to be rendered in that the mesh to be rendered is divided based on a number of vertices to be rendered, and the grid to be rendered is divided based on the region. And, the specified number is a number of threads in a thread bundle, such as, 32*32, 64*64.

[00115] S5014. Create a second fluctuation data resolving task for each of the vertices to be rendered in each of the meshes to be rendered to obtain a second fluctuation data resolving task group corresponding to the mesh to be rendered, and obtain a plurality of second fluctuation data resolving task groups corresponding to the plurality of meshes to be rendered.

[00116] In embodiments of the present disclosure, the GPU creates a task for resolving corresponding vertex fluctuation data for each of the vertices to be rendered, that is to say, the second fluctuation data resolving task is also created. Accordingly, for the specified number of vertices to be rendered which are included in the mesh to be rendered, a specified number of second fluctuation data resolving tasks are also created. Here, the specified number of second fluctuation data resolving tasks are grouped into the second fluctuation data resolving tasks group. Accordingly, for the plurality of meshes to be rendered, the plurality of second fluctuation data resolving tasks groups are also created. The second fluctuation data resolving task is used for resolving the vertex

fluctuation data corresponding to each of the vertices to be rendered, and the plurality of meshes to be rendered correspond to the plurality of second fluctuation data resolving task groups one by one.

[00117] S5015. Perform the plurality of second fluctuation data resolving task groups to obtain the vertex fluctuation data set corresponding to the liquid surface to be rendered.

[00118] In embodiments of the present disclosure, the GPU may perform the plurality of second fluctuation data resolving task groups in parallel, or may perform the plurality of second fluctuation data resolving task groups in series, or may be a combination of parallel and serial, etc., and embodiments of the present disclosure do not limit this. Here, the GPU may correspond each of the second fluctuation data resolving task groups to one thread bundle, and perform each of the second fluctuation data resolving task groups by performing one thread bundle.

[00119] A process of the GPU performing the plurality of second fluctuation data resolving task groups is a process of performing each of the second fluctuation data resolving tasks to resolve the vertex fluctuation data corresponding to each of the vertices to be rendered. Accordingly, when the GPU completes performing the plurality of second fluctuation data resolving task groups, the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered are also obtained, and thus the vertex fluctuation data set composed of the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered is also obtained.

[00120] In embodiments of the present disclosure, S5013 and S5015 may be implemented by the compute shader in the GPU. In addition, the GPU of the rendering device may obtain the vertex fluctuation data set through S5011 and S5012 when the vertex fluctuation data set cannot be obtained through S5013 and S5015.

[00121] Through using the thread bundle processing of the GPU, the thread bundle is corresponding to the mesh to be rendered of a certain size (a specified number of vertices to be rendered), batch computing is realized, and thus lightweight thread scheduling (a number of threads is less than a specified number of threads) is realized, and the efficiency of resolving the fluctuation data of the fine granularity of the liquid surface to be rendered can be improved.

[00122] Referring to FIG. 6d, FIG. 6d is a flow diagram 4 of a virtual scene-based rendering method provided by an embodiment of the present disclosure. As shown in FIG. 6d, in embodiments of the present disclosure, S501 is followed by S504. That is to say, after the GPU in the rendering device resolves the vertex fluctuation data corresponding to each of the vertices to be rendered of the liquid surface to be rendered to obtain the vertex fluctuation data set corresponding

to the liquid surface to be rendered, this virtual scene-based rendering method also includes S504, and this step is described below.

[00123] S504. Determine, by the GPU, shading rendering data corresponding to the vertex fluctuation data set.

[00124] In addition to rendering the ripples of the liquid surface to be rendered, the rendering device also renders the shading effect of the liquid surface to be rendered. Accordingly, the GPU of the rendering device is also configured to determine the shading rendering data corresponding to the vertex fluctuation data set, where the shading rendering data is used for rendering the shading effect of the liquid surface to be rendered.

[00125] Accordingly, in embodiments of the present disclosure, in S503, the GPU renders the liquid surface to be rendered based on the vertex fluctuation data set, including: the GPU combines the shading rendering data and the vertex fluctuation data set to render the liquid surface to be rendered in the virtual scene. That is to say, the GPU renders the shading effect of the liquid surface to be rendered based on the shading rendering data, and renders the ripples of the liquid surface to be rendered based on the vertex fluctuation data set, thereby completing the rendering of the liquid surface to be rendered.

[00126] The GPU renders a process of light propagation in the liquid by acquiring the shading rendering data, so it can improve a confidence of liquid ripple color rendering. Furthermore, since the rendering of ripples of the liquid surface to be rendered and the rendering of postures of the virtual object interacting with the liquid surface to be rendered are realized by the parallel processing of the GPU and the CPU, a confidence of the rendering of the liquid surface to be rendered can be improved with stable performance.

[00127] With continued reference to FIG. 6d, in embodiments of the present disclosure, S504 is preceded by S505 to S507. That is to say, before the rendering device determines the shading rendering data corresponding to the vertex fluctuation data set through the GPU, this virtual scene-based rendering method also includes S505 to S507, and each step is respectively described below.

[00128] S505. The CPU gathers lighting environmental data.

[00129] A rendering module gathers data related to shading through the CPU, and then gathers the lighting environmental data. Here, the lighting environmental data includes at least one of scene environmental data, lighting intensity, lighting color, lighting angle, and lighting behavior data. The scene environmental data is, such as, sky environmental data or indoor environmental

data of the virtual scene. The lighting behavior data is, such as, a state of the virtual object in the virtual scene.

[00130] S506. The CPU converts the lighting environmental data into a light rendering parameter.

[00131] The CPU pre-processes and pre-computes the gathered lighting environmental data to convert the lighting environmental data into data for realizing light rendering, and thus the light rendering parameter is obtained.

[00132] S507. The CPU transmits the light rendering parameter to the GPU.

[00133] In embodiments of the present disclosure, the CPU uploads the converted light rendering parameter to the GPU through a parameter uploader, and thus the process of transmitting the light rendering parameter to the GPU is completed.

[00134] Accordingly, in embodiments of the present disclosure, in S504, the GPU determines the shading rendering data corresponding to the vertex fluctuation data set, including: the GPU determines the shading rendering data corresponding to the vertex fluctuation data set based on the light rendering parameter. That is to say, when the CPU transmits the light rendering parameter to the GPU, the GPU receives the light rendering parameter, and the GPU determines data for rendering the shading of the liquid surface to be rendered based on the received light rendering parameter, and obtains the shading rendering data.

[00135] With continued reference to FIG. 6d, in embodiments of the present disclosure, S504 may be realized by S5041 and S5042. That is to say, the GPU determines the shading rendering data corresponding to the vertex fluctuation data set, including S5041 and S5042, and each step is respectively described below.

[00136] S5041. Traverse the vertex fluctuation data set.

[00137] In embodiments of the present disclosure, the GPU traverses a solved vertex fluctuation data set in units of the vertex fluctuation data corresponding to the vertex to be rendered.

[00138] S5042. Perform lighting processing on each traversed vertex fluctuation data to obtain the shading rendering data corresponding to the vertex fluctuation data set.

[00139] In embodiments of the present disclosure, when the GPU performs the lighting processing on each traversed vertex fluctuation data, it may be realized based on the received light rendering parameter transmitted by the CPU. Here, when the GPU completes the lighting processing on each vertex fluctuation data in the vertex fluctuation data set, the shading rendering

data corresponding to the vertex fluctuation data set are also obtained. Here, the lighting processing includes at least one of under-surface scattering processing, subsurface scattering processing, and surface reflection processing. The under-subsurface light scattering process refers to light scattering process below the surface of the liquid surface to be rendered, and the surface reflection process refers to light reflection process on the surface of the liquid surface to be rendered.

[00140] The rendering device performs the lighting processing on the vertex fluctuation data of each of the vertices to be rendered through at least one of the under-surface scattering process, the subsurface scattering process, and the surface reflection process, realizes the rendering of a ripple color of the liquid surface to be rendered from energy conservation and geometrical optics, enhances the fidelity of the liquid surface to be rendered, and enhances the rendering effect of the liquid surface to be rendered.

[00141] Referring to FIG. 6e, FIG. 6e is a flow diagram 5 of a virtual scene-based rendering method provided by an embodiment of the present disclosure. As shown in FIG. 6e, in embodiments of the present disclosure, S501 is preceded by S508 to S510; That is to say, before the GPU of the rendering device resolves the vertex fluctuation data corresponding to each of the vertices to be rendered of the liquid surface to be rendered, the rendering method of the liquid surface also includes S508 to S510, and each step is respectively described below.

[00142] S508. The CPU acquires fluctuation environmental data.

[00143] The fluctuation environmental data refers to data affecting the ripples of the liquid surface to be rendered, including at least one of a wind direction, a wind power, a wave height, a wavelength, and fluctuation behavior data. The fluctuation behavior data is the behavior of the virtual object associated with the fluctuation of the liquid surface to be rendered in the virtual scene.

[00144] S509. The CPU converts the fluctuation environmental data into a fluctuation environmental parameter.

[00145] The CPU pre-processes and pre-computes the gathered fluctuation environmental data to convert the fluctuation environmental data into data for realizing ripple rendering, and thus the fluctuation environmental parameter is obtained.

[00146] S510. The CPU transmits the fluctuation environmental parameter to the GPU.

[00147] In embodiments of the present disclosure, the CPU uploads the converted fluctuation environmental parameter to the GPU through a parameter uploader, that is to say, transmitting the fluctuation environmental parameter to the GPU is completed. Here, when the CPU transmits the

fluctuation environmental parameter to the GPU, the GPU receives the fluctuation environmental parameter.

[00148] In embodiments of the present disclosure, S508 to S510 are performed by the CPU in response to an i^{th} frame rendering request of the liquid surface to be rendered in the virtual scene, where i is a monotonically increasing positive integer variable. That is to say, the virtual scene-based rendering method provided by embodiments of the present disclosure performs rendering of the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered frame-by-frame, and can feed back the rendering parameter obtained in real time to a rendering result.

[00149] Accordingly, in embodiments of the present disclosure, in S501, the GPU of the rendering device resolves the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered, including: the GPU resolves the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered based on the fluctuation environmental parameter. And in S502, the CPU of the rendering device resolves the grid fluctuation data corresponding to each of the grids to be rendered, including: the CPU resolves the grid fluctuation data corresponding to each of the grids to be rendered based on the fluctuation environmental parameter.

[00150] During the rendering of the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, the CPU renders the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered in the virtual scene in real-time in response to the rendering request of each frame, reducing the rendering delay and improving the rendering performance.

[00151] Referring to FIG. 6f, FIG. 6f is a flow diagram 6 of a virtual scene-based rendering method provided by an embodiment of the present disclosure. As shown in FIG. 6f, in embodiments of the present disclosure, in S502, the CPU of the rendering device divides the model to be rendered into the plurality of grids to be rendered, including S5024 and S5025, and each step is respectively described below.

[00152] S5024. Determine a grid division unit based on a thread processing size and the liquid surface to be rendered.

[00153] The thread processing size is the maximum number of threads processed by the CPU, and the grid division unit is negatively correlated with the thread processing size, and is positively

correlated with a size of the liquid surface to be rendered. Here, the grid division unit determined by the CPU is a size of each grid to be rendered.

[00154] S5025. Divide the model to be rendered into the plurality of grids to be rendered based on the grid division unit.

[00155] A number of the grids to be rendered in the plurality of grids to be rendered divided by the CPU is positively correlated with the thread processing size.

[00156] The CPU in the rendering device can maximize the use of the processing resources of the CPU by determining the grid division unit for dividing the model to be rendered based on the processing scale of the thread thereof, and improve the acquisition efficiency of the grid fluctuation data set, thereby improving the rendering efficiency.

[00157] Referring to FIG. 6g, FIG. 6g is a flow diagram of rendering a virtual object provided by an embodiment of the present disclosure. As shown in FIG. 6g, in embodiments of the present disclosure, in S503, the CPU of the rendering device renders the virtual object interacting with the liquid surface to be rendered based on the grid fluctuation data set, including S5031 to S5034, and each step is respectively described below.

[00158] S5031. Determine a buoyancy parameter based on the grid fluctuation data set.

[00159] The CPU determines the buoyancy parameter, including parameters such as a displacement, for determining buoyancy of the liquid surface to be rendered.

[00160] S5032. Determine buoyancy information corresponding to the liquid surface to be rendered based on the buoyancy parameter.

[00161] S5033. Call the physics engine to perform the following processing: determining the motion information of the virtual object interacting with the liquid surface to be rendered based on the buoyancy information.

[00162] S5034. Render the virtual object based on the motion information.

[00163] In embodiment of the present disclosure, the CPU determines an acting force (such as the buoyancy information) of the interaction between the liquid surface to be rendered and the virtual object through a grid fluctuation data set resolved by itself, and then renders the state or posture of the virtual object interacting with the liquid surface to be rendered based on the acting force. The physics engine is called by the CPU.

[00164] In the following, exemplary application of embodiments of the present disclosure in one practical application scene will be described. The exemplary application describes a rendering

method of a water surface (a water surface, referred to as the liquid surface to be rendered) in the field of water rendering, such as rendering of the water surface by a three dimension (3D) rendering module running in the game application on a client or a cloud server.

[00165] The rendering of the water surface includes water ripple simulation and ripple lighting shading. The water ripple simulation refers to the simulation of the ripples that occur when the water surface is affected by environmental forces such as a wind direction, a wind power, a wave height, and a wavelength, which generate the fluid mechanics effect. Here, water fluctuation data corresponding to the ripples generated by the water surface driven by the environmental forces are resolved simultaneously by the CPU and the GPU. The water is a complex medium, and the optical shading property is embodied by ripple lighting shading, where the ripple lighting shading refers to simulating a process of light propagating in the water to improve the confidence of a rendered water ripple color.

[00166] Referring to FIG. 7, FIG. 7 is a flowchart of an exemplary rendering of a liquid surface provided by an embodiment of the present disclosure. As shown in FIG. 7, firstly, environmental parameters 7-3 are collected from the state of the virtual object (referred to as a behavior of the virtual object in the virtual environment, for example, driving of a virtual ship) 7-1 and the virtual environment 7-2 where the virtual object is located. A water wave simulation module 7-41 and a shading simulation module 7-42 in a water surface rendering system 7-4 then jointly render a water surface 7-5 based on the environmental parameters 7-3. The water surface rendering system 7-4 is a functional module configured to realize the rendering of the water surface. The water wave simulation module 7-41 is configured to realize the simulation of water surface ripples by simulating the situation of water surface fluctuation, and the shading simulation module 7-42 is configured to realize the ripple lighting shading by simulating the shading effect of water surface ripples under a lighting environment. In addition, besides being influenced by environmental forces such as the wind direction, the wind power, the wave height, and the wavelength, the water ripple simulation may also be influenced by a behavior of the virtual object in the virtual environment. This is because the behavior of the virtual object in the virtual environment changes the environmental forces and thus affects a rendering result of the water surface, so the behavior of the virtual object in the virtual environment may affect the simulation of the water surface.

[00167] In embodiments of the present disclosure, the water wave simulation module 7-41 in FIG. 7 includes a sub-module running in the GPU and a sub-module running in the CPU. Referring to FIG. 8, FIG. 8 is a functional implementation diagram of an exemplary water wave simulation

module provided by an embodiment of the present disclosure. As shown in FIG. 8, on the one hand, a sub-module 8-1 running in the CPU in the water wave simulation module 7-41 is configured to realize a grid division of a water surface region to obtain a grid division result 8-11 (referred to as the plurality of grids to be rendered); on the other hand, is configured to resolve the fluctuation data at a grid level to obtain a coarse granularity fluctuation resolving result 8-12 (referred to as the grid fluctuation data set). On the one hand, a sub-module 8-2 running on the GPU in the water wave simulation module 7-41 is configured to read a water surface vertex 8-21 (referred to as the set of vertices to be rendered). On the other hand, it is configured to traverse the water surface vertex 8-21, resolve fluctuation data vertex-by-vertex (referred to as the vertex to be rendered) to obtain a fine granularity fluctuation resolving result 8-22 (referred to as the vertex fluctuation data set). Here, the coarse granularity fluctuation resolving result 8-12 is used for computing parameters such as the displacement (referred to as buoyancy parameters) through a buoyancy module 8-3 running on the CPU to obtain buoyancy 8-4 (referred to as the buoyancy information), and then a physics engine 8-5 running on the CPU determines motion information 8-6 of the virtual object interacting with the water surface based on the buoyancy 8-4. The fine granularity fluctuation resolving result 8-22 is used for rendering the water fluctuation effect, i.e., rendering a water surface ripple 8-7, through a water surface rendering module of the GPU.

[00168] The water wave simulation module 7-41 in FIG. 7 ensures the correctness of parameters such as the wave geometric effect of the water surface rendering and the displacement in physical resolving. Through the water wave simulation module 7-41, a real simulation of the water surface ripple may be performed according to parameters such as the wind direction, the wind power, the wave height, and the wavelength, and the water surface ripple can be correctly fed back to a rendering animation, and physical forces and motion postures of the interactive object (referred to as the virtual object interacting with the liquid surface to be rendered) at the same time.

[00169] An implementation flow of the sub-module 8-1 running in the CPU in FIG. 8 is described below. Referring to FIG. 9, FIG. 9 is a flow diagram of an exemplary implementation of a sub-module running in a CPU provided by an embodiment of the present disclosure. As shown in FIG. 9, the exemplary implementation flow of the sub-module running in the CPU includes S901 to S907, and each step is respectively described below.

[00170] S901. The terminal (referred to as the rendering device) starts a water surface rendering system.

[00171] In embodiments of the present disclosure, the water surface rendering system 7-4 is started to start the flow of rendering of the water surface.

[00172] S902. The CPU determines whether a frame iteration of the water surface rendering ends. If so, S907 is performed. If not, S903 is performed.

[00173] The water surface and the virtual object interacting with the water surface, are rendered frame-by-frame.

[00174] S903. The CPU performs a grid division on the water surface.

[00175] Here, the grid division is a voxelization process.

[00176] S904. The CPU creates a resolving task of the Gerstner wave (referred to as the first fluctuation data resolving task) for each of the grids.

[00177] S905. The CPU divides the plurality of resolving tasks corresponding to the plurality of grids into different resolving threads for execution to obtain coarse granularity fluctuation resolving data.

[00178] S906. The CPU transmits the coarse granularity fluctuation resolving data to the buoyancy module. Thereafter, S902 is performed.

[00179] The buoyancy module is a module running on the CPU configured to compute the buoyancy. Here, the buoyancy module computes the buoyancy based on the coarse granularity fluctuation resolving data and transmits the buoyancy to the physics engine such that the physics engine determines motion data of the virtual object on the water surface based on the buoyancy. The physics engine is also a module running on the CPU configured to determine motion information of the interactive object.

[00180] S907. The terminal turns off the water surface rendering system.

[00181] The terminal ends the water surface rendering flow through turning off the water surface rendering system.

[00182] An implementation flow of the sub-module 8-2 running in the GPU in FIG. 8 is described below. Based on FIG. 9, referring to FIG. 10, FIG. 10 is a flow diagram of an exemplary implementation of a sub-module running in a GPU provided by an embodiment of the present disclosure. As shown in FIG. 10, the exemplary implementation flow of the sub-module running in the GPU includes S901, S902 and S907, and S1001 to S1008, and each step is respectively described below.

[00183] S1001. The GPU receives the fluctuation environmental parameters (including the environmental forces and the behavior of the virtual object in the virtual environment) uploaded by the CPU.

[00184] Here, when a result of performing S902 is determined to be NO, S1001 is performed.

[00185] S1002. The GPU determines whether the compute shader is available. If so, S1003 is performed. If not, S1006 is performed.

[00186] S1003. The GPU reads a mesh of the water surface (referred to as the mesh to be rendered) through the compute shader.

[00187] S1004. The GPU fixes the thread bundle on the mesh.

[00188] S1005. The GPU performs batch resolving on fluctuation data for each of the vertices on the mesh of a certain size (referred to as a specified number of vertices to be rendered) through the thread bundle. Thereafter, S1008 is performed.

[00189] S1006. The GPU traverses each of the vertices through the geometric pipeline.

[00190] S1007. The GPU resolves the fluctuation data of each of the vertices.

[00191] S1008. The GPU submits the fluctuation data corresponding to all vertices to the water surface rendering module through the geometric pipeline to render the water surface ripple. Thereafter, S902 is performed. The water surface rendering module is a module running on the GPU in the water wave simulation module 7-41.

[00192] The execution processes of S903 to S906 and the execution processes of S1001 to S1008 are processed in parallel.

[00193] In embodiments of the present disclosure, the shading simulation module 7-42 in FIG. 7 includes a GPU functional module (such as a GPU program), a parameter uploader, and a parameter collector; The GPU functional module runs on the GPU, and the parameter uploader and the parameter collector run on the CPU. Referring to FIG. 11, FIG. 11 is a diagram of an exemplary shading simulation module implementing water shading provided by an embodiment of the present disclosure. As shown in FIG. 11, a parameter collector 11-1 in the shading simulation module 7-42 is configured to gather rendered and simulated shading data 11-2 (referred to as the lighting environmental data). A parameter uploader 11-3 in the shading simulation module 7-42 is configured to pre-process and pre-compute the collected shading data 11-2 to obtain a shading parameter 11-4 (referred to as the light rendering parameter), and is also configured to upload the shading parameter 11-4 to a shading rendering module 11-5 in the shading simulation module 7-42. The shading rendering module 11-5 (referred to as the GPU functional module) in the shading simulation module 7-42 runs on the GPU and is configured to perform the lighting processing (including underwater scattering processing, water subsurface scattering processing, and water surface reflection processing) on fine granularity fluctuation resolving data read from the geometric

pipeline based on the shading parameter 11-4 to obtain shading rendering data 11-6. It is also configured to render a water surface ripple color 11-7 based on the shading rendering data 11-6.

[00194] An implementation flow of the shading rendering module 11-5 running in the GPU in FIG. 11 is described below. Based on FIG. 9, referring to FIG. 12, FIG. 12 is a flow diagram of an exemplary implementation of a shading rendering module provided by an embodiment of the present disclosure. As shown in FIG. 12, the exemplary implementation flow of the shading rendering module running in the GPU includes S901, S902 and S907, and S1201 to S1205, and each step is respectively described below.

[00195] S1201. Read a shading parameter uploaded by the CPU.

[00196] Here, when a result of performing S902 is determined to be NO, S1201 is performed.

[00197] S1202. Perform the water surface reflection processing based on the shading parameter and the fine granularity fluctuation resolving data.

[00198] S1203. Perform the water surface scattering processing on a water surface reflection processing result.

[00199] S1204. Perform the subsurface scattering processing on a water surface scattering processing result.

[00200] S1205. Render the water surface ripple based on a subsurface scattering processing result. Thereafter, S902 is performed.

[00201] Rendering and other processing is implemented using the rasterization engine and the geometric pipeline on the GPU, and environmental data are collected by the CPU, and then the collected environmental parameters are uploaded to the GPU through the parameter uploader for reading and accessing by a shader program on the GPU.

[00202] The water fluctuation realized by the water wave simulation module 7-41 in FIG. 7 is described below with reference to FIG. 13 to FIG. 16. Referring to FIG. 13, FIG. 13 is a diagram of an exemplary simulation of water fluctuations provided by an embodiment of the present disclosure. As shown in FIG. 13, a water surface 13-1 is a rendering result when a wind direction 13-21 is forward (-6), a wind power 13-22 is level 1, and a water surface fluctuation intensity coefficient 13-23 is 0.238, where the wind direction 13-21, the wind power 13-22, and the water surface fluctuation intensity coefficient 13-23 are the environmental parameters 7-3 in FIG. 7.

[00203] Referring to FIG. 14, FIG. 14 is a diagram of another exemplary simulation of water fluctuations provided by an embodiment of the present disclosure. As shown in FIG. 14, a water

surface 14-1 is a rendering result when a wind direction 14-21 is rightward (80), a wind power 14-22 is level 5, and a water surface fluctuation intensity coefficient 14-23 is 0.456.

[00204] FIG. 13 and FIG. 14 are representations of the water fluctuations driven by the environmental parameter, that is to say, different fluctuation representations may be generated according to the wind power, the wind direction, and the water surface fluctuation intensity coefficient in the virtual environment.

[00205] Referring to FIG. 15, FIG. 15 is a diagram of yet another exemplary simulation of water fluctuations provided by an embodiment of the present disclosure. As shown in FIG. 15, a water surface 15-1 is a shore wave simulated in the case where wind conditions change.

[00206] Through the water wave simulation module 7-41 in FIG. 7, it is possible to generate various water surface ripples in response to rapid changes in the environment.

[00207] Referring to FIG. 16, FIG. 16 is a diagram of an exemplary interaction of water fluctuations with a virtual object provided by an embodiment of the present disclosure. As shown in FIG. 16, there is shown a ship 16-2 on a water surface 16-1 that is inclined by an incoming sideward wave.

[00208] The water surface rendering method based on the GPU and the CPU parallel rendering of fluctuation data provided by embodiments of the present disclosure can reduce the delay of a physical simulation and can respond to the physics engine in real time, reducing the rendering delay and improving the fidelity of the buoyancy effect.

[00209] The shading effect of the water fluctuation in the lighting environment realized by the shading simulation module 7-42 in FIG. 7 is described below with reference to FIG. 17 and FIG. 18.

[00210] Referring to FIG. 17, FIG. 17 is a diagram of an exemplary water surface lighting rendering effect provided by an embodiment of the present disclosure. As shown in FIG. 17, a water surface 17-1 is a lighting shading simulation result in a clear daylight environment.

[00211] Referring to FIG. 18, FIG. 18 is a diagram of another exemplary water surface lighting rendering effect provided by an embodiment of the present disclosure. As shown in FIG. 18, a water surface 18-1 is a lighting shading simulation result in a dusky environment with heavy waves.

[00212] Through the shading simulation module provided by embodiments of the present disclosure, rendering effects in various lighting environments can be rendered based on changes,

such as the scene (the sky environment), a main lighting intensity, a color, an angle, and a user setting.

[00213] The simulation method of the water surface ripple and the ripple color provided by embodiments of the present disclosure can combine with geometric and optical properties to improve the confidence of the water surface rendering and to improve the rendering effect of the water surface. In addition, through resolving the fluctuation data in parallel in the CPU and the GPU, the posture of the virtual object interacting with the water surface can be quickly obtained, and the rendering efficiency of the virtual object interacting with the water surface can be improved.

[00214] The following continues to describe an exemplary structure of the virtual scene-based rendering apparatus 455 implemented as the software module provided by embodiments of the present disclosure, in some embodiments, as shown in FIG. 4, the software module stored in the rendering apparatus 455 of the memory 450 may include:

[00215] a first resolving module 4551, configured to call the GPU to perform the following processing: resolving the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered;

[00216] a second resolving module 4552, configured to call the CPU to perform the following processing: dividing the model to be rendered into the plurality of grids to be rendered, and resolving the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered; and

[00217] a data rendering module 4553, configured to call the GPU to perform the following processing: rendering a liquid surface to be rendered in a virtual scene based on the vertex fluctuation data set, and call the CPU to perform the following processing: rendering the virtual object interacting with the liquid surface to be rendered based on the grid fluctuation data set.

[00218] In embodiments of the present disclosure, the second resolving module 4552 is further configured to call the CPU to perform the following processing: creating the first fluctuation data resolving task for each of the grids to be rendered to obtain the plurality of first fluctuation data resolving tasks corresponding to the plurality of grids to be rendered, the first fluctuation data resolving task being used for resolving the grid fluctuation data of the grids to be rendered; dividing the plurality of first fluctuation data resolving tasks into at least one fluctuation data resolving thread; and performing the at least one fluctuation data resolving thread to obtain the grid fluctuation data set.

[00219] In embodiments of the present disclosure, the first resolving module 4551 is further configured to call the GPU to perform the following processing: traversing the set of vertices to be rendered corresponding to the model to be rendered; and resolving the vertex fluctuation data corresponding to each traversed vertex to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered.

[00220] In embodiments of the present disclosure, the first resolving module 4551 is further configured to call the GPU to perform the following processing: dividing the set of vertices to be rendered corresponding to the model to be rendered based on the division unit to obtain the plurality of meshes to be rendered, the division unit including a specified number of the vertices to be rendered; creating the second fluctuation data resolving task for each of the vertices to be rendered in each of the meshes to be rendered to obtain the second fluctuation data resolving task group corresponding to the meshes to be rendered, the plurality of meshes to be rendered corresponding to the plurality of second fluctuation data resolving task groups, and the second fluctuation data resolving task being used for resolving the vertex fluctuation data corresponding to each of the vertices to be rendered; and performing the plurality of second fluctuation data resolving task groups to obtain the vertex fluctuation data set corresponding to the model to be rendered.

[00221] In embodiments of the present disclosure, the rendering apparatus 455 further includes the shading rendering module 4554, configured to call the GPU to perform the following processing: determining the shading rendering data corresponding to the vertex fluctuation data set.

[00222] In embodiments of the present disclosure, the data rendering module 4553 is further configured to call the GPU to perform the following processing: combining the shading rendering data and the vertex fluctuation data set to render the liquid surface to be rendered.

[00223] In embodiments of the present disclosure, the shading rendering module 4554 is further configured to call the CPU to perform the following processing: gathering the lighting environmental data, the lighting environmental data including at least one of scene environmental data, lighting intensity, lighting color, lighting angle, and lighting behavior data; converting the lighting environmental data into the light rendering parameter; transmitting the light rendering parameter to the GPU.

[00224] In embodiments of the present disclosure, the shading rendering module 4554 is further configured to call the GPU to perform the following processing: determining the shading rendering data corresponding to the vertex fluctuation data set based on the light rendering parameter.

[00225] In embodiments of the present disclosure, the shading rendering module 4554 is further configured to call the GPU to perform the following processing: traversing the vertex fluctuation data set; and performing lighting processing on each traversed vertex fluctuation data to obtain the shading rendering data corresponding to the vertex fluctuation data set, the lighting processing including at least one of under-surface scattering processing, subsurface scattering processing, and surface reflection processing.

[00226] In embodiments of the present disclosure, the rendering apparatus 455 further includes the parameter acquisition module 4555, configured to call the CPU to perform the following processing: in response to an i^{th} frame rendering request of the liquid surface to be rendered in the virtual scene, performing, the CPU performing the following processing, i being a monotonically increasing positive integer variable: acquiring the fluctuation environmental data, the fluctuation environmental data including at least one of the wind direction, the wind power, the wave height, the wavelength, and the fluctuation behavior data; converting the fluctuation environmental data into the fluctuation environmental parameter; transmitting the fluctuation environmental parameter to the GPU.

[00227] In embodiments of the present disclosure, the first resolving module 4551 is further configured to call the GPU to perform the following processing: resolving the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered based on the fluctuation environmental parameter.

[00228] In embodiments of the present disclosure, the second resolving module 4552 is further configured to call the CPU to perform the following processing: resolving the grid fluctuation data corresponding to each of the grids to be rendered based on the fluctuation environmental parameter.

[00229] In embodiments of the present disclosure, the second resolving module 4552 is further configured to call the CPU to perform the following processing: determining the grid division unit based on the thread processing size and the model to be rendered, the thread processing size being a maximum number of threads processed by the CPU; and dividing the model to be rendered into the plurality of grids to be rendered based on the grid division unit.

[00230] In embodiments of the present disclosure, the data rendering module 4553 is further configured to call the CPU to perform the following processing: determining the buoyancy parameter based on the grid fluctuation data set; determining the buoyancy information corresponding to the liquid surface to be rendered based on the buoyancy parameter; calling the physics engine to perform the following processing: determining the motion information for the

virtual object interacting with the liquid surface to be rendered based on the buoyancy information, the physics engine being called by the CPU; rendering the virtual object based on the motion information.

[00231] In embodiments of the present disclosure, the second resolving module 4552 is further configured to call the CPU to perform the following processing in the process of calling the GPU to acquire the vertex fluctuation data set: dividing the model to be rendered into the plurality of grids to be rendered, and resolving the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered.

[00232] Embodiments of the present disclosure provide a computer program product or computer program including a computer-executable instruction stored in a computer-readable storage medium. A processor (including the GPU and the CPU) of an electronic device (referred to as the rendering device) reads the computer-executable instruction from the computer-readable storage medium, and the processor executes the computer-executable instruction to cause the electronic device to perform the above-mentioned virtual scene-based rendering method in embodiments of the present disclosure.

[00233] Embodiments of the present disclosure provide a computer-readable storage medium storing a computer-executable instruction, the computer-executable instruction, when executed by a processor, cause the processor to perform the virtual scene-based rendering method provided by embodiments of the present disclosure, such as the virtual scene-based rendering method shown in FIG. 5.

[00234] In some embodiments, the computer-readable storage medium may be a memory such as FRAM, ROM, PROM, EPROM, EEPROM, flash memory, magnetic surface storage, optical disk, or CD-ROM; may also be various devices including one or any combination of the above-mentioned memories.

[00235] In some embodiments, the computer-executable instruction may be written in the form of programs, software, software modules, scripts, or code in any form of programming language (including compiled or interpreted languages, or declarative or procedural languages), and may be deployed in any form, including as stand-alone programs or as modules, components, subroutines, or other units suitable for use in a computing environment.

[00236] As an example, the computer-executable instructions may, but need not, correspond to files in a file system, may be stored in a portion of a file that holds other programs or data, e.g., in

one or more scripts in a hyper text markup language (HTML) document, in a single file dedicated to the program in question, or in multiple coordinated files (e.g., files that store one or more modules, subroutines, or portions of code).

[00237] As an example, the computer-executable instructions may be deployed to be executed on one electronic device (in this case, the one electronic device is the rendering device), or on a plurality of electronic devices located at one site (in this case, the plurality of electronic devices located at one site are the rendering devices), or on a plurality of electronic devices distributed at a plurality of sites and interconnected by a communication network (in this case, the plurality of electronic devices distributed at the plurality of sites and interconnected by the communication network are the rendering devices).

[00238] In embodiments of the present disclosure, reference to relevant data, when embodiments of the present disclosure are applied to a particular product or technology, require user permission or consent and that the collection, use, and processing of relevant data requires compliance with relevant national and regional laws and regulations and standards.

[00239] In summary, in the process of resolving the fluctuation data of the liquid surface to be rendered by the GPU, embodiments of the present disclosure also perform the resolving of the coarse granularity on the fluctuation data of the liquid surface to be rendered by the CPU. Accordingly, while the GPU realizes the high confidence rendering of the liquid surface to be rendered, the CPU can render the virtual object interacting with the liquid surface to be rendered based on the fluctuation data of the coarse granularity resolved by the CPU. Thus, since the rendering of the liquid surface to be rendered implemented by the GPU is implemented based on data resolved by the GPU, and the rendering of the virtual object interacting with the liquid surface to be rendered implemented by the CPU is implemented based on data resolved by the CPU; That is to say, the CPU and the GPU may implement rendering only based on data resolved by themselves, thereby improving the stability of rendering performance. In addition, while the GPU resolves the fluctuation data for rendering the ripples of the liquid surface to be rendered, the CPU resolves the fluctuation data for rendering the posture of the virtual object interacting with the liquid surface to be rendered, thereby enabling parallel rendering of the liquid surface to be rendered and the virtual object interacting with the liquid surface to be rendered, improving resource utilization.

[00240] The foregoing descriptions are merely preferred embodiments of the present disclosure and are not intended to limit the protection scope of the present disclosure. Any modifications, equivalent substitutions, and improvements made within the spirit and scope of the present disclosure are covered by the scope of protection of the present disclosure.

CLAIMS

What is claimed is:

1. A virtual scene-based rendering method, comprising:

resolving, by a graphics processing unit (GPU), vertex fluctuation data corresponding to each vertex to be rendered in a model to be rendered to obtain a vertex fluctuation data set corresponding to the model to be rendered;

dividing, by a central processing unit (CPU), the model to be rendered into a plurality of grids to be rendered, and resolving grid fluctuation data corresponding to each of the grids to be rendered to obtain a grid fluctuation data set corresponding to the plurality of grids to be rendered;

rendering, by the GPU, a liquid surface to be rendered in a virtual scene based on the vertex fluctuation data set; and

rendering, by the CPU, a virtual object for interacting with the liquid surface to be rendered based on the grid fluctuation data set.

2. The method according to claim 1, wherein the resolving the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered comprises:

creating a first fluctuation data resolving task for each of the grids to be rendered to obtain the plurality of first fluctuation data resolving tasks corresponding to the plurality of grids to be rendered, the first fluctuation data resolving task used to resolve the grid fluctuation data of the grids to be rendered;

dividing the plurality of first fluctuation data resolving tasks into at least one fluctuation data resolving thread; and

performing the at least one fluctuation data resolving thread to obtain the grid fluctuation data set.

3. The method according to claim 1 or 2, wherein the resolving vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered comprises:

traversing a set of vertices to be rendered corresponding to the model to be rendered; and

resolving the vertex fluctuation data corresponding to each traversed vertex to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered.

4. The method according to claim 1 or 2, wherein the resolving vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered comprises:

dividing the set of vertices to be rendered corresponding to the model to be rendered based on a division unit to obtain a plurality of meshes to be rendered, the division unit comprising a specified number of the vertices to be rendered;

creating a second fluctuation data resolving task for each of the vertices to be rendered in each of the meshes to be rendered to obtain a second fluctuation data resolving task group corresponding to the meshes to be rendered, the plurality of meshes to be rendered corresponding to a plurality of second fluctuation data resolving task groups, and the second fluctuation data resolving task used to resolve the vertex fluctuation data corresponding to each of the vertices to be rendered; and

performing the plurality of second fluctuation data resolving task groups to obtain the vertex fluctuation data set corresponding to the model to be rendered.

5. The method according to claim 1 or 2, wherein after the resolving vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered, the method further comprises:

determining, by the GPU, shading rendering data corresponding to the vertex fluctuation data set;

wherein the rendering the liquid surface to be rendered in the virtual scene based on the vertex fluctuation data set comprises:

rendering the liquid surface to be rendered in the virtual scene in conjunction with the shading rendering data and the vertex fluctuation data set.

6. The method according to claim 5, wherein before the determining shading rendering data corresponding to the vertex fluctuation data set, the method further comprises:

gathering, by the CPU, lighting environmental data, the lighting environmental data comprising at least one of scene environmental data, lighting intensity, lighting color, lighting angle, and lighting behavior data;

converting, by the CPU, the lighting environmental data into a light rendering parameter;

transmitting, by the CPU, the light rendering parameter to the GPU;

wherein the determining the shading rendering data corresponding to the vertex fluctuation data set comprises:

determining the shading rendering data corresponding to the vertex fluctuation data set based on the light rendering parameter.

7. The method according to claim 5, wherein the determining shading rendering data corresponding to the vertex fluctuation data set comprises:

traversing the vertex fluctuation data set; and

performing lighting processing on each traversed vertex fluctuation data to obtain the shading rendering data corresponding to the vertex fluctuation data set, the lighting processing including at least one of under-surface scattering processing, subsurface scattering processing, and surface reflection processing.

8. The method according to claim 1 or 2, wherein before the resolving the vertex fluctuation data corresponding to each vertex to be rendered in the model to be rendered, the method further comprises:

in response to an i^{th} frame rendering request of the liquid surface to be rendered in the virtual scene,

acquiring, by the CPU, fluctuation environmental data, the fluctuation environmental data comprising at least one of a wind direction, a wind power, a wave height, a wavelength, and fluctuation behavior data;

converting, by the CPU, the fluctuation environmental data into a fluctuation environmental parameter; and

transmitting, by the CPU, the fluctuation environmental parameter to the GPU,

wherein i is a monotonically increasing positive integer variable,

wherein the resolving vertex the fluctuation data corresponding to each vertex to be rendered in the model to be rendered comprises:

resolving the vertex fluctuation data corresponding to each of the vertices to be rendered in the model to be rendered based on the fluctuation environmental parameter, and

wherein the resolving grid fluctuation data corresponding to each of the grids to be rendered comprises:

resolving the grid fluctuation data corresponding to each of the grids to be rendered based on the fluctuation environmental parameter.

9. The method according to claim 1 or 2, wherein the dividing the model to be rendered into

the plurality of grids to be rendered comprises:

determining a grid division unit based on a thread processing size and the model to be rendered, the thread processing size being a maximum number of threads processed by the CPU; and

dividing the model to be rendered into the plurality of grids to be rendered based on the grid division unit.

10. The method according to claim 1 or 2, wherein the rendering the virtual object for interacting with the liquid surface to be rendered based on the grid fluctuation data set comprises:

determining a buoyancy parameter based on the grid fluctuation data set;

determining buoyancy information corresponding to the liquid surface to be rendered based on the buoyancy parameter;

calling a physics engine to determine motion information for the virtual object interacting with the liquid surface to be rendered based on the buoyancy information; and

rendering the virtual object based on the motion information.

11. The method according to claim 1 or 2, wherein the dividing, by the CPU, the model to be rendered into the plurality of grids to be rendered, and resolving the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered comprises:

while the GPU acquires the vertex fluctuation data set, dividing, by the CPU, the model to be rendered into the plurality of grids to be rendered, and resolving the grid fluctuation data corresponding to each of the grids to be rendered to obtain the grid fluctuation data set corresponding to the plurality of grids to be rendered.

12. A virtual scene-based rendering apparatus, comprising:

a first resolving module, configured to call a graphics processing unit (GPU) to resolve vertex fluctuation data corresponding to each vertex to be rendered in a model to be rendered to obtain the vertex fluctuation data set corresponding to the model to be rendered;

a second resolving module, configured to call a central processing unit (CPU) to: divide the model to be rendered into a plurality of grids to be rendered, and resolve grid fluctuation data corresponding to each of the grids to be rendered to obtain a grid fluctuation data set corresponding to the plurality of grids to be rendered; and

a data rendering module, configured to call the GPU to render a liquid surface to be rendered

in a virtual scene based on the vertex fluctuation data set, and call the CPU to render a virtual object for interacting with the liquid surface to be rendered based on the grid fluctuation data set.

13. An electronic device for rendering based on a liquid surface, comprising:

a memory, configured to store a computer-executable instruction; and

a processor comprising a graphics processing unit (GPU) and a central processing unit (CPU), the processor configured to implement, when executing the computer-executable instruction stored in the memory, the virtual scene-based rendering method according to any one of claims 1 to 11.

14. A computer-readable storage medium have stored therein a computer-executable instruction that, when executed by a processor, causes the processor to perform the virtual scene-based rendering method according to any one of claims 1 to 11.

15. A computer program product comprising a computer program or a computer executable instruction, the computer program or the computer executable instruction, when performed by a processor, implementing the virtual scene-based rendering method according to any one of claims 1 to 11.